# 3. PROBLEM MODELING

Speech recognition research in the recent decades has shown more exploration and applications built for many languages like English, French, Spanish, Chinese, Malay, etc. around the world and languages like Hindi, Urdu, Malayalam, Telugu, Tamil, etc. in India. The complexity and challenges keep on increasing with the high variability in the speech features of various languages, people, characteristics of channel considered, etc. [67]. The formative status observed in most south Indian languages' speech recognition models trigger to study and build an efficient acoustic speech recognition model for Tamil continuous speech. Some popular speech databases are TI digits, TIMIT, NTIMIT, RM1, RM2, ATIS0, ATIS2, SwitchBoard, ATC, CMU-Census and LDC. Linguistic Data Consortium for Indian Languages (LDC-IC) run by Central Institute of Indian Languages has a collection of speech corpus for 16 Indian languages including Tamil.

This chapter explains the architecture of proposed model to solve the phoneme recognition problem for Tamil continuous speech. It demonstrates the steps involved in building the speech corpus, the pre-processing phase including filtering and speech segmentation. The chapter also elucidates in detail about the proposed graph cut based segmentation algorithm which is used to segment the input continuous speech into phonetic segments.

## 3.1 OVERVIEW OF PROPOSED PHONEME RECOGNITION MODEL

The effort to meet the objective of the research is proceeded by formulating the phoneme pattern classification problem in Tamil continuous speech as a pattern recognition task. The proposed methodology shown in Fig. 3.1 is followed to build the acoustic model for phoneme classification in Tamil continuous speech. Various building blocks of the proposed model includes corpus development, filtering, feature extraction, segmentation of speech into phonemes, creation of phoneme datasets and building acoustic model.

The Tamil speech corpus has been developed by collecting speech utterances from various speakers of Tamil language. The collected speech utterances are then subjected to second order finite impulse response (FIR) filters that help to enhance the speech signal quality by removing the noise.

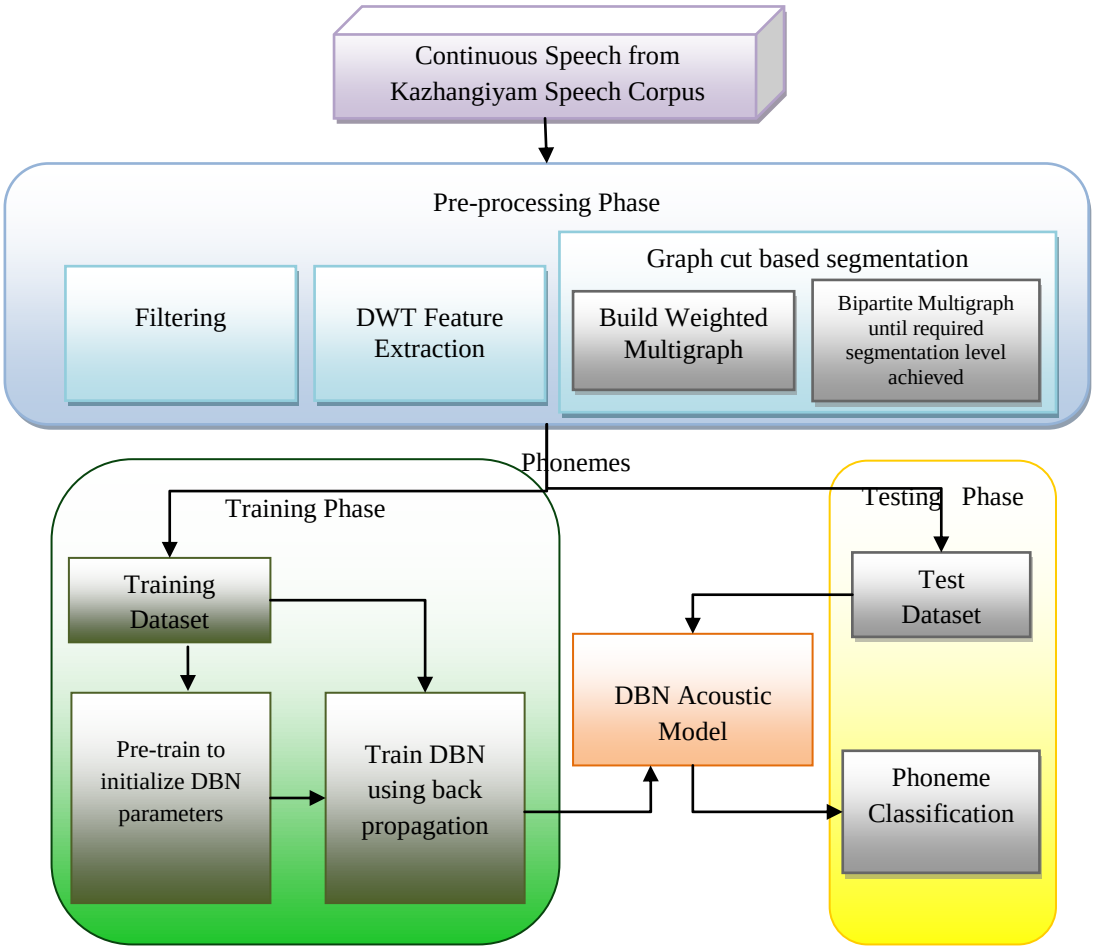



**Fig. 3.1 Architectural Diagram – Phoneme Recognition Model**

Discrete Wavelet Transform (DWT) feature extraction technique is applied to extract DWT features from the continuous speech. Then the phonetic segments are determined using the proposed graph cut based method, an unsupervised segmentation algorithm. The DWT features of the extracted phonemes are derived and the dataset is developed. The acoustic models are built by learning the dataset using DBN.

## 3.2 KAZHANGIYAM: THE SPEECH CORPUS

Speech databases play an important role in building a successful speech recognition application. Due to lack of benchmark speech corpus in Tamil, a speech corpus named 'Kazhangiyam' Tamil Corpus is created. Tamil speech is directly dependent on grapheme. It is a grapheme based language. Orthography of Tamil language comprises twelve vowels and eighteen consonants. The alphabet system of Tamil language is believed to derive from an ancient script, the Brahmi. Each sound in Tamil is represented by a syllable. A syllable is a combination of consonant and vowel, where the vowels are represented using secondary symbols

and conjoined with the consonants. Syllables of two hundred and sixteen different characters that are otherwise termed as graphemes are derived from vowels and consonants. In addition, a letter coined as aayutham, ஃ is also available in the Tamil language contributing to overall characters of Tamil alphabet set of size 247. Apart from the graphemes of Tamil language, few characters called Grantha letters or Agamas are used to represent sounds that penetrated to Tamil language from Sanskrit are also under use today. Earlier these letters were mainly used to write the rules and conducts of Hindu temple rites and rituals. Tamil speech corpus built here comprises of 39 phonetic sounds related to 33 graphemes and silence units. The graphemes along with their phonemic representation discovered from the corpus are listed in Table II.

**Table II Phonemes of Tamil Language**

| Tamil Grapheme | Phoneme Representation | | Tamil Grapheme | Phoneme Representation |
|---|---|---|---|---|
| அ | A | | ஞ் | nj/gh |
| ஆ | Aa | | ட் | t/d |
| இ | i | | ண் | Nn |
| ஈ | ii | | த் | th/dh |
| உ | u | | ந் | N |
| ஊ | uu | | ப் | p/b |
| எ | e | | ம் | M |
| ஏ | ee | | ய் | Ei |
| ஐ | ai | | ர் | R |
| ஒ | o | | ல் | L |
| ஓ | oo | | வ் | V |
| ஔ | au | | ழ் | Zh |
| க் | k/g | | ள் | L |
| ங் | ng | | ற் | Rr |
| ச் | ch/s | | ன் | n |
| ஸ் | s | | ஜ் | J |
| ஷ் | sh | | | Sil |

The speech is collected from thirty nine speakers whose mother tongue is Tamil. The speakers are scattered in the age group from 18 to 45. Table III shows the distribution of speakers over gender and age. Totally, forty five Tamil sentences that deliberately cover all the phonemes are taken into consideration. All the forty five sentences used to build the speech corpus are listed in Appendix I out of which few are given below.

- ஒருவரின் யூகம் எப்போதும் சரியாக இருக்காது.
- ரவி மிகவும் வீரமானவன்.
- முல்லைக்குத் தேர் ஈந்தவன் பாரி.
- ரேவதியின் வீட்டில் ரோஜா செடி உள்ளது.
- ரௌத்திரம் பழகு என்பது பாரதியின் கூற்று.
- வைகை நதி மதுரையில் பாய்கிறது.
- தேனீ நமக்குத் தேனை உணவாகத் தருகிறது.
- பொன்னூஞ்சல் மிகவும் விலை மதிப்பானது.
- நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும்.
- திணை என்னும் சொல்லுக்கு ஒழுக்கம் என்பது பொருள்.

**Table III Distribution of Speakers**

| Age/Gender | Male Speakers | Female Speakers |
|---|---|---|
| **18 – 25** | 13 | 15 |
| **26 – 40** | 4 | 7 |
| **Total** | 17 | 22 |

Five utterances of all the sentences are collected from all the 39 speakers. Each speaker contributed 225 sentences summing up to 8775 instances of wav files. The speech samples are collected in a controlled environment i.e. in closed room, using laptops and mobile phones. Audacity software is used while recording the speech using laptops. Audacity is a multi-tract audio editing and recording software available in open source for a Windows, Mac and other operating systems. The sampling frequency is set to 16000 Hz, the input channel is set to

'mono', and the Realtek High Definition microphone is used during the speech recording using Audacity. This corpus collectively comprises of 29 hours of speech, which is used entirely to build the datasets that are used in this research.

### 3.3 FEATURE EXTRACTION

In spite of various feature extraction methods like MFCC, LPC, FFT, STFT, etc., discrete wavelet transform is used in this research for speech feature extraction. DWT extracts both the time and frequency domain features as it uses the DWT basis function localized to both time and frequency and thus advantageous to other feature extraction methods which either produce time domain feature or frequency domain features. This is due to the adaptive window size in DWT rather than using fixed window size as in STFT, FFT, etc.

#### DWT Feature Extraction

Wavelet Transform is a feature extraction technique that transforms the sampled data into a new feature space that captures both the temporal and spatial information of the given data, thus allowing time-frequency analysis. The wavelet transforms are broadly classified as discrete wavelet transform and Continuous Wavelet Transform (CWT). The DWTs use orthogonal wavelet, whereas CWT used non-orthogonal wavelets. The DWTs have the property to return a data vector equal to the length of input data. It decomposes the input into set of wavelets that are orthogonal to its scaling and translation. Here, the input signal is decomposed into wavelet coefficient spectrum with data points equal to or less than those seen in input signal. Generally, the wavelet transform is represented as follows,

$$F(a,b) = \int_{-\infty}^{\infty} f(x)\psi^{*}_{(a,b)}(x)dx \tag{3.1}$$

where * represents the complex conjugate, ψ represents some arbitrary wavelet function. This research uses DWT for feature extraction from the input speech signal as it does not have any redundant information and is suitable for signal processing [68]. Various wavelet families like Haar, Daubechies, Symlets, Coiflets and Biorthogonal exist in the literature [69]. This research uses Daubechies 2 (db2) wavelet from Daubechies family which was formulated by Ingrid Daubechies in the year 1988 to extract the features of the given speech signal. Fig. 3.2 shows the scaling function, wavelet function of db2, the low pass and high pass filters used for signal decomposition and reconstruction. The features of the speech are extracted by cascading the DWT transforms for the required number of levels which is 6, in our case. In each level of transform, the input is given to a low pass filter, *g(n)* and a high pass filter, *h(n)* simultaneously.

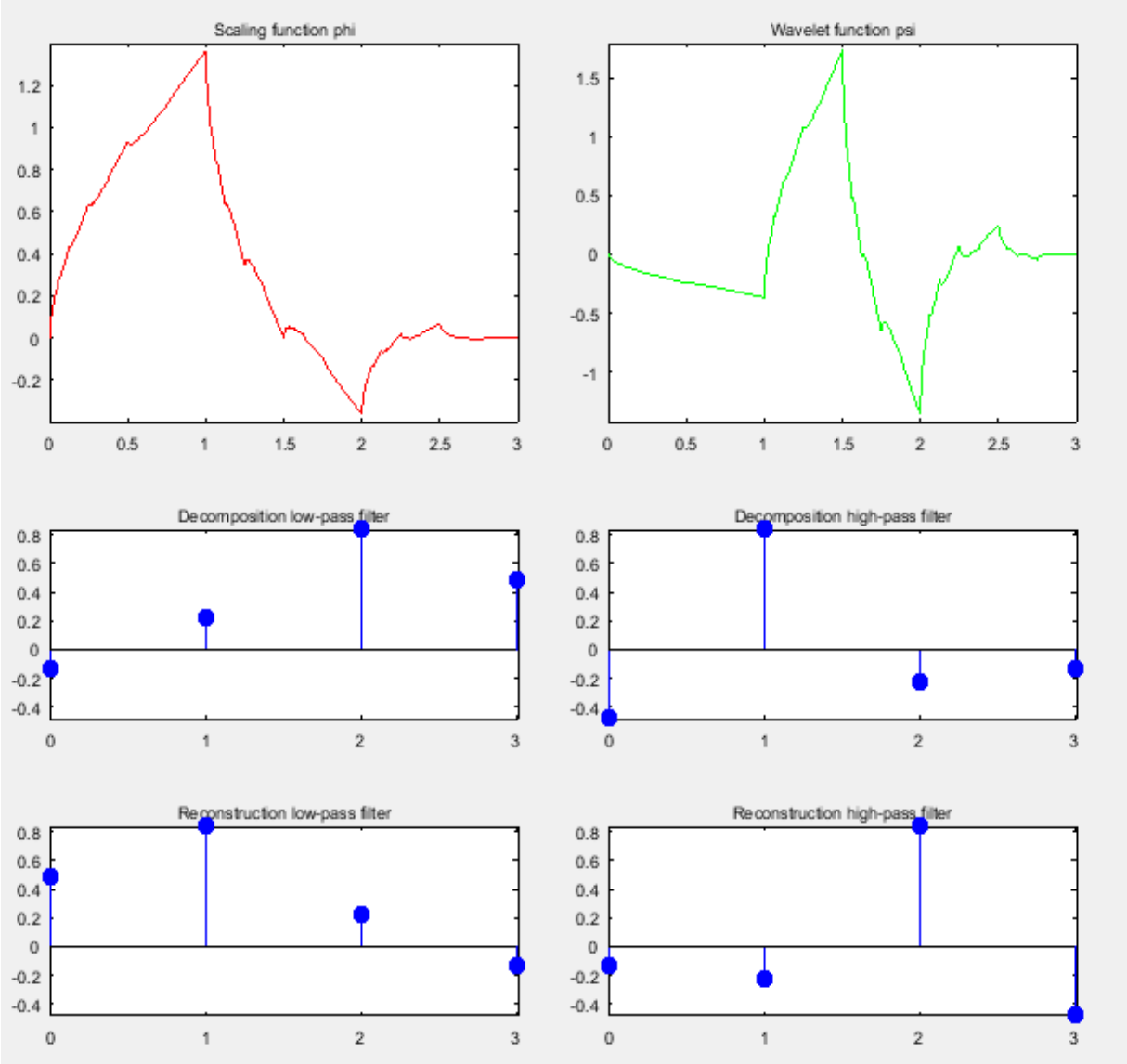



**Fig. 3.2 Scaling and Wavelet Function of db2, the Decomposition and Reconstruction Filters**

The outputs received from the low pass and high pass filters are approximation coefficients and detail coefficients respectively. For the given signal s(n), the coefficients of the approximation of the signal s(n) is given as series and is obtained using the following,

$$s_{m+1}(n) = \sum_i c_{m+1,i}\, \phi_{m+1,i}\,(n) \tag{3.2}$$

where $\phi_{m+1,i}(n)$ is the $i$-th wavelet function at the (m+1)-th resolution level. The wavelet function approximation is given by,

$$c_{m+1,i} = \sum_{n \in D_i} s(n)\phi_{m+1,i}\,(n) \tag{3.3}$$

where $D_i$ are supports of $\phi_{m+1,i}(n)$. The coefficients of the lower level are calculated by using,

$$c_{m,k} = \sum_i h_{i-2k} c_{m+1,i} \tag{3.4}$$

$$d_{m,k} = \sum_i g_{i-2k} c_{m+1,i} \tag{3.5}$$

Where $h$ and $g$ are the constant coefficients that depend on the scale function $\phi$ and wavelet $\psi$. The coefficients of m-th resolution level are computed by using the coefficients of $m+1$-th resolution level. The DWT of each level is given by

$$d_m = (d_{m,1}, d_{m,2}, \ldots.)^T \tag{3.6}$$

The multiresolution analysis is done for the given speech signal by

$$DWT(s) = \{d_M, d_{M-1,\ldots,} d_1 c_1\} \tag{3.7}$$

As the DWT has the property to produce the output data of same size as that of the input, the outputs of both the low pass filter and the high pass filter are downsampled by 2. At each level, the frequency resolution increases and it is cascaded to the next level by passing the approximate coefficients to the low pass and high pass filters, followed by downsampling process. Two important considerations for DWT are:

1. The opted low pass and high pass filters should be quadature mirror filters.
2. The length of input signal must be a multiple of $2^n$, where n is the number of levels to enable decomposition of signal.

Cascading of 6-level DWT filter bank is showcased in Fig. 3.3.

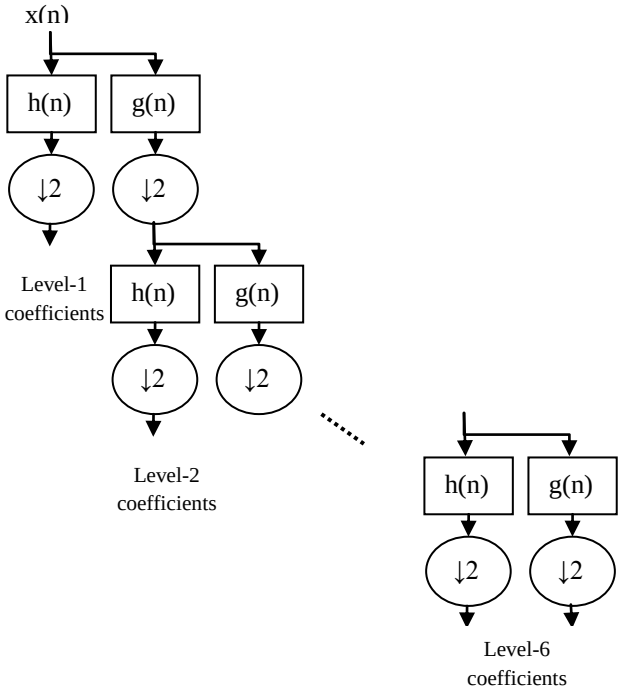



**Fig. 3.3 6-level Filter Bank used to Extract the DWT Features of the Speech Signal**

For an input speech signal recorded at 16000 Hz frequency, if the raw speech signal made up of 256 samples, the six levels of decomposition produces 7 output scales which is detailed in the following Table IV. The frequency decomposition obtained as a result of 6-level decomposition of the input speech signal with a frequency rate of 16000Hz is shown visually in the Fig. 3.4. Once six levels of decomposition of the input speech signal is done, the approximation coefficients from six levels are considered as the DWT features at a particular time frame of the speech signal, discarding detail coefficients as they contain least discriminative information for voiced signal [70].The DWT features extracted for the sample speech '**ரேவதியின் வீட்டில் ரோஜா செடி உள்ளது**' is visualized as DWT spectrum in the Fig. 3.5.

**Table IV The Frequency Bands and the Output Scales Produced in the DWT Spectra**

| Level | Frequencies | Samples |
|---|---|---|
| 6 | $0\ to\ \frac{f_n}{64}$ (0 Hz – 250 Hz) | 4 |
| | $\frac{f_n}{64}\ to\ \frac{f_n}{32}$ (250 Hz – 500 Hz) | 4 |
| 5 | $\frac{f_n}{32}\ to\ \frac{f_n}{16}$ (500Hz – 1000Hz) | 8 |
| 4 | $\frac{f_n}{16}\ to\ \frac{f_n}{8}$ (1000Hz – 2000Hz) | 16 |
| 3 | $\frac{f_n}{8}\ to\ \frac{f_n}{4}$ (2000Hz to 4000Hz) | 32 |
| 2 | $\frac{f_n}{4}\ to\ \frac{f_n}{2}$ (4000Hz to 8000Hz) | 64 |
| 1 | $\frac{f_n}{2}\ to\ f_n$ (8000Hz to 16000Hz) | 128 |

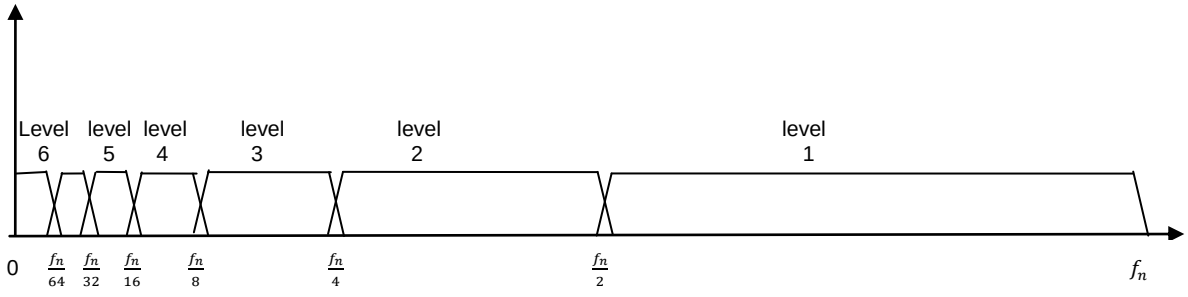



**Fig. 3.4 Frequency Bands in the Resulting DWT Spectrum of the Input with 16000Hz**

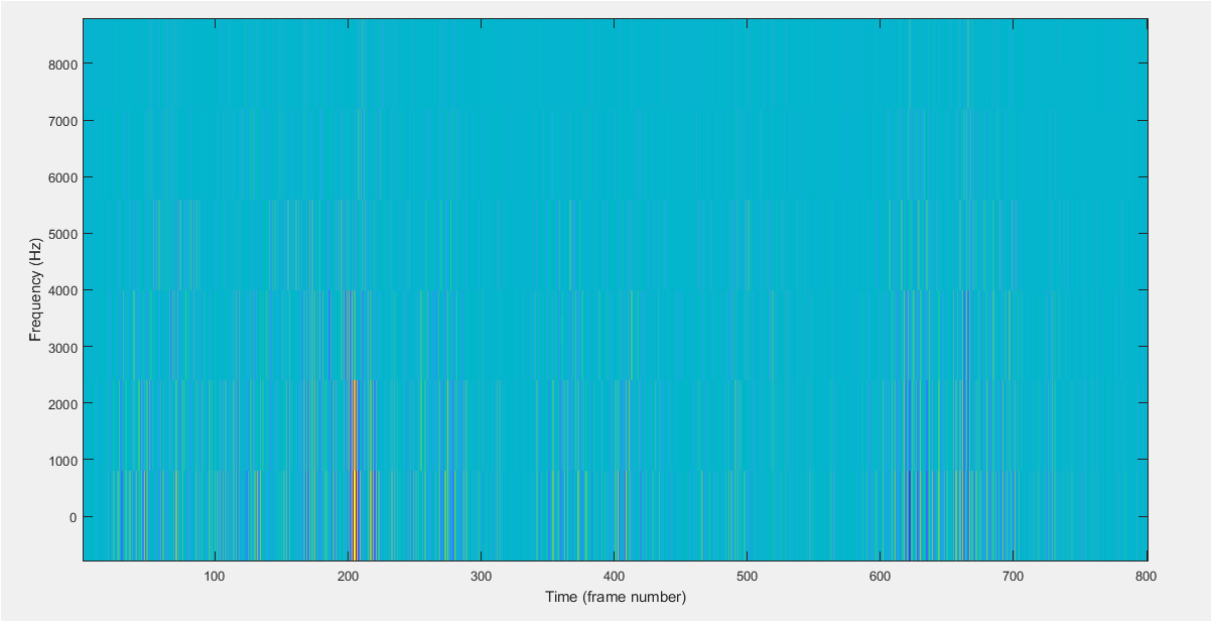



**Fig. 3.5 DWT Feature Spectrum of the Sample Speech 'ரேவதியின் வீட்டில் ரோஜா செடி உள்ளது'**

### 3.4 SPEECH SEGMENTATION

Segmentation is a crucial task in various speech processing applications. There are various methods to segment the speech data. Speech segmentation involves the process of identifying the boundaries of spoken speech units under consideration based on the requirement of the problem such as word-level segmentation, syllable level segmentation and phoneme-level segmentation. Extraction of words from the continuous speech is called as word-level segmentation. This segmentation uniquely identifies the meaningful words from the continuous speech. This level is a best choice for small size speech recognition systems where the dictionary is smaller, but not good for larger vocabulary systems. Extraction of syllables from the continuous speech signals is called as syllable-level segmentation. This segmentation identifies the valid syllables of the speech data based on the language used. Extraction of phonemes from the continuous speech signals is called as phoneme-level segmentation. This segmentation identifies the valid phonemes in the speech data of the language considered. Based on the knowledge that the segmentation algorithm has, it can be broadly classified as supervised and unsupervised segmentation. If the segmentation task is carried out using some prior knowledge then it is called as supervised segmentation. If the segmentation task is carried out without any prior knowledge of the data then it is called as unsupervised segmentation. Depending on the size of the dictionary for the given problem, the type of segmentation can be chosen. For problems with smaller dictionary, word-level segmentation will perform well. If the dictionary size is huge

then, syllable-level or phoneme level segmentations are more suitable. The type of segmentation also depends on the nature of the language, the problem is concerned with.

Although different speech segmentation approaches have been used so far for segmenting continuous speech in Tamil which includes group delay [71], zero crossing rate and absolute energy, and discrete wavelet transform. But in this research the spoken sentences collected using Audacity/mobile audio recorders are segmented into phonetic units using proposed graph cut based segmentation algorithm whose results have been validated against manual segmentations done using Praat tool by segmenting one sentence at a time into word level, syllable level and phonetic level segments.

### 3.4.1 Proposed Graph Cut based Segmentation

Graph theoretic approach is a type of methodology that uses graph structures to model the problem in hand and perform operations on graph to achieve the required solution. A plethora of graph based techniques are available in the literature for clustering analysis which includes removing inconsistent links from a graph or a spanning tree are constructed using proximity measures or likelihood information [72, 73], forming clusters with decision trees [74], etc. Graph cut is one such technique. Graph cut is an approach where the edges are removed from the weighted graph to form optimized clusters [75]. This approach produced good results in image segmentation problem. The pixels of the image forms the vertices of the graph with the neighborhood between them representing the edges of the graph. A normalized cut criterion is introduced to identify the minimal cut for better partition of the image [76].

These approaches are observed to produce meticulous outcomes on various problems where the problems are structured as a graph, with the data points forming the vertex set of the graph and the relationship between the vertices forming their edge set. In this research the speech segmentation problem is transformed to a clustering problem by converting the input speech signal into a graph with the frames of the speech forming the vertex set and relationship between the frames forming the edge set. This follows the fact that the frames of a phoneme are placed together and phonemes of the same class lying apart in the continuous speech are not connected through edges. Thus it is clear that the time frames in the speech of the speech segmentation problem can be considered as nodes and the likelihood of the frames as the edges of the graph.

***Graph Cut***

Let $\mathbb{G}= (\mathbb{V}, \mathbb{E})$ be a graph with $\mathbb{V}$ defining the vetex set of the graph $\mathbb{G}$ and $\mathbb{E}$ the edge set of $\mathbb{G}$. Graph cut is nothing but an operation performed on $\mathbb{G}$ that creates partitions like $\mathbb{G}_1$ and $\mathbb{G}_2$, so that the two subgraphs $\mathbb{G}_1= (\mathbb{V}_1, \mathbb{E}_1)$ and $\mathbb{G}_2= (\mathbb{V}_2, \mathbb{E}_2)$ conforming $\mathbb{V}_1 \cup \mathbb{V}_2=\mathbb{V}$ and $\mathbb{V}_1 \cap \mathbb{V}_2= \emptyset$, by simply discarding few edges connecting subgraphs $\mathbb{G}_1$ and $\mathbb{G}_2$ in $\mathbb{G}$. Such partitioning of graphs are evaluated based on the degree of dissimilarity in the subgraphs, the greater dissimilarity gives optimal partitioning. The optimal partition of the graph depends on the subset of edges that is selected for removal. The dissimilarity between two subgraphs is evaluated as follows,

$$cut(\mathbb{V}_1, \mathbb{V}_2) = \sum_{u \in \mathbb{V}_1, v \in \mathbb{V}_2} w(u, v) \tag{3.8}$$

where, w(u, v) gives the similarity of nodes u and v which can be referred in equation 3.12.

***Optimal cut***

A graph cut that yields the best partition of the graph is called the optimal cut. Thus, an optimal segmentation of speech can be achieved by performing graph cut that is optimal. Fig. 3.6 shows the importance of optimal cut over non-optimal one. A measure of disassociation is defined in [76], which is termed as normalized cut (Ncut). It is used to identify if the cut is optimal or not and is defined in as follows.

$$\text{Ncut}(\mathbb{V}_1, \mathbb{V}_2) = \frac{\text{cut}(\mathbb{V}_1, \mathbb{V}_2)}{\text{assoc}(\mathbb{V}_1, \mathbb{V})} + \frac{\text{cut}(\mathbb{V}_2, \mathbb{V}_1)}{\text{assoc}(\mathbb{V}_2, \mathbb{V})} \tag{3.9}$$

In the above equation $assoc(\mathbb{V}_1, \mathbb{V}) = \sum_{u \in \mathbb{V}_1, t \in \mathbb{V}} w(u, t)$ denotes the sum of all edge weights that connects the nodes of the graph $\mathbb{G}_1$ with all the nodes in the graph $\mathbb{G}$ and $assoc(\mathbb{V}_2, \mathbb{V}) = \sum_{v \in \mathbb{V}_2, t \in \mathbb{V}} w(v, t)$ defines the sum of all edge weights that connects the nodes of graph $\mathbb{G}_2$ with each node present in the graph $\mathbb{G}$.

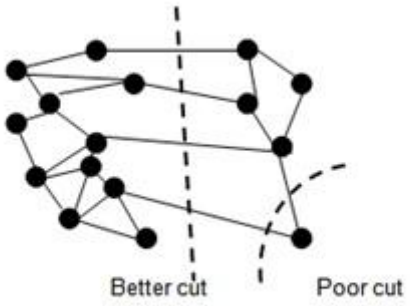



**Fig. 3.6 Case showing the importance of optimal cut for better partition**

Similarly, normalized association is a measure defined to analyze the strength of association of nodes within subgraphs of a graph. For the subgraphs $\mathbb{G}_1$ and $\mathbb{G}_2$ of graph $\mathbb{G}$, the normalized association $Nassoc$, can be calculated as:

$$\text{Nassoc}(\mathbb{V}_{1,}\mathbb{V}_2) = \frac{\text{assoc}(\mathbb{V}_1,\mathbb{V}_1)}{\text{assoc}(\mathbb{V}_1,\mathbb{V})} + \frac{\text{assoc}(\mathbb{V}_2,\mathbb{V}_2)}{\text{assoc}(\mathbb{V}_2,\mathbb{V})} \tag{3.10}$$

where, $assoc(\mathbb{V}_1, \mathbb{V}_1)$ and $assoc(\mathbb{V}_2, \mathbb{V}_2)$ defines the within-nodes association of $\mathbb{V}_1$ and $\mathbb{V}_2$ respectively. Obviously the measures normalized cut and normalized associations are inter related, which allows us to represent normalized cut in terms of normalized association as follows,

$$Ncut(\mathbb{V}_1, \mathbb{V}_2) = \frac{cut(\mathbb{V}_1, \mathbb{V}_2)}{assoc(\mathbb{V}_1, \mathbb{V})} + \frac{cut(\mathbb{V}_2, \mathbb{V}_1)}{assoc(\mathbb{V}_2, \mathbb{V})}$$

$$= \frac{assoc(\mathbb{V}_1, \mathbb{V}) - assoc(\mathbb{V}_1, \mathbb{V}_1)}{assoc(\mathbb{V}_1, \mathbb{V})} + \frac{assoc(\mathbb{V}_2, \mathbb{V}) - assoc(\mathbb{V}_2, \mathbb{V}_2)}{assoc(\mathbb{V}_2, \mathbb{V})}$$

$$= \frac{assoc(\mathbb{V}_1, \mathbb{V})}{assoc(\mathbb{V}_1, \mathbb{V})} - \frac{assoc(\mathbb{V}_1, \mathbb{V}_1)}{assoc(\mathbb{V}_1, \mathbb{V})} + \frac{assoc(\mathbb{V}_2, \mathbb{V})}{assoc(\mathbb{V}_2, \mathbb{V})} - \frac{assoc(\mathbb{V}_2, \mathbb{V}_2)}{assoc(\mathbb{V}_2, \mathbb{V})}$$

$$\therefore \text{Ncut}(\mathbb{V}_{1,}\mathbb{V}_2) = 2 - \text{Nassoc}(\mathbb{V}_{1,}\mathbb{V}_2) \tag{3.11}$$

Thus, selecting the best edge set for partition helps to perform a best partition of the graph. The implementation of graph cut based segmentation has already proved itself better in image segmentation problem. The normalized cut criterion has been applied to achieve promising solutions for the problems like static image segmentation and motion sequence segmentation where the image is represented as a graph with each pixel in the image represented as a node of the graph and the likelihood between the pixels depicting the edges of the graph [76]. A dense

stereo matching algorithm which takes advantage the disparity between the regions in the images for segmenting epipolar rectified images is proposed and developed [77].

***Phonetic Level Segmentation of Continuous Speech***

The process of extracting the phonetic segments for the given continuous speech by identifying the boundaries of the phonemes using graph cut based segmentation algorithm is portrayed in Fig. 3.7. Let S be the continuous speech wave. The input speech is subjected to a second order filter is initially applied the removal of undesired noise. Discrete wavelet transform, a speech feature extraction technique explained in the section 3.3 is applied on the speech S, to extract the DWT features of the speech. The DWT features extracted from speech are represented as a set of feature vectors, $F = \{F_1, F_2, \ldots . . F_n\}$ where $n$ represents the total number of frames in the given continuous speech. A graph is build by considering each feature vector as a node and similarity between the feature vectors as the edge connecting the nodes. The distance of the one feature vector from the other in the feature set $F$ is called the physical distance. The physical distance between the nodes is considered as a factor to restrict the number of edges in the graph. An edge between nodes is allowed if their physical distance is less than the distance factor $\zeta$. The graph is represented as a weight matrix $W$, where the weight of edge is the likelihood between the feature vectors representing the nodes connected by their respective edge.

The whole system that is represented as a weight matrix is now converted to an eigen value problem. The eigenvectors for the whole system is thus calculated and few eigenvectors are chosen to perform the graph cut. From the selected eigen vector an optimal cut is identified out of the proposed candidate cuts *0*, *mean($E_i$)* and *median($E_i$)*. Then the identified optimal cut is applied to segment the graph into two subgraphs. This process is iteratively applied to each subgraph obtained in the previous level to achieve the required segmentation of speech. The algorithm explaining the procedure to perform the phonetic level segmentation of continuous speech using the proposed graph cut based segmentation is given in Fig. 3.7 which takes speech signal as input and produces the phonetic segments as output.

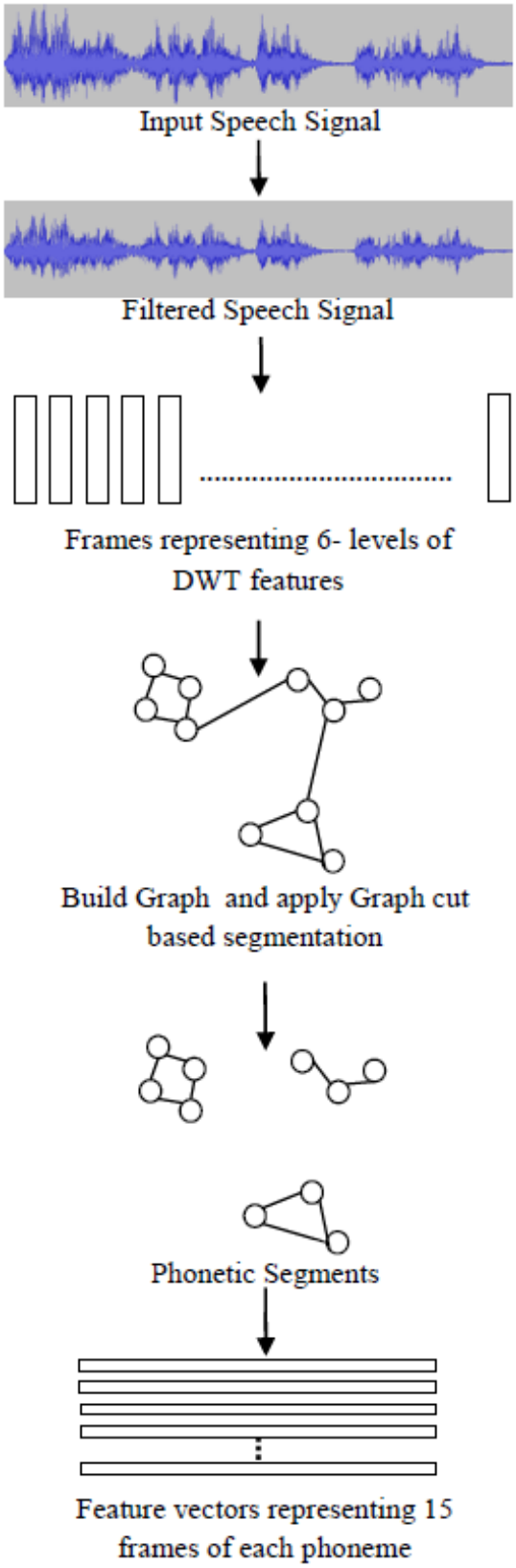



**Fig. 3.7 Framework Showing the Steps in Building the Phonetic Datasets**

**Algorithm 3.1** Graph cut based segmentation for continuous Tamil speech into phonetic units

Step 1: Extract the DWT features of the speech and represent it as, $F = \{F_1, F_2, \ldots\ldots F_n\}$, where $F_i$ is the $i^{th}$ feature vector of the given speech S.

Step 2: Build a multigraph $\mathbb{G}$= ($\mathbb{V}$, $\mathbb{E}$), consider each feature vector as a node and the relationship between the vectors as edge. Each edge is represented through a similarity measure between each node and graph is represented as a weight matrix W as follows,

$$\mathrm{W(i,\ j)} = \begin{cases} e^{\frac{-\left\|F_i - F_j\right\|_2^2}{\sigma_F^2}} & if\ distance(\mathrm{F}_i - \mathrm{F}_j) < \gamma \\ 0 & Otherwise\ . \end{cases} \tag{3.12}$$

Step 3: Find the eigenvectors with the median eigenvalues of the system by representing it as a standard eigenvalue problem,

$$(D - W)x = \lambda \mathrm{D}x \tag{3.13}$$

where D a diagonal matrix with elements $d_i = \sum_j W(i,j)$, the weight of $w_i$. Let $\varepsilon_1$, $\varepsilon_2$, .....$\varepsilon_n$ be the sorted list of eigenvalues and $E_1, E_2, \ldots\ldots E_n$ be their corresponding eigenvectors.

Step 4: Starting from the eigenvector of second median eigenvalues, an eigenvector is selected for each level of segmentation and then an optimal cut is identified for the graph considered.

a. If $E_i$ is the selected eigenvector, then *0*, *mean($E_i$)* and *median($E_i$)* are considered as candidate values to bipartite the graph using the selected eigenvector.

b. The Ncut values for the candidate options in step 4(a) are calculated using equation 3.9. The candidate having a greater Ncut value is then considered as an optimal candidate.

Step 5: The optimal candidate identified in step 4 is used to bipartite the graph. Then step 4 is repeated for each partition or subgraph of the graph until the required level of segmentation is achieved.

A spoken sentence in its raw form stored in .wav file with the sample speech "நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும்", shown in Fig. 3.8 is taken as input to the algorithm. The speech is subjected to the second order filter for noise removal, whose mathematical representation is given by,

$$Y(i) = b_0 S(i) + b_1 S(i-1) + b_2 S(i-2) \tag{3.14}$$

where S represents the input speech and Y, the filtered speech. The output of the sample speech in Fig. 3.8 subjected to the second order filter is shown in Fig. 3.9.

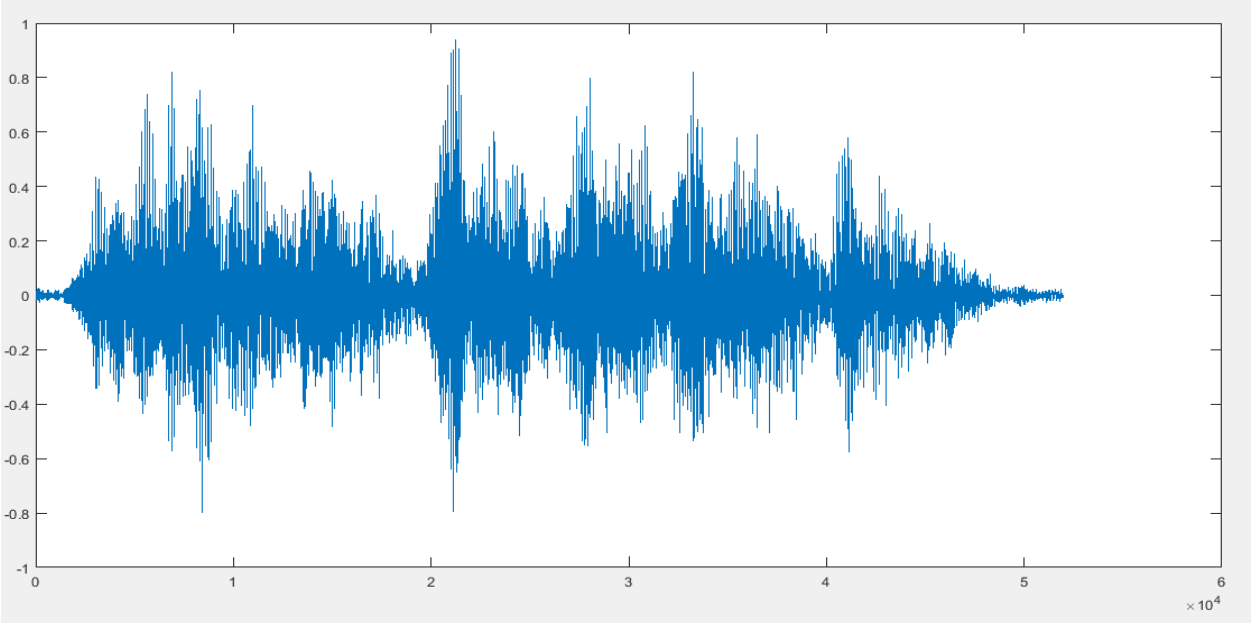



**Fig. 3.8 Sample Input Speech Signal**

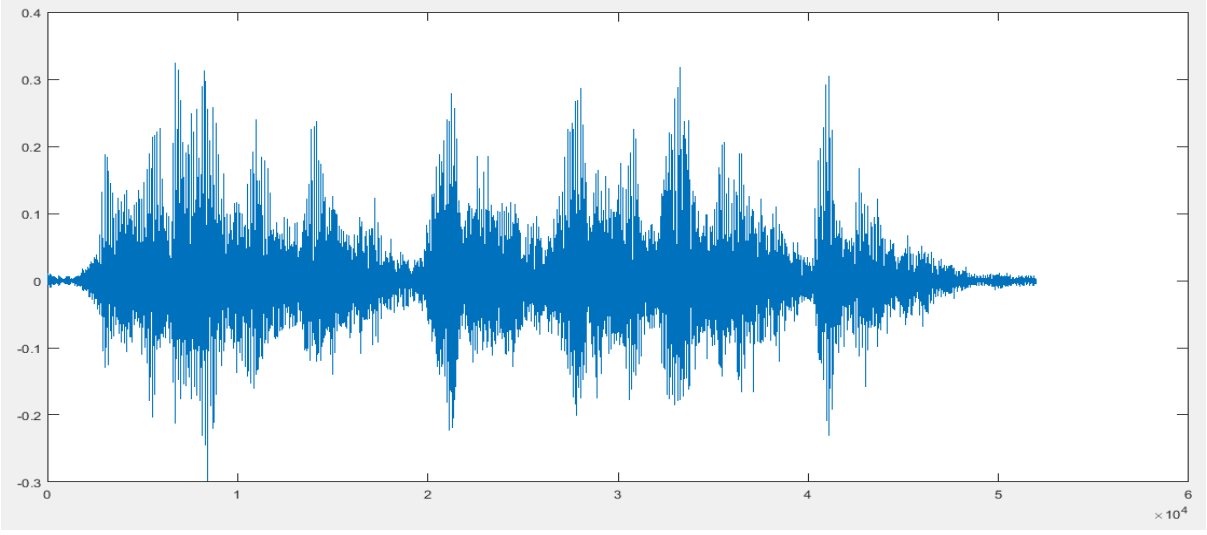



**Fig. 3.9 Filtered Sample Speech**

The filtered speech is then passed as the input to the graph cut based segmentation algorithm, where it starts with the process of DWT feature extraction. Here, Daubechies wavelet (db2) is applied on the speech to extract the features at different frequency bands as described in section 3.3. The speech is decomposed to six levels shown in Fig. 3.10 and 6 low frequency band features, also called the approximate features, are considered for each time frame of the speech. The approximate co-efficient values of the DWT spectrum of the sample speech is portrayed in Fig. 3.11 depicting the first 10 such frames.

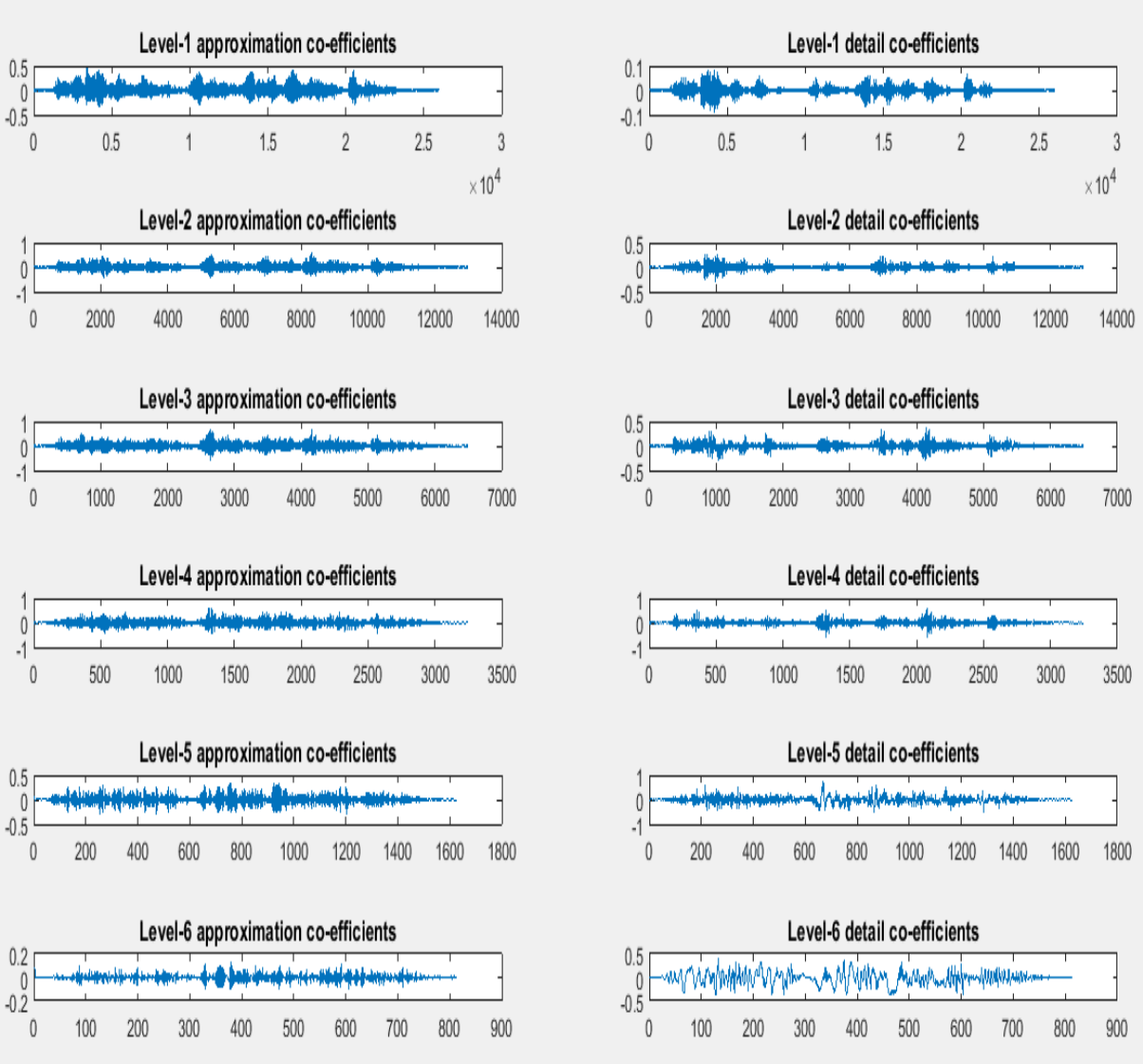



**Fig. 3.10 DWT Approximate and Detail Co-efficients of the Sample Speech**

6x801 double

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | -0.0088 | -6.9519e-04 | -0.0012 | 0.0012 | 0.0290 | -0.0344 | 0.0214 | 0.0091 | 0.0280 | -5.3140e-04 |
| 2 | -0.0211 | 0.0083 | -0.0079 | 0.0132 | 0.0089 | -0.0291 | 0.0092 | 0.0173 | 0.0383 | -0.0043 |
| 3 | -0.0308 | 0.0108 | -0.0053 | -0.0015 | 0.0049 | -0.0146 | 6.5741e-04 | 0.0070 | 0.0020 | -0.0023 |
| 4 | -0.0480 | 0.0076 | -0.0036 | 0.0089 | -0.0050 | -0.0034 | 0.0173 | -0.0083 | -0.0278 | 0.0062 |
| 5 | -0.0601 | 0.0040 | -0.0026 | -0.0011 | -0.0012 | 0.0102 | -0.0112 | 0.0010 | 0.0072 | -0.0022 |
| 6 | -0.0439 | -0.0406 | 0.0045 | -0.0017 | 0.0014 | -0.0030 | 0.0045 | -0.0025 | -0.0011 | 4.8938e-04 |

**Fig. 3.11 First Ten Frames Representing the Sample Speech**

The frames extracted so, are then considered as nodes of the graph and the likelihood of the frames represents the edge weights of the graph. The physical location of the frames in the speech influences the existence of edges between nodes in the graph. The node distance ζ decides the existence of an edge in the graph. In this experiment, the distance factor ζ is set to 50 with the knowledge gained by observing the number of frames covering a phonetic unit for manually segmented phonemes of continuous speech. The range of frames per phoneme generally ranges from 10 to 30 frames for representing phonemes from shorter to longer duration i.e from nasals, stops to long vowels. Partial elements of weight matrix for the graph built for the sample 'நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும்' is shown in Fig. 3.12. The complete weight matrix of the sample is presented in Fig. 3.13. The weight matrix shows the existence of edges in the diagonal part of the matrix which is due to the restriction of the node distance on the existence of edges in the graph through the distance factor ζ.

801x801 double

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0.9126 | 0.9243 | 0.9062 | 0.9034 | 0.9151 | 0.8963 | 0.9094 | 0.8887 | 0.9164 |
| 2 | 0.9126 | 1 | 0.9751 | 0.9842 | 0.9738 | 0.9568 | 0.9730 | 0.9825 | 0.9581 | 0.9811 |
| 3 | 0.9243 | 0.9751 | 1 | 0.9939 | 0.9879 | 0.9829 | 0.9876 | 0.9910 | 0.9659 | 0.9988 |
| 4 | 0.9062 | 0.9842 | 0.9939 | 1 | 0.9905 | 0.9680 | 0.9941 | 0.9958 | 0.9747 | 0.9970 |
| 5 | 0.9034 | 0.9738 | 0.9879 | 0.9905 | 1 | 0.9459 | 0.9937 | 0.9953 | 0.9865 | 0.9887 |
| 6 | 0.9151 | 0.9568 | 0.9829 | 0.9680 | 0.9459 | 1 | 0.9480 | 0.9580 | 0.9172 | 0.9800 |
| 7 | 0.8963 | 0.9730 | 0.9876 | 0.9941 | 0.9937 | 0.9480 | 1 | 0.9897 | 0.9700 | 0.9918 |
| 8 | 0.9094 | 0.9825 | 0.9910 | 0.9958 | 0.9953 | 0.9580 | 0.9897 | 1 | 0.9886 | 0.9919 |
| 9 | 0.8887 | 0.9581 | 0.9659 | 0.9747 | 0.9865 | 0.9172 | 0.9700 | 0.9886 | 1 | 0.9646 |
| 10 | 0.9164 | 0.9811 | 0.9988 | 0.9970 | 0.9887 | 0.9800 | 0.9918 | 0.9919 | 0.9646 | 1 |

**Fig. 3.12 Sample Weight Matrix**

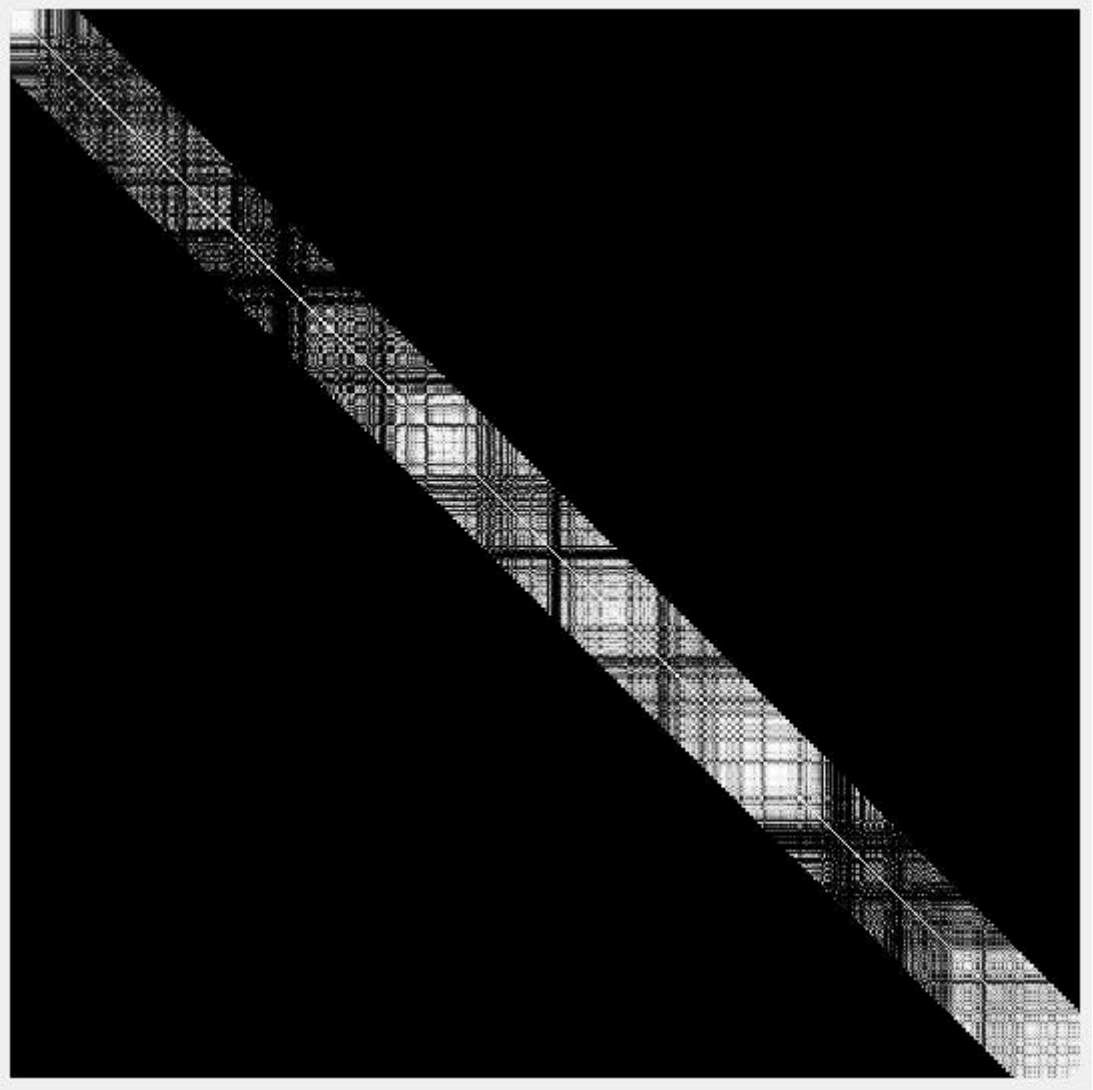


**Fig. 3.13 Weight matrix of the Graph G Constructed from the Frames Representing the Sample Speech**

The set of vertices in the graph built for a sample speech is visualized as a scatter plot in Fig. 3.14. At this point, the problem of segmentation is transformed to a standard eigenvalue problem as pointed in step 3 of the segmentation algorithm, through which the eigenvalues and their corresponding eigenvectors are evaluated. Totally 801 eigen vectors with their corresponding eigen values are generated for the sample being portrayed. Fig. 3.15 shows the sorted eigenvalues for the corresponding eigenvectors generated. From those eigenvectors, appropriate eigenvectors are selected to undergo the graph cut operation on the graph representing the speech. The selection of eigenvectors and identifying the optimal cut plays a crucial role in the segmentation accuracy. The selected eigenvector with median eigenvalues for the sample speech is shown in Fig. 3.16.

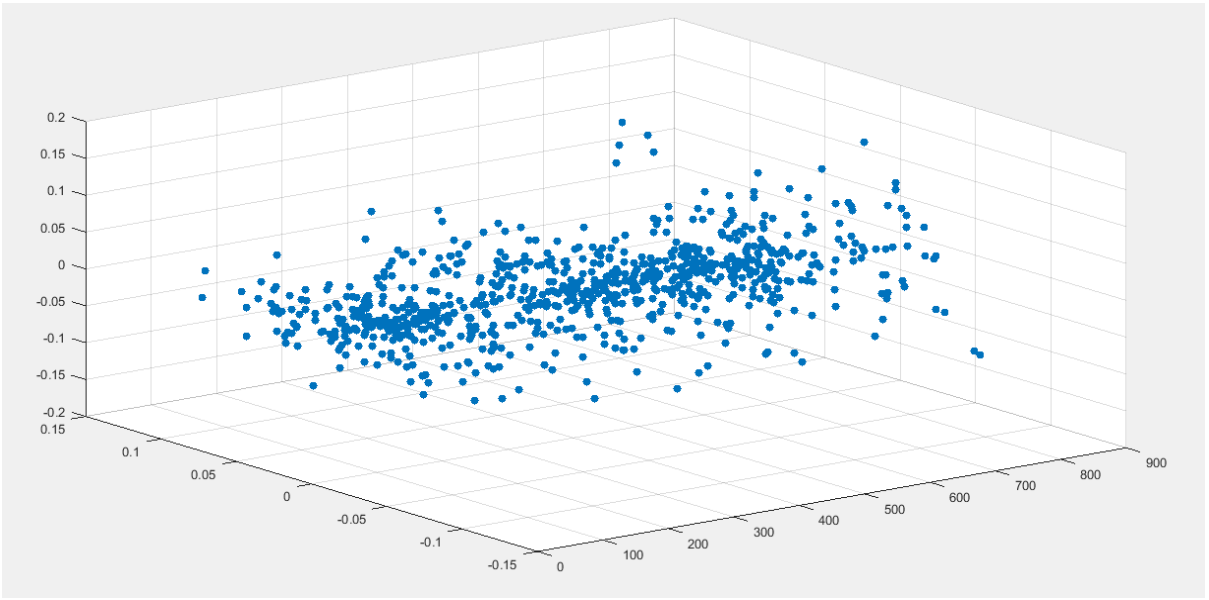



**Fig. 3.14 Vertex Set of Graph for a Sample Speech**

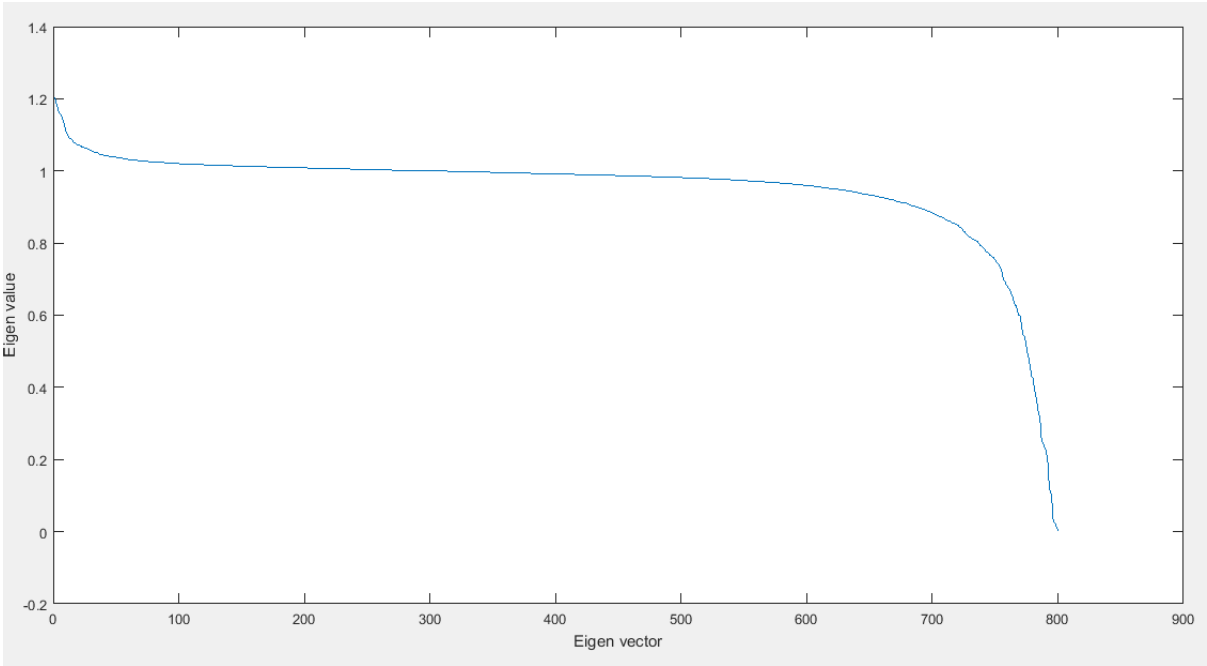
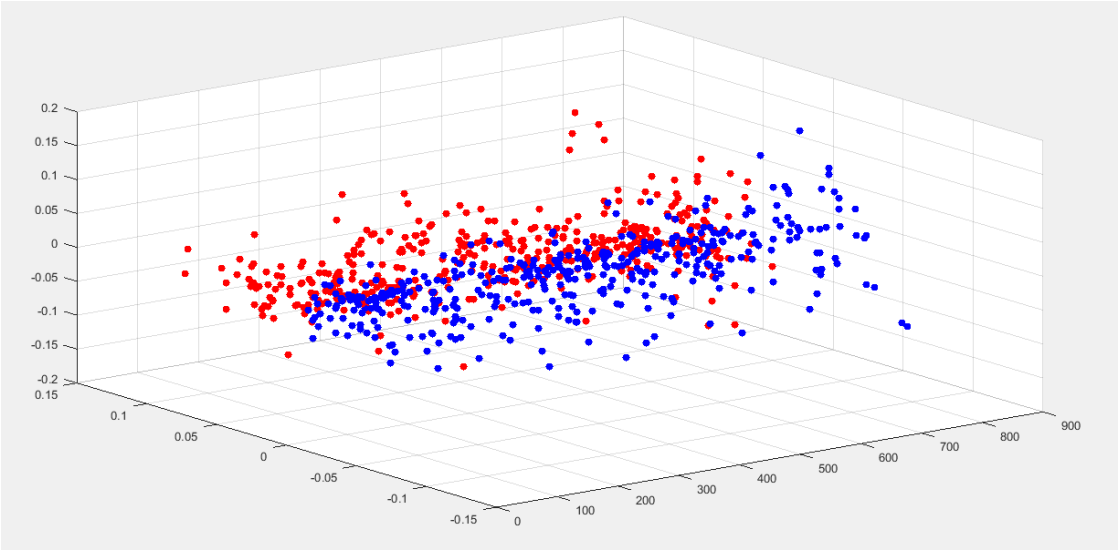



**Fig. 3.15 Sorted Eigenvalues for Sample Speech ‘நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும்’ (male)**

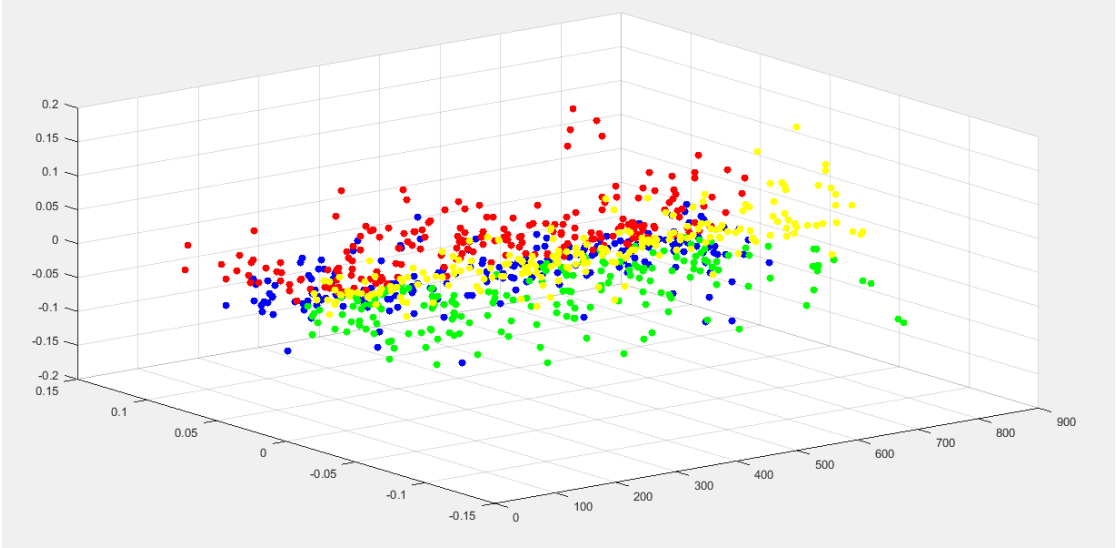
| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| -0.00783 | 0.00309 | -0.00470 | -0.01238 | 0.01566 | -0.00613 | -0.00396 | -0.00175 | -0.00896 | 0.01477 | -0.01638 | -0.01800 |
| 0.02372 | 0.07222 | -0.05787 | -0.03392 | 0.01948 | -0.02061 | -0.01668 | 0.01324 | 0.00658 | -0.00130 | 0.00493 | -0.00454 |
| 0.05016 | 0.02942 | 0.00698 | 0.02376 | -0.01081 | 0.00287 | -0.01184 | 0.02180 | -0.06269 | -0.00984 | 0.01645 | 0.01479 |
| -0.02973 | 0.01543 | 0.00059 | -0.05465 | 0.03113 | 0.00588 | -0.01558 | 0.00898 | 0.00355 | 0.02434 | -0.00031 | -0.01095 |
| 0.00906 | -0.00158 | 0.00522 | -0.00573 | -0.00555 | 0.01459 | -0.00633 | -0.00822 | 0.00746 | -0.00106 | 0.00993 | -0.00971 |
| -0.02926 | 0.03232 | 0.02453 | -0.05941 | 0.01242 | 0.01080 | -0.01087 | 0.00943 | -0.00104 | 0.01216 | -0.00017 | -0.01835 |
| -0.00539 | 0.01164 | -0.00783 | -0.00486 | 0.01691 | 0.00143 | -0.01984 | -0.00022 | -0.01202 | 0.00711 | 0.01027 | -0.00310 |
| -0.00531 | -0.00984 | 0.00946 | -0.03171 | -0.01351 | 0.05364 | 0.00450 | -0.04322 | 0.04937 | 0.01932 | -0.03975 | -0.00678 |
| -0.00799 | -0.00462 | 0.02201 | -0.00695 | -0.00219 | 0.00101 | 0.01733 | 0.01344 | -0.03666 | -0.02318 | 0.00666 | 0.03632 |
| -0.00567 | -0.01319 | 0.01560 | -0.03021 | 0.02711 | -0.00411 | -0.00969 | -0.01257 | -0.00298 | 0.01364 | -0.03369 | 0.03419 |
| -0.03280 | 0.03782 | -0.00859 | 0.01116 | 0.03764 | 0.01002 | -0.01604 | 0.00657 | -0.01095 | -0.00131 | -0.01091 | -0.00306 |
| 0.01313 | 0.00992 | -0.01826 | 0.00569 | -0.00784 | -0.00036 | 0.00054 | -0.02498 | 0.00327 | 0.01723 | 0.01260 | -0.02701 |
| 0.02636 | 0.00231 | 0.00536 | 0.01159 | 0.01661 | -0.01697 | 0.02716 | 0.01939 | -0.02203 | -0.00890 | 0.01528 | 0.00364 |
| -0.01096 | -0.04709 | 0.04288 | -0.02510 | 0.00279 | 0.05013 | -0.04991 | -0.00049 | -0.00508 | 0.00337 | 0.00965 | 0.00153 |
| -0.00179 | -0.00750 | 0.00454 | -0.02869 | -0.01646 | 0.02437 | -0.00279 | -0.00892 | 0.00257 | 0.00168 | -0.00110 | -0.00529 |
| 0.00521 | 0.00613 | -0.03899 | 0.01203 | 0.00883 | -0.01065 | -0.03113 | 0.00676 | 0.01163 | 0.00051 | 0.00690 | 0.01607 |
| 0.03323 | -0.00460 | -0.02054 | -0.00502 | 0.00470 | 0.00895 | 0.00054 | 0.00016 | 0.00022 | -0.00133 | 0.00003 | 0.00000 |
| 0.00000 | 0.00001 | -0.00002 | 0.00160 | -0.00013 | -0.00210 | 0.00091 | 0.00054 | 0.00664 | -0.00009 | 0.00424 | 0.00428 |
| 0.00199 | 0.00396 | 0.00008 | -0.00656 | -0.00009 | 0.00458 | 0.00080 | -0.01982 | -0.01725 | 0.02537 | 0.02457 | -0.01774 |
| 0.01298 | -0.02229 | 0.01050 | -0.01009 | -0.00828 | 0.00734 | 0.00031 | -0.02136 | 0.01342 | 0.04830 | -0.04742 | 0.00184 |
| -0.02427 | 0.00846 | -0.00719 | 0.01252 | 0.01299 | -0.02359 | 0.00486 | 0.00124 | 0.01243 | 0.03669 | -0.00706 | -0.02045 |
| 0.02702 | 0.01346 | 0.01854 | -0.01903 | 0.00769 | -0.04098 | -0.01300 | 0.01226 | 0.01045 | -0.01224 | 0.00210 | -0.03034 |
| 0.00991 | 0.04535 | -0.00711 | -0.03585 | 0.00474 | -0.00606 | 0.00525 | -0.00189 | 0.00059 | 0.03534 | 0.00002 | -0.01493 |
| 0.01760 | -0.02193 | 0.00133 | -0.00254 | -0.07497 | -0.03251 | -0.00471 | 0.04265 | -0.00481 | 0.02272 | 0.00639 | -0.00722 |
| 0.01122 | -0.01958 | 0.06212 | 0.01487 | -0.02964 | 0.03453 | -0.02469 | -0.00414 | 0.00216 | 0.00214 | 0.01497 | -0.05341 |
| 0.02979 | 0.00095 | 0.05929 | -0.06567 | 0.04092 | -0.06760 | 0.01865 | 0.01464 | 0.02467 | 0.00600 | -0.03954 | -0.00288 |
| 0.01707 | 0.04286 | 0.03416 | -0.00822 | -0.07919 | 0.03811 | -0.00057 | -0.02348 | -0.01534 | 0.01424 | 0.00214 | -0.04481 |
| -0.01563 | 0.07596 | -0.01342 | 0.08297 | 0.00944 | -0.04137 | -0.04017 | -0.03995 | 0.03403 | -0.01405 | -0.05025 | 0.02530 |
| 0.01414 | -0.03950 | 0.03499 | 0.00855 | -0.01179 | -0.03242 | 0.02544 | -0.02778 | 0.02888 | 0.06030 | -0.00069 | -0.04515 |
| 0.02820 | -0.00479 | 0.00548 | 0.04052 | -0.02104 | 0.00937 | -0.04342 | 0.03401 | -0.00749 | -0.00718 | 0.03134 | -0.04049 |
| -0.02979 | -0.02193 | 0.05046 | -0.00803 | -0.02521 | 0.04290 | -0.03260 | -0.01962 | 0.02533 | -0.00610 | 0.00973 | 0.01228 |
| 0.00294 | 0.00241 | 0.04575 | -0.03676 | 0.00997 | -0.02390 | -0.04649 | -0.07349 | -0.02849 | 0.04530 | 0.00601 | -0.00332 |
| 0.05246 | 0.01387 | -0.00228 | -0.01283 | -0.00970 | 0.04600 | 0.01315 | 0.00858 | 0.00114 | 0.00159 | 0.01097 | -0.05117 |
| -0.01721 | 0.01932 | 0.02923 | -0.02100 | 0.02054 | -0.00751 | -0.00124 | 0.04118 | -0.07411 | -0.01232 | -0.00261 | -0.00067 |
| 0.00032 | 0.01370 | -0.00150 | 0.00139 | 0.00908 | 0.05010 | 0.08275 | -0.05374 | -0.01445 | -0.05122 | 0.01620 | 0.00541 |
| -0.02821 | 0.00589 | -0.02300 | -0.00906 | 0.00697 | 0.06386 | 0.00467 | 0.02929 | -0.02533 | -0.00957 | -0.01354 | -0.02213 |
| 0.02373 | -0.00968 | -0.07308 | 0.03279 | -0.02228 | 0.06614 | -0.00996 | 0.03522 | -0.06951 | 0.00547 | 0.01829 | 0.02580 |
| -0.04364 | 0.09106 | -0.03212 | -0.04286 | 0.01143 | 0.08141 | -0.00717 | -0.04623 | -0.05082 | 0.00102 | 0.02454 | 0.05026 |
| -0.00445 | -0.12283 | 0.06078 | 0.01479 | 0.14633 | -0.03984 | -0.06474 | -0.03187 | -0.04067 | 0.00740 | -0.00436 | 0.08263 |
| -0.09596 | 0.02461 | 0.01448 | 0.00028 | 0.01968 | 0.04582 | -0.01060 | 0.00809 | -0.08952 | -0.05515 | 0.01271 | -0.08542 |
| 0.02326 | 0.03817 | -0.04519 | 0.05188 | 0.06726 | 0.01185 | 0.00545 | -0.01196 | 0.01222 | -0.00729 | 0.02892 | 0.02779 |
| -0.00819 | -0.00958 | -0.00630 | -0.09903 | 0.05782 | 0.02254 | 0.01039 | 0.00094 | -0.01394 | -0.01151 | 0.03381 | -0.06211 |
| 0.02428 | -0.04738 | -0.07471 | 0.09835 | 0.02191 | 0.07453 | -0.12715 | -0.00017 | 0.09766 | -0.01679 | 0.00236 | -0.02441 |
| -0.03713 | 0.04384 | 0.04655 | -0.01147 | -0.04249 | -0.05650 | 0.07483 | -0.03634 | 0.09015 | -0.07588 | 0.02375 | 0.05145 |
| -0.11234 | 0.04052 | 0.00630 | -0.06520 | 0.13152 | -0.08787 | -0.06674 | 0.07209 | 0.00484 | 0.03591 | -0.01128 | -0.03055 |
| -0.01444 | -0.02964 | 0.00861 | 0.08197 | -0.00453 | 0.04220 | -0.02300 | -0.02246 | 0.07292 | -0.02485 | -0.09905 | 0.00157 |
| -0.04284 | 0.11222 | -0.02379 | -0.01182 | -0.00738 | 0.00485 | 0.07045 | -0.01952 | 0.07603 | -0.03296 | -0.10262 | -0.01888 |
| -0.00437 | 0.00562 | 0.14571 | -0.01831 | -0.05463 | -0.05938 | 0.00479 | 0.08787 | -0.01343 | 0.04771 | -0.12189 | 0.02059 |
| -0.02628 | -0.12107 | 0.06617 | -0.03313 | -0.01370 | 0.07744 | -0.09330 | 0.07832 | 0.07398 | -0.01659 | 0.02401 | -0.03771 |
| -0.00465 | 0.02382 | 0.01194 | 0.02777 | 0.03330 | -0.08049 | -0.01682 | -0.01191 | 0.05971 | -0.00909 | 0.00828 | -0.10154 |
| 0.06963 | -0.00531 | 0.04445 | -0.04499 | -0.08345 | -0.01156 | 0.06706 | 0.06335 | -0.00367 | -0.04779 | 0.00694 | -0.01134 |
| 0.03292 | 0.01396 | 0.00194 | -0.00621 | -0.02573 | -0.00163 | -0.00498 | -0.00233 | 0.00460 | -0.00001 | 0.00456 | -0.00384 |
| -0.00130 | -0.00584 | 0.00252 | 0.00006 | -0.00041 | -0.00008 | 0.00565 | 0.01190 | 0.00400 | -0.02696 | 0.00076 | 0.01265 |
| 0.00941 | -0.00169 | -0.00821 | 0.00482 | -0.02117 | 0.00707 | 0.00844 | -0.00834 | 0.01158 | -0.00100 | 0.02327 | -0.00948 |
| -0.02126 | 0.05150 | -0.01499 | -0.00667 | -0.02253 | 0.01187 | 0.00041 | 0.01722 | 0.00825 | 0.01688 | -0.01924 | -0.00118 |
| -0.00662 | 0.00098 | 0.00115 | -0.01270 | -0.00076 | 0.00017 | -0.00264 | -0.00682 | 0.01059 | -0.03333 | -0.00651 | 0.00019 |
| 0.02948 | -0.00685 | -0.00708 | -0.00954 | -0.01692 | -0.01248 | 0.00911 | -0.00040 | -0.01273 | -0.01118 | 0.00859 | 0.02343 |
| -0.00753 | 0.01932 | 0.01387 | 0.01511 | 0.02524 | -0.00586 | 0.02495 | 0.00394 | -0.02284 | -0.02842 | -0.00094 | 0.00119 |
| 0.01528 | -0.03077 | 0.00839 | -0.00744 | -0.00286 | 0.02406 | 0.00313 | -0.01367 | 0.00062 | 0.00792 | 0.03652 | -0.07529 |
| 0.03058 | 0.07075 | -0.10442 | 0.03400 | 0.04751 | -0.06604 | 0.02865 | 0.03147 | -0.05350 | -0.00071 | 0.06258 | -0.00154 |
| -0.12543 | 0.07388 | 0.02951 | 0.01952 | -0.03289 | -0.01984 | 0.03215 | -0.02851 | 0.01465 | 0.00342 | -0.02809 | -0.00757 |
| 0.00032 | 0.04101 | -0.04004 | 0.02586 | -0.03368 | 0.01588 | 0.02905 | -0.02927 | 0.02371 | -0.03262 | 0.05640 | -0.03008 |
| 0.01922 | -0.05277 | 0.04188 | 0.02020 | -0.05271 | -0.01388 | 0.05580 | 0.03056 | -0.01606 | 0.00454 | 0.00009 | -0.00949 |
| -0.05329 | 0.11555 | -0.07641 | -0.05815 | 0.03488 | 0.04698 | -0.07280 | 0.07817 | -0.03649 | -0.05432 | 0.06865 | 0.00538 |
| -0.06509 | -0.05990 | 0.06086 | 0.06114 | -0.01833 | -0.06514 | -0.00801 | 0.01167 | -0.01986 | -0.01530 | -0.00329 | -0.03821 |
| 0.03632 | 0.04100 | 0.00057 | 0.01206 | -0.00264 | 0.01499 | 0.03872 | 0.02078 | -0.02751 | 0.03275 | -0.03575 | 0.06486 |
| -0.05243 | -0.02902 | -0.01420 | 0.03612 | -0.00480 | -0.03727 | 0.01339 | 0.05054 | -0.03297 | | | |

**Fig. 3.16 A Selected Sample Eigenvector**

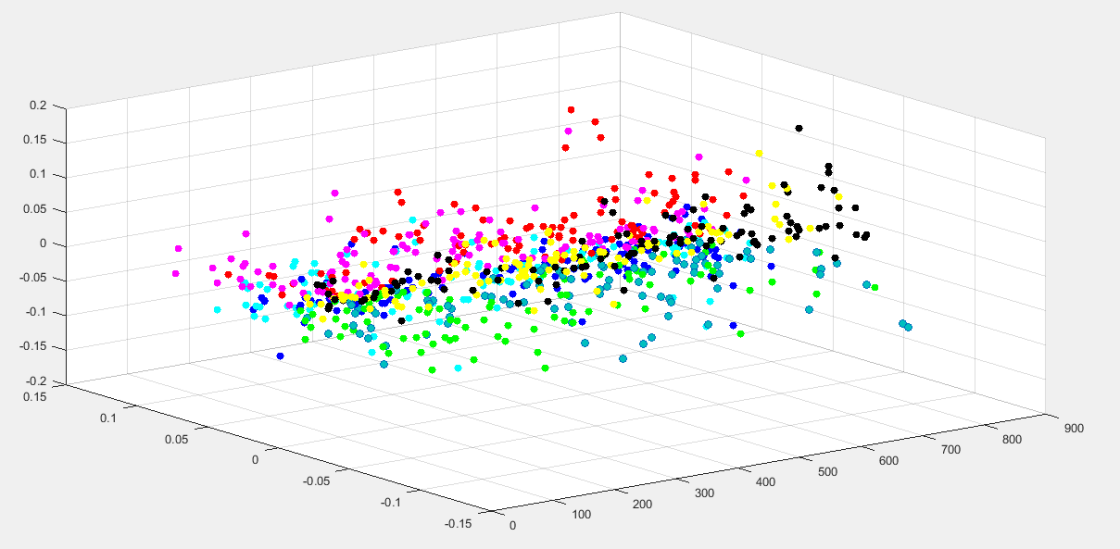
The first level, second level and third level partition of the graph representing the sample speech formed into clusters denoting the vertex sets of subgraphs is shown in Fig. 3.17. As a result of partitioning the graph into clusters, the speech signal is broken down into phonetic unit by identifying the boundary points. Each subgraph is a forest with a collection of connected graphs, where each graph in that forest represents a phonetic unit of the spoken sentence.




(a) First level partition




(b) Second level partition




(c) Third level partition

**Fig. 3.17 Partitioning Graph Vectors Representing Speech**

The boundary points are identified in terms of frame number. The boundary points identified for the sample speech '**நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும்**', referred as sample 1 is shown in Fig 3.18. The segmentation is performed hierarchically until the desired level of segmentation is achieved. Fig. 3.19 shows the segmentation results of the sample sentence along with the hand segmentation. The red vertical lines in the plot shows the boundary points of hand segmentation and the green vertical line shows the boundary points acquired through graph cut based segmentation. The hand segmentation of the sample speech has 37 boundary points.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 17 | 33 | 49 | 65 | 82 | 99 | 115 | 131 | 147 | 163 | 179 | 195 |
| 211 | 227 | 243 | 259 | 275 | 291 | 307 | 323 | 339 | 355 | 371 | 387 |
| 403 | 419 | 435 | 451 | 467 | 483 | 499 | 515 | 531 | 547 | 563 | 579 |
| 595 | 611 | 627 | 643 | 659 | 675 | 691 | 707 | 723 | 739 | 801 | |

**Fig. 3.18 Frame Numbers of Boundary Points Extracted using Graph Cut Algorithm for Sample 1**

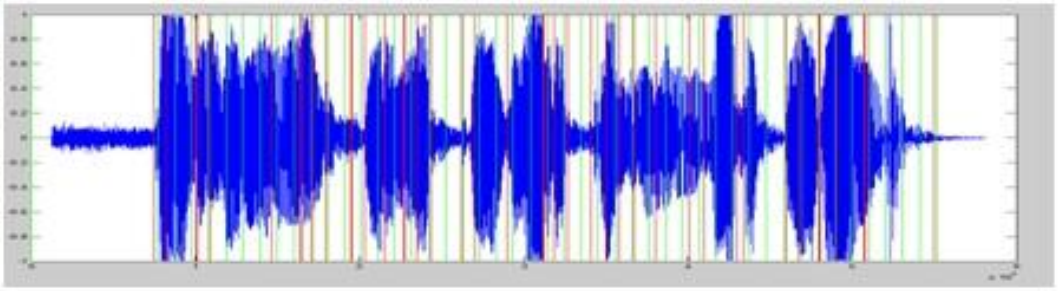


**Fig. 3.19 The Sample Speech 'Thinai ennum sollukku ozhukkam enbathu porul' Showing the Hand Segmentation Points (red) and Graph Cut based Segmentation Points (green)**

### 3.4.2 Performance Evaluation of Graph Cut Based Segmentation

The performance of graph cut based segmentation is analyzed to study the influence of selecting eigenvectors corresponding to high, low and median eigenvalues respectively. The Table V shows the number of boundary points identified in manual segmentation, boundary points identified in graph cut based segmentation algorithm on three different speech samples listed below spoken by three different individuals contributed by two males and one female while selecting eigenvector with low, median and high eigenvalues respectively.

Sample 1: நம்முடைய எண்ணம் எப்போதும் உயர்ந்ததாக இருக்க வேண்டும் (male)

Sample 2: திணை என்னும் சொல்லுக்கு ஒழுக்கம் என்பது பொருள் (female)

Sample 3: வைகை நதி மதுரையில் பாய்கிறது (male)

The actual number of boundary points of the three sample speech 1, 2 and 3 are 47, 37 and 27 respectively. The number of boundary points identified by the graph cut segmentation algorithm for sample 1 is 45, 47 and 50 respectively for eigenvectors chosen with low, median and high eigenvalues. Out of which the correctly identified boundary points are 37, 45 and 32 respectively.

Similarly, for sample 2 the correctly identified boundary points are 30, 37 and 23 out of total number of retrieved boundary points 38, 48 and 30 for low, median and high eigenvectors using graph cut algorithm, where it is 20/25, 25/28 and 9/19 for sample 3. At the same time, the number of boundaries missed by the algorithm is observed as 10/47, 7/37 and 2/27 while selecting high eigenvalues vectors for sample 1; 0/47, 0/37 and 2/27 while selecting median eigenvalues vectors; and 15/47, 14/37 and 10/27 while selecting low eigenvalue vectors. Thus,

eigenvectors with median eigenvalues seem to miss only minimal amount of boundary points. When coming to false positives or the number of wrongly predicted boundaries, the algorithm turned out with values 8/45, 8/38 and 5/25 for high eigenvalue vector; 3/47, 11/48 and 3/28 for median eigenvalues vector; and 18/50, 7/30 and 3/19 for low eigenvalues vector conforming median eigenvalues vectors for the contribution of detecting wrong boundaries. The study showed that the eigenvectors with median eigenvalues supports to achieve optimal segmentation and thus used throughout this research.

**Table V Segmentations Achieved as Result of Choosing Eigenvectors with High, Median and Low Eigenvalues for Sample Speech 1, 2 And 3**

| Input Speech | Eigen Value Selection | Manual Segmentation | Graph cut based segmentation | True Positives | False Negatives | False Positives |
|---|---|---|---|---|---|---|
| **Sample 1** | High | 47 | 45 | 37 | 10 | 8 |
| | Median | | 47 | 44 | 0 | 3 |
| | Low | | 50 | 32 | 15 | 18 |
| **Sample 2** | High | 37 | 38 | 30 | 7 | 8 |
| | Median | | 48 | 37 | 0 | 11 |
| | Low | | 30 | 23 | 14 | 7 |
| **Sample 3** | High | 27 | 25 | 20 | 7 | 5 |
| | Median | | 28 | 25 | 2 | 3 |
| | Low | | 19 | 9 | 10 | 3 |

The efficiency of proposed graphcut based segmentation in terms of precision, recall and F-measure of phoneme segmentation is compared with existing Blind Segmentation using Non-Linear Filters (BSNLF) [78] and Non-Uniform Segmentation using DWT (NUSDWT) [79]. The comparative results are shown in Table VI. It is found that the proposed graph cut based segmentation method outperforms than the other two as precision, recall and F-measure are 0.9259, 0.8928 and 0.9090 respectively. These outcomes fortified the use of proposed graph cut

based segmentation for segmenting the phonetic units of the continuous speech in Kazhangiyam speech corpus and to construct the dataset required to train the acoustic models.

**Table VI Comparing the accuracy of Graph Cut Based Segmentation with BSNLF and NUSDWT**

| Algorithm | Precision | Recall | F-measure |
|---|---|---|---|
| **BSNLF** | 0.7977 | 0.8189 | 0.8067 |
| **NUSDWT** | 0.6156 | 0.5860 | 0.7172 |
| **Graph cut** | **0.9259** | **0.8928** | **0.9090** |

### 3.5 KAZHANGIYAM DATASET

Once the phoneme segments are extracted using graph cut based segmentation from the wav file as discussed above, dataset for the segmented phonemes in the continuous speech has been built. The dataset is formed by initially consolidating the six low dimensional DWT features in each individual frame of phoneme segmented from continuous Tamil speech. Each phoneme is represented or defined by 15 frames with a total of 90 attributes contributed by 6 features from each frame. For short phonemes, whose actual number of frames is less than 15, zeros are appended to the trailing features of the respective phonemes. In case of longer phonemes, whose actual number of frames is greater than 15, the features contributed by the additional frames are truncated. The feature vectors formed are then subjected to min-max normalization in order to put in the data to a more suitable form for training models like neural networks. A feature vector of phoneme /a/ is shown in Fig. 3.20. The sample feature vectors for the remaining phonemes are shown in Appendix A.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| -0.31742 | -0.03589 | 0.34661 | -0.00356 | -0.00485 | 0.01887 | -0.42121 | -0.26165 | -0.30463 |
| 0.26840 | -0.13945 | 0.04324 | -0.13505 | -0.08544 | -0.02347 | -0.16341 | 0.05407 | -0.04594 |
| 0.41701 | 0.20288 | 0.01646 | -0.03295 | 0.08554 | -0.03309 | 0.22900 | -0.18302 | -0.12041 |
| 0.11799 | -0.04094 | 0.04953 | 0.01719 | 0.29205 | -0.01567 | -0.09883 | -0.00758 | 0.00272 |
| -0.31481 | 0.16094 | 0.06465 | -0.02673 | 0.03129 | -0.02804 | -0.10616 | -0.09768 | -0.02862 |
| 0.07307 | -0.05729 | 0.02950 | 0.20157 | 0.18040 | 0.01839 | -0.15720 | 0.07401 | -0.03030 |
| 0.15488 | 0.08216 | 0.09935 | 0.07594 | -0.00692 | 0.00569 | 0.08107 | -0.59539 | -0.15088 |
| 0.15071 | -0.10100 | 0.04439 | -0.18458 | 0.26983 | -0.24479 | -0.03569 | 0.04657 | -0.04682 |
| 0.26981 | -0.31344 | 0.38504 | -0.12369 | 0.07086 | -0.01497 | -1.00519 | -0.08045 | 0.15342 |
| 0.02100 | -0.00505 | 0.02551 | 0.32446 | 0.16301 | -0.03945 | -0.02501 | 0.00207 | 0.00041 |

**Fig 3.20 Feature Values of Phoneme /a/**

The phonetic instances segmented from the speech in Kazhangiyam corpus were grouped under various phonetic classes and found to contribute as listed in Table VII in building datasets. The total number of phonetic instances sums up to 6,67,260 samples.

**Table VII Phonetic Instances in each Class of Kazhangiyam Database**

| Class Name | Tamil Grapheme | Instances |
|---|---|---|
| a | அ | 71495 |
| aa | ஆ | 22940 |
| i | இ | 37565 |
| ii | ஈ | 7925 |
| u | உ | 34445 |
| uu | ஊ | 7535 |
| e | எ | 12410 |
| ee | ஏ | 9485 |
| ai | ஐ | 14555 |
| o | ஒ | 7145 |
| oo | ஓ | 10070 |
| au | ஔ | 5390 |
| K | க் | 29730 |
| G | க் | 17640 |
| Ng | ங் | 9645 |
| Ch | ச் | 8450 |
| S | ச், ஸ் | 11765 |
| Sh | ஷ் | 6890 |
| nj | ஞ் | 7085 |
| gh | ஞ் | 7280 |

| Class Name | Tamil Grapheme | Instances |
|---|---|---|
| t | ட் | 15275 |
| d | ட் | 12545 |
| nn | ண் | 10205 |
| th | த் | 35445 |
| dh | த் | 9510 |
| N | ந் | 13995 |
| p | ப் | 19065 |
| b | ப் | 7755 |
| m | ம் | 26865 |
| ei | ய் | 13115 |
| R | ர் | 24720 |
| L | ல் | 21990 |
| V | வ் | 22575 |
| zh | ழ் | 7755 |
| L | ள் | 17370 |
| rr | ற் | 13410 |
| n | ன் | 19015 |
| J | ஜ் | 4975 |
| sil | | 32230 |
| **Total** | | 667260 |

The 39 phonemes segmented from the speech samples of the Kazhangiyam corpus have been grouped as per the phonetic sounds of Tamil alphabet system is shown in Table VIII. The category vowel is formed with twelve phonetic classes comprising short vowels, long vowels and dipthongs, nasals with seven phonetic classes, stops with eight and the remaining eleven classes grouped under others with fricatives, glides and grantha category.

**Table VIII Categories of Phonemes in Tamil Language**

| Broad Categories | Phonetic Classes | Number of Samples |
|---|---|---|
| Vowels | /a/,/aa/,/i/,/ii/,/u/,/uu/,/e/,/ee/,/ai/,/o/,/oo/,/au/ | 240960 |
| Nasals | /m/,/n/,/N/,/nn/,/nj/,/gh/,/ng/ | 94090 |
| Stops | /p/,/b/,/t/./d/,/th/,/dh/,/k/,/g/ | 19065 |
| Others | /s/,/zh/,/ch/,/sh/,/ei/,/r/,/rr/,/L/,/j/,/v/,/l/ | 280915 |

Thus 667260 samples of dimension 90 have been generated and created the dataset called Discrete Wavelet Transform Feature Set (DWTFS).

### 3.6 TRAINING AND TESTING

Generally, the dataset is split into training set and testing test or validation test. The training set is used to build or fit a model. The testing set is used to validate the model by estimating the error or evaluating the accuracy of the model that is built using the training set. To define an effective test set, two points needs to be considered:

- The test set should be large enough that is capable of producing consequential statistical results.
- It acts as a representative of the complete dataset catering samples of various kinds and thus showing the characteristics through samples similar to the ones in the original dataset.

The model or the hypothesis is considered to be efficient if it works fine on the test set. In this research the proportion of the train and test sets are considered to be 70% and 30% respectively.

#### Model Building

Once the training and testing datasets are derived, the next step is to design a model infrastructure to build the classification model, and decide the algorithms to train the model. This research is concerned with building competent machine learning based acoustic models efficient enough to classify the phonetic units segmented from the continuous Tamil speech. Deep

learning technique namely, DBN is used in this research to build the acoustic models through training dataset, which is preceded by building and analyzing simple acoustic models using ANN and ANFIS. The models are then tested for its performance by subjecting it to the test data set. The performances of the models are validated using the metrics discussed in the subsequent section.

**Model Evaluation**

There are various metrics available to estimate the efficiency of the model being built. Classifier's accuracy is one such metric that helps to evaluate how good the classifier is capable to identify the designated class of the incoming data. It is also used to compare the efficiency of various classifiers and identify the suitable one to solve the given problem. Error rate is another popular metric that is generally used to evaluate the efficiency of the model by calculating the contribution of error. This estimate is considered to provide more accurate results when the test dataset is larger in size. Several techniques are available for assessing the accuracy of the model few of which are hold-out method, cross-validation method, k-fold cross-validation method and leave-one-out cross-validation method. This research uses hold-out method throughout for testing the model being built. Hold-out is an assessing technique that splits the dataset into train set and test set. The train set is used to build the model using some training procedure and the test set is used to test the model that was trained using the training set by passing the unseen data to the model and predicting the outcome. The evaluation of the model based on this method is highly variable and is highly dependent on the data points placed in the train and test sets.

The efficiency of the models built can be evaluated using various standard metrics. The various metrics like precision, recall, F-measure and accuracy. These measures can be computed using confusion matrix.

**Confusion Matrix**

Confusion matrix is the matrix representation of the observed results in classification of test data. The confusion matrix of a binary classification takes the structure defined in Fig. 3.23. True positive counts if the classifier classifies the data point to actual true class. False positive counts if the classifier has predicted the data point to true class which is actually in false class. False negative counts if the classifier predicted the input to belong to false class which is actually in true class. True negative counts if the classifier classifies the data point to false class which is the actual case.

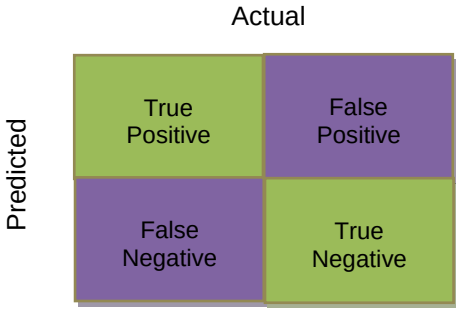



**Fig. 3.21 Confusion Matrix**

Several measures that are defined with confusion matrix as a core are used in this research and are defined below:

***Precision.*** It defines the proportion that has been correctly identified as positives to the total number of identified positive classifications and is given as follows,

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{3.15}$$

***Recall.*** It defines the proportion of actual positives that was identified correctly as positives and is given as follows,

$$Recall = \frac{True\ positives}{True\ positives + False\ Negatives} \tag{3.16}$$

***F-measure.*** It is a score that is defined to be the harmonic mean of both precision and recall and is given by,

$$F - measure = 2.\frac{Precision.Recall}{Precision + Recall} \tag{3.17}$$

***Accuracy.*** It defines how often the classifier gives the correct outcome and is calculated as follows,

$$Accuracy = \frac{(True\ positives + True\ Negatives)}{Total} \tag{3.18}$$

**Receiver Operating Characteristics (ROC) Curve**

It is a graphical representation of the ratio between True positive rate and false positive rate as shown in Fig. 3.24. True positive rate is the ratio between total numbers of positive predictions of instances in true class to the total number of actual instances in true class, whereas false positive rate is the ratio between total number of negative predictions of instances in true class to total number of actual instances in true class. More the area under curve then the model is considered to be more accurate. The diagonal dotted line shows the ratio equivalent to 0.5. The

curve below this level or equal designates the model to be less accurate and not good to use. At the same time, when the ratio is 1, then the model is considered to be fit.

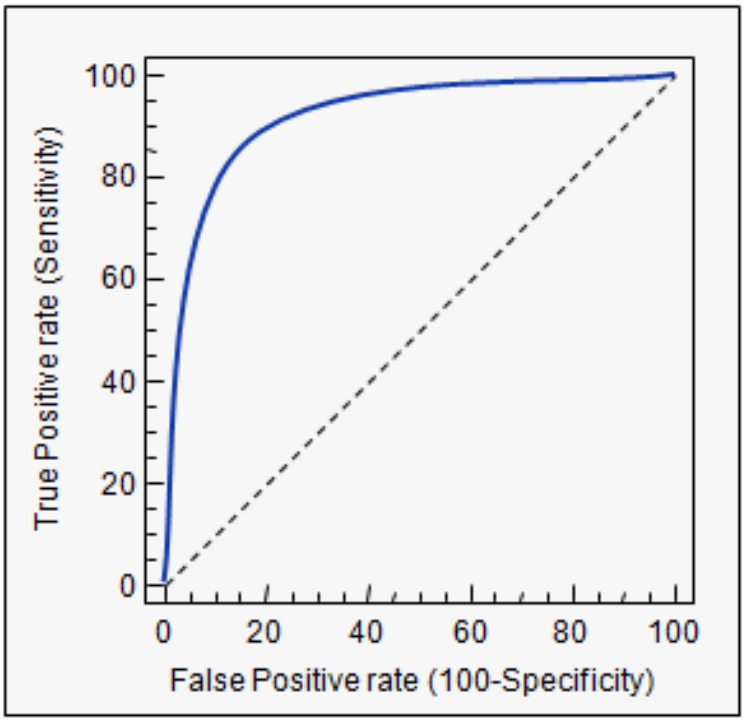



**Fig. 3.22 Sample ROC Curve**

**Mean Square Error (MSE) and Root Mean Square Error (RMSE)**

It is a measure that estimates average of squared error which is the square difference between the predicted output, $y_i$ and the actual output, $o_i$ for *n* predictions. It is given as follows,

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - o_i)^2 \tag{3.19}$$

Whereas, RMSE is defined as a square root of MSE and is given as follows,

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}} \tag{3.20}$$

**Phoneme Error Rate (PER)**

This error is defined as a percentile of ratio of sum of all phoneme errors like insertion *IP*, deletion *DP* and misclassification *MC* to the total number of phonemes expected in output *EP*. It is given as follows,

$$\mathrm{PER} = \frac{IP+DP+MC}{EP} \times 100 \tag{3.21}$$

## SUMMARY

The process of modelling phoneme recognition in Tamil continuous speech with various tasks was demonstrated in detail in this chapter. The Tamil speech corpus developed by collecting the speech utterances from the speakers were presented appropriately. A detailed explanation of feature extraction using DWT on speech signal was presented precisely. The proposed graph cut segmentation method and its performance analysis was illustrated in detail. The dataset representing the phonetic units and their categorization were detailed. The performance metrics used throughout this research to evaluate the efficiency of the various proposed models were also mentioned in this chapter. Various models proposed for Tamil phoneme recognition in continuous speech will be described in the subsequent chapters.

### *Remarks*

- The article titled 'Graph Cut Based Segmentation Method for Tamil Continuous Speech' is published in proceedings of 51st Annual Convention of the Computer Society of India: Digital Connectivity – Social Impact, Communications in Computer and Information Science 679, CSI Coimbatore Chapter, Springer, Singapore, 2016 ISBN: 978-981-10-3273-8, ISBN: 978-981-10-3274-5 (ebook), pp: 257-267. **(Scopus Indexed)**