EE 678: Wavelets

Wavelet Theory & Deep Learning Based Detection of Fetal QRS Complex in Noisy Abdominal ECG Signal

Arbaz Shaikh
Electrical Engineering, IIT Bombay
Email: arbazsh10@gmail.com

Aman Butoliya
Electrical Engineering, IIT Bombay
Email: amanbutoliya16@gmail.com

Nithin Lakshmisha (Mentor)
Biosciences and Bioengineering, IIT Bombay
Email: nithin.lakshmisha@gmail.com

Abstract

We propose a methodology for detection of fetal QRS complex in a noisy abdominal ECG signal using concepts from wavelet theory, and deep learning. Noisy ECG signal are obtained by placing electrodes on mother's abdomen, thus the ECG signal is obtained in a non-invasive way. Our methodology employs continuous wavelet transform to visualize the time-frequency components of the noisy ECG signal. Later we have used pre-trained VGG16 deep convolutional neural network architecture for extracting feature vectors of scalograms generated for the ECG signal in consideration. Further, we have used a neural network model (Multilayer Perceptron) to classify these feature vectors. Finally, we take ensemble of the predictions from four different abdominal ECG signals recorded for the subject in consideration. Our analysis through the simulations performed shows that the proposed model gives us precision, sensitivity, and specificity values of 0.80, 0.85, and 0.97 respectively. We have also provided various improvements possible in the proposed framework, both from wavelet theory, and deep learning perspective. All simulations were performed using Python. Code snippets are included to facilitate a better understanding from the implementation point of view.

Key Words: Fetal QRS detection, Continuous Wavelet Transform, Scalogram, Transfer Learning, Feature Extraction, CNN, Multilayer Perceptron

1 Brief Introduction of Pursued R&D Problem

Worldwide, an estimated 2.65 million stillbirths occur annually, of which about half occur in the intrapartum period such as childbirth [2]. Hence, it is of utmost importance to monitor the fetal ECG signal during the early months of pregnancy and at delivery too. One of the ways to monitor fetal ECG is through fetal scalp electrode (FSE), where an electrode is to the scalp of the fetus. Although this method posses the risk of infection to the fetus as well as it is an invasive method indeed. Alternative method is to collect the fetal ECG signal by attaching the electrodes to maternal abdomen. This is indeed a non-invasive method. However, the major challenge in employing this method for monitoring the fetal activity (mainly fetal heart rate) is the fact that, apart from fetal ECG (FECG) signal, the maternal abdominal signal is composed of maternal ECG (MECG), and noise signals from sources such as baseline wandering, powerline interference, and various other muscular activities in mother's body etc. Hence it is evident that the FECG signal present in maternal abdominal signal has very low signal to noise ratio. Thus, indeed, extracting FECG from the maternal abdominal recordings is a difficult task restricting its use so far.

A widely employed two-step approach is to remove the MECG component first by using, for example, an adaptive filtering algorithm, followed by an attempt to extract the fetal QRS complexes from ECG residuals, which would contain the FECG and noise components. Although, a reference MECG (which is affected by noise) is an essential requirement in this type of algorithms. Since the noisy MECG signal does not satisfy the hypothesis of regressive model which is indeed a case with adaptive filtering algorithm, therefore the performance in MECG cancelling is reduced. Recently, AM_{esn} [1] has been shown as a promising adaptive technique.

Other techniques to tackle the problem under inspection, include but are not limited to linear/non-linear decomposition where the abdominal signal is decomposed into three categories i.e. MECG, FECG and noise. Most commonly used linear decomposition techniques are Principal Component Analysis (PCA) [4], Independent Component Analysis (ICA) [5], Periodic Component Analysis (πCA) [8], and Singular Value Decomposition (SVD) [4]. On the other hand, non-linear decomposition algorithms for fetal ECG extraction usually use deflation method of subspace decomposition and non-linear projection [7]. Although their high computational complexity limits their use in real-time setting. Non-linear decomposition algorithms for fetal ECG extraction are a subject of considerable research. For more details about the reader may refer to an excellent review article by Sameni and Clifford [6].

Major goals of our study for the problem of fetal QRS complex detection in the maternal abdominal signal are as mentioned below:

- Approach the problem under inspection without cancelling the MECG component in maternal abdominal signal.
- Investigate if the problem under consideration can be approached by employing scalogram based time-frequency properties of the (maternal abdominal) signal snippets, instead of using conventional

noise removal techniques used to remove the noise components.

- Built a framework for problem under consideration, using continuous wavelet transform and deep learning architectures and investigate its effectiveness.
- Explore the possibility of using several advanced deep learning algorithms in the later versions of the framework proposed in this article.

The plan of this article is as follows. Section 2 provides the background needed for major topics used in this work. Section 3 is presents the details of the scheme used for the proposed framework along with its Python based implementation, code snippets for which are provided in the appendix. Section 4 contains the numerical experiments performed and the results obtained. Section 5 is dedicated to present the plausible improvements and extensions of the proposed framework. In addition, possible improvements which can be incorporated are stated at various places throughout the article. In Finally some concluding remarks are made in section 6.

2 Background Information

2.1 Continuous Wavelet Transform

Formally, the definition of continuous wavelet transform (CWT) states that, for a function f at any scale s and position u its CWT can be written as a projection of f on the corresponding wavelet atom, ie

$$Wf(u,s) =< f, \Psi_{u,s} >= \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} f(t)\Psi^*(\frac{t-u}{s})dt \tag{1}$$

where, Ψ is a mother wavelet function with zero average, i.e.

$$\int_{-\infty}^{+\infty} \Psi(t)dt = 0 \tag{2}$$

which is dilated with scale parameter s and translated by u to get the wavelet dictionary D i.e.,

$$D = \Psi_{u,s}(t) = \frac{1}{\sqrt{s}}\Psi(\frac{t-u}{s})_{u\in\mathbb{R},s>0} \tag{3}$$

Thus, CWT results in many wavelet coefficients $Wf(u,s)$, where, in theory scale $s > 0$ and $u \in \mathbb{R}$. Indeed these coefficients enable us to represent time series signals as time-scale images in (u,s).

The scale factor s is inversely proportional to frequency. Thus enabling us to capture various frequency components present in the time series function of interest. Below figure illustrates this phenomenon. Whereas, the shift parameter u translates the mother wavelet function in time for different values of u.

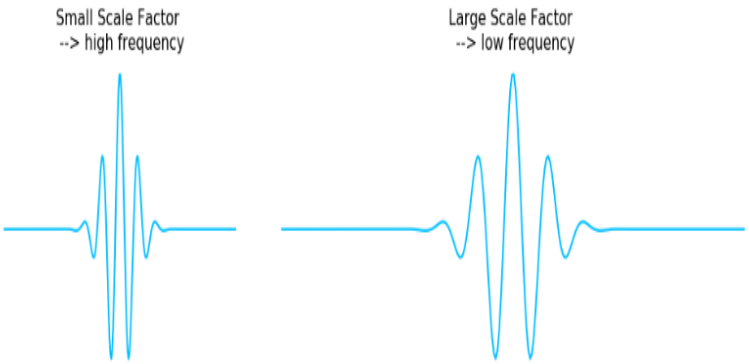

Figure 1: Shrinked and Stretched Morlet Mother Wavelet Function in Time

This definition essentially means that CWT is nothing but correlation between the function and (scaled and translated) mother wavelet function. These coefficients are then used to construct a 2D scalogram. Below figure obtained from [9] helps visualize the definition of the CWT.

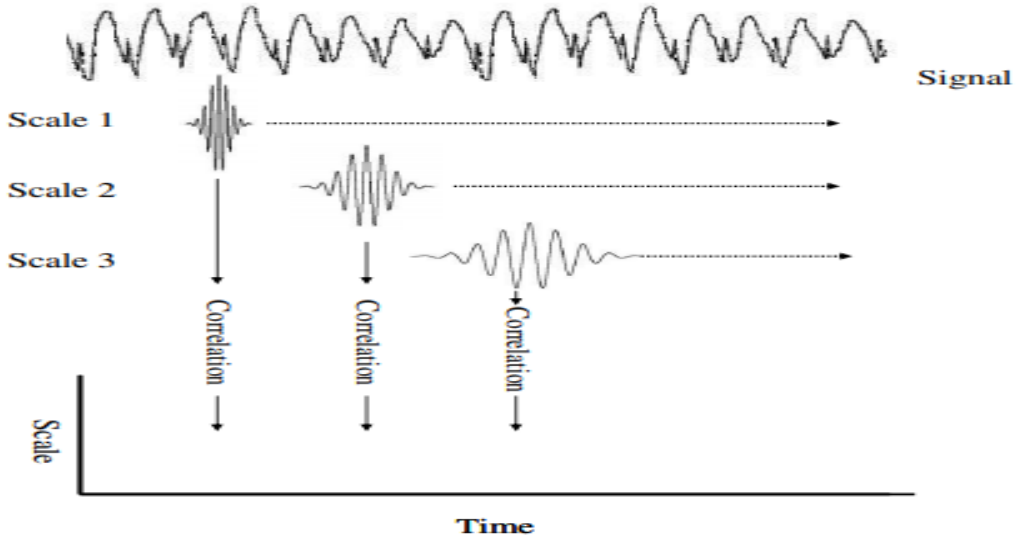

Figure 2: Constructing a Scalogram: A scalogram illustrates how signal activity within a range of time-scales evolves over time. The scalogram is constructed by evaluating the correlation between a signal and wavelets with different scales, and then plotting how the correlation with each wavelet varies over time.

Thus, it is clear that wavelet analysis is a powerful tool for analyzing non-stationary time series because decomposition of time series into time-frequency space allows one to determine both the dominant modes of variability and how those modes vary in time.

2.2 Dataset Used

For our framework, we have used "The Abdominal and Direct Fetal Electrocardiogram Database" [3] which contains multi-channel fetal electrocardiogram (FECG) recordings obtained from 5 different women in labor, between 38 and 41 weeks of gestation. Each recording comprises of four differential signals acquired from maternal abdomen and the reference direct fetal electrocardiogram registered from the fetal head.

Data Description: The configuration of the abdominal electrodes comprised four electrodes placed around the navel, a reference electrode placed above the pubic symphysis and a common mode reference electrode (with active-ground signal) placed on the left leg.

These recordings constitute an excellent material for testing and evaluation of efficacy of new FECG processing techniques, e.g. algorithms for suppression of maternal electrocardiogram in abdominal signals or for detection of fetal QRS complexes. Thus indeed we have chosen to work with this dataset.

Recording information:

- Signals recorded in labor, between 38 and 41 weeks of gestation.
- Four signals acquired from maternal abdomen.
- Direct electrocardiogram recorded simultaneously from fetal head.
- Positioning of electrodes was constant during all recordings.
- Ag-AgCl electrodes (3M Red Dot 2271) and abrasive material to improve skin conductance (3M Red Dot Trace Prep 2236).
- Bandwidth: 1Hz - 150Hz (synchronous sampling of all signals).
- Additional digital filtering for removal of power-line interference (50Hz) and baseline drift.
- Sampling rate: 1 kHz.
- Resolution: 16 bits.
- Input ranges are included in the records in EDF format.

2.3 VGGNet Architecture

VGGNet is the name of a pre-trained deep convolutional neural network (CNN) invented by Simonyan and Zisserman from Visual Geometry Group (VGG) at the University of Oxford in 2014 and it was able to be the 1st runner-up of the ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2014 in the classification task[10]. VGG Net has been trained on ImageNet ILSVRC data set which includes the images of 1000 classes split into three sets of 1.3 million training images, 100,000 testing images and 50,000 validation images.

VGG was invented with the purpose of enhancing classification accuracy by increasing the depth of the CNNs, for our framework we have used VGG16, having 16 weight layers.

- **Input:** VGG Net takes input of 224×224 RGB images and passes them through a stack of convolutional layers.
- **Convolutional Layers:** The convolutional layers in VGG use a very small receptive field (3x3, the smallest possible size that still captures left/right and up/down). There are also 1x1 convolution filters which act as a linear transformation of the input, which is followed by a ReLU unit. The convolution stride is fixed to 1 pixel so that the spatial resolution is preserved after convolution.
- **Fully-Connected Layers:** VGG has three fully-connected layers: the first two have 4096 channels each and the third has 1000 channels, 1 for each class.
- **Hidden Layers:** All of VGG's hidden layers use ReLU (a huge innovation from AlexNet that cut training time). VGG does not generally use Local Response Normalization (LRN), as LRN increases memory consumption and training time with no particular increase in accuracy.

In our study, we employ VGG16 architecture for extracting the features of scalogram images obtained. Although VGG16 was not trained on scalograms images but we it can be used to extract features for scalogram images. This type of mechanism is famously known as transfer learning in the field of machine learning and deep learning. To achieve the aforementioned task we remove the top layer from the VGG16 architecture which then allows us to extract 512 dimensional feature vectors for scalogram images.

2.4 Multilayer Percentron Model

The field of artificial neural networks is often just called neural networks or multi-layer perceptrons after perhaps the most useful type of neural network. A perceptron is a single neuron model that was a precursor to larger neural networks. A MLP is a class of feed forward Artificial Neural Network (ANN). It can distinguish data that is not linearly separable.

- **Activation Function:** If a multilayer perceptron has a linear activation function in all neurons, that is, a linear function that maps the weighted inputs to the output of each neuron, then linear

algebra shows that any number of layers can be reduced to a two-layer input-output model. In MLPs some neurons use a nonlinear activation function that was developed to model the frequency of action potentials, or firing, of biological neurons. The more frequently used activation function is Rectifier Linear Unit (ReLU).

- **Layers:** The MLP consists of three or more layers (an input and an output layer with one or more hidden layers) of nonlinearly-activating nodes. Since MLPs are fully connected, each node in one layer connects with a certain weight w_{ij} to every node in the following layer.
- **Learning:** Learning occurs in the perceptron by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result. This is an example of supervised learning, and is carried out through backpropagation, a generalization of the least mean squares algorithm in the linear perceptron.

An example of MLP model with one input, one output and one hidden layer is shown below.

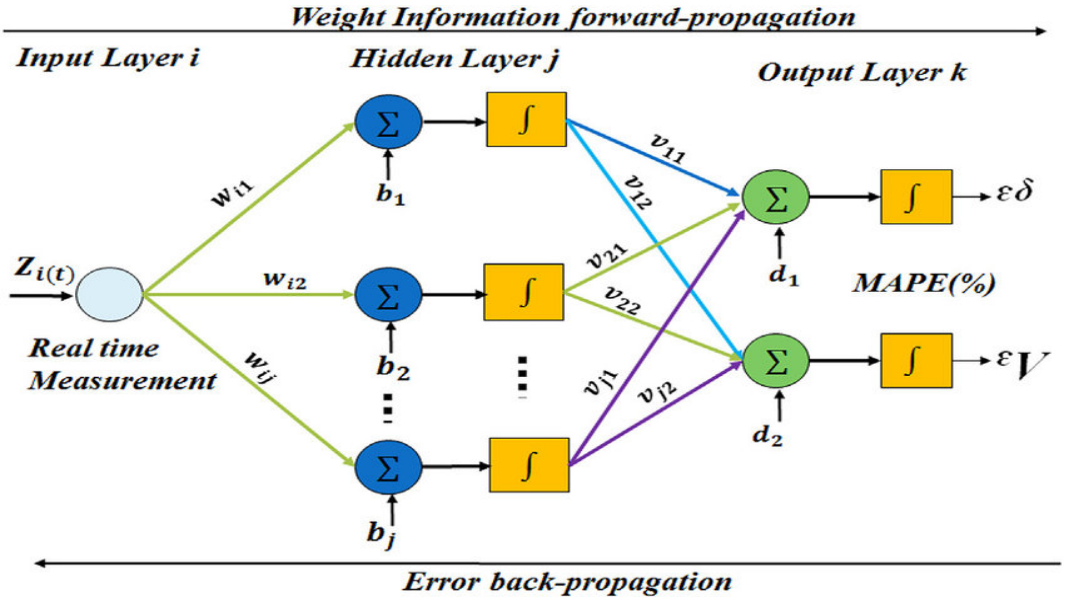

Figure 3: MLP Model

We have chosen MLP for the task of classification of feature vectors obtained. More details about how its usefulness and implementation have been provided in the later sections.

3 Outline of the Proposed Methodology

In this section we describe the pseudocode for the proposed methodology in a step by step easy to follow manner. Explanation and the details of the implementation are provided in the next section.

1. Consider one of the recordings (corresponding to a unique medical subject) from the Physionet dataset [3] (data description is provided in section 2.2).
   This recording consists of four abdominal signals, and one direct signal.

2. For one of the four abdominal signals, we generate the scalograms in the following manner.

   (a) Let us say that this abdominal signal has total of n data points i.e. it spans over a time interval of n milliseconds.

   (b) Taking the datapoints in a sliding window of fixed length (l milliseconds) and sliding stride (s milliseconds), obtain a scalogram corresponding to datapoints in each sliding window using continuous wavelet transform (CWT).

   (c) Depending on the value of stride s, we will have to consider only $n_{final} = int(\frac{n-1-l+1}{s})s$ datapoints and shall exclude the remaining few datapoints (i.e. last $n - n_{final}$ datapoints) from the further analysis.
   Note that this is an offline analysis, when implemented in real-time, there is no need to worry about loosing any datapoints as this point won't have any effect in real-time analysis.

   (d) Thus, for one of the abdominal signals of length n, $\frac{n_{final}}{s}$ number of scalograms would be generated.

3. Label each of the scalograms obtained in the previous step as Fetal or Non-Fetal. The labelling scheme is described in detail in the next section.

4. Using deep learning based VGG16 convolutional neural network architecture, extract (512 dimensional) feature vectors for each of the scalogram obtained in step 2.

   At this point we have obtained the feature vectors for each of the scalograms generated corresponding to sliding time windows of one of the abdominal signals. Since these feature vectors have been obtained for scalograms generated for the sliding windows of the abdominal signal in consideration, it could be said that, they essentially contain the time-frequency information of the abdominal signal's snippet in the sliding window.

5. Use a multilayer perceptron model (i.e. a neural network model) with these (512 dimensional) feature vectors as input and two output classes i.e. Fetal or Non-Fetal.

6. Perform all of the above steps for remaining three abdominal signals as well and obtain the output of the MLP classifier for each of the abdominal signals.

7. Take an ensemble (mode value) of the classification predictions obtained for all four abdominal signals.

8. Lastly, we look at the confusion matrix and accuracy parameters such as precision, sensitivity (recall), specificity for individual models on each of the four abdominal signals and the ensemble model.

   Based on the results obtained and proposed improvements, one could work towards increasing the accuracy parameters further and thus finally it would make sense to proceed towards obtaining the fetal heart rate using our model, which is indeed the main problem being investigated.

4 Numerical Experiments & Results Obtained

In this section we provide the detailed explanation concerning the implementation of proposed framework and summarize the results where we mainly focus on the accuracy parameters.

4.1 Detailed Explanation and Implementation of the Proposed Methodology

Here, we proceed on the lines of the outline mentioned in previous section, to provide the details of the proposed methodology, both from theoretical and implementation point of view.

4.1.1 Obtaining & Visualizing the Recording Signal Being Considered

We have considered recording 1 from the Physionet Abdominal and Direct Fetal ECG Databse [3]. As described in section 2.2, this dataset contains different recordings for different medical subjects. Within the `"r01.edf"` file of recording 1, there are four abdominal signals and one direct fetal ECG signal. We obtain these signals as `".csv"` files and plot them for basic time-domain visualization.

Figures 3, 4, 5, 6, and 7 show the time-domain visualization of a part (i.e. a time interval of length 2000 milliseconds or basically 2000 datapoints) of these five signals. Kindly refer appendix section to view these figures.

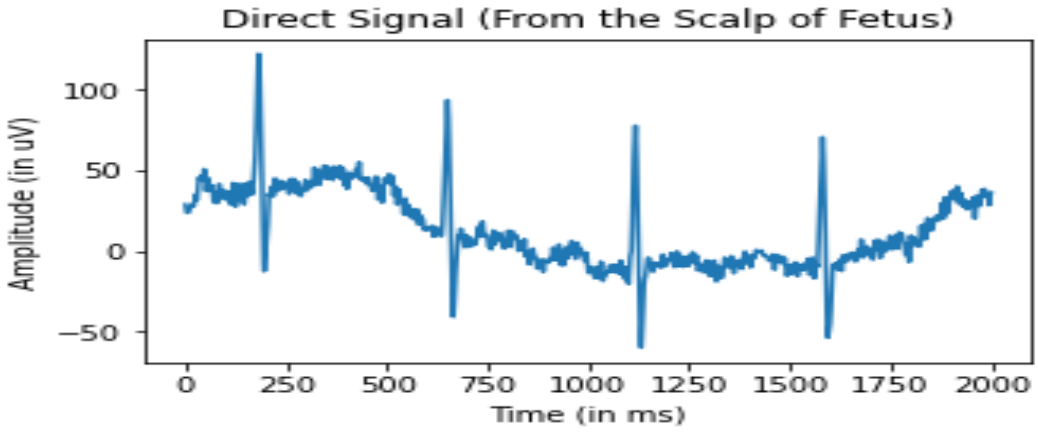

Figure 4: Direct Fetal Signal 1

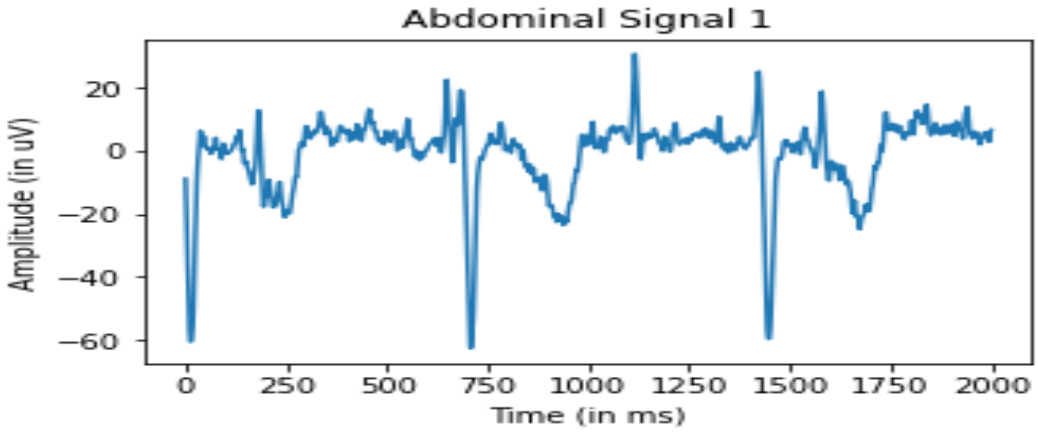

Figure 5: Abdominal Signal 1

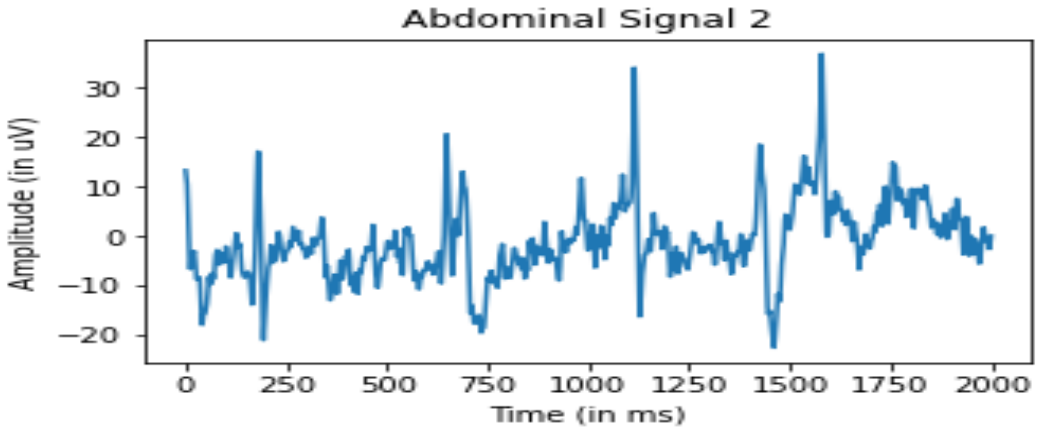

Figure 6: Abdominal Signal 2

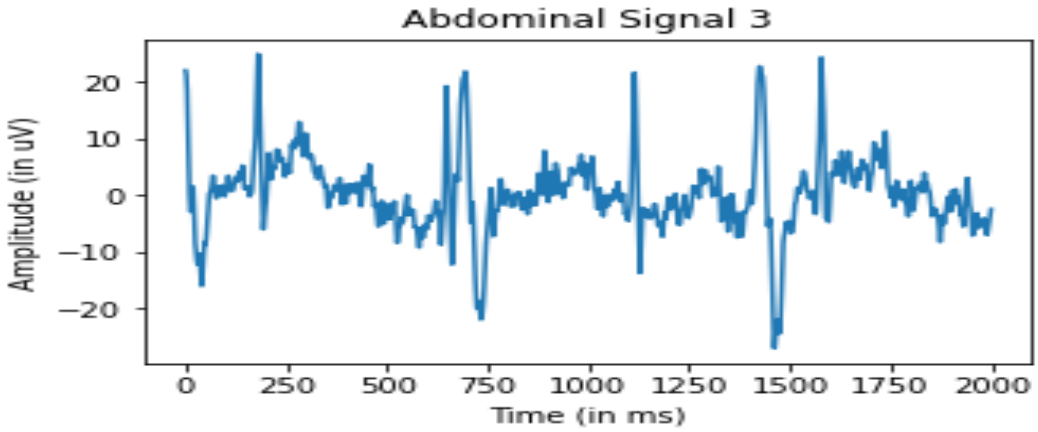

Figure 7: Abdominal Signal 3

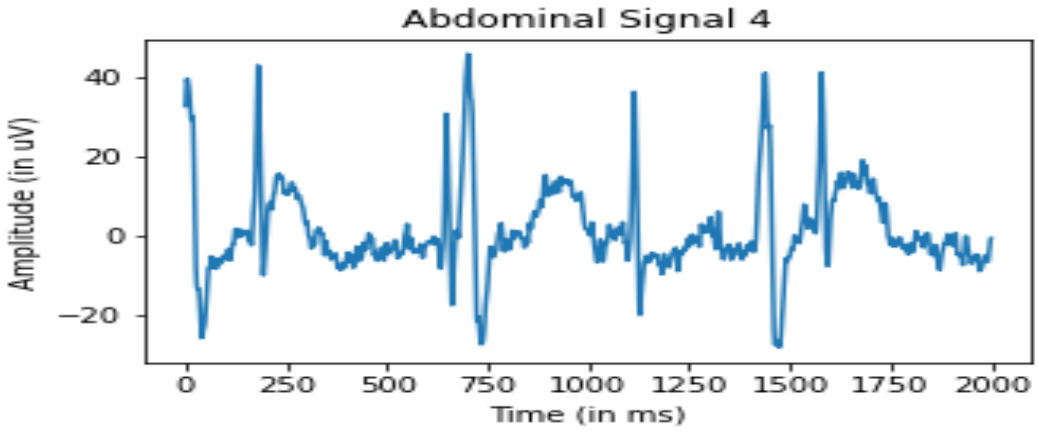

Figure 8: Abdominal Signal 4

4.1.2 Constructing the Scalograms

For each of the four abdominal signals, we construct the scalograms following the procedure mentioned in the previous section, i.e. for each of the abdominal signals, we take rolling windows of length $l = 512$ milliseconds and sliding stride of length $s = 50$ milliseconds, and then construct the scalograms for each of the rolling windows. We follow the same procedure of each of the abdominal signal. Gaussian wavelet was chosen for constructing the scalograms based on our mid-term analysis. The colour-map chosen to represent the absolute values of CWT coefficients using different colors is "`plt.cm.seismic`".

The figure below shows an illustrative scalogram for first 512 datapoints for each of the abdominal signal.

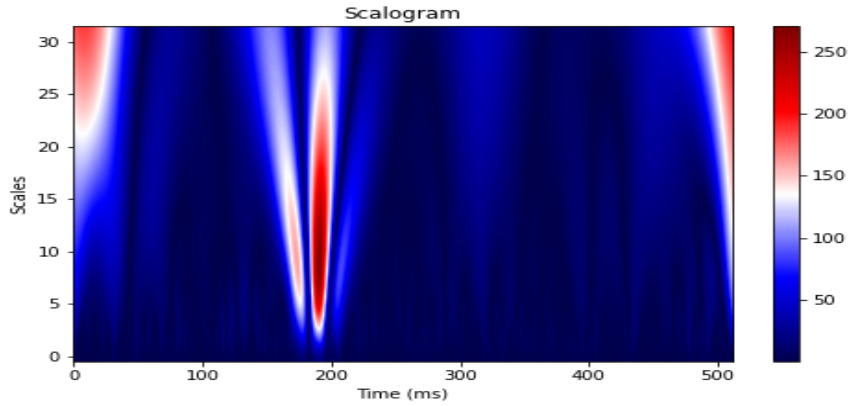

Figure 9: Scalogram for Direct Fetal Signal (with local vmin and vmax)

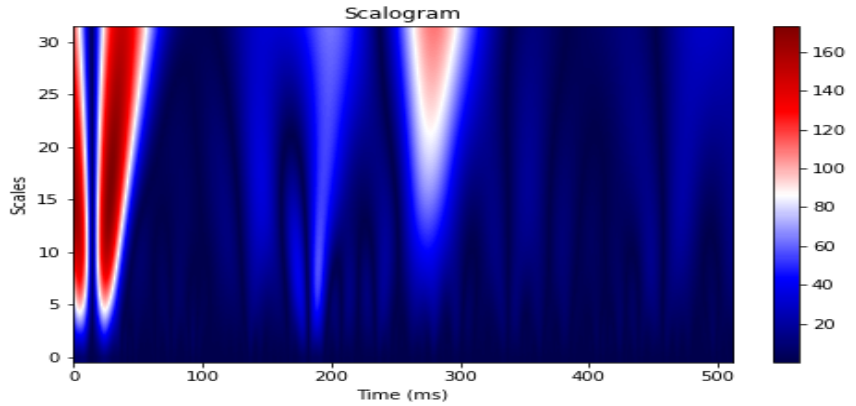

Figure 10: Scalogram for Abdominal Signal 1 (with local vmin and vmax)

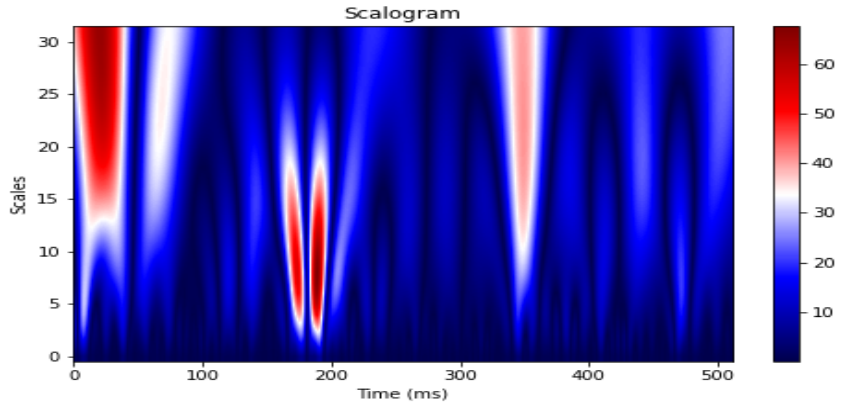

Figure 11: Scalogram for Abdominal Signal 2 (with local vmin and vmax)

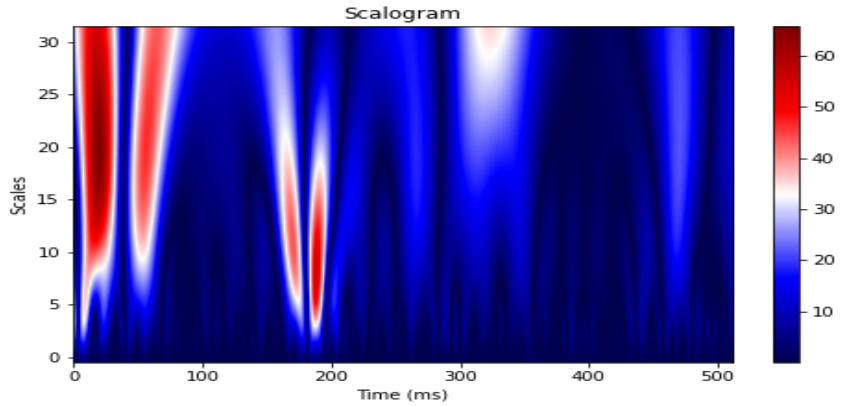

Figure 12: Scalogram for Abdominal Signal 3 (with local vmin and vmax)

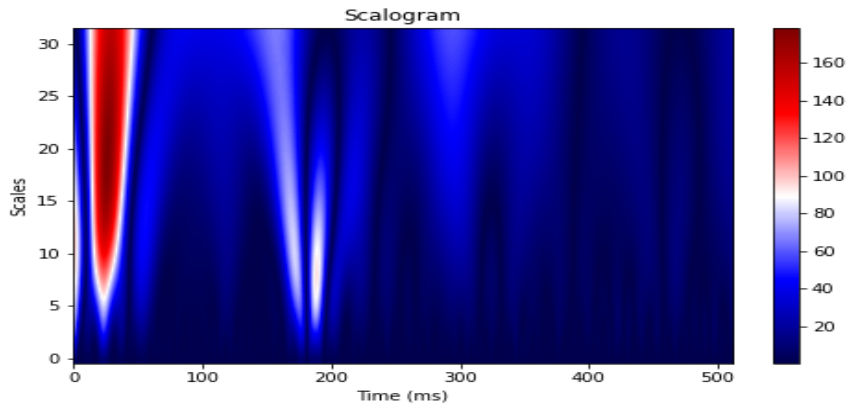

Figure 13: Scalogram for Abdominal Signal 4 (with local vmin and vmax)

It should be noted that, the choice of window length l and sliding stride s can be varied and is indeed meant to be decided by the user of this methodology. Also, the reason behind choosing the sliding stride as $s = 50$ milliseconds is, the fact that the duration of the fetal QRS complex is around 50 milliseconds for a normal fetus. Thus taking the sliding window of 50 milliseconds allows us to deal with the fragmentation of fetal QRS complex if taking place due to windowing operation. Furthermore, as one of the plausible improvement could be to consider the value of sliding window length l as 1024, which should result in better scalogram generation, reason being we are already plotting a discritized version of scalogram, so if we increase the number of points along the time axis, we would get more smoother scalograms. But in our study, considering the length of the abdominal signals we have chosen $l = 512$ and refrained from using $l = 1024$ because we need sufficient number of scalograms (thus in turn sufficient number of feature vectors) for the training of MLP model used in the final part of our framework.

Here, we will explain one of the most important points to keep in mind while constructing the scalograms from the point of view of implementation of our proposed framework. As mentioned before, for constructing the scalograms using the absolute values of CWT coefficients, we have used `"plt.cm.seismic"` colormap in Python. But in addition to this we have to give two arguments (i.e. vmin and vmax) while plotting the scalogram, which are essentially the maximum and minimum CWT coefficient value which shall correspond to the extreme colors in the colorbar used. Now, in the scalograms showed above, vmin and vmax values were set locally (i.e. vmin and vmax were calculated using only the CWT coefficients of the first rolling window of length 512 milliseconds). Whereas, if we consider the global vmin and vmax values (i.e. calculated the vmin and vmax values based on the CWT coefficients for all the rolling windows for the signal considered, then we get the scalograms as shown below. These scalograms are generated for exactly same signal snippets as that of the scalograms shown above.

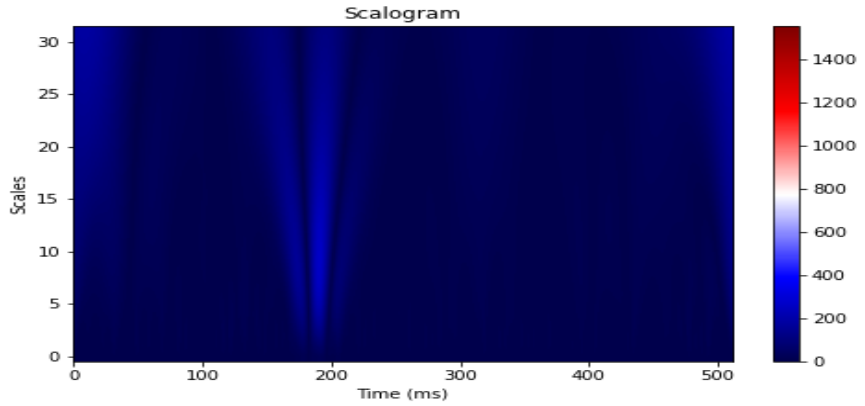

Figure 14: Scalogram for Direct Fetal Signal (with global vmin and vmax)

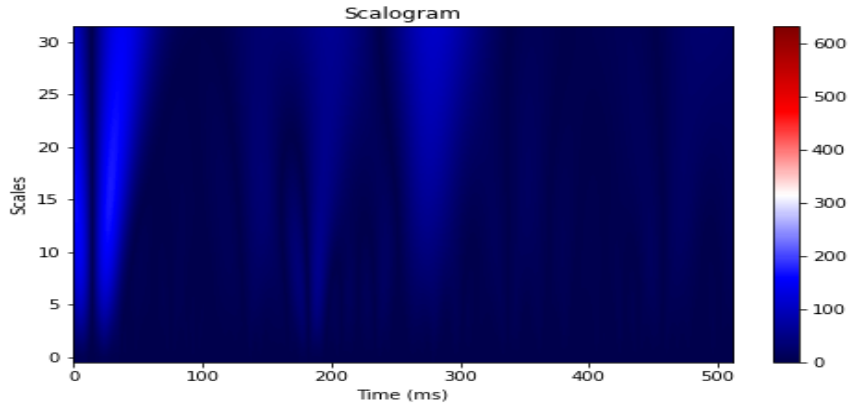

Figure 15: Scalogram for Abdominal Signal 1 (with global vmin and vmax)

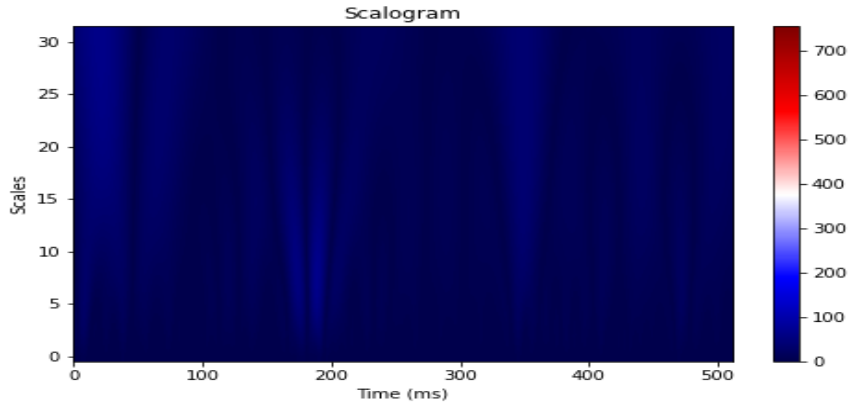

Figure 16: Scalogram for Abdominal Signal 2 (with global vmin and vmax)

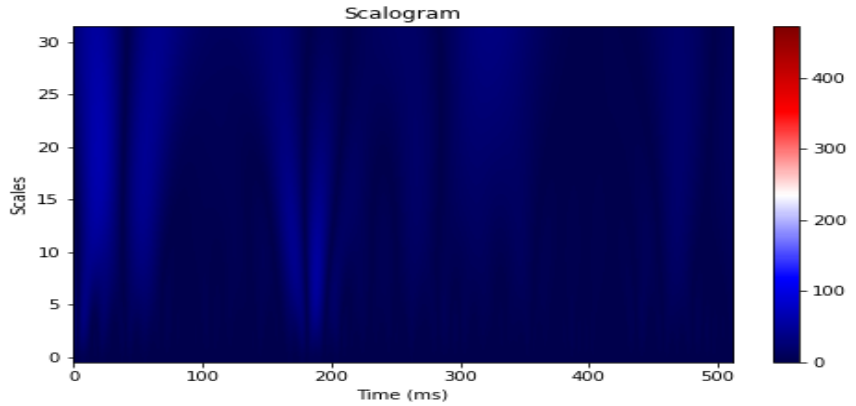

Figure 17: Scalogram for Abdominal Signal 3 (with global vmin and vmax)

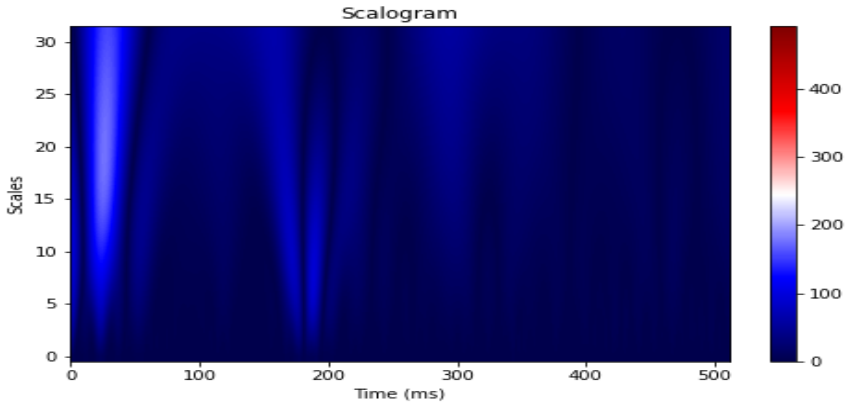

Figure 18: Scalogram for Abdominal Signal 4 (with global vmin and vmax)

The reason to explain this point is because, indeed, in our framework we need to use global vmin and vmax while constructing the scalograms. To understand the reason behind this requirement, reader should note that we are essentially extracting the feature vectors of the scalogram in the subsequent stages of our framework by using the deep learning methods CNN architectures. Thus if we use local vmin and vmax the deep learning architecture may extract incorrect features as it is taking the scalogram images as inputs.

4.1.3 Labelling the Scalograms Obtained

Here, we demonstrate the labelling scheme used in our proposed methodology. We label the scalograms generated for an abdominal signal into two classes i.e. Fetal (1) or Non-Fetal (0). Since the ultimate goal of this study is to compute the fetal heart rate we are interested in detecting the time instances of fetal QRS complexes or more precisely fetal R peaks. To aid in achieving this detection task, we label such scalograms as 1 which have the time instance of fetal QRS complex towards the end of their window. Whereas all other scalograms are labelled as 0. This labelling scheme is demonstrated below with the help of some illustrative scalograms.

Example 1 and 2 shown in the figures 19 and 20 respectively, shows 2 scalograms from fetal class. Reader may note that these scalograms are labelled as 'Fetal', because they have fetal QRS complex (or a part of it) towards the end of their time axis.

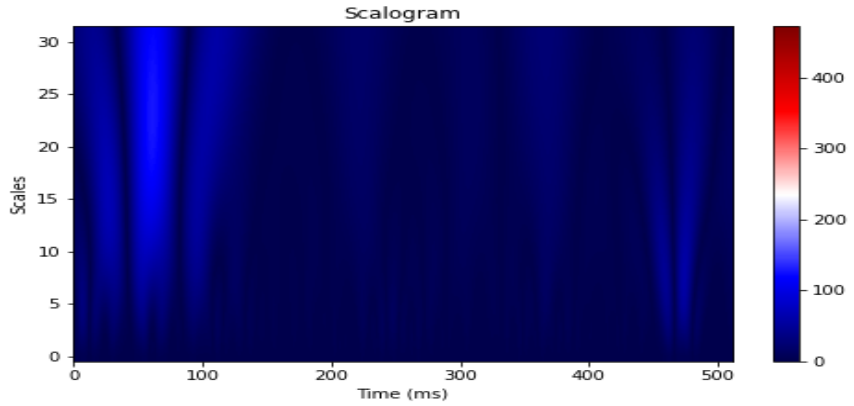

Figure 19: Example 1 of Scalogram labelled as Fetal Class (i.e. a scalogram with fetal QRS complex towards the end of scalogram window)

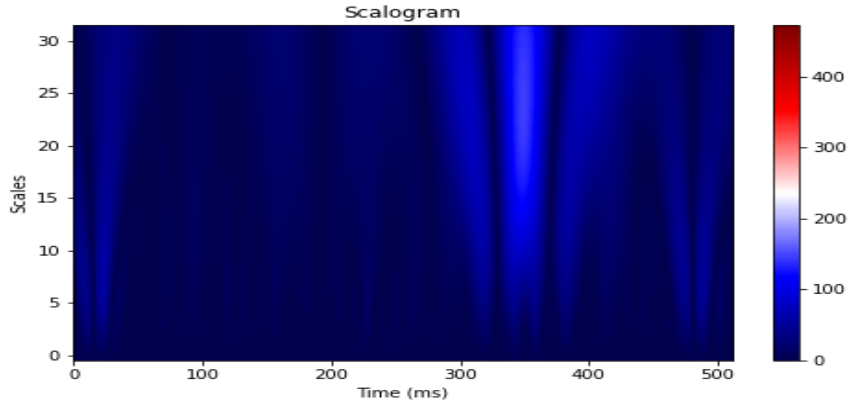

Figure 20: Example 2 of Scalogram labelled as Fetal Class (i.e. a scalogram with fetal QRS complex towards the end of scalogram window)

Example 1 from the non-fetal class shown in the figure 21 below has a fetal QRS complex somewhere in the middle of the scalogram window and noise everywhere else. Example 2 from the non-fetal class shown in the figure 22 below has a fetal QRS complex at the beginning of the scalogram window and also a maternal QRS complex somewhere in the right half (its exact location doesn't really matter for us while) of the scalogram window, of course there is noise everywhere else. Example 3 from the non-fetal class shown in the figure 23 below has a fetal QRS complex in the right half plane (but not towards the

end of scalogram window. One may imagine that the fetal QRS complex visible in this window would have been towards the end of scalogram in one of the previous scalograms) and maternal QRS complex in the middle of the scalogram window.

Thus reader may again note that these scalograms are labelled as 'Non-Fetal', because they don't have fetal QRS complex (or a part of it) towards the end of their time axis. Thus having the fetal QRS complex towards the end of scalogram window (along the time axis) is the only requirement for labelling a scalogram as "Fetal". This idea has been proposed because the ultimate goal of this study is not to classify the scalograms containg or not containing the fetal QRS peaks into different classes, but rather to detect the time instances of fetal QRS complexes with the aim of finally obtaining the fetal heart rate (which is indeed the bigger and main problem that our study is aimed at). With this remark, we end this subsection about labelling of scalograms.

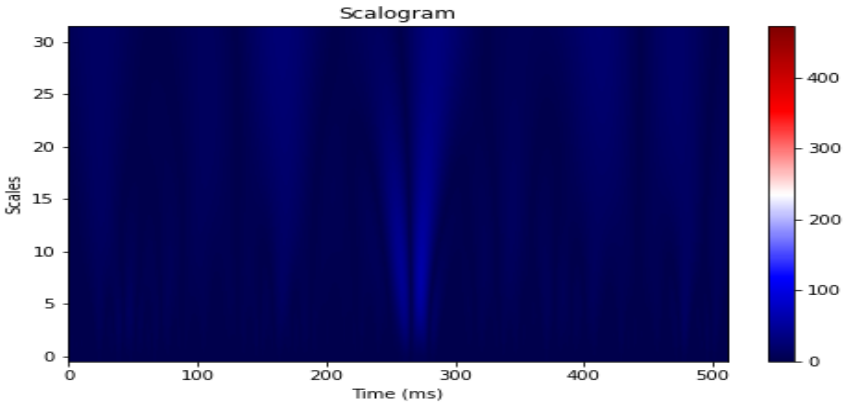

Figure 21: Example 1 of Scalogram labelled as Non-Fetal Class

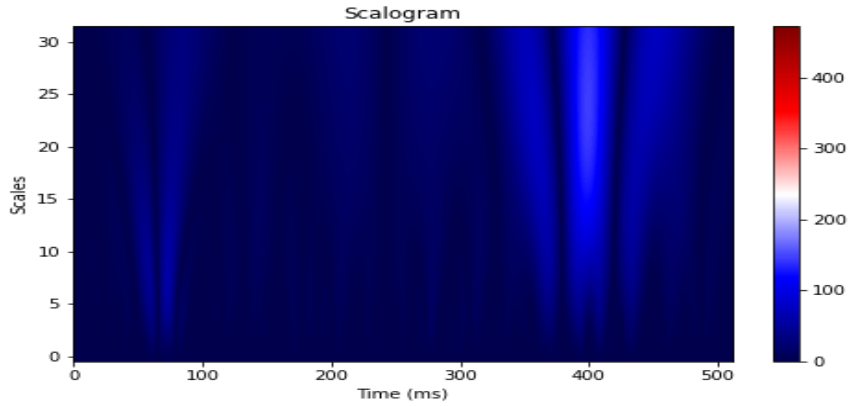

Figure 22: Example 2 of Scalogram labelled as Non-Fetal Class

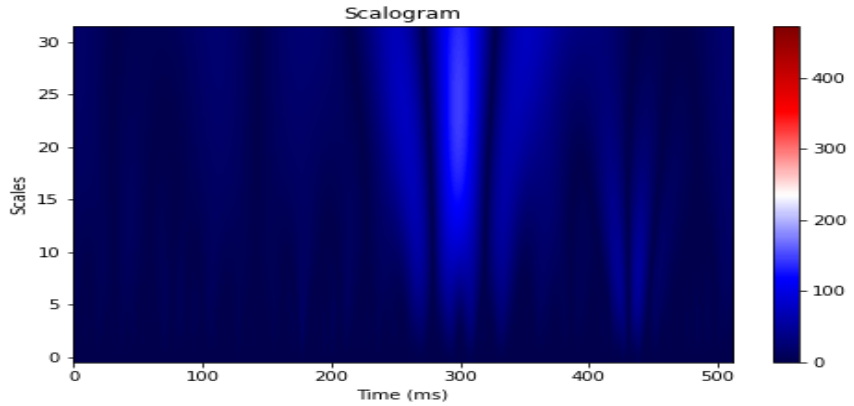

Figure 23: Example 3 of Scalogram labelled as Non-Fetal Class

4.1.4 Feature Extraction with Transfer Learning

Here, we state the process of extracting features (512 dimensional feature vectors) for the scalograms generated so far. We use VGG16 [10] deep convolutional neural network pre-trained on Imagenet dataset for the task of feature extraction. Thus, corresponding to each scalogram we have one feature vector (512 dimensional). Reader may refer to section 2.3 for further details on VGG16 architecture.

4.1.5 Classification using Multilayer Perceptron Model

After extracting the features as described above, we use a multilayer perceptron (MLP) which is a neural network with multiple layers. The inputs to the MLP are 512 dimensional feature vectors and the outputs are two classes for for feature vectors (and thus consequently for corresponding scalograms), which are either fetal or non-fetal. The structure of the MLP is self explanatory and can be seen in the model summary provided below. We have chosen to show the structure of MLP with the help of model summary as it is much more convenient and easier to understand in this way.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 1024)              525312
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_2 (Dense)              (None, 512)               524800
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 1026
_________________________________________________________________
activation (Activation)      (None, 2)                 0
=================================================================
Total params: 2,100,738
Trainable params: 2,100,738
Non-trainable params: 0
_________________________________________________________________
```

Figure 24: Model Specification for Considered MLP Model

4.1.6 Classification using Ensemble Model Employing All Four Abdominal Signals

Furthermore, we proposed to use an ensemble model which uses the classification prediction results for all of the abdominal signal using the steps discussed so far. Ensemble predictions are obtained by taking the mode of predictions of four different abdominal signals. This essentially means that if 3 out of 4 predictions (as we have obtained 4 predictions for four abdominal signals) are classifying the scalogram as class 1 then we take the ensemble prediction as class 1, as simple as that. We call this ensemble model as Ensemble Model 1.

Furthermore, we also tried relaxing the ensemble criteria, i.e. even if only 2 out of 4 predictions (as we have obtained 4 predictions for four abdominal signals) are classifying the scalogram as class 1 then we take the ensemble prediction as class 1.

So far, in this subsection we have described in detail the methodology and the tools used. Results for implementation of this methodology are provided in the next subsection.

4.2 Results Obtained

Here, we state the implementation results obtained for our proposed methodology. Note that the entire datset has been divided into train, validation, and test datasets in 50%, 20%, and 30% ratio respectively.

4.2.1 Results for Univariate Models

If we use our proposed methodology on one of the abdominal signals only, then indeed, the resulting model can be termed as a "Univariate Fetal QRS Detection Model", so without loss of gnenerality we stick to this name. It is worthwhile to recall that in this univariate model, we first generate the scalograms for the abdominal signal in consideration, followed by feature vector extraction using transfer learning with VGG16 architecture, and finally, MLP based classification of scalograms.

Here, we have shown the model (MLP) accuracy and model loss function graphs when we trained the MLP model on each of the four abdominal signals, in an attempt to justify the accuracy results which we have stated just below these plots. Accuracy measures have also been stated along with the confusion matrices. It should be noted that, since we have low positive class rate we stick to "F1 Score" as our accuracy measure amongst all available.

1. Model Accuracy and Loss Function Plots for Abdominal Signal 1

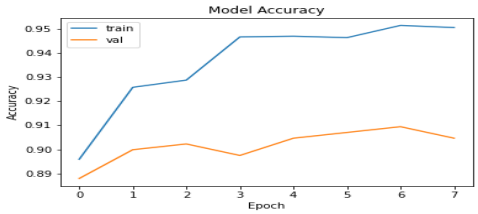

Figure 25: Model Accuracy Plot for Abdominal Signal 1

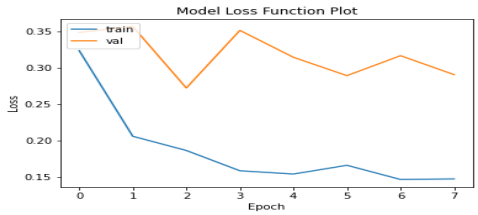

Figure 26: Model Loss Function Plot for Abdominal Signal 1

2. Confusion Matrix and Accuracy Parameters for Abdominal Signal 1

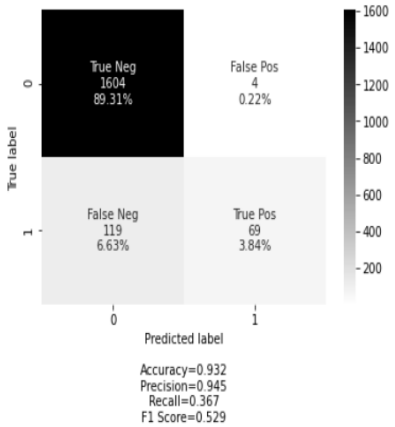

Figure 27: Confusion Matrix and Accuracy Parameters for Abdominal Signal 1

3. Model Accuracy and Loss Function Plots for Abdominal Signal 2

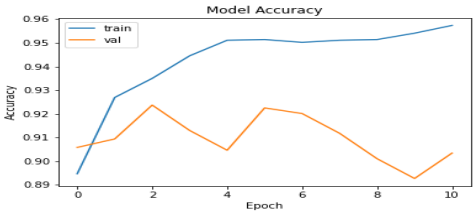

Figure 28: Model Accuracy Plot for Abdominal Signal 2

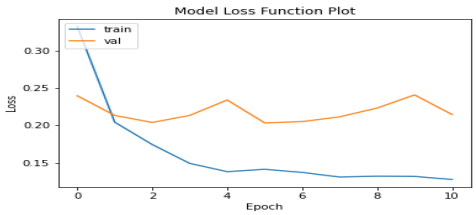

Figure 29: Model Loss Function Plot for Abdominal Signal 2

4. Confusion Matrix and Accuracy Parameters for Abdominal Signal 2

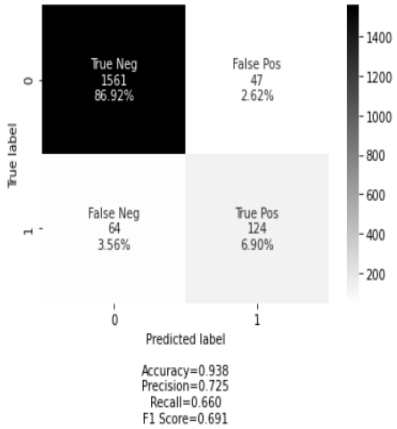

Figure 30: Confusion Matrix and Accuracy Parameters for Abdominal Signal 2

5. Model Accuracy and Loss Function Plots for Abdominal Signal 3

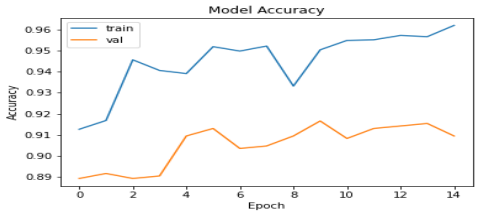

Figure 31: Model Accuracy Plot for Abdominal Signal 3

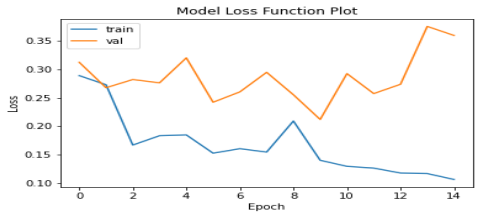

Figure 32: Model Loss Function Plot for Abdominal Signal 3

6. Confusion Matrix and Accuracy Parameters for Abdominal Signal 3

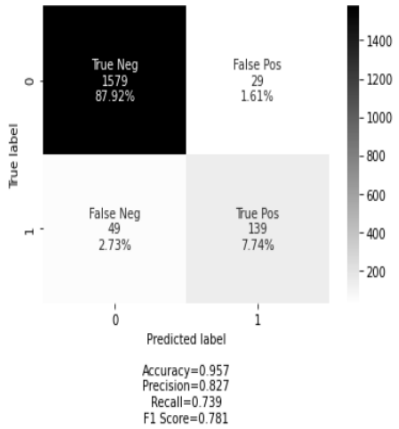

Figure 33: Confusion Matrix and Accuracy Parameters for Abdominal Signal 3

7. Model Accuracy and Loss Function Plots for Abdominal Signal 4

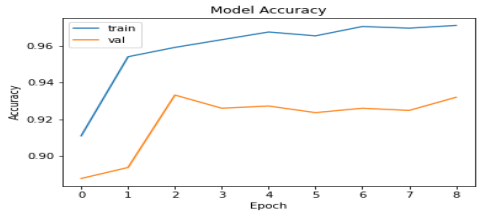

Figure 34: Model Accuracy Plot for Abdominal Signal 4

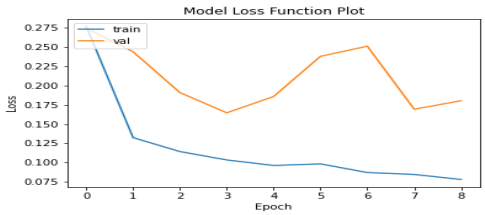

Figure 35: Model Loss Function Plot for Abdominal Signal 4

8. Confusion Matrix and Accuracy Parameters for Abdominal Signal 4

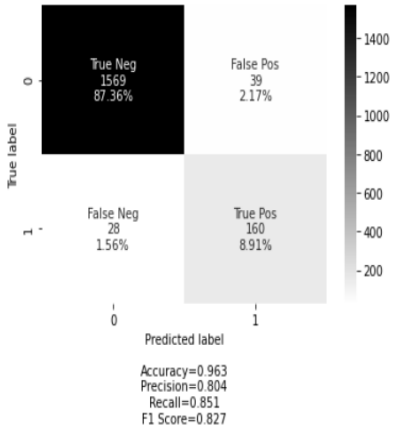

Figure 36: Confusion Matrix and Accuracy Parameters for Abdominal Signal 4

4.2.2 Results for (Multivariate) Ensemble Model

Here, we call our ensemble model as "Multivariate Fetal QRS Detection Model" on the similar lines. Since Our proposed ensemble model involves taking the ensemble of the predictions obtained for four different abdominal signals using the univariate model, we have only provided the confusion matrix and accuracy parameters for our ensemble model.

1. Confusion Matrix and Accuracy Parameters for Ensemble Model 1

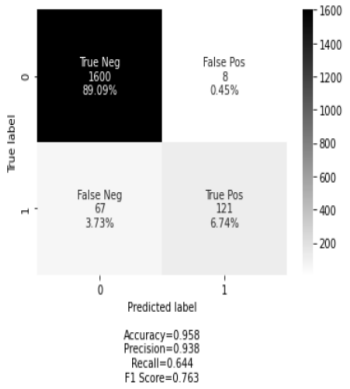

Figure 37: Confusion Matrix and Accuracy Parameters for Ensemble Model 1

2. Confusion Matrix and Accuracy Parameters for Ensemble Model 2

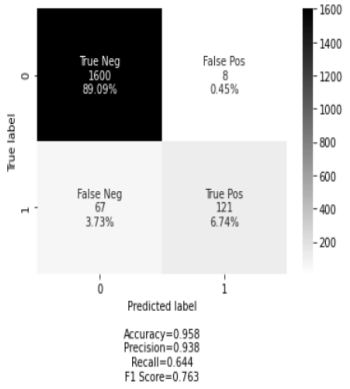

Figure 38: Confusion Matrix and Accuracy Parameters for Ensemble Model 1

3. Recall that the Ensemble Model 1 takes the mode of all the four abdominal signal predictions, whereas the Ensemble model 2 predicts a class label as '1' if at least 2 out of 4 abdominal signal predictions are '1'. To further explain this criterion for Ensemble Model 1 and 2 we have added a figure below which contains the first few rows of the dataframe containing predictions from all 4 univariate models and 2 ensemble models.

	Predictions for Abdominal Signal 1	Predictions for Abdominal Signal 2	Predictions for Abdominal Signal 3	Predictions for Abdominal Signal 4	Predictions for Ensemble with Criteria 1	Predictions for Ensemble with Criteria 2	True Labels
0	0	0	0	0	0	0	0.0
1	0	0	0	0	0	0	0.0
2	0	0	0	0	0	0	0.0
3	0	0	0	0	0	0	0.0
4	0	0	0	0	0	0	0.0
5	0	0	0	0	0	0	0.0
6	0	0	0	0	0	0	0.0
7	0	0	0	1	0	0	1.0
8	0	1	0	0	0	0	0.0
9	0	0	0	0	0	0	0.0
10	0	0	0	0	0	0	0.0
11	0	0	0	0	0	0	0.0
12	0	0	0	0	0	0	0.0
13	0	0	0	0	0	0	0.0
14	0	0	0	0	0	0	0.0
15	0	1	0	1	0	1	1.0
16	0	0	0	1	0	0	0.0
17	0	0	0	0	0	0	0.0
18	0	0	0	0	0	0	0.0
19	0	0	0	0	0	0	0.0

Figure 39: Dataframe for Predictions from 4 Univariate and 2 Ensemble Models

As we can see from all of the results above, the accuracy measure (i.e. Model F1 Score) of ensemble models (F1 Score = 76.3 % for both the ensemble models) is better than the univariate model trained and tested on abdominal signals 1, 2, and 3. Whereas, the F1 score for univariate model trained and tested on abdominal signal 4 is even better than the ensemble models (F1 Score = 82.7% for univariate model on abdominal signal 4). This could be explained by the fact that abdominal signal 4 has high SNR as compared to other three abdominal signals. In fact the SNR is high enough that the results are even better than the ensembel model 2. Although, note that this was not the case with abdominal signals for other medical subjects. Thus the only conclusion that we draw from this observation is that, if the signal under consideration has high SNR, our proposed framework would better.

5 Plausible Improvements & Extensions for the Proposed Framework

In this section, we propose a few extensions and improvements which could be incorporated in our proposed methodology. Note that, in all of the extensions, we should use exactly same framework till point 3 in section 3 (i.e. till the generation and then labelling of scalograms).

5.1 Improved Transfer Learning with CNN instead of MLP

In our study we have used pre-trained VGG16 architecture for feature extraction, followed by MLP for classification of feature vectors (and thus consequently for classification of scalograms). In this extension, we propose to use the same pre-trained VGG16 architecture for feature extraction, but instead of MLP classification model, we propose to employ a CNN (convolutional neural network) model. Reader may note that, since we propose to employ the CNN architecture for the task of classification, the details of the CNN architecture would have to be decided through the simulations. We had tried to implement this CNN based classification approach too during our project, but due to limited computational power and huge amount of data we could not take the task to completion.

5.2 Fine Tuning with Pre-Trained VGG16 (Without altering the weights of VGG16)

In this extension, we propose to use the pre-trained VGG16 architecture as a base model (by removing the top layer), followed by a head model which would be a CNN architecture (needs to be built with the help of simulations again). Note that, indeed there is a difference between the previous extension and this one. In the previous extension we were using the CNN architecture only as a classification model which takes the feature vectors extracted using VGG16 architecture as inputs. Whereas, in this case we would have to look at a complete model comprising of base model and the head model, and rather than training the complete model we just train the head model and do not alter the weights of the base model. The reason for this partial training could be explained with the figure below.

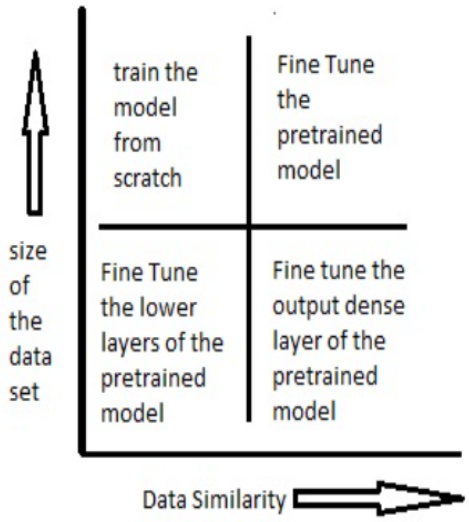

Figure 40: Tradeoff Between Data Similarity and Size of the Training Data

As we can see, this kind of framework is indeed suitable and beneficial when we have less training data. Although we would have relatively poorer model performance as the similarity between our training data (scalogram images) and the data on which VGG16 has been trained is less, whereas based on the figure provided above, the data similarity should be higher for achieving greater model performance.

5.3 Fine Tuning with Pre-Trained VGG16 (Alter the weights in all except a few initial layers of VGG16

In this extension, we essentially propose to train both the base model and the head model (this base and head model naming criteria is same as the one used in previous subsection). With one condition that the training for base model should be carried out on all except a few initial layers. The reason behind taking this approach is that the VGG16 architecture has been designed to extract some of the initial features in any given image by training it on a large number of images in Imagenet dataset. We don't want to distort these initial weights by retraining the base model for all the layers. Although retraining the base model for remaining layers (except a few initial layers) allows our complete model to learn the weights specific to the input images being considered for our task (i.e. scalograms).

It is worthwhile to note again that, this type of framework would be the best suited framework for our training data. This can be verified from the figure 40 provided in the previous subsection, where we can see that, indeed, this type of framework is suitable when we have less data similarity and at the same time, less training data at our disposal.

5.4 CNN Based Approach with Scalograms as Inputs

In this extension, we propose to train a CNN from scratch using the scalogram images as inputs. The structure of this CNN architecture would have to be decided with the help of simulations. The major challenge in this approach is that, one would need a huge amount of data (input scalograms) and huge computational power too for training such a CNN architecture from scratch (since our training dataset comprises of scalogram images). Again, this too is evident from the figure 40 provided.

It should be noted that, the main reason for providing these extensions without implementing them, is the limited computational power available to us at the moment. So even if we would have managed to use huge and perhaps multiple dataseets in our analysis, we don't have the computational power to run the simulations for such cases. We hope that these extensions could be tried upon with the help of higher computational power as a part of our mentor's (Mr. Nithin Lakshmisha) PhD thesis.

6 Concluding Remarks

In this article, we have proposed an approach for detecting the fetal R peaks in the maternal abdominal signal. This study is supposed to serve towards the problem of fetal heart rate detection which would be the final part once we successfully detect the fetal R peaks in the maternal abdominal signal. Our proposed methodology neither uses any noise removal techniques, nor any algorithms for removal of MECG components from the maternal abdominal signal. Our proposed methodology relies heavily on the use of CWT and the deep learning architectures for feature extraction. Concept of continuous wavelet transform has been used to visualize the time-frequency properties of the maternal ECG signal (and consequently its components), followed by pre-trained deep learning architectures for extracting features of these scalograms, and finally a neural network classifier has been used for classification of scalograms. The proposed methodology has been implemented in Python to run the simulations and obtain the results. Results obtained are not extraordinarily great but are considerably good which could be seen from the accuracy parameters obtained for the model. Furthermore, we believe that these accuracy parameters can be improved further, by making some changes in the hyperparameters of the model where it uses wavelet theory concepts. Also, we believe that the extensions provided from the employed deep learning architecture perspective would certainly help in improvising the accuracy parameters further.

We have tried to combine wavelet theory concepts with deep learning architectures to tackle the problem of fetal R peaks detection (and consequently fetal heart rate detection). Python implementations have been provided to facilitate a better understanding of the proposed framework. While tackling this problem statement, the innovative thinking was mainly used while employing the wavelet theory concepts and while combining the concepts from aforementioned two fields. We hope that this work could be continued further by our mentor Mr. Nithin Lakshmisha, and if given a chance we would like to contribute and extend our framework further.

References

[1] Andreotti et al. “An open-source framework for stress-testing non-invasive foetal ecg extraction algorithms”. In: *Physiological Measurement* 37.5 (2016), p. 627.

[2] Bhutta Z et al. “Stillbirths: What difference can we make and at what cost?” In: *The Lancet* 377 (2011), pp. 1523–1538.

[3] Goldberger et al. “Components of a new research resource for complex physiologic signals”. In: *PhysioBank, PhysioToolkit, and PhysioNet: Circulation [Online]* 101.23 (2000), e215–e220.

[4] Kanjilal et al. “Fetal ecg extraction from single-channel maternal ecg using singular value decomposition”. In: *IEEE Transactions on Biomedical Engineering* 44.1 (1997), pp. 51–59.

[5] Najafabadi et al. “Fetal heart rate monitoring based on independent component analysis”. In: *Computers in Biology and Medical* 36.3 (2006), pp. 241–252.

[6] Sameni R and Clifford G. “A review of fetal ecg signal processing issues and promising directions”. In: *The Open Pacing, Electrophysiology Therapy Journal* 3 (2010), pp. 4–20.

[7] Schreiber T Richter M and Kaplan D T. “Fetal ecg extraction with nonlinear state-space projections”. In: *IEEE Transactions on Biomedical Engineering* 45.1 (1998), pp. 133–137.

[8] Jutten C Sameni R and Shamsollahi M B. “Multichannel electrocardiogram decomposition using periodic component analysis”. In: *IEEE Transactions on Biomedical Engineering* 55.8 (2008), pp. 1935–1940.

[9] Ali Shoeb and G Clifford. “Lecture notes on "Chapter 16 - Wavelets; Multiscale Activity in Physiological Signals", Course: Biomedical Signal and Image Processing Spring 2005”. In: (2006).

[10] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *ICLR 2015 Conference* (2015), pp. 1409–1556.

Author Details

1. **Arbaz Shaikh** is currently pursuing Dual Degree in Electrical Engineering with Master’s degree in Communication and Signal Processing from IIT Bombay. For his Dual Degree Project he is working under the supervision of Prof. Vivek Borkar on the problem of non-stationary time-series analysis.

2. **Aman Butoliya** is currently pursuing Master’s degree in Communication and Signal Processing from IIT Bombay. For his master’s thesis, he is working under the supervision of Prof. Vikram Gadre on the problem of extraction and anomaly detection of fetal ECG signal.