A Binary Audio Fingerprint Based Music File Retrieval System

Sarvesh Rao[1], Rahul Kodag[1] and Chandrakant Gaikwad[1]

[1] Ramrao Adik Institute of Technology (D. Y. Patil Deemed to be University)
Email: sarveshgrao@gmail.com

Abstract.

The music industry around the globe has been witnessing huge, consistent growth over the past couple of decades after the internet boom. Music is a popular source of entertainment and has made a very large contribution to the growth of the global cinema industry. Each music record stored in the database of music companies has an ID or a combination of metadata that acts as a primary key for retrieving the user-specified music file. The music file retrieved from the database using metadata can get compromised due to the eavesdropping of hackers attacking the enterprise system. To prevent the sensitive metadata of music files from getting compromised, many music file retrieval systems make use of audio fingerprints (AFP) in place of metadata to identify the audio file being requested by the customer. The prime objective is to ensure that sensitive metadata is not compromised and even if AFPs get compromised due to hacker's eavesdropping, the feature values created from the read sample of audio are not sufficient information to get the complete audio file compromised. The proposed system uses a numeric pattern matching recognition system with decision boundaries for querying the database to find the train sample having an AFP match to the query sample. The AFPs of the train samples are stored in the MongoDB ATLAS database. The experiment results show that the proposed AFP system has more accurate recognition compared to two existing systems.

Keywords. Audio Fingerprint, eavesdropping, metadata, database, enterprise, MongoDB ATLAS, decision boundaries.

1. INTRODUCTION

The music files are audio recordings with specific patterns of beats that do not possess phonemes or words [1]. Hence, they do not possess any lexical or linguistic features that are conducive to identifying music files efficiently [1]. The majority of music produced is registered for copyright protection, then organized and subsequently categorized into numerous hierarchical levels, starting from groups of notes, bars, and phrases to sections and then movements [2].The Music Data Storage Process (MDSP) is a three-step process [3]. The initial step of the procedure is generating the feature vectors to create an AFP using

audio data of the music file whose metadata is to be stored in a database such that the size of the AFP is smaller than the audio data [3]. The AFP has to be converted to a database (DB) compatible AFP format [2]. The AFP will act as unique index for the set of metadata of that music file [2]. The metadata will be stored along with AFP separately in a DB table. The AFP will be used as primary key for accessing the set of metadata of that music file [2]. The third step of the process is storing AFP with the audio content of music file in a DB-compatible format in a separate DB table with the AFP set as the primary key [2, 3]. The Music Data Fetch Process (MDFP) is also a three-step process [3]. The initial step is the creation of the AFP of the query sample. The second step is the AFP search algorithm execution to find the file with an AFP match [3]. The third step is fetching of specified metadata associated with that file. The AFP search technique may be audio content-dependent. A very general example is that of the music recommendation algorithms in music streaming services that use AFPs for querying the DBs when the user selects an option from the recommendation list [3]. The proposed music file retrieval (MFR) system comprises of both MDSP and MDFP intended for retrieving the data related to the music file.

1.1. Audio Fingerprint

AFP storage is a method that uses the entire audio content under observation to make a smaller-sized vector which is mapped one-to-one to the actual filename in the database [4, 5]. The AFPs can serve the purpose of checking the number of broadcasts of specific music made in a day or for a quick search of the metadata of the music composed and stored in a very large database [4, 6]. They can also be used to identify plagiarized content in a music or a song to protect digital rights [6]. The purpose of generating fingerprints is to decrease the amount of data needed for representing complete audio in a way that it can be mapped in the audio database [7]. The prerequisites for any AFP based system are efficient distinction of each AFP over a large set of AFPs searched in the DB [5]. The generated AFP should remain significantly robust to acoustic distortions [6]. An AFP generating algorithm is considered time-efficient if the algorithm has a very low computation time [5]. Such AFP algorithms are well-suited to process very large-sized databases [5].As the percentage of accurate detection of the actual music file in the audio DB has the highest priority for every AFP based audio system, the importance of true positives precedes that of true negatives [5, 8, 9].The organization of the paper is summarized here. Section I gives a brief introduction to the characteristics of music files and music information retrieval challenges, along with a brief explanation of processes in AFP based system. Section II gives a literature survey of a few of the existing AFP based music systems that are robust to environmental disturbances. The complete functioning of our proposed system is provided in Section III. The Section IV has all the details of the test results of our proposed system, followed by conclusions drawn from the observations in Section V.

2. LITERATURE SURVEY

The majority of the research works proposed shortly after the first proposed AFP based Music systems focused deeply on making the symbolic representations of the music segments with help of instrument digital interface [3]. There was much signal processing research in the very early years of the 2000s aimed at applying domain transformation directly to the read music data without audio pre-processing [3].Cha Guang-Ho analyzed

Miller et al. proposed system [10]. Miller et al. has used a 256-ary tree to guide the AFP search in an AFP database [10]. As each node has 32-bit, the tree has an overall 256 X 32 = 8192 bits in the AFP frame [10]. Each 8192-bit AFP frame is represented as 1KB of data. Every successive byte in the AFP is a factor that decides the path to descend to which node of the tree out of the 256 child nodes [10]. When the bits are traced from the root node to a specific child node, a unique fingerprint of an audio file will be obtained for their proposed system [10].George and Ashok proposed an AFP algorithm robust to many distortions like the time-stretching effects on audio [11]. Their AFP based system operates by finding the three largest frequencies. The bins are chosen for every overlapping time frame within the spectra of the signal. The authors have encoded the bins to get 30-bit binary codes. Every bin within the 30 bits of binary code gets 10 encoded bits. Hence, the data can be displayed as a binary sequence of Ci codes. The 'i' suffix used in Ci is to indicate the instantaneous bit in the sequence. The AFP search algorithm deployed in their system extracts every feature code of the train sample which found a match with the test sample codes codes [11]. Chang et al., 2021 has used an AFP which converts segmented features to the L2 normalize vector. The inner-product is used to identify matching segment. Their system has a pre-processor followed by neural networks [5].

3. PROPOSED METHODOLOGY

The proposed AFP based MFR system has two processes which have the block diagram as shown in figure 1 and figure 2. The MFR system execution flow has two processes, namely, the Batch Creation process and the Query Fetch process.

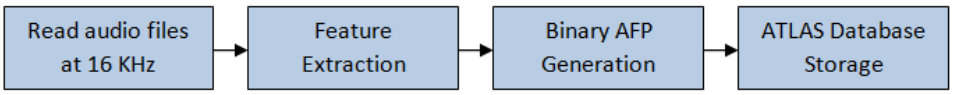

Fig. 1. Batch Entry creation process

3.1. Batch Creation process

The directory path containing the audio files of the training dataset is passed as input to the algorithm of the process. Each audio file in that directory is processed serially in the batch creation process. The AFPs of each train sample are generated in a serial manner and stored with metadata in a collection of MongoDB database. A file-Id value is one-to-one mapped to the AFP stored in ATLAS collection. The audio of the music file from which the AFP is generated and gets stored in a separate collection present in the same database with the file-Id mapped to that corresponding music file.

3.2. Query Fetch process

The directory path of a specific test sample is passed as a query input to the process algorithm. The audio files of training set and test set are referred to as ‘train audio files’ and ‘test audio files’ respectively in the explanation ahead. The test audio files have a length ranging from 1 to 6 seconds. The algorithm of the process reads the audio content at a fixed sampling rate of 16 kHz and extracts the features which are used to create AFP corresponding to that query music file.

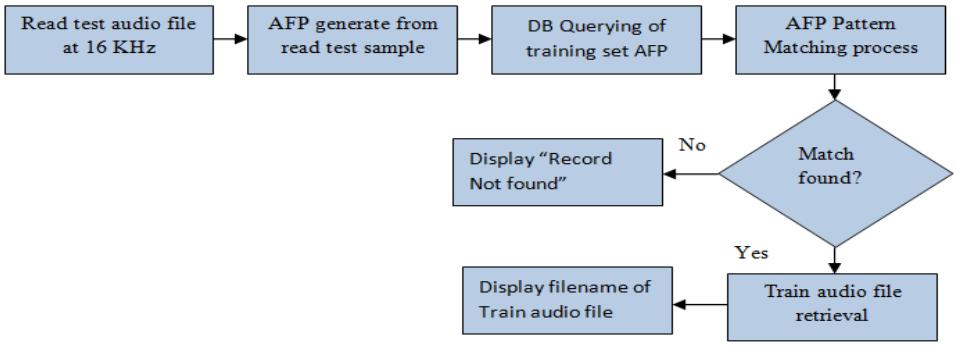

Fig. 2. Query Fetch process

All the binary data of AFPs within the collection of the database is fetched by the algorithm which is converted to their numeric equivalent using binary-to-ASCII conversion. The process algorithm transforms the binary data of trained sample AFP serially to their numeric equivalent values. The numeric equivalents of entire binary data are stored in a two-dimension array with each frame of the array containing 16 samples which we will refer to as the "rms samples". Every frame of the train rms sample AFP gets searched for an absolute match of the test sample being iterated.If the algorithm finds a matching rms sample between test and train samples of the AFP being compared, then the counter is incremented by one. When every rms sample of the test AFP has been matched with the train sample AFP under observation, then the Query Fetch process calculates the percentage of rms samples with matches. If the percentage of matching samples is greater than 90%, the test sample is classified as a match for the training sample in the current iteration. The hop size of the sliding window is set to 100 rms samples. The iteration count with AFP pattern match is multiplied by eight to find the second of the train sample for which match was found. The algorithm will fetch the file-Id value of the corresponding music file from the ATLAS database. The music file audio data will be fetched from the ATLAS collection using the file-Id and will be stored in the project folder.

3.3. Audio Fingerprint Extraction

The entire sequence of steps for extracting the feature based on audio content is provided in the figure 3. The audio files used in the training and testing dataset are in ".wav" format. The audio content of the music file is read into an array using a 16 kHz sampling rate [12]. The audio array is trimmed from the start such that the count of audio samples is an integral multiple of 16000. The root mean square(rms) of 20 successive samples is taken to reduce the storage cost of AFP by a factor of 20 and to normalize the disturbances, if present. The audio data sampled at a frequency of 8 kHz is sufficient to make an audio fingerprint as specified by Sonnleitner and Widmer in their Quad based AFP system. It would reduce the storage requirement of the proposed system with a minor decrease in the accuracy [12]. Hence, there will be a trade-off between storage requirements and the accuracy of the system.

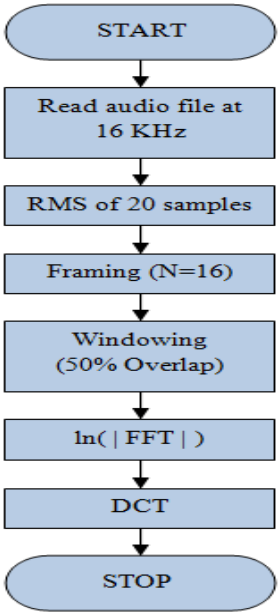

Fig. 3. Feature Extraction Flowchart

As we are taking the rms of 20 consecutive samples in our proposed system, the size of the AFP of the proposed system would be smaller than many existing systems and so, the sampling rate is set at 16 kHz. The rms audio array has 'N' no. of rms samples which is fragmented into 'M' rms frames of size '16'. Each rms frame corresponds to a 20 ms time period.

$$N\ X\ 20\ =\ T \tag{1}$$

$$M\ X\ 16\ =\ N \tag{2}$$

Here,
M - count of rms frames corresponding to 20ms,
N - Overall count of RMS sample,
T - Total samples stored in the array from the read audio file.

The frame size is 16 rms samples for a sampling rate of 16 kHz. Hence, the Hanning window used in the proposed system has a size (M) of 16 with 50% overlap. The equation of Hanning window for M=16 as per Podder et al.(2014) would be,

$$H(n)\ =\ 0.5\ -\ 0.5\cos\left(\frac{2n\pi}{15}\right) \tag{3}$$

The zeroth sample of the Hanning window is replaced by 0.01 to prevent data loss of each zeroth sample of the output frame of the windowing operation.

$$H(0)\ =\ 0.01 \tag{4}$$

Each rms sample of the windowing operation output A(m) will undergo a 1-D FFT magnitude transformation [13, 14]. The resultant array will contain only real parts of the

A(m) in the frequency domain. The FFT magnitude of RMS sample $F_r(k)$ in close relation to (7) of [13] can be expressed as,

$$F_r(k) = \Delta(t) \left| \sum_{m=0}^{15} A(m) e^{\frac{-j(2mk\pi)}{16}} \right| \quad \dots \quad (k = 0,1\dots,15) \tag{5}$$

As the sampling rate is a fixed value, the value of 'T' and 'N' will remain constant. Hence, Δ(t) is a ratio of constant given in accordance with [13] as $\Delta(t) = \frac{20\ X\ 10^{-3}}{16}$ which gives the value $\Delta(t) = 1.25\ X\ 10^{-3}$. The frequency magnitude of the rms sample is then subjected to compression through natural logarithmic transformation followed by Discrete Cosine transform (DCT). Let, $L_f(k)$ be the natural log transformation values of each Frequency sample written as,

$$L_f(k) = ln(F_p(k) \tag{6}$$

In Let, $C_f(\bar{k})$ be the output array of the DCT signals. The FLCC co-efficient at the output using DCT of [15] is calculated as,

$$C_f(\bar{k}) = \frac{2}{16} \sum_{k=0}^{15} L_f(k) \cos \frac{(2k+1)\bar{k}\pi}{32} \quad \dots \quad (\bar{k} = 1,2,\dots,15) \tag{7}$$

The equation (8) gives the value of zeroth sample of DCT [15].

$$C_f(0) = \frac{\sqrt{2}}{16} \sum_{k=0}^{15} L_f(k) \tag{8}$$

The zeroth sample of DCT output is replaced with a unit value. The output of the DCT operation is an array of numbers, which can't be stored in the ATLAS database. The compatible data formats for storing data chunks in ATLAS are binary or string formats. Hence, the DCT output array is initially subjected to ASCII-to-binary conversion to convert the numeric array to binary data.The binary data is then transmitted to the ATLAS server over an SSL secured channel during the batch creation process. The binary data of AFP when received from a database is subjected to binary-to-ASCII conversion to get the original numeric FLCC array during the query fetch process. The AFP pattern matching algorithm takes the AFP numeric array of test and train sample as the input. The list of recognized audio files with the exact seconds of the train sample where the AFP pattern match has been found is passed at the output.

4. RESULTS AND DISCUSSION

The experimental plots of initial frame for each step of the FLCC feature extraction procedure are shown from figure 4 to figure 7. From the comparison of the samples of FFT magnitude in figure 6 and the FLCC samples magnitude in figure 7, it is evident that the FLCC samples are compressed significantly which reduces the AFP storage cost of the proposed system and gives the cepstral co-efficient.

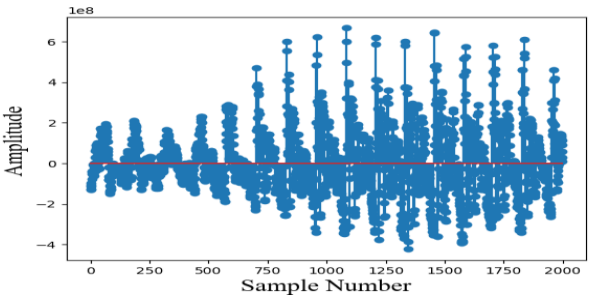

Fig.4. Stem plot of test audio file

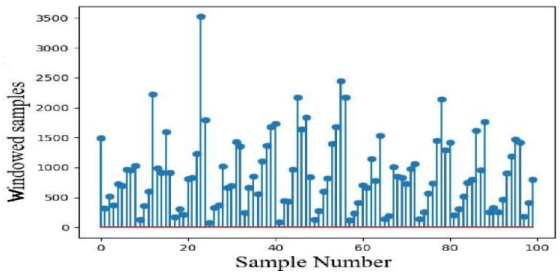

Fig. 5. Stem plot for Windowing operation output

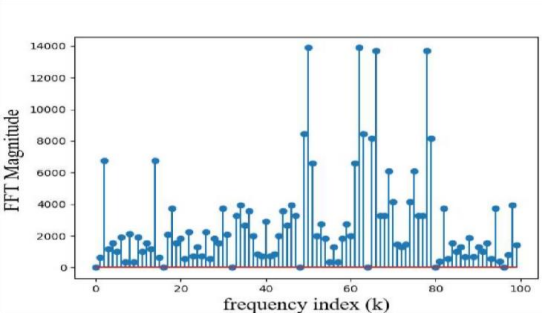

Fig.6. FFT magnitude output stem plot

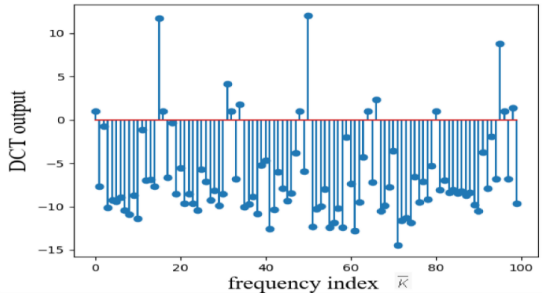

Fig.7. FLCC Feature stem plot

4.1. Test Result Analysis

The accuracy of our proposed system was compared with the Panako and RAFS AFP system on the subset of Voxceleb1 dataset [16]. The dataset was further separated into three sub-sets in the proportion of 60:20:20 for training, validation and testing of Voxceleb1 dataset samples [9]. The training dataset possesses 2,799 clean noise-free audio files while the validation and test sets each have 1,812 and 1,184 clean audio clips with length ranging from 1 to 6 second [9].Panako AFP system uses a constellation algorithm in which a sample is initially pre-processed before generating a constant-Q spectrogram which is robust to time-frequency domain disturbance [9]. The local maximas are combined to form the AFP of their system [9]. A hash value is calculated using AFP for a single frame. Panako uses a hash matching algorithm which queries the database for matching hashes and returns the train samples with exact matches [9, 16].The RAFS AFP system uses 32-bit frames equivalent to 11.6 ms for a frame. The AFP has a consecutive 256 frames, which equals to three second in time domain [17]. The algorithm initially segments the entire audio data into overlapping frames [17]. The overlapping frames of 37 ms are windowed with a Hanning window with overlapping of 31/32. This overlap strategy causes one sub-AFP to be extracted for every 11.6ms and uses bit error rate pattern matching [9, 17].The name of each test audio file along with exact second value was noted in advance before the start of the testing process. If the recognized test file during the testing process has the same name as the actual file and the predicted second same as the actual second, then that test sample is classified as a ”true positive”. If the list of the recognized file does not have the name of the actual file, then it is classified as a ”false negative” sample. If the proposed system recognizes the name of the actual file but the predicted second is not same as the exact second of the actual file, then it is marked as a ”false positive” sample. If the test audio file does not occur in any train audio file sample and there is no file recognized from the fetched data due to AFP mismatches then the sample is noted as ”True negative” else as a ”False positive” sample.Table I shows the comparison of the granular identification accuracy of our proposed system with that of the Panako and RAFS AFP system using Suarez recorded readings [9]. The Panako and RAFS system were tested for accuracy using test audio files ranging from 1-6 seconds.

TABLE I. ACCURACY COMPARISON FOR DIFFERENT AUDIO LENGTH

Test sample length	**Proposed**	**Panako [9]**	**RAFS [9]**
1 second	64.5%	0%	0%
2 second	85.35%	15.29%	23.07%
3 second	89.75%	60.70%	67.02%
4 second	94.3%	91.22%	79.49%
5 second	99.6%	97.61%	85.24%
6 second	99.8%	98.03%	82.74%

Test sample length	Proposed	Panako [9]	RAFS [9]
Average	88.883%	60.475%	56.26%

The testing process is carried out to identify granular identification accuracy of proposed system compared to the Panako and RAFS systems. The proposed system was able to recognize test samples with length of 1 second.

5. CONCLUSION

A frequency-domain feature named as Frequency Log Cepstral Coefficient (FLCC) has been proposed in this paper to make a binary AFP of our proposed system. The test results of the proposed system gave better granular identification accuracy than the Panako and RAFS AFP systems on the subset of Voxceleb1 dataset.

6. REFERENCES

[1] Wahyuni, R. and Budi, I. (2018)"Combining linguistic, semantic and lexicon feature for emoji classification in twitter dataset"in Procedia Computer Science, 135, 194-201, DOI:10.1016/j.procs.2018.08.166.

[2] Lerch, A.(2021)"Audio content analysis"in arXiv preprint arXiv:2101.00132,

[3] Kuldeep, G., and Yang-Sae, M.(2018)A comparative analysis of music similarity measures in music information retrieval systems. Journal of Information Processing Systems, 14, vol. 1, 32-55, DOI: 10.3745/JIPS.04.0054.

[4] Jiang, T., Kang X., Jing L., Rihui W., Xin L. and Feng D. (2013)"A large scale audio fingerprinting system"in Pacific-Rim Conference on Multimedia,. Springer, Cham, 866-875. DOI: 10.1007/978-3-319-03731-8_81.

[5] Chang, S., Donmoon L., Jeongsoo P., Hyungui L.,Kyogu L., Karam K. and Yoonchang H. (2021)"Neural audio fingerprint for high-specific audio retrieval based on contrastive learning"inIEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 3025-3029.

[6] J.Williams, Dominic, Akash P., and Jesse S. (2017) Efficient music identification using ORB descriptors of the spectrogram image. EURASIP Journal on Audio, Speech, and Music Processing, Vol. 1, 1-17. DOI: 10.1186/s13636-017-0114-4.

[7] Struharová, N. (2021) Performance of the Dejavu audio fingerprinting framework in music identification in movies. DOI: 10.1145/3380828.

[8] Bang, G., Xiaoou C. and Deshun Y. (2014) "Efficient music identification by utilizing space-saving audio fingerprinting system" in 2014 IEEE International Conference on Multimedia and Expo (ICME), 1-6.

[9] Báez Suárez, A. (2020) Unsupervised Deep Learning Recurrent Model for Audio Fingerprinting.PhD thesis, Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM), Monterrey, México. DOI: 10.1145/3380828.

[10] Cha, Guang-Ho. (2011) "An effective and efficient indexing scheme for audio fingerprinting"in IEEE2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering, 48-52. DOI:10.1109/MUE.2011.20.

[11] George, J. and Ashok, J. (2015) "Scalable and robust audio fingerprinting method tolerable to time stretching." in 2015 IEEE International Conference on Digital Signal Processing (DSP), 436-440. DOI:10.1109/ICDSP.2015.7251909.

[12] Sonnleitner R.andWidmer,G. (March 2016)"Robust Quad-Based Audio Fingerprinting," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 24, no. 3, 409-421, DOI: 10.1109/TASLP.2015.2509248.

[13] Brigham E. O. and Morrow R. E. (Dec. 1967) "The fast Fourier transform" in IEEE Spectrum, vol. 4, 12, 63-70.

[14] Zhu, X., Gerald, T. B., and Lonce, L. W. (2007) "Real-time signal estimation from modified short-time Fourier transform magnitude spectra" in IEEE Transactions on Audio, Speech, and Language Processing 15, no.5, 1645-1653, DOI:10.1109/TASL.2007.899236.

[15] Ahmed, N., Natarajan, T. and Kamisetty, R. R. (1974) "Discrete cosine transform" in IEEE transactions on Computers 100, no.1, 90-93. Six, J. and Marc, L. (2014) "Panako: a scalable acoustic fingerprinting system handling time-scale and pitch modification" in 15th International Society for Music Information Retrieval Conference (ISMIR-2014).

[16] Cano, P., Eloi, B., Ton, K. and Jaap, H. (2002) "A review of algorithms for audio fingerprinting" in 2002 IEEE Workshop on Multimedia Signal Processing, 169-173, DOI:10.1109/MMSP.2002.1203274.

[17] Podder, P., Tanvir Z. K., Mamdudul, H. K., and Rahman M. M., (2014) Comparative performance analysis of hamming, hanning and blackman window. International Journal of Computer Applications 96, no.18.

Biographies

Mr. Sarvesh Rao received his B.E degree in electronics engineering from University of Mumbai in 2017 and currently persuing M.Tech degree from Ramrao Adik Institute of Technology, DY Patil Deemed to be University, Nerul, Navi-Mumbai.

Mr. Rahul Kodag is Assistant Professor at Ramrao Adik Institute of Technology, DY Patil Deemed to be University, Nerul, Navi-Mumbai. His areas of interest are Machine learning, Audio and Music Signal Processing. DY Patil Deemed to be University, Nerul, Navi-Mumbai.

Dr. Chandrakant Gaikwad is Professor and Head of Electronics and Telecommunication Engineering department at Ramrao Adik Institute of Technology, DY Patil Deemed to be University, Nerul, Navi-Mumbai. He has over 24 years of teaching and research experience. Before joining RAIT, he did his Ph.D. from the Department of Electrical Engineering, Indian Institute of Technology Kanpur, U.P. , India. He has received his MTech degree in the Electrical Engineering department of the IIT Kanpur. His areas of interest are Time-Frequency Analysis, Singers Voice Analysis, Speech, Image and Video Processing.