International Journal of Modern Communication Technologies & Research (IJMCTR)
ISSN: 2321-0850, Volume-1, Issue-1, March 2013

Efficient Speech/Music Discrimination in Real Time Application

Dheeraj Kumar Joshi, Sandhya Sharma, Prashant Moud

***Abstract*— *While* many efforts have been made in the audio signal classification field, the noise embedded signal problem is seldom concerned so far, especially in many telecommunication applications, where a real-time and noise robust approach is needed. In this paper a 'Four Class Speech/Music Classifier' is proposed which gives 95% classification accuracy in real world noise with less computation time.**

Keywords— White noise, Colored noise, Pink noise, Signal to Noise Ratio (SNR), Support Vector Machine (SVM).

I. INTRODUCTION

Many available speech/ music classifier gives very good classification accuracy for clean audio input. But when we use these systems in noisy atmosphere like: army or military field areas, highways, traffic signals, process industries and railway stations, their classification accuracy are decreased. So there should be a classifier which gives good accuracy for these real world noisy applications. To design such type of noise robust system for classification, energy, pitch and cepstrum based feature are used. For noise embedding purpose colored and pink noise signals are used and for classification purpose Support Vector Machine [10, 11] is used, as it is most efficient and accurate classifier.

II. MOTIVATION

In real far-field area, available input audio signal is not only speech or music type, it may be speech containing music or other signal. So designed system should classify these four classes. As available surrounding environmental noise is not pure white Gaussian noise, it is colored noise. Pink noise is a type of colored noise that is appropriately resembles with real human audible noise. Thus classification should be done on this noise. Further in real world noise-embedding is not on fix SNR. So discrimination should be done on random signal to noise ratio.

Manuscript received March 14, 2013.
Dheeraj Kumar Joshi, M.Tech Scholar DWCE, Suresh gyan vihar University, Jaipur, India, (e-mail:erdj07@gmail.com).
Sandhya Sharma, Associate Professor ,Electronics Department , Suresh gyan vihar University, Jaipur, (e-mail:sandhyasharma.mbm@gmail.com).
Prashant Moud, B.E, M.Tech (e-mail: prashantmoud24@gmail.com).

III. PROPOSED WORK

Feature Extraction- For analysis purpose features are extracted on clean then different-different type of noise is embedded to input signal on random SNR and again same feature values are extracted. Following clip level features are extracted and analysed:

Average Pitch Density (APD) - It represents the differences of tones between speech and music. Real cepstrum is used to analysis the pitch information, since cepstrum is a powerful tool to show the details of spectrum by separating pitch information from spectral envelope. The real cepstrum is usually defined as

$$\mathbf{rc_x(n)} = \mathbf{real}\left(\frac{1}{2}\cdot\int_{-\pi}^{\pi}\log(\mathrm{abs}(X_n(jw))e^{jwn}dw\right) \qquad (1)$$

. Where $\mathbf{X_n(jw)}$ is the short-time Fourier transform of the n^{th} windowed audio frame, *n* is the frame index and *real* (·) denotes extracting the real part of the complex value. $\mathbf{rc_x(n)}$ is a vector that contains all real cepstral coefficients of the n^{th} frame signal. The low-order coefficients of $\mathbf{rc_x(n)}$ refer to the big scale information of spectrum like formants, and the high-order coefficients contain the detail information like pitches. High order coefficients are captured to distinguish between speech and music. Since in most telecommunication applications, the audio data are usually disturbed by unpredictable noises, the estimation of the accurate pitch positions and the holding lengths of pitches in real-time might be very difficult. So the pitch density (PD) is used to roughly characterize the pitch properties of music and speech, which is defined as

$$\mathrm{PD(n)} = \frac{1}{L}\sum_{m=l_1}^{l_2}\mathrm{abs}(\mathbf{rc_x(n,m)}) \qquad (2)$$

Where $L = l_2 - l_1 + 1$

Where $\mathbf{rc_x(n,m)}$ is the *m*-th coefficient of$\mathbf{rc_x(n)}$. *PD* (*n*) is the mean of absolute values of high-order coefficients of $\mathbf{rc_x(n)}$ within [*l1*, *l2*]. Based on empirical analysis, the average of overall high-order cepstrum content is used. For music signals, due to the characteristics of musical instruments and the existence of polyphony, the PD is tend to be higher than that of speech signals. To get a more reliable

estimation, the average PD (APD) within an audio segment is used, which is defined as

$$APD(k) = \frac{1}{N}\sum_{n=k\beta N+1}^{k\beta N+N} PD(n) \qquad (3)$$

Where *N* is the number of frames contained in an audio segment, *k* is the segment index, and β is the overlapping factor of each segment. Note that *APD* (*k*) is a scalar extracted from the k^{th} audio segment.

Relative Tonal Power Density (RTPD) - RTPD especially focuses on the distinct properties of the percussion instruments. For the noise-like music RTPD is considered. Firstly, every audio frame is marked as a tonal-frame or a non-tonal-frame according to the maximum of the high-order coefficients of $rc_x(n)$. That is, if the maximum value is bigger than a predefined threshold θ, indicating a significant peak in the high-order part, the frame is then marked as a tonal-frame. Secondly, within the current audio segment, we compute the relative power density ratio of the tonal-frames to overall frames, i.e. RTPD, as

$$RTPD(k) = \left(\frac{1}{M}\cdot\sum_{n=\theta_k} RMS_x(n)\right) \Big/ \left(\frac{1}{N}\cdot\sum_{n=k\beta N+1}^{k\beta N+N} RMS_x(n)\right) \qquad (4)$$

Where θ*k* denotes all tonal-frames inside the k^{th} analysis segment. *RMSx*(*n*) is the root mean square of the n^{th} windowed audio frame [1]. Note that *RTPD* (*k*) is also a scalar extracted from the *k*-th short audio segment.

The voiced speech usually has stronger energy than the unvoiced speech and the background noise, so that if the RTPD value is small, the audio signal may not be speech, which might be a clip of noisy music, such as rock music.

Variance of Zero Crossing Rate (varZCR) - It is defined as the variance of zero crossing rates for a one second clip, whereas zcr is defined to be the number of time domain zero crossings within a processing window. A zero crossing is said to occur if successive samples have different algebraic signs. Thus, the zero-crossing rate [2] is the rate of sign-changes along a signal, i.e., the rate at which the signal changes from positive to negative or vice-versa.

$$ZCR = \frac{1}{2\,(M-1)}\sum_{m=1}^{M}\left|\text{sign}(x(m)) - \text{sign}(x(m-1))\right| \qquad (5)$$

Where, M is total number of samples in a processing window and

x (m) is the value of m^{th} sample.

High ZCR values correspond to a higher frequency signal portion and vice-versa.

$$\text{var}(ZCR) = \frac{1}{N}\sum_{i=1}^{N}[ZCR(i) - \text{avg}(ZCR)] \qquad (6)$$

Where avg (ZCR) is average value of the clip and ZCR (i) is the i^{th} frame value.

Percentage of Low Energy Frame (POLEF)- STE is defined to be the sum of squared time domain data. This feature can be used in discrimination of audio on the basis of energy. The short time energy of a frame is given as

$$STE = \sum_{m=0}^{M-1} x^2(m) \qquad (7)$$

Where, M = total no of samples in a processing window, and x (m) = value of the m^{th} sample of input speech signal

POLEF is defined as the percentage of frames whose STE value is below 0.5 times average Short Time Energy of a particular window .

$$POLEF = \frac{1}{2N}\sum_{n=0}^{N-1}[\text{sign}(0.5\,avSTE - STE(n)) + 1] \qquad (8)$$

where N is total number of frames, n is the frame index, STE(n) is the Short Time Energy at nth frame, avSTE is the average value of STE over the entire window length and sign ( ) is the sign function.

Variance of Spectral Flux (varSF) –

It is defined as the variance of the spectral flux of a clip whereas spectral flux is the average variation value of a spectrum between the adjacent two frames of one second duration.

$$SF = \frac{1}{(N-1)(K-1)}\sum_{n=1}^{N-1}\sum_{k=1}^{K-1}(|X_n(k)| - |X_{n-1}(k)|)^2 \qquad (9)$$

Where $|X_n(k)|$ is the Discrete Fourier Transform of n^{th} frame of input signal. K is the order of DFT, N is the total number of frames in a clip and n and n-1 are the frame indices.

$$varSF = \frac{1}{N}\sum_{i=1}^{N}[SF(i) - \text{avg}(SF)] \qquad (10)$$

Where avg (SF) is average value of the clip and SF (i) is the i^{th} frame value.

It is a measure of how quickly the power spectrum of a signal is changing which is calculated by comparing the power spectrum for one frame against the power spectrum from the previous frame. It is usually calculated as the 2-norm (also known as the Euclidean distance) between the two normalized spectra.

Variance of RMS (varRMS) - It is defined as the variance of root mean squire value (RMS) for a one second clip. For this purpose first buffer the one second clip into frames of 32ms at 8 kHz. Then evaluate root mean squire value for each sample and then find the variance using following formula.

$$RMS = \sqrt{\sum_{i}^{K} x(i)} \qquad (11)$$

$$varRMS = \frac{1}{N}\sum_{i=1}^{N}[RMS(i) - \text{avg}(RMS)] \qquad (12)$$

Where avg (RMS) is average value of the clip and RMS (i) is the i^{th} frame value.

Dynamic Range (DR) - It is defined as the ratio between the largest and smallest possible values of a changeable quantity,

such as sample value audio signal. It is measured as a ratio, or as a base-10 (decibel) or base-2 (doublings, bits or stops) logarithmic value. For this purpose first signal is normalized.

$$DR = 20\{log10(\max(100 * y)) - log10(\min(100 * y))\} \quad (13)$$

Where, *y* is one second audio clip.

Average Mel-Frequency Cepstrum Coefficients (avgMFCC) - The motivation for using MFCC is due to the fact that the auditory response to the human ear resolves frequencies non -linearly. MFCC's are based on the known variation of the human ear's critical bandwidths with frequency; filters spaced linearly at low frequencies and logarithmically at high frequencies have been used to capture the phonetically important characteristics of speech.

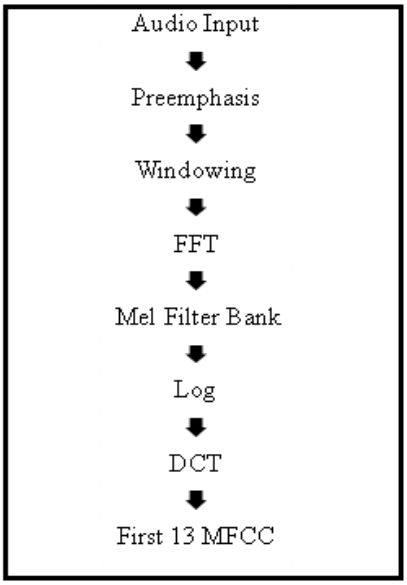

Fig.1 Steps in MFCC computation

The computation of Mel-frequency cepstrum is similar to that of cepstral coefficients**.** The difference lays on mel-frequency warping before doing logarithmic and inverse DFT.

Take average of all 31 frames and retrieve 12 MFCC coefficients for a clip. Then finally find out average MFCC of these 12 coefficients and get a single value for linear SVM input.

Mean of Minimum Cepstral Distance (MMCD) - The MMCD parameter is based on cepstral distances using Mel frequency cepstral coefficients (MFCC). The MMCD parameter finds a minimum cepstral distances among the neighbor frames.

Since the low-order coefficients of cepstrum represent the spectral envelope, the cepstral distances between two frames becomes a parameter to measure the difference between them. The cepstral distance between the n^{th} and $(n+d)^{th}$ frame is defined as follows:

$$CD(n,d) = \sqrt{\sum_{k=1}^{K} (c(n+d) - c(n))^2} \quad (14)$$

Where K is the order of cepstrum, c (n, k) is k^{th} cepstral coefficient of n^{th} frame and d represents the frame interval between the two frames to be compared.

The mean of cepstral distances is defined as follows:

$$\mu_{CD} = \sum_{n=1}^{N-d} CD(n,d) \Big/ N - d \quad (15)$$

Then modified cepstral distance (MCD) is given by

$$MCD(n, d_1, d_2) = \min_{d_1 \le d \le d_2} CD(n,d) \quad (16)$$

Where d1 and d2 indicate the range of candidate frames to be searched for minimum cepstral distance. Then we compute the MMCD, the mean of MCD, as follows:

$$\mu_{MCD}(d_1, d_2) = \sum_{n=0}^{N-d_1} MCD(n, d_1, d_2) \Big/ N - d_1 \quad (17)$$

Average Delta Cepstral Energy (avgDCE) - The delta cepstrum measures the temporal change in audio characteristics and can be used to track energy change in speech or music over time. The energy variation can be observed by analyzing the sum of the squares of the delta cepstral coefficients for each frame. This sum of squares of the delta coefficients is termed the Delta Cepstral Energy (DCE). The DCE is computed using

$$E_{di} = \sum_{j=1}^{N} (d_{ij})^2 \quad (18)$$

Where d_{ij} is the j^{th} delta cepstral coefficient of the i^{th} frame, N is the number of delta coefficients and E_{di} is the DCE for the i^{th} frame. The computation of the delta MFCC coefficients is given by:

$$d_t = \sum_{k=1}^{N} k(c_{t+k} - c_{t-k}) \Big/ 2.\sum_{k=1}^{N} k^2 \quad (19)$$

Where N represents the delta window size, c_t represents the MFCC at frame t and d_t is the delta coefficient for frame t. Then finally find out average DCE of these 29 coefficients and get a single value for linear SVM input.

Average Power Spectrum Deviation (avgPSDev) - Speech has greater energy at low frequencies, however, in the case of music, the higher frequencies also have significant energy. Thus, the energy in each filter of filter bank can also be used for speech and music discrimination. Power Spectrum Deviation (PSDev) is computed as the standard deviation of filter bank energies in each band. Thus, PSDev can be found using

$$P_i = \frac{1}{n-1} \cdot \sum_{j=1}^{N} (E_{ij} - avgE_i)^2 \quad (20)$$

Where P_i is the PSDev for the i^{th} frame, N is the number of filters in a filter bank and E_{ij} is the energy in the j^{th} filter of i^{th} frame. Where $avgE_i$ is the mean energy value for all filters in the i^{th} frame and can be computed using

$$avgE_i = \frac{1}{n} \cdot \sum_{j=1}^{N} E_{ij} \quad (21)$$

Then finally find out average PSDev of these 29 coefficients and get a single value for linear SVM input.

Noise - Generally in communication, white noise is used for noise representation. But for real world noise presentation white noise is not correct option as true white noise has infinite power with infinite bandwidth. To represent real world noise, colored noise is used as it has finite power for limited bandwidth. To represent physical noise (audible) pink noise is used as it has constant energy per constant percentage bandwidth. Comparision of white, colored and pink noise can see from figure 2.

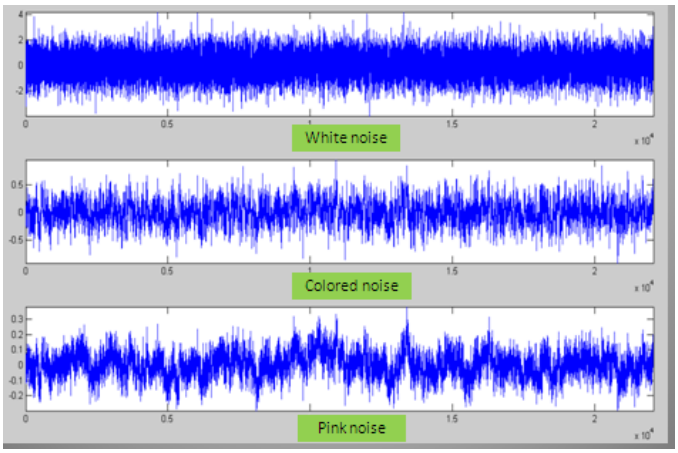

Figure2: Wave form of white, Colored and Pink noise

Colored noise – Colored noise is mixture of all type of available environmental noises like pink, red, gray etc. From figure 3, it can easily notice that white noise has constant power spectral density across the entire frequency spectrum (extending up to infinity).There is no correlation between the samples of a white noise process at different time instances i.e. the auto correlation or the auto covariance of white noise is zero for all lags except for lag L=0. But for colored noise power spectral density of the noise is not uniform across the entire frequency spectrum. There exist non-zero values for auto correlation or auto covariance at different time instances for the colored noise. The auto covariance is maximum for zero lag (L=0) and decreases gradually for increasing and decreasing values of lag (L).

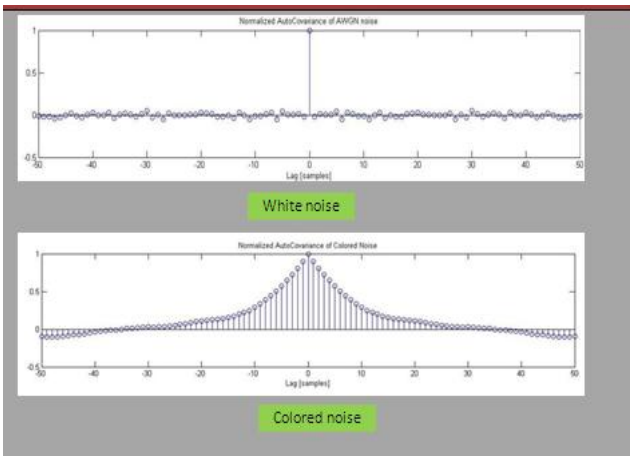

Figure 3: Normalized auto-covariance of white and colored noise

The frequency spectrum of white and colored noise is shown in figure 4. For white noise power spectrum is constant for all frequency band but for colored noise it is gradually decrease as frequency is increased and maximum for 0Hz.

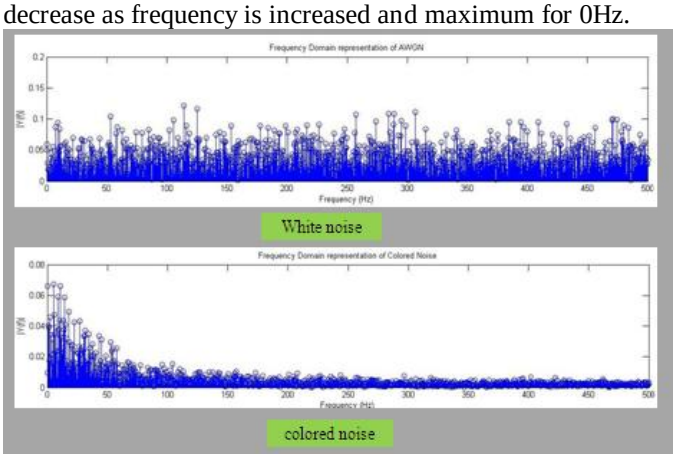

Figure 4: frequency spectrum of white and colored noise

Colored noise is generated by passing the white noise through a shaping filter. The shaping filter is a dynamic filter usually a low pass filter. The response of the colored noise can be varied by adjusting the parameters of the shaping filter.

```
whiteNoise=sqrt(variance)*randn(1,length(z))
;
[coloredNoise]=filter(1-a,[1
-a],whiteNoise);
Colorednoise_embedded_signal=z+coloredNoise
```

Where, 'z' is input audio clip, 'a' is filter parameter and variance for white noise is zero.

Pink noise – Pink noise is a specific type of colored noise in which spectrum is inversely proportional to frequency. From figure 5 it can easily notice that white noise has constant power spectrum irrespective of frequency whereas pink noise has power spectrum inversely proportional to frequency.Pink noise has uniform power density for a relative bandwidth (octave, decade). It has constant energy per constant percentage bandwidth. This equates to a -3dB/octave frequency response.

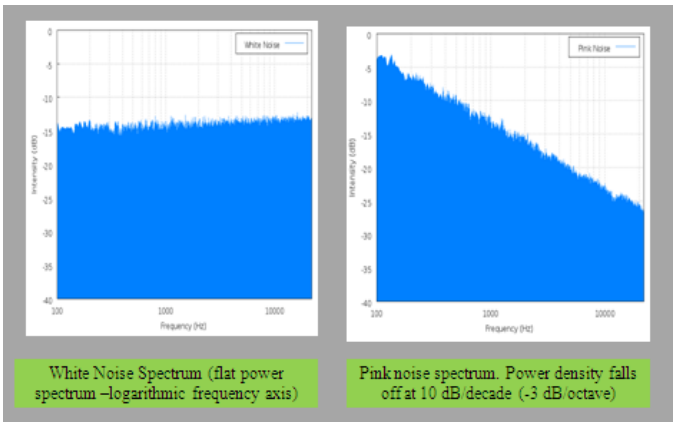

Figure5: Power Spectrum of White and Pink noise

Pink noise is generated by passing the white noise through a -3dB per octave filter. This filter has parameter 'A' and 'B',

those values are specific and generate pink noise with +/-0.05dB tolerance [17].

```
B = [0.049922035 -0.095993537 0.050612699
-0.004408786];
A = [1 -2.494956002 2.017265875 -0.522189400];
pinkNoise = filter(B,A,whiteNoise);
pinknoise_embeddedsignal= z+pinkNoise;
```

where 'z' is input audio signal.

Classifier – Support Vector Machine (SVM) is used as a classifier due to its reduced computational complexities and greater classification accuracies.

Support vector machines use supervised learning methods for classification. SVMs map input vectors to a higher dimensional space if the data is not linearly separable. Then a hyper plane is constructed to separate the input vectors. Two parallel hyper planes are constructed on each side of the hyper plane. The hyper plane that maximizes the distance between the two parallel hyper planes is found to be the solution. In linear non-separable cases, a kernel function is required to transform the original feature space to a higher dimensional space in an implicit way such that the mapped data is linearly separable. Common kernels include Polynomial, Gaussian Radial Basis Function, Sigmoid, etc. The choice of kernel is an important issue in SVM classification.

Optimal Hyper Plane for Linearly Separable Patterns: Consider the training sample $\{\mathbf{x}_i, d_i\}_{i=1}^{N}$ where $\mathbf{x}_i$ is the input pattern for the i^{th} example and d_i is the corresponding desired response (target output). Let us assume that the pattern (class) represented by the subset $d_i = +1$ and the subset $d_i = -1$ are linearly separable. The equation of a decision surface in the form of a hyper plane that does the separation is

$$\mathbf{w}^T\mathbf{x} + b = 0 \qquad (22)$$

Where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is an adjustable weight vector and b is the bias.

Thus,

$$\mathbf{w}^T\mathbf{x}_i + b > 0 \quad \text{for } d_i = +1 \quad (23)$$

$$\mathbf{w}^T\mathbf{x}_i + b < 0 \quad \text{for } d_i = -1 \quad (24)$$

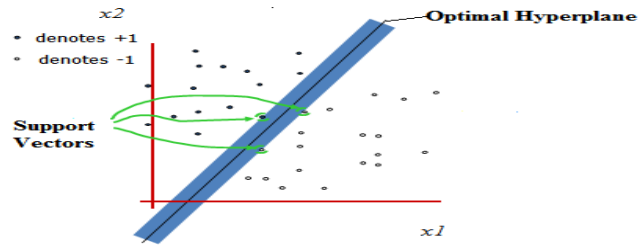

Figure 6: Geometric construction of optimum hyper plane for two dimensional input space

Hyper Plane for Non-Separable Patterns: To set the stage for formal treatment of non separable data points, a new set of non negative scalar variables $\{\xi_i\}_{i=1}^{N}$ is introduced in the definition of separating hyper plane. The ξ_i are called *slack variables*, they measure the deviation of data point from the ideal condition of pattern separability.

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{for } i = 1, 2, \ldots \ldots N \quad (25)$$

The support vectors are those particular data points that satisfy equation (25) precisely even if $\xi_i > 0$.The primal problem in case of non separable case may thus be formally defined as follows, where C is user specified positive parameter also called as SVM penalty/cost parameter.

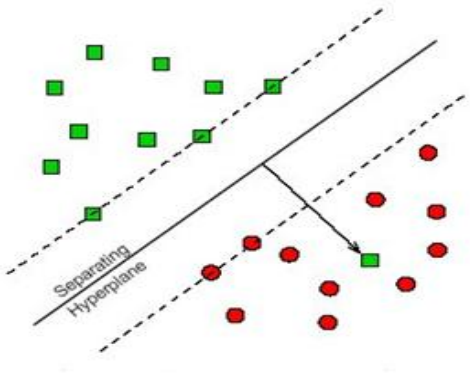

Figure 7: Non-Separable Training Sets introduces misclassification

And weight vector $\mathbf{w}$ minimizes the cost function

$$\varphi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \qquad (26)$$

The parameter C controls the tradeoff between complexity of the machine and the number of the non separable points; it may be therefore viewed as regularization parameter .The parameter C has to be selected by the user .This can be done in one of the two ways. The parameter C is determined experimentally via the standard use of training / (Validation) test set, which is crude form of resampling. It can be determined analytically.

For patterns that are not linearly separable, the following mathematical operations are performed in construction of SVM optimal hyper plane.

1. Non linear mapping of input vector into *high-dimensional feature space* that is hidden from both input and output. The low dimensional input data $\mathbf{x}$ is mapped into a high-dimensional feature space by mapping function $\boldsymbol{\varphi}(\mathbf{x})$.
2. Construction of optimum hyper plane for features obtained for separating the features discovered in step 1.

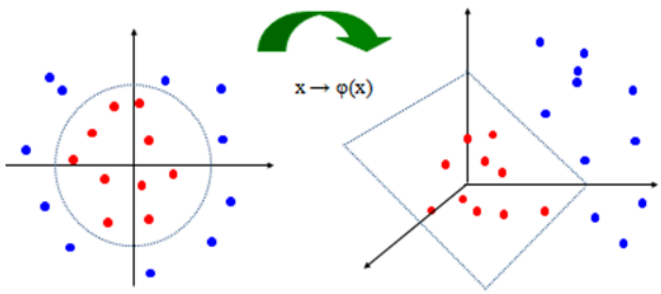

Figure8: Non linear mapping from the input space to higher dimension feature space

The separating hyper plane is now defined as a linear function of vectors drawn from the feature space rather than original input space.

Inner product kernel: The term $\varphi^T(\mathbf{x}_1)\varphi(\mathbf{x})$ represents the inner product of two vectors induced in the feature space by the input vector $\mathbf{x}$ and the input pattern $\mathbf{x}_1$ pertaining to the i^{th} example the inner product kernel denoted by $K(\mathbf{x}, \mathbf{x}_1)$ and defined by

$$K(\mathbf{x}, \mathbf{x}_1) = \varphi^T(\mathbf{x})\varphi(\mathbf{x}_1) \quad \text{for } i = 1,2,\ldots..N \quad (27)$$

In multiclass SVM, Radial Basis Function (RBF) kernel is used. As the RBF kernel nonlinearly maps samples into a higher dimensional space, so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear. In RBF kernel number of hyper parameter is less which influence model complexity. Its equation is given as

$$K(\mathbf{x}, \mathbf{x}_1) = e^{-\gamma\|\mathbf{x}-\mathbf{x}_1\|^2} \quad \text{for } i = 1,2,\ldots..N \quad (28)$$

The width γ is kernel parameter specified apriori by the user ($\gamma > 0$).

Cross-validation and Grid-search: There are two parameters while using RBF kernels: C and γ. It is not known beforehand which C and γ values are the best for one problem; consequently some kind of model selection (parameter search) must be done. The goal is to identify good (C, γ) so that the classifier can accurately predict unknown data (i.e., testing data). Note that it may not be useful to achieve high training accuracy (i.e., classifiers accurately predict training data whose class labels are indeed known). Therefore, a common way is to separate training data into two parts of which one is considered unknown in training the classifier. Then the prediction accuracy on this set can more precisely reflect the performance on classifying unknown data. An improved version of this procedure is cross-validation.

In v-fold cross-validation, the training set is first divided into v subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining v-1 subsets. Thus, each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified. The cross-validation procedure can prevent the over fitting problem. It is recommended to use "grid-search" on C and γ using cross-validation.

"Grid.py" is a program which performs a "grid-search" on C and γ using cross-validation. Basically pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked. An exponentially growing sequences of C and γ is a practical method to identify good parameters (for example, C = 2^-5, 2^-3, …., 2^15; and γ = 2^-15, 2^-13,…., 2^3).The grid-search is a straightforward approach to determine the optimum values of C and γ.

There are three reasons of preferring grid-search approach over other methods:

1. It does an exhaustive parameter search by approximations or heuristics.
2. The computational time to find good parameters by grid-search is comparable to that by advanced methods, since there are only two parameters to be determined.
3. Unlike the advanced iterative processes, grid-search can be easily parallelized because each (C, γ) is independent.

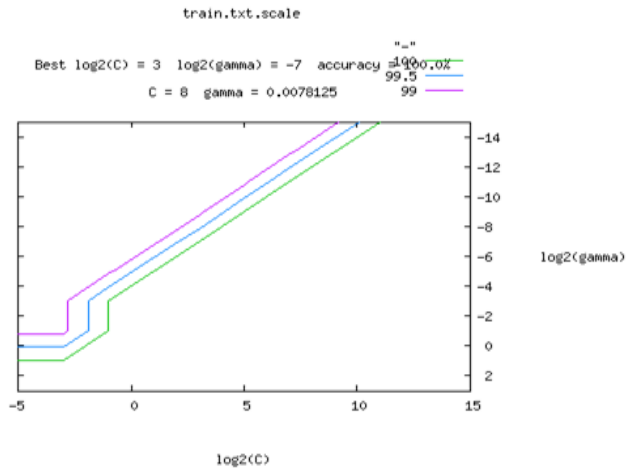

Figure 9: Plot of $\log_2$ C vs $\log_2$ gamma

IV. EXPERIMENTAL SETUP

For analysis purpose, four different type of database are used [DB1, DB2, DB3, DB4]. For DB1, standard database of Schierer [4] is used. For training purpose 40 files of Speech audio samples(S), 40 files of Music audio samples (M), 40 files of Speech mixed Music audio samples (SM) and 40 files of other type audio samples (O) are created. For testing purpose 20 files of each class are created. Total 240 files are used for training and testing. Each audio clip is one second long and in *.wav* format. This database is recorded at 22.050 kHz sampling frequency and at 16 bit, mono PCM. The fourth class of audio files includes all remaining types of audio signal for example silence, bird's sound, surrounding environmental noise etc.

For noise embedded input database, first white, colored and pink noise are generated in MATLAB environment and then added these noise with input audio database [DB1] to generate white noise embedded database [DB2], colored noise embedded database [DB3] and pink noise embedded database [DB4] respectively. These noisy databases also created in MATLAB environment at sampling frequency 22.050 kHz and at 16 bit, mono PCM. In these noisy databases clip size is one second and in *.wav* format.

For feature extraction purpose, first these clips are down-sampled to 8 kHz, in order to simulate with real time telecommunication. Frame size of 32millisecond with 10millisecond overlapping, is used to avoid data loss in framing process. Then features are extracted for each clip.

For classification purpose SVM with a radial basis function (RBF) kernel is used due to its outstanding performance [10, 11]. Specifically, a toolbox called LibSVM is utilized. A feature set of 40 down-sampled audio clips of each class are used to train SVMs. And remaining 20 audio clips are used for the tests. Each feature component is normalized to make their scale similar from 0 to 1.

V. RESULTS

When number of classes is increased for classification purpose then single feature or combination of two features is not sufficient to classify all classes, results are given in table 1. 75.63% is the highest classification accuracy, when single

feature (average Delta Cepstral Energy) is used for clean database DB1.This classification accuracy can be increased up to 93.75%, when combination of APD (Average Pitch Density) and MMCD (Mean Minimum Cepstral Distance) is used.

Table 1: SVM Classification Accuracy for Four Classes (Speech(S), Music(M), Speech containing Music(MS) and Other audio signal(O)) using single feature and combination of two features for clean audio input (DB1)

Feature	Accuracy
APD	50%
RTPD	49.38%
MMCD	65.63%
POLEF	55.63%
varZCR	56.87%
avgPSDev	62.5%
SF	45.63%
varDR	39.38%
avgMFCC	50%
avgDCE	75.63%
APD+RTPD	70%
APD+MMCD	93.75%

For combination of APD and MMCD, results of classification for noise embedded database (DB2, DB3, and DB4) are given in table 2. The classification accuracy for this combination is decreased when noise is embedded to input audio signal. For colore noise embedded input database DB3 and for pink noise embedded input database DB4, classification accuracy is decreased to 73.13% and 76.25% respectively.

Table 2: SVM Classification Accuracy for FourClasses (Speech,Music, Speech containing Music and Other audio signal) using APD+MMCD features

Database	Classification accuracy
DB1	93.75%
DB2	61.82%
DB3	73.13%
DB4	76.25%

When noise is embedded to audio input signal, then its feature's value is changed from its clean state value. Due to this change, feature's discriminative power is decreased. Classification accuracy is directly proportional to the distance between centroid and inversely proportional to overlapping of two class features. In figure 10, APD is used as classifying feature for speech and music classification. At clean input, the centroid distance between speech and music feature's values is 0.006 unit and at colored noise embedded input, the centroid distance between speech and music feature's values is 0.001 unit. The centroid distance is decreased by 6 times means overlapping of two classes is increased by 6 times. Classification accuracy at clean audio input is 92.5% but when colored noise is embedded to input audio signal, classification accuracy is decreased to 62.25% (figure 10).

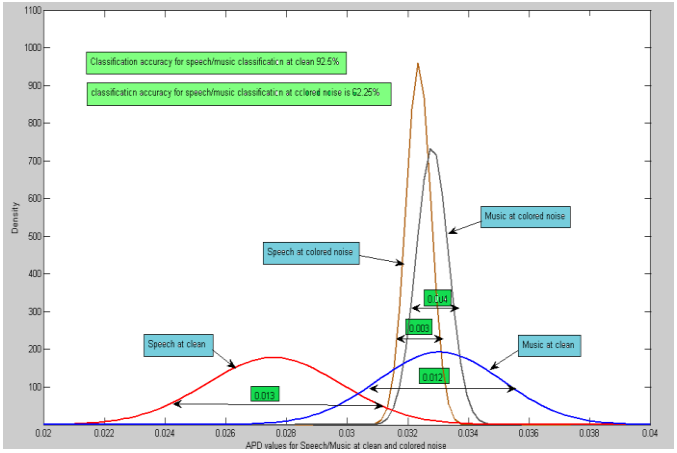

Figure 10: Effect of Noise on Speech/Music classification

Form table 3 results show that when all features are used for classification then accuracy is increased for both clean input database DB1 and noise embedded input databases (DB2, DB3, and DB4). For DB1 classification accuracy is 96.5%. For white noise embedded input database classification accuracy is increased to 98.13%. For colored and pink noise embedded input databases DB3 and DB4 classification accuracy are 88.13% and 95% respectively.

Table 3: SVM Classification Accuracy for Four Classes (Speech, Music, Speech containing Music and Other audio signal) using all features

Database	Classification accuracy
DB1	96.5%
DB2	98.13%
DB3	88.13%
DB4	95%

This 'four class classifier' takes 0.96 second CPU runtime on 4 Gb RAM. This computation time is less than input audio clip duration. So this classifier system can use for real time applications.

VI. CONCLUSION AND FUTURE SCOPE

This paper proposes a noise robust four class classification system with accuracy of 96.5% at clean audio input and 95%

at pink noise embedded audio input. As proposed system deals with real world noise (colored and pink noise) so it is applicable in real world application. Also its computation time is 0.96 second which is less in comparison of clip time (1 second). So this system is also useful in real time application. Future work will focus on optimizing number of features used for classification purpose.

REFERENCES

[1] Zhong-Hua Fu, Jhing-Fa Wang, Lei Xie,"Noise Robust feature for speech/music discrimination in real time application", IEEE international conference on multimedia, ICME2009,pp.574-577,2009.

[2] C. Panagiotakis, G. Tziritaz, "A Speech/Music Discriminator Based on RMS and Zero-Crossings," *IEEE Trans. on Multimedia*, vol.7 (1), pp. 155-166, 2005.

[3] M. J. Carey, E. S. Parris, and H. Lloyd-Thomas, "A comparison of features for speech, music discrimination," in Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, ICASSP-99, Phoenix, AZ, pp. 149-152, 1999.

[4] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multi feature speech/music discriminator," in Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, ICASSP-97, Munich, Germany, pp. 1-28, 1997.

[5] W.Q. Wang, W. Gao, D.W. Ying, "A Fast and Robust Speech /Music Discrimination Approach," *Information, Communications and Signal Processing, 2003*, vol.3, pp. 1325-1329, 2003.

[6] J. Wang, Q. Wu, H. Deng, and Q. Yan, "Real-Time Speech/Music Classification with a Hierarchical Oblique Decision Tree," *Int. Conf. Acoustic, Speech, and Signal Processing (ICASSP'08)*, Las Vegas, USA, pp. 2033-2036, 2008.

[7] M.Y. Choi, H.J. Song, and H. S. Kim, "Speech/Music Discrimination for Robust Speech Recognition in Robots," *Int. Symposium on Robot & Human Interactive Communication*, Jeju, Korea, pp.118-121, 2007.

[8] O. M. Mubarak, E. Ambikairajah, and J. Epps, "Novel Features for Effective Speech and Music Discrimination," *Int. Conf. Engineering of Intelligent Systems*, pp. 1-5, 2006.

[9] 8.Y. Zhu, Q. Sun, and S. Rahardja, "Detecting Musical Sounds in Broadcast Audio Based on Pitch Tuning Analysis," *Int. Conf. Multimedia and Expo (ICME'06)*, Toronto, Canada, pp. 13-16,2006.

[10] L. Chen, S. Gunduz, and M. T. Qzsu, "Mixed Type Audio Classification with Support Vector Machine," *Int. Conf. Multimedia and Expo (ICME'06)*, Toronto, Canada, pp. 781-784, 2006.

[11] Vladimir N. Vapnik, "An Overview of Statistical Learning Theory"," IEEE Trans. Neural Networks, vol. 10, No. 5, pp. 988-999, 1999.

[12] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery 2, pp. 121-167, 1998.

[13] www.wikipedia.org

[14] www.mathwork.com/matlabcentral

[15] www.edaboard.com

[16] http://gaussianwaves.blogspot.com/

[17] https://ccrma.stanford.edu/