Introduction

With the rapid growth of computational power and complex algorithms, we propose a novel approach to detect arrhythmias in Phonocardiograms (PCGs). Typically, Electrocardiograms are used to diagnose arrhythmias, requiring medical-grade equipment to recognize cardiac illnesses accurately. However, PCGs provide ease of access to everyone who has a device capable of recording audio, allowing medical professionals to treat arrhythmias in the developmental stages. The new design comprises two subsystems; one is based on the relationship between Electrocardiograms (ECGs) and PCGs, and the other between PCGs and arrhythmias. The association between ECGs and PCGs is amended to translate from one space to another, where ECGs become dimensionally reduced, then reconstructed into a PCG signal. The second subsystem uses a Generative Adversarial Networks (GAN), in which both arbitrary PCG signals and generated signals are fed into a discriminator that detects if an arrhythmia is present or if the signal is false.

Background

An estimated three million cases of arrhythmia occur in the United States yearly (Mayo Clinic), with 300,000 sudden deaths per year – an incidence rather higher than stroke, lung cancer, or breast cancer (American Heart Association). Traditionally, non-invasive arrhythmia analysis is based on multiple electrodes that reflect the electrical activity on ECGs. This method, despite being accurate, limits the use case to hospitals and clinics with specialized equipment; thus, limiting the portability of diagnosing, let alone classification of the type of pathology.

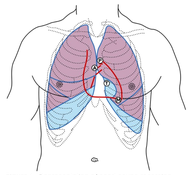


Figure 1. Representation of heart sound recording positions.

Phonocardiograms (PCGs) are sounds that are created by the mechanical movement of the heart. This physical movement produces four distinct sounds: S1, S2, S3, S4, and murmurs. S1 and S2 are sounds created by a healthy heart; whereas, S3, S4, and murmurs refer to diseases or anomalies.

The first heat sound, S1, marks the start of Systole. Systole occurs when the heart muscle contracts and pumps blood from the chambers into the arteries. The second heart sound, S2, marks the end of Systole and the start of Diastole. Diastole is a phase of the heartbeat when the heart muscle relaxes and allows the chambers to fill with blood.

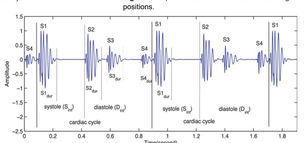
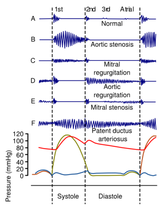


Figure 2. Illustrates the S1, S2, S3, and S4 biomarkers of heart sounds.



Phonocardiograms from normal and abnormal heart sounds

Figure 3. Representation of different abnormalities in sound and pressure.

Although heart sound databases do exist, these datasets are still limited by the number of pathologies that are collected, often having to divide the dataset into two categories: normal and abnormal. Currently, only three major supervised PCG datasets exist: PhysioNet Classification of Heart Sound Recording Challenge dataset, PASCAL Heart Sound Challenge dataset, and the Heart Sound and Murmur Library. The presently available PCG datasets have a limited number of samples and do not cover the complete range of pathologies that are likely to be encountered in clinical settings.

In diagnosing heart sounds, two major challenges arise: localization and classification. Localization aims to find the position of the aforementioned biomarkers in heart sounds. By doing this, heart sounds can be segmented into signals containing a single heart sound. Furthermore, classification attempts to categorize heart sounds into normal and abnormal groups by exploiting the information extracted from localization.

Conventional heart sound localization and classification methods involve time, frequency, or both, and are typically dependent on machine learning algorithms to enhance the results. These algorithms typically include artificial neural networks (ANNs), support vector machines (SVMs), self-organizing maps (SOMs), and are limited to the number of samples and pathologies covered in a given dataset. This leads to a surface-level analysis of the heart sounds.

The main challenge of ineffective heart sound detection stems from an analysis of noisy heartbeats, e.g., background noise. For clean datasets, e.g., the PhysioNet Challenge dataset, a varieties time and frequency of methods converged on localization accuracy of 96.9% (Fernando et al.) and 96.0% classification accuracy (Mostafa et al.). For large datasets with noisy signals, e.g., the PASCAL Challenge dataset, the performance of time and frequency methods remained inconsistent at a localization accuracy of 93.3% (Singh et al.) and 93.3% classification accuracy (Singh et al.).

From the viewpoint of practical applications, the development of computationally efficient solutions is extremely important to the success of a model's deployment. Many studies have negated to comment on the practicality of their proposed methods. From our research, we have concluded only two studies have noted their time efficiency, (Fernando et al.) and (Messner et al.). The fastest model processed 1000 heart state classifications in 56.88 seconds (Fernando et al.), suggesting the model can process 18 bps. The average heart rate of a human heart is around 60-100 bps; thus, current models need severe optimization to achieve near to real-time analysis. These results are excluding the classification of heart arrhythmias.

Thus, the problem of computationally **efficient** and **accurate** classification of noisy heartbeats, especially with datasets with a **variety of pathologies** still remains a problem.

Engineering Goals

1) **Develop a System for End-to-End Heart Sound Arrhythmia Detection**— Create a system that is able to record and analyze heart sounds for Cardiovascular modalities. The system should implement an adversarial model that is both time and space-efficient, and accurate

2) **Increase the Number of Cardiovascular Pathologies**—Develop a model to construct heart sounds from pre-existing data.

3) **Real-World Testing** — Test the end-to-end system in a real-world environment to ensure practicality and generality of the system.

Methodology

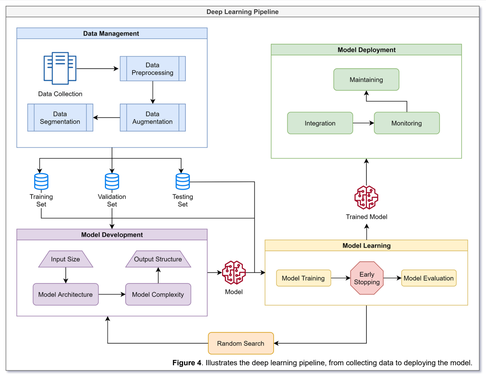


Figure 4. Illustrates the deep learning pipeline, from collecting data to deploying the model.

PCG-Net & VQGAN:

Generative Adversarial Networks for PCG Arrhythmia Detection

Data Management

Data Collection

Although PCG signals are analyzed less often than ECG signals, these signals are rather analyzed in real-time by physicians and healthcare workers. Preliminary studies were done on PCG segmentation and classification primarily used private datasets. Hence, there existed no publicly available datasets until recently. Since then, many public datasets have been developed aiding researchers in their studies and creating open benchmarks for researchers to use in comparing similar findings. However, these datasets are still limited by the number of classes that are collected, when compared to ECG datasets.

Currently, only three major supervised PCG datasets exist: PhysioNet Classification of Heart Sound Recording Challenge dataset, PASCAL Heart Sound Challenge dataset, and the Heart Sound and Murmur Library. These datasets are all anonymized and de-identified for the safety of their subjects, and thus includes no personal information such as name, income, age, etc.

The PhysioNet Classification of Heart Sound Recording Challenge dataset was produced as a part of the 2016 PhyisoNet Computing in Cardiology Challenge. The heart sounds were collected from both clinical and non-clinical environments (in -home visits). The challenge focused on creating an accurate dataset of normal and abnormal heart sound recordings, especially in real-world (extremely noisy and low signal quality) scenarios. These recordings were sourced from nine independent databases and in total, contain 4,593 heart sound recordings from 1072 subjects, lasting from 5-120 seconds. Of which, 409 recordings that were collected from 121 patients contain one PCG lead and one simultaneously recorded ECG. Though, all recordings were resampled to 2,000 Hz using an anti-alias filter. Furthermore, the dataset is comprised of 3 classes: normal, abnormal, and unsure (this is due to poor recording quality), and have the following proportion respectively: 77.1%, 12,0%, 10.9%.

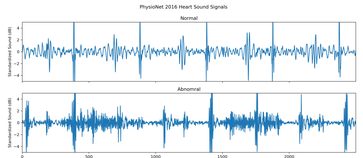


Figure 5. Plots the classes of the PhyioNet 2016 heart sound dataset (Normal and Abnormal).

The PASCAL Classifying Heart Sounds Challenge dataset was released to the general public in 2011. The challenge consisted of two sub-challenges: heart sound segmentation, and heart sounds classification; these sub-challenges corresponded with dataset A, and dataset B respectively. Both datasets have recordings of varying lengths, between 1 second and 30 seconds. Dataset A was collected via the iSethoscope Pro iPhone app, and contained 176 heart sound recordings. 124 of which are divided into four classes: Normal (31 recordings), Murmur (34 recordings), Extra heart sound (19 recordings), and Artifact (40 recordings); the rest of the records are unlabeled for testing purposes. Dataset B was collected using a DigiScope (a digital stethoscope), and included 656 heart sounds. All except 370 were separated into three classes: Normal (320 recordings), Murmur (95 recordings), and Extra-systole (46 recordings). Both datasets A and B vary in sound recordings between lengths of 1 second and 30 seconds.

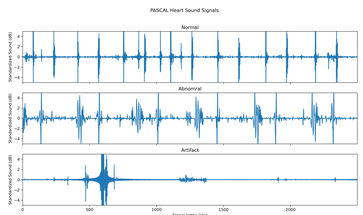


Figure 6. Plots the classes present in the PASCAL dataset (Normal, Abnormal, and Artifact).

More than 300 million ECG recordings are analyzed yearly, and thus create an exceptional tool for arrhythmia classification. Coupled with the recent surge in research interest in 2015, many massive publicly available datasets have been published, notable by PhysioNet - the moniker of the Research Resources for Complex Physiologic Signals. Numerous, datasets ECG exist, however, many are limited to few classes (Normal and Abnormal). At present, three public datasets exist that have more than 4 classes: AF Classification Challenge 2017, PTB Diagnostic ECG, and PTB-XL dataset. Additionally, iRhythm Technologies have developed a semi-public dataset, that is available upon request, that contains 12 classes.

The PTB-XL is the largest publicly available dataset for ECGs and contains 21,837 clinical 12-lead ECG recordings from 18,885 patients of 10 second length. These recordings are separated into 5 super-classes: Normal, Myocardial Infraction, Hypertrophy, ST/T-Change, and Conduction Disturbance. These super-classes are further split into 71 sub-classes that range from AV Block to Posterior Myocardial Infraction. The raw signal data were downsampled to 100 Hz and annotated by up to two cardiologists, who assigned potentially multiple ECG statements to each record.

iRhythm Technologies developed a large, 12 classes ECG dataset using raw single-lead ECG inputs. The 12 classes include Atrial fibrillation and flutter, AVB, Bigeminy, EAR, IVR, Junctional rhythm, Noise, Sinus rhythm, SVT, Trigeminy, Ventricular tachycardia, and Wenckebach. The dataset consists of 91,232 ECG recordings from 53,549 patients. This training dataset is available upon request under license from iRhythm Technologies, Inc. The publicly available test dataset contains 328 records collected from 328 unique patients, split between 6 classes. Both datasets were recorded using a Zio monitor, which monitors the heart through a single-lead sensor at 200 Hz. The annotation was done by a consensus committee of expert cardiologists.

The PhysioNet AF Classification database, presented in 2017 for the Computing in Cardiology Challenge, contains 8,528 ECG recordings, divided into 4 classes: Normal (5154 recordings), Atrial Fibrillation (771 recordings), Other arrhythmias (2557 recordings), and Noisy (46 recordings). The single-lead recordings last from 9 - 61 seconds, with a mean of 32.5 seconds and a standard deviation of 10.9 seconds. The ECG recordings were sampled to 300 Hz and provided in MATLAB V4 WFDB-compliant format.

Data Preprocessing

PCG recordings often are recording in non-ideal environments that are filled with unwanted background noise and interference. Data preprocessing is the process of altering the data in the signal, often by denoising, normalizing, standardizing, and transforming the signal. These steps are crucial for automatic localization and classification tasks. Preprocessing the data allows a model to extract meaning features efficiently and reveals the physiological structure of the heart sounds [Latif et al.]. Furthermore, preprocessing helps ensure that the data that is fed into the model is always in the same domain. This allows the model to generalize more easily.

We first resample the data to 500 Hz, to decrease the spatial resolution of the heart sound recordings, but still retain important features. Thus, helping the model to converge faster. The resampled data is standardized using the standard score equation. This scales the mean of the distribution to 0, artificially scaling all data into similar ranges, thus, helping combat the exploding gradient problem.

The standardized data is then fed into a CycleGAN that has learned to denoise data. The CycleGAN is fed synthetically noised PCG signal and attempts to construct the denoised data from the noisy data. This synthetic noise consists of white noise, pink noise, and real background noise collected from audio recordings. The noise is added to each PCG signal recording and then treated as the input to the CycleGAN. The CycleGAN's output is compared to the original, non-noise, PCG recording. In this way, the CycleGAN eventually learns to denoise PCG recordings.

Data Segmentation

Data segmentation refers to the process of creating cross-validation datasets. This process assists in validating if the model is overfitting to the dataset. These datasets include the training set, validation set, and testing set. Typically, the training set is 70%-80% of the dataset, the reset of the dataset is split among the validation set and testing set. Here, we split the data 80% training, 10% validation, and 10% testing.

Data Augmentation

Data augmentation is a strategy that enables a significant increase in the diversity of data available while training a model, without actually collecting new data. Data augmentation techniques aim to slightly alter existing data to a point where the model cannot recognize the augmented data as one it has trained on before, but still retains the characteristics of the data's category. This helps in reinforcing important features within the data and is only done during the training portion of the workflow.

A common misconception arises when comparing preprocessing and augmentation. To be clear, preprocessing aims to clean the data of unwanted artifacts that are not meant for classification and is done in place. Augmentation, on the other hand, is solely done for expanding the dataset's size, often to combat overfitting. Augmenting the data before preprocessing further obscures the data unrealistically, and beyond classification.

We use to resample the heart sound recordings to different frequencies to simulate slower and faster beats per minute (bpm). The normal bpm for a human is between 60-100 bpm. Thus, measuring the sample distance between the first S1 (the start of systole) and the second S1, we calculate the bps and resample accordingly.

Furthermore, we use noise injection directly to preprocessed PCG recordings [Messner et al.]. This process is identical to the process of synthetically adding noise to PCG recordings described in the preprocessing step. A variety of noises, like white noise, is added to the signal to increase the sample of recordings per class. This method is extremely beneficial for training on small datasets, like the PASCAL dataset.

Model Development

Model Architecture

Here we propose using Generative Adversarial Networks (GANs) for increased success in PCG heart sound detection. GANs pose a unique advantage over traditional machine learning and deep learning methods, in that a model learns to mimic a dataset by creating its own data, and tries to fool a discriminator into thinking the generated data is real.

In a supervised approach, a GAN consists of two parts, a generator and a discriminator. The generator is responsible for creating fake heart sound data, while the discriminator tries to predict where the incoming data is fake or real. In a semi-supervised approach, however, the is fed data from a real dataset and the generator. Here, the discriminator tries to classify the generator's fake data, as well as predict the classes form the real dataset.

Model Complexity

Traditionally, generators are dense layers that slowly increase the dimensionality of the generated data to match that of the real dataset. Discriminators, on the other hand, are commonly CNNs because the majority of their applications work with images. However, it is possible to use a wide variety of architectures; such as LSTMs, RNNs, SVMs, DNNs, ANNs, Transformers.

As mentioned above, there are many types of model architecture, some are used for classification, and others for feature extraction. Optimizing the combination of feature extraction layers and classification layers is extremely time-consuming and computationally taxing. This is because there exist many combinations of hyperparameters, thus making it difficult to optimize each parameter. To optimize hyperparameters, we used hyperparameter sweeps to make the optimization process more efficient. This method involves using one of three methods: grid search, random search, and Bayesian search. Grid search computes each possible combination of all hyperparameters and tests them all. Although this is very effective, it can be computationally costly. Random search selects a new combination at random, provided a distribution of values. This method is surprisingly effective and scales very well. Bayesian search creates a probabilistic model of metrics and suggests parameters that have a high probability of improving metrics. This works well for small-scale projects, but scales poorly as the complexity of parameter relationships increases. Here, we used a random search to optimize our hyperparameters.

Model Learning

Model Training

During the training phase, the model is trained using backpropagation in conjunction with a cost function. Backpropagation attempts to calculate the gradient of the cost function with respect to the weight and biases of the model. This process involves an optimizer, which optimizes the model's parameters and a cost function that measure the correctness or incorrectness of the model. The goal of the optimizer is to minimize the cost function's error by adjusting the parameters to the given label. In this study, we used the Adam optimizer in union with Cross-Entropy Loss. The Adam optimizer uses a hyperparameter that dictated the change in the model's parameters on each backpropagation step, this is called the learning rate. Here we choose a learning rate of 0.0001.

The model is only trained on the training set; thus, backpropagation only occurs on the training set. Additionally, for each step in the training set, the optimizer backpropagates and optimizes the parameters and calculates metrics to further evaluate the model. The amount of steps in the training set is dictated by the batch size, the number of signals the model is trained on, in a single forward pass. Here we use a batch size of 32, meaning that the model is fed 32 signals per input. This significantly speeds up the process of training as more signals are passed through the model every time the model is optimized. A full pass of the training set is called an Epoch, here we train the model on 100 Epochs.

Model Training

To ensure the model is not overfitting, but generalizing to the training set, we use a validation set to track the metrics of the model. In theory, the metrics on the training set equal to that of the validation set. In practicality, after many epochs of training the metrics of the validation set become static, but the metrics of the training set still increase. This suggests that the model is overfitting. Thus, we stop training the model on the training set and test it as a testing set.

Model Evaluation

Testing sets or hold-out sets are used to validate the metrics of the model, this is because both the validation set and the testing set have been tested by the model; thus, the model has developed a latent bias to both sets. Therefore, a third set is needed to assess the model's ability to generalize on an independent dataset. The model is tested with the metrics below:

Specificity:

Tp = True Positives
Fn = False Negatives

$$\frac{T_p}{T_p+F_n}$$

Sensitivity:

Tn = True Negatives
Fn = False Positives

$$\frac{T_n}{T_n+F_p}$$

Accuracy:

Tp = True Positives
Fn = False Negatives
Tn = True Negatives
Fp = False Positives

$$\frac{T_n+T_p}{T_n+T_p+F_n+F_p}$$

Cross-Entropy Loss:

N = Number of Classes
c = Correct Classes
x = Predicted Classes

$$\mathcal{L}(x,c) = -\ln\frac{e^{x[c]}}{\sum_{i=1}^{N} e^{x[i]}}$$

Model Deployment

Model Deployment is one of the last stages of any machine learning project and involves releasing the model to the public.

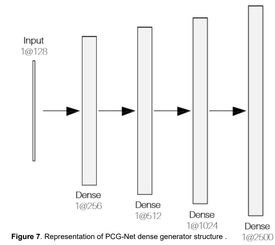
Integration

Integration consists of implementing the model in a system, whether it happens on the client-side or the backend. The most popular backend model integration tools involve Flask, Azure, and FastAPI. These tools create APIs that encapsulate the model prediction, given a GET request with the desired input.

Monitoring & Maintaining

Following model integration and deployment, we move onto the next phase, monitoring and maintaining the system. As more and more data passes through the model, it increases the opportunity for the model to learn from a more generalized dataset. Though such data would be unsupervised, we could use unsupervised techniques to categories the data. Based on the improvement of the model, the model and be reintegrated and deployed. In essence, looping the whole process from data management to model learning.

PCG-Net: Dense Generator

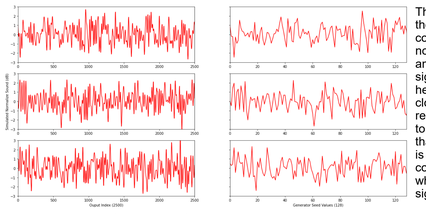


The input consists of a 2-dimensional tensor (batch size x signal) with a length of 128. This input is randomly generated.

The Dense layers upscale the length of the input vector until the length reaches that of a real input signal (2500). This allows for the output of the generator to be directly fed into the discriminator.

The Rectified Linear activation function alters the range of the incoming data by setting all numbers below 0 to 0 and leaving all positive numbers intact.

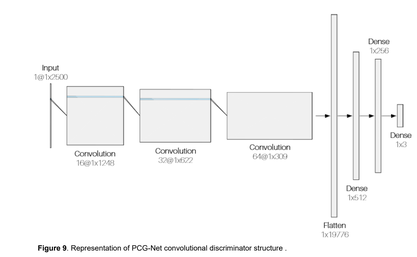
Figure 7. Representation of PCG-Net dense generator structure .



The graph to the right shows the input values (seeds) and corresponding generated noise. The variance in amplitudes and periods in the signal mimics that of real heartbeats changing. At a closer examination of the results, the generator appears to resample the seed values to that of the output (2500). This is expected as the generator consists of dense layers, which linearly increase the signal length and complexity.

Figure 8. Plot of the noise (artifact class) generated from the generator.

PCG-Net: Convolutional Discriminator



The input consists of a 3-dimensional tensor (batch size x channel size x signal) with a length of 2500.

Convoluting decreases the size of the input vector. A vector (with a size of 1x5) filters across the data by multiplying all values in the input vector by the filter vector. This method also assists the model in finding features.

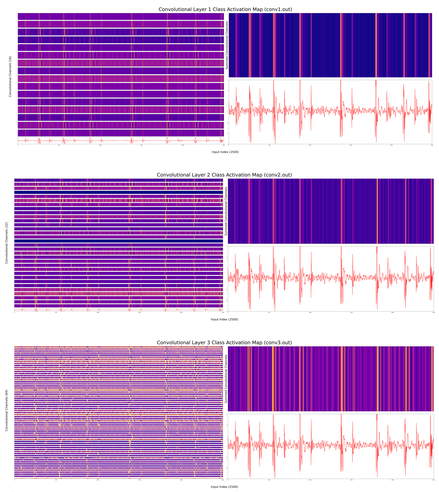
Figure 9. Representation of PCG-Net convolutional discriminator structure .

The Rectified Linear activation function alters the range of the incoming data by setting all numbers below 0 to 0 and leaving all positive numbers intact.

The liner function flattens the incoming result (batch size x 64 x 309) into a 2D tensor (batch size x19776).

The Dense layers downscale the length of the input vector until the length reaches that of the number of classes. This allows for the output of the discriminator to be directly interpreted by the cost function.

The Linear Output layer transforms the Linear layer output into a 1x3. Each column in the tensor represents a class's likelihood of being the correct class in the dataset. Thus, the column with the largest values is the model prediction for the input.



The graphs to the left visualize the channels of the convolutional layers, which are responsible for extracting features from an inputted signal. The color of the vertical line represents the importance of the feature relative to other features. Meaning, the lighter the color, the more important the feature. As the input travels deeper into the model, the extractions get more complex. This is clearly shown by the district increase in coloration between layers one and two. This indicates that the layers are extracting latent features from the signal.

The first convolutional plot depicts the extraction of important biomarkers, such as S1 and S2. In the next two layers, the extractions make it clear that the S1 sound is more important, as those are the brightest throughout all layers. This parallels medical knowledge, as most cardiovascular anomalies occur in Systole, or at the start of S1.

Figure 10-12.Nomral activation maps of convolutional layers of the PCG-Net discriminator .

PCG-Net: Raw Results

Table

Table 1. The table shows the accuracy, sensitivity, and specificity of 150 trials of training the GAN model on the dataset. The metrics displayed are the results of the testing set.

Accuracy: percent of correctly identified normal and abnormal heart sounds.

Sensitivity: percent of correctly identified abnormal heart sound recording from a sample of abnormal heart signals.

Specificity: percent of correctly identified normal heart sounds recordings from a sample of normal heart signals.

PCG-Net: Testing Sample Distributions

The graphs on the right illustrate that the GAN's accuracy, specificity, and sensitivity are skewed to the left. This suggests that the model is extremely successful at differentiating between abnormal and normal heart sounds.

The specificity distribution has a mean of **0.9030**, with a standard deviation of **0.0547.** The best model reached a specificity of **0.9672.** This suggests that ~**90.3**% of normal heart sounds were correctly identified.

The sensitivity distribution has a mean of 0.9952, with a standard deviation of 0.0197. The best model reached a sensitivity of 1.0. This suggests that 99.5% of abnormal heart sounds were correctly identified. In detecting pathologies in medicine, we often attempt to maximize sensitivity, the rate at which a subject with a disease is correctly identified amongst other ill subjects. This is because we want to ensure that all potential subjects with a disease are sent for further examination. Essentially, weighting sensitivity higher than specificity, the rate of correctly identified normal or healthy patients from a sample of healthy patients. Thus, from the results, the proposed model is robust, in that it can detect abnormalities nearly ~100% of the time.

The accuracy distribution has a mean of 0.9498, with a standard deviation of 0.0293. The best model reached an accuracy of 0.9875. This suggests that ~95.0% of both abnormal and normal sounds were classified correctly.

Figure 13. Plot of testing set results on specificity, sensitivity, and accuracy.

PCG-Net: Generalization Statistics

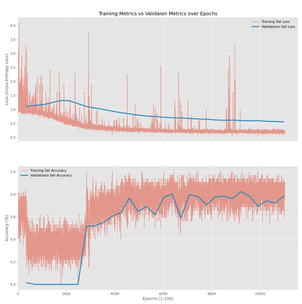


Figure 14. Plot of validation and training set accuracy and loss over 100 epochs.

Generalization is important in creating accurate predictions, as it establishes that the model is learning meaningful features that are not just applicable to the training data, but signals overall. The graphs to the left plot the loss and accuracy of the training and validation set over each epoch. The large fluctuations in the training set metrics are caused by logging the metrics after each step in the training set (on every change in the model parameters). Thus, the optimizer is bound to decrease gradients in the wrong direction, thus correcting for such variations cause those fluctuations. Both lines on the loss plot resemble an exponential curve, which suggests the model continues to learn as training progresses. The average deviation between the validation and training set for each epoch is 0.4817; though this deviation is high, it is due to the lack of meaningful surface-level features that would lead to accurate detection. Meaning, the model's deep feature extraction layers are responsible for the gap. Furthermore, the accuracy plots follow a logarithmic curve. In other words, as the training accuracy increasing, the validation correspondingly increases, though at a slower rate. Both graphs illustrate the model has reached convergence by the end of the training phase. This is confirmed by the static change in metrics in both datasets.

PCG-Net: Dataset Feature Visualization

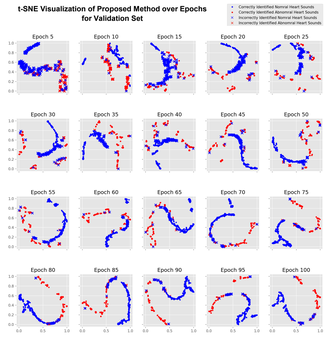


Figure 15. Plot of t-SNE visualizations over 100 epochs of the validation set.

Dataset visualization is critical in understanding the dataset's complexity and model's effectiveness. Here, we use t-distributed stochastic neighbor embedding (t-SNE), a statistical method for visualizing multi-dimensional data in less computationally expensive dimensions. The method is presented with the raw prediction values for each input of the validation set and maps the corresponding predictions into a 2-dimensional space. Tracked over epochs, the visualization allows us to view the progression of the model's competence while training. The visualization highlights clear clustering within the dataset, which suggests the model is stable. Though, from epochs 70 and onwards, it is evident that there is overlapping between abnormal and normal signals. Assuming these signals as ground truth, this implies that additional feature engineering is required to adequately classify heart sounds.

PCG-Net: Time Complexity

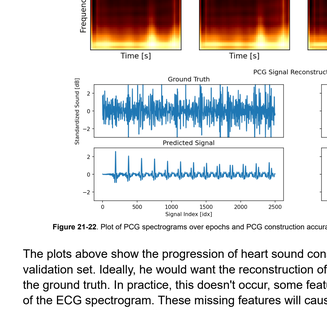


Figure 17. Time complexity of discriminator in classifying heart sounds.

Model complexity is used to gauge and evaluate the efficacy of a model against an increase in data (n). We mainly focus on time complexity as it is most relevant to the problem at hand (space complexity is O(1)). Depending on model deployment and integration, the complexity can vary. For example, GPUs have parallel processing capabilities, which allow them to process multiple signals at once, efficiently decreasing the model complexity to O(1). For this reason, we use the worst-case scenario (a CPU), for analysis of the proposed method's time complexity. The graph to the left implies the model's time complexity is directly and linearly correlated to the input size, suggesting the complexity is O(n). Thus, the model on average, can predict 2800 heart sounds in the worst case scenario. This will prove to be greatly helpful in real-time detection.

VQGAN: Data Transformation

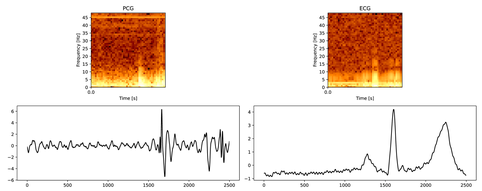


Figure 19. Comparison of PCG and ECG log spectrograms.

Traditional machine learning reconstruction techniques use images because they contain dense information in a low dimensionality. However, time-series data, such as heart sound signals, are one-dimensional. Thus, we use time and frequency analysis, specifically log spectrograms to represent heart sound recordings in a two-dimensional manner. These spectrograms show the frequency of the signal against time. This process is done on both the PCG and ECG signals and results in a 48 x 48 image.

VQGAN: PCG Construction Visualization

Figure 21-22. Plot of PCG spectrograms over epochs and PCG construction accuracy.

The plots above show the progression of heart sound construction from ECGs over epochs of the validation set. Ideally, he would want the reconstruction of the PCG spectrogram to identical to that of the ground truth. In practice, this doesn't occur, some features may be lost in the latent representation of the ECG spectrogram. These missing features will cause a spatial anti-aliasing effect, as the latent space doesn't have the dimensionality to extract pixel-to-pixel information. The series of spectrograms show the development of features in the latent space through the epochs. For example, the frequency of the first peak (S1) varies significantly until the ~86th epoch. This feature is important because it determines the rate of the S1 or "lub" sound; thus, creating the illusion that the sound is occurring faster relative to the ground truth. Furthermore, the S2 marker is severely softened, this is due to the rapid change in frequencies surrounding the marker and the light vertical bars to the right in each spectrogram. This suggests that much of the information regarding S2 will be lost when converting the spectrogram into a wave signal.

PCG-Net: Confusion Matrix

The confusion matrix to the right aids illustrates the performance for each class of the proposed method. Specifically, we evaluated the model's success on the grounds of average accuracy, specificity, and sensitivity of the classification. We calculated the average true positives, false positives, etc. for all 150 trials of the testing set. Using these floored values, we normalized each label along the y-axis. The matrix reveals that the most common misunderstanding occurs in predicting a normal heart sound, this is not surprising as the model is trained to be biased in detecting abnormal heart sounds. However, this occurs at the cost of specificity, which decreases the average normal misclassification rate to 8.11%. Overall, we conclude that the average accuracy of abnormal heartbeat detection is ~95% with a misclassification rate of just 5%. Thus, the model is extremely accurate in detecting abnormalities in heart sounds and has the capabilities to further classify abnormal heart sounds into labeled arrhythmias.

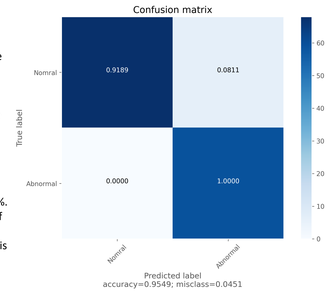


Figure 16. Matrix of accuracy between labels in the dataset.

PCG-Net: Statistical Significance

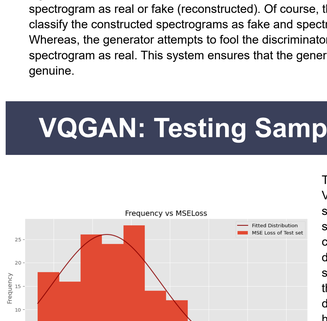


Figure 18. Comparison pervious state-of-the-art and proposed method in specificity, sensitivity, and accuracy.

P(T<=t) one-tail:

Specificity: 1.65E-10
Sensitivity: 1.54E-77
Accuracy 1.07E-11

From the p-values shown above, we can conclude that all metrics were statistically significant because all p-values were less than 0.05. This implies that the null hypothesis can be rejected and the alternative hypothesis is accepted. The decrease in specificity is expected because we prioritized sensitivity over specificity to maximize the true positive rate, the percent of correctly identified abnormal heart sounds from a sample of only abnormal heart sounds.

VQGAN: Architecture

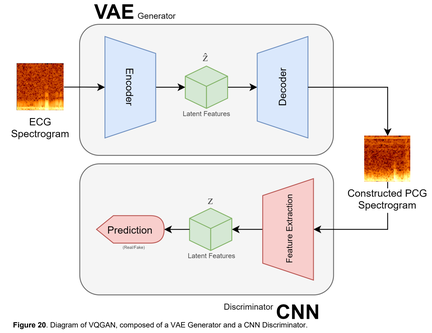


Figure 20. Diagram of VQGAN, composed of a VAE Generator and a CNN Discriminator.

Variational Autoencoders (VAE) is an architecture that attempts to map the initial data into an encoded space (ż) that stores latent features, the encoder is responsible for doing this task. The encoded space is regularized to avoid overfitting. In the context of ECG and PCG signals, both signals come from different latent spaces, which makes it difficult to travel from one domain to another. Regularizing this encoded latent space ensures that the latent space has good properties. This space is then reconstructed by a decoder into an encoded-decoded space, identical to the initial data in shape. With respect to ECG and PCG signals, the input, a spectrogram representation of the ECG signals, is encoded into the encoded space. The encoder consists of ResNet and Multi-head attention blocks that introduce a method to negate the vanishing gradient problem and weights for each pixel in the spectrogram respectively. The encoded space represents important structural elements, such as the positions of the PQRST complex. This space is then decoded using a revered structured encoder, into an encoded-decoded state that respectable a spectrogram of the PCG signal. These reconstructed PCG signals are sent to a convolutional discriminator, which attempts to classify the generated PCG spectrogram as real or fake (reconstructed). Of course, the goal of the discriminator is to classify the constructed spectrograms as fake and spectrograms from a PCG dataset as real. Whereas, the generator attempts to fool the discriminator into classifying the generated spectrogram as real. This system ensures that the generated spectrogram looks authentic and genuine.

VQGAN: Testing Sample Distributions

Figure 23. Distribution plot of testing results from PCG spectrogram construction.

The graph to the left shows that the VQGAN's loss is skewed to the right, but still has a large amount of variability. This suggests that the model has the ability to construct PCG spectrograms, but has a difficult time accurately constructing all spectrograms. This is supported by both the mean loss - 1.34, and the standard deviation - 0.258, as both values convey a high level of variance. We believe the inconsistencies are caused by the narrow gap of meaning full data. Markers, such as S1 and S2, only occur for a narrow amount of time relative to the signal size. Thus, the majority of the signal's data is nonimportant and contains background noise.

VQGAN: Raw Results

Table

Table 2. The table shows the loss of 150 trials of training the VQGAN model on the ECG and PCG datasets. The metrics displayed are the results of the testing set.

Loss: A vector that represents the correctness and confidence of the model's prediction. A low loss suggests the model was correct and confident in its answer.

Real-World Test

Testing the model's viability is crucial for ensuring the model's success in the real world. Ideally, recording heart sounds are recorded with digital stethoscopes. These tools use transducer technology to convert sound into an electrical signal. Over the past decade, this technology has grown immensely (by cause of speech recognition). Modern phones have the potential to record the sounds at a high resolution, given the microphone is located at the correct position relative to the heart. Such a device will prove to extremely beneficial in providing diagnosis without the need for specialized equipment. The graph to the right shows a heart sound recording from a phone microphone. The plot shows that import biomarkers, like S1 and S2, remain visible. This ensures that the recording doesn't contain excessive amounts of noise that may hinder the performance of the detection system. Hence, feeding it into the proposed method resulted in a normal classification.

Figure 24. Plot of heart sound recording taken from a phone microphone.

Conclusion

We proposed a Generative Adversarial Network (GAN), composed of a Dense Generator and a Convolutional Neural Network (CNN) Discriminator to detect abnormal heart sounds in a recording. The model achieved an accuracy of 94.98%, a specificity of 90.30%, and a sensitivity of 99.52% on the testing set. The previous state-of-the-art achieved an accuracy of 86.02%, a specificity of 77.81%, and a sensitivity of 94.24%. This data, along with results from the t-test revealed that the proposed alternative hypothesis was correct and that the null hypothesis should be rejected. This is because the proposed method reached better performance than the previous state-of-the-art methods. Additionally, the model attained a staggering ~2500 classification per second in the worst-case scenario. This is because of the nature of the CNN architecture; unlike other methods, the CNN's reduce the data dimensionality as it forward propagates through the model. Furthermore, the proposed method showed real-world deployment capabilities for autonomous heart sound abnormality detection with recordings collected from a phone microphone. This test shows extremely promising results for future applications and integrations.

We also set out to introduce new pathologies for increased arrhythmia labels in classification. We proposed using a VQGAN for constructing PCG signals from existing ECG datasets that contain a surplus amount of arrhythmia-specific data. The results were promising, in that the VQGAN discriminator was able to construct the general shape of the PCG spectrogram, but missed import details in the fluctuation of important biomarkers (S1 and S2). This caused the PCG waveform representation extracted from the PCG spectrogram to miss rapid oscillations present in the biomarkers.

The object of this study was to create a fast and accurate end-to-end heart sound arrhythmia detection system, capable of detecting abnormalities in real-time without specialized equipment. While also increasing the number of cardiovascular pathologies classified. With the data shown, our proposed method accomplishes exemplary statistics in abnormalities detection and shows promising results in increased arrhythmia construction. Hopefully, this study will shed light on PCG construction techniques and give birth to applications with autonomous abnormality detection.

Further Exploration and Application

- Deploying the model with an app that is available to 3rd world countries that can't afford to conduct in-depth testing regularly
  - Integrate and serve the model using FastAPI
- Use abnormal heart sound unsupervised datasets as a basis of categorical arrhythmia classification
  - Using low dimensional visualization techniques like t-SNE or UMAP
  - Cluster data using methods like K means and hierarchical clustering
- Create a classifiable latent representation of PCG signal biomarkers that can be represented with accuracy and precision
  - Created by VAE that are fed the PCG signals directly instead of a spectrogram
- Investigate training a heart sound discriminator from generated PCG data from ECG datasets
- Reconstructing speech (wav) recordings from the human auditory cortex (EEG) using techniques used for PCG construction