CLASSIFICATION OF SOUND SIGNALS VIA COMPUTATIONALLY EFFICIENT SUPERVISED AND UNSUPERVISED LEARNING SCHEMES

by

Fatemeh Saki

APPROVED BY SUPERVISORY COMMITTEE:

Dr. Nasser Kehtarnavaz, Chair

Dr. Carlos Busso

Dr. Issa M. S. Panahi

Dr. Mehrdad Nourani

To my beloved family

CLASSIFICATION OF SOUND SIGNALS VIA COMPUTATIONALLY EFFICIENT SUPERVISED AND UNSUPERVISED LEARNING SCHEMES

by

FATEMEH SAKI, BS, MS

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN

ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT DALLAS

May 2017

ACKNOWLEDGMENTS

First of all, I would like to express my deepest appreciation and thanks to my supervisor, Professor Nasser Kehtarnavaz, for his great mentorship, fundamental guidance, and constant support. I went through a very difficult phase in my life during my PhD, but his support and confidence in me made it possible for me to move on and achieve what I have today. Professor Kehtarnavaz has a great research vision and he taught me to look at things in a more complete way. In addition to our academic collaboration, I greatly value the personal support that Professor Kehtarnavaz provided me over these years. Without his guidance and assistance, this research and the results achieved would not have been possible. I quite simply cannot imagine a better advisor.

Here, I gratefully acknowledge the members of my PhD committee for their time and valuable feedback. I am extremely grateful to Professor Carlos Busso, Professor Issa Panahi and Professor Mehrdad Nourani for their support, in the form of their scientific advice, insightful discussions and suggestions. I am also thankful to them for their constructive feedback during my PhD and providing concise advice for my future career. I would also like to thank my friends and labmates in SIP lab, Chih-Hsiang Chang, Chen Chen, Reza Pourreza, Kui Liu, and Abhishek Sehgal for all the great times that we have shared. I am particularly thankful to Taher Mirzahasnloo for his infinite mentorship, help and advice. I am so grateful for my wonderful friendship with Neha Dawar that made the last two years of my PhD unforgettable. Neha is one of the kindest people I have ever seen in my life. I would like to express my extreme gratitude to my dearest friend and mentor Dr. Sarah Ostadabbas for her infinite and unconditional support, advice and mentorship throughout all these years. Sarah taught me to stay motivated, focused and passionate during the difficulties to achieve my goals.

I am deeply thankful to my family for their love, support, and sacrifices. My mom provided her unconditional love and care to me, and I would not have made it this far without her constant encouragement, support and sacrifices. I am also blessed to have my eldest brother, Reza, who always has been a father, a brother and a great friend to me. He is the reason behind all my confidence and hope. My parents, my brothers, my sisters and my nieces and nephews are the love of my life and the reason for my everyday breath.

March 2017

CLASSIFICATION OF SOUND SIGNALS VIA COMPUTATIONALLY EFFICIENT SUPERVISED AND UNSUPERVISED LEARNING SCHEMES

Fatemeh Saki, PhD
The University of Texas at Dallas, 2017

Supervising Professor: Dr. Nasser Kehtarnavaz

Classification of sound signals is increasingly being used in hearing improvement devices such as hearing aids, cochlear implants, and smart headphones. Classification of sound signals enables adapting the speech enhancement/noise reduction algorithms in such devices to different sound environments in an automatic manner. The thrust of this dissertation research has been on the development of sound signal classification approaches that are computationally efficient, thus enabling their real-time deployment in hearing improvement devices. Both supervised and unsupervised learning schemes have been examined. For the supervised case, effective and computationally efficient features and classifiers have been developed. For the unsupervised case, an online clustering algorithm has been developed without knowing the number of clusters. Experimental results obtained indicate that the developed classification approaches outperform the existing sound classification approaches in terms of both classification rates and computational efficiency.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Environmental sound signals classification constitutes a major component in machine listening systems. A major goal of machine listening systems is to achieve human-like auditory classification of sound signals, in particular speech, music, and different types of noise. A sound classification component often appears with other components or subsystems in the signal processing pipeline of hearing improvement devices. For example, it is well established that the hearing sensation of hearing aid users degrades considerably in noisy environments. Thus, there have been attempts at developing speech enhancement/noise reduction algorithms that are adaptive to different sound environments. Classification of sound signals enables adapting the speech enhancement/noise reduction algorithm in such devices to different sound environments in an automatic manner.

Many sound signal classifiers have appeared in the literature. However, one aspect that has not been adequately addressed in the literature is the real-time computational efficiency aspect. Hence, the thrust of this dissertation research is placed on developing a real-time as well as a reliable classifier in real-world settings. In addition, a limitation of the existing sound classifiers is the supervised nature of the classification, meaning that a training procedure is first conducted based on a collected dataset. A challenge that arises as a result is how many sound classes should one consider. One possible solution would be to consider as many classes as possible. This would require extensive data collection and training. Furthermore, the sound environments encountered may vary from user to user. A more effective solution would be to make a hearing device user-specific, in other words, by allowing the device to learn the sound environments on its own for a

specific user. Towards this objective, an on-the-fly clustering algorithm, which is capable of defining different clusters in real-time with no knowledge of the number of clusters is developed in this dissertation. This algorithm does not require any data collection for training. Clusters or sound classes get generated in real-time and in an on-the-fly manner.

The contributions made in this dissertation consist of seven papers, five of which have already been published and two of which are under review at the time of this writing. These papers appear as the seven chapters of this dissertation. Each chapter provides an abstract of the contribution made, an introduction and literature review, the developed methodology, the results obtained together with discussion, and the conclusion associated with the corresponding chapter.

Chapter 2 covers the development of computationally efficient features, named band-periodicity and band-entropy, for noise signals classification in cochlear implants. The experimental results show that the devised features along with the use of a Random Forest (RF) classifier outperform the state-of-the-art classifiers in terms of both classification rate and computational efficiency. A real-time implementation of this classification approach on Android smartphones is then covered in Chapter 3.

In Chapter 4, a computationally efficient Voice Activity Detector (VAD) is developed to enable automatic switching between noise classification and speech enhancement for hearing aid applications. The developed VAD consists of a computationally efficient feature extractor and an RF classifier. This switching approach is compared to two popular VADs. The results obtained indicate the introduced approach outperforms these existing approaches in terms of both detection rate and processing time.

A hierarchical classification algorithm which is designed to classify sound signals in a computationally efficient manner is described in Chapter 5. The developed classification hierarchy consists of three levels to classify speech, music and different noise types. A distinguishing attribute of this hierarchical approach is that effective features are computed as needed at different levels of the hierarchy making the classification process computationally efficient. The results obtained show higher classification rates as well as higher computational efficiency of this hierarchical classification approach compared to the conventional one-step classification approach.

As an application of the supervised classification approaches in the previous chapters, Chapter 6 covers a multi-band environment-adaptive approach to noise suppression for cochlear implants to improve the speech processing pipeline in cochlear implants.

Chapter 7 covers the unsupervised approach to sound signals classification by introducing an Online Frame-based Clustering algorithm named OFC, for applications in which data are received in a streaming manner as time passes by, with the number of clusters being unknown. This algorithm consists of a number of steps including density-based outlier removal, new cluster generation, and cluster updating. This algorithm is designed for applications when data samples are received in an online manner in frames. Experiments involving four synthetic and two real datasets are conducted to show the performance of the introduced clustering algorithm in terms of cluster purity and normalized mutual information. Comparison results with similar clustering algorithms are also reported exhibiting the effectiveness of the introduced online frame-based clustering algorithm.

In Chapter 8, a modification is made to the OFC algorithm by adding a feature extraction, a smoothing step and a fading step to perform real-time unsupervised classification of environmental noise signals without knowing the number of noise classes or clusters. The results obtained for actual noise signals exhibit the effectiveness of the introduced unsupervised classification in terms of both classification rate and computational efficiency.

CHAPTER 2

BACKGROUND NOISE CLASSIFICATION USING RANDOM FOREST TREE CLASSIFIER FOR COCHLEAR IMPLANT APPLICATION *

Authors- Fatemeh Saki, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents improvements made to the previously developed noise classification path of the environment-adaptive cochlear implant speech processing pipeline. These improvements consist of the utilization of subband noise features together with a random forest tree classifier. Three commonly encountered noise environments of babble, street, and machinery are considered. The results using actual noise signals indicate that this classification method provides 10% improvement in the overall classification rate compared to the previously developed classification while maintaining the real-time implementation aspect of the entire speech processing pipeline.

2.1 INTRODUCTION

Since the introduction of cochlear implants (CIs) that has brought hearing sensation to profoundly deaf people, many advances have been made to improve their capabilities. It is well known that the hearing sensation of patients wearing cochlear implants degrades considerably in noisy background environments. In [1-5], we developed a speech processing pipeline that performs automatic classification of different background noises for the purpose of tuning the speech enhancement component of CIs according to the classified noise type.

Many studies have been reported on noise classification consisting of the two major components of feature extraction and classification. Table 2.1 provides a representative listing of recent studies where different features and classifiers have been used to achieve background noise classification.

Table 2.1. Previous works on noise classification

References	Year	Features	Classifier
Khunarsal et al. [6]	2013	spectrogram, LPC and MP	NN (neural network) classifier
Chu et al. [7]	2012	MFCC and matching pursuit (MP)	Deep belief network classifier
Li et al. [8]	2010	MFCC, rhythm pattern (RP) and matching pursuit (MP)	SVM
Lozano et al. [9]	2010	MFCC, zero crossing rate, centroid and roll-off point with multi-resolution window size	GMM
Chu et al. [10]	2009	matching pursuit (MP) and MFCC	GMM
Byeong et al. [11]	2009	traditional features (TFs), change detection features (CDFs), and acoustic texture features (ATFs)	SVM
Ntalampiras et al. [12]	2008	MFCC and MPEG-7 features	Hidden Markov Model (HMM)
Kraetzer et al. [13]	2007	63 statistical features computed by AAST	Bayes classifier
Eronen et al. [14]	2006	zero-crossing rate (ZCR), MFCCs, delta-MFCC, band energy, spectral roll-off, linear prediction coefficients (LPCs) and linear prediction cepstral coefficient	k-NN (k nearest neighbor) and one-state HMM
Wang et al. [15]	2006	spectral centroid, spectral spread, and spectral flatness	Hybrid SVM and k-NN classifier
Malkin et al. [16]	2005	64 dimenstional MFCC and spectral centriod	Auto-encoder NN and GMM
Toyoda et al. [17]	2004	instantaneous spectrum at power peak and the power pattern in the time domain	NN

The classification rates for the listed studies varied from 80% to 95% using different datasets. One issue that has not been specifically addressed in these studies is the computational complexity or the real-time implementation aspect for actual deployment on a CI processing platform. In [1], it was shown that the use of mel-frequency cepstral coefficients (MFCC) features together with a Gaussian mixture model (GMM) classifier provided a balance between noise classification rate and real-time implementation on the PDA platform approved by the US Food and Drug Administration (FDA) for cochlear implant studies [3].

This work involves improving the results reported in [1] in two ways: First, in place of a GMM classifier, a tree classifier is used in order to improve the overall classification rate. This is the first time tree classifier is utilized for the purpose of achieving background noise classification. Second, alternative features to MFCC are considered in order to improve the overall classification rate. In this study, the noise classes have been limited to three widely encountered noise environments of babble noise (e.g., restaurant, mall), street noise, and machinery noise. Although there are other noise types or classes, by limiting the noise classes to the above three major noise environments, the computational complexity is kept low making the real-time deployment feasible.

2.2 PREVIOUSLY DEVELOPED ENVIRONMENT-ADAPTIVE COCHLEAR IMPLANT PIPELINE

The previously developed environment-adaptive speech processing pipeline for cochlear implants described in [1-5] is briefly mentioned here to provide an overview of the components involved. The pipeline consists of two parallel paths, see Figure 2.1, a speech processing path and a noise classification path, both running in real-time. The speech processing path includes a parameterized noise suppression component whose parameters are set according to the noise class identified by

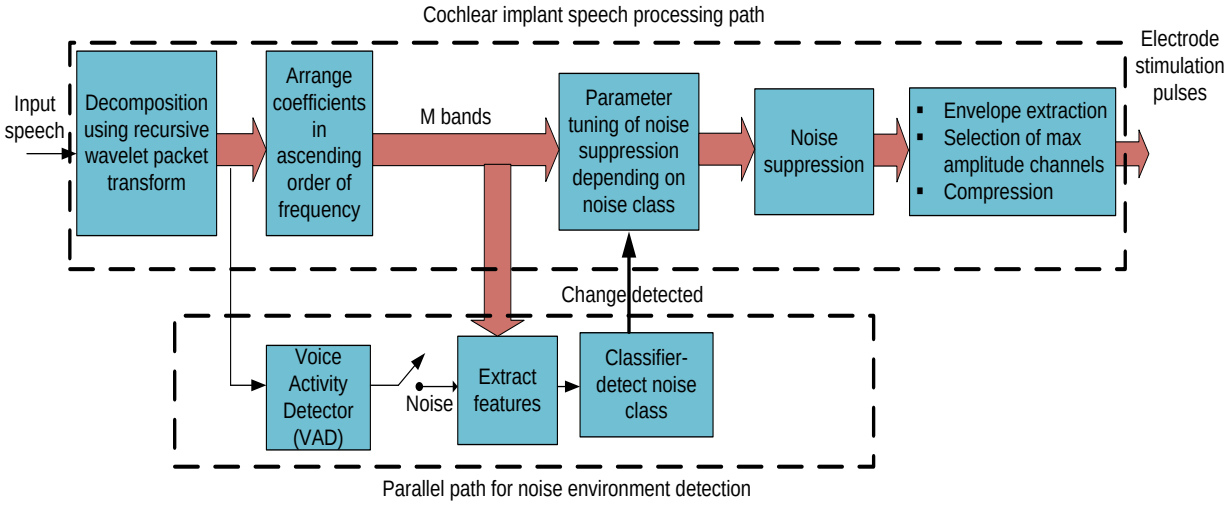

Figure 2.1. Cochlear implant speech processing pipeline implemented in real-time [2]

the noise classification path. This path includes a voice activity detector (VAD) to determine whether signal frames are pure noise or speech+noise. Then, MFCC features are extracted from these durations and fed into a GMM classifier to determine the noise class or noise type.

In sections 2.3 and 2.4, the modification of the noise classification path is presented. The performance results of the modifications are then reported in section 2.5.

2.3 MODIFIED NOISE CLASSIFICATION

The use of tree classifier has been growing for real-time applications due to their recall computational efficiency. There are different training methods for tree classifiers. It has been shown that ensemble training methods such as boosting and bagging are effective training methods for tree classifiers. In particular, in [18], the method of random forest (RF) was shown to provide higher or more accuracy than the other ensemble methods. In this method, ensemble of trees is grown independently using randomly selected subsets of the training data. For training, the entropy of the root node (starting point of training) starts high since all the training samples from all the

classes are included at this level. Then, the tree is grown in a way that the amount of entropy is decreased at each level, finally reaching the leaves having the lowest entropy.

Let $H(Q)$ denote the entropy at node Q,

$$H(Q) = -\sum_c P(\omega_c) \log_2(P(\omega_c)) \quad (2.1)$$

where $P(\omega_c)$ represents the portion of samples from class ω_c at node Q. It is desired for this value to be 0, that is all the samples reaching this node corresponding to the same class; otherwise this value would be high when all the classes appear equally. Let us consider being at node Q and the samples are to be split between the left and right side of node Q. The information gain after the split can be expressed by

$$I_Q = H(S_Q) - \sum_{d\in\{Right,Left\}} \frac{|S_Q^d|}{|S_Q|} H(S_Q^d) \quad (2.2)$$

where S_Q is the number of samples at node Q, and $Right$, $Left$ indicate the left and right side. Among all possible splitting values, the value which provides the maximum information gain is selected. In other words, the split point which leads to a higher entropy reduction is used for growing the tree.

The classification decision is then made based on the most voted class over all the trees. Each decision tree of RF is grown on a bootstrap training sample using a learning algorithm such as CARTS [19]. During the recall, a test input X gets pushed through all the trees (starting at the root) until it reaches the leaves. Figure 2.2 shows the recall or test process of RF.

As the first contribution of this work, the RF tree classifier is used in place of the GMM classifier previously used in [1] in order to improve the classification outcome. As the second contribution of this work, alterative noise features are considered in place of the MFCC features previously

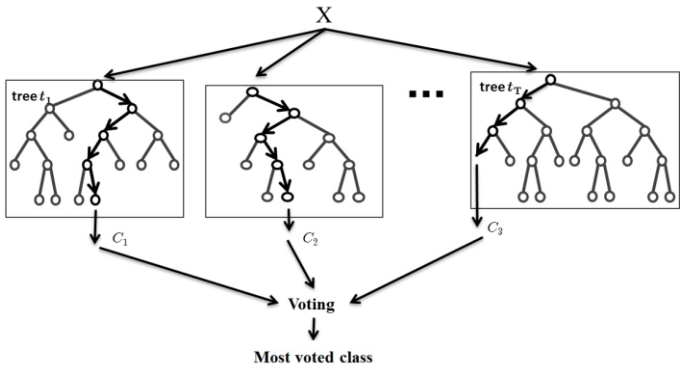

Figure 2.2. Recall process of Random Forest

used in [1] in order to improve the classification outcome. These alternative noise features are discussed next.

2.3.1 Subband Noise Features

The alternative noise features considered include band periodicity (BP) and band entropy (BE). Band periodicity was used in [20] to distinguish between music and background noise based on the periodicity characteristics in each subband of a signal. This feature is utilized here to capture the periodicity aspect of the machinery noise signal whose characteristics remain mostly constant or stationary over time. Figure 2.3 illustrates the difference in the probability densities of this feature in the first subband for the three noise classes considered.

The periodicity of each subband can be represented by the maximum local peak of the normalized correlation function. The normalized correlation function between two adjacent frames is calculated as follows:

$$C_{b,n}(k) = \frac{\sum_{m=0}^{M-1} f(m-k)f(m)}{\sqrt{\sum_{m=0}^{M-1} f^2(m-k) \sum_{m=0}^{M-1} f^2(m)}} \tag{2.3}$$

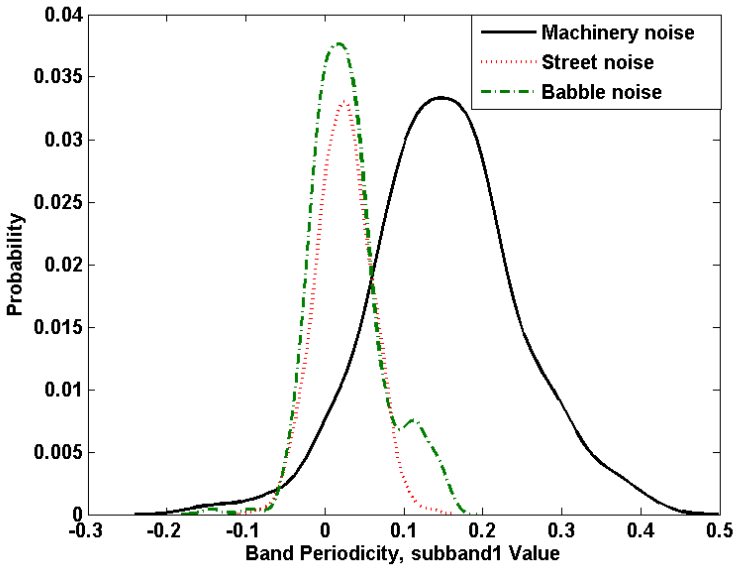

Figure 2.3. Probability density curves of band periodicity for babble, machinery and street noise classes

where $C_{b,n}(k)$denotes the normalized correlation function between two frames with b denoting band index and n frame index, $f(.)$ is the subband signal associated with the two consecutive frames, and M indicates the frame length. Let the maximum local peak of the correlation of two adjacent frames be $C_{b,n}(k_p)$. Then, the band periodicity of noise signal frames at each subband is calculated as follows:

$$BP_b = \frac{1}{N}\sum_{n=1}^{N} C_{b,n}(k_p) \tag{2.4}$$

where N indicates the total number of frames.

Band entropy is a feature that provides a measure of entropy at each subband of noise signal frames, that is

$$BE_b = \frac{1}{N}\sum_{n=1}^{N} H(n) \tag{2.5}$$

where $H(n)$ denotes the entropy of nth frame. This feature is meant to capture the non-stationary characteristics of the babble and street noise types.

2.4 CLASSIFICATION RESULTS

To examine the effectiveness of the modifications made, noise data corresponding to the three noise environments of babble, street, and machinery were collected at a sampling frequency of 44,100Hz. Randomly selected 80% of the dataset was used for training and the remaining 20% of the dataset was used for testing. This selection was repeated 100 times and the classification outcomes were averaged.

For extracting MFCC features, the signals were windowed into 11msec frames as done in [1] via a Hamming window with 6 msec overlap. 50 CART trees were used for the RF tree classifier. Table 2.2 provides the classification outcome when using the GMM and the tree classifier while keeping the features the same as the ones in [1], i.e. 13 MFCC features. These results show that the tree classifier provided a higher classification rate than the GMM classifier. In addition, it was found that the computation time associated with the RF tree classification was approximately 30% lower than that of the GMM classifier. In other words, the entire speech processing pipeline could still be run in real-time.

When using the subband features, as done in [20], the noise signals were segmented into 1s window frames across 8 subbands. These segments were then divided into forty 25-ms non-overlapping frames. It was found that the band periodicity of the first 6 subbands and the band entropy of the first 4 subbands provided the highest discriminatory power than the other bands. As a result, these 10 subband features were used for the classification. Table 2.3 shows the classification confusion matrix via the tree classifier when using the 10 subband features in place of the original MFCC

Table 2.2. Comparison of GMM and RF Tree classification

	Babble		**Street**		**Machinery**	
	GMM	***Tree***	***GMM***	***Tree***	***GMM***	***Tree***
Babble	**90.1%**	**94.6%**	7.2%	1.7%	2.7%	3.7%
Street	5.5%	2.6%	**91.6%**	**97.2%**	2.9%	0.2%
Machinery	10.2%	1.3%	4.8%	0.2%	**85 %**	**98.5%**

Table 2.3. Confusion matrix using subband noise features

	Babble	**Street**	**Machinery**
Babble	**98.3%**	1.7%	0.0%
Street	1.5%	**98.5%**	0.0%
Machinery	0.1%	0.0%	**99.9%**

Table 2.4. Treatment of other noise environments

Noise Environment	**Mapped Noise Environment**		
	Babble	**Street**	**Machinery**
Quiet office with PC fan running	0%	0%	100%
Bus on road	0%	3%	97%
Party	100%	0%	0%
Hood in kitchen	0%	0%	100%
Market	90%	0%	10%
Church	96%	4%	0%
Airport	70%	10%	20%

features. As can be seen from this table, the overall classification rate was improved by 10% over the previous classification rate as a result of the two improvements made in this work.

As stated earlier, to keep the complexity low, the noise classes were limited to the three widely encountered noise environments of babble noise, street noise, and machinery noise. Another experiment was carried out to examine the performance of our developed classification approach in the presence of other noise types. Table 2.4 provides the classification results for this experimentation. As can be seen from this table, other noise types were placed into the closest noise class with similar noise feature characteristics.

2.5 CONCLUSION

In this chapter, two modifications were made to the previously developed noise classification path of the environment-adaptive speech processing pipeline of cochlear implants. The first modification involved the utilization of a random forest tree classifier in place of a GMM classifier. The second modification involved the utilization of subband features to capture periodicity and entropy of noise signals. These modifications led to 10% increase in the overall classification rate while at the same time generating a lower computational burden, thus maintaining the real-time implementation aspect on the FDA approved PDA research platform for cochlear implant studies. It is planned to carry out a study on patients by turning on and off the classification path developed in this work.

2.6 ACKNOWLEDGEMENT

This work was supported by a contract from Cochlear Limited to The University of Texas at Dallas.

2.7 REFERENCES

[1] V. Gopalakrishna, N. Kehtarnavaz, T. Mirzahasanloo, and P. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, pp. 1691-1700, 2012.

[2] T. Mirzahasanloo and N. Kehtarnavaz, "Real-time dual-microphone noise classification for environment-adaptive pipelines of cochlear implants," *Proceedings of the 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 5287-5290, July 2013.

[3] T. Mirzahasanloo, V. Gopalakrishna, N. Kehtarnavaz, and P. Loizou, "Adding real-time noise suppression capability to the cochlear implant PDA research platform," *Proceedings of the 34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2271-2274, August 2012.

[4] T. Mirzahasanloo, N. Kehtarnavaz, and I. Panahi, "Adding quiet and music detection capabilities to FDA-approved cochlear implant research platform," *Proceedings of the 8th International Symposium on Image and Signal Processing and Analysis*, pp. 399- 403, September 2013.

[5] V. Gopalakrishna, N. Kehtarnavaz, P. Loizou, and I. Panahi, "Real-time automatic switching between noise suppression algorithms for deployment in cochlear implants," *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 863- 866, September 2010.

[6] P. Khunarsal, C. Lursinsap, and T. Raicharoen, "Very short time environmental sound classification based on spectrogram pattern matching," *Journal of Information Sciences*, vol. 243, pp. 57-74, 2013.

[7] S. Chu, S. Narayanan, C. Kuo, "Composite-dbn for recognition of environmental contexts," *Proceedings of the Signal Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1 -4, 2012.

[8] Y. Li, Y. Li, "Eco-environmental sound classification based on matching pursuit and support vector machine," *Proceedings of the 2nd International Conference on Information Engineering and Computer Science (ICIECS)*, pp. 1-4, 2010.

[9] H. Lozano, I. Hernaez, A. Picon, J. Camarena, and E. Navas, "Audio classification techniques in home environments for elderly/dependant people," *Proceedings of the 12th International Conference on Computers Helping People with Special Needs*: *Part I, ICCHP'10*, Springer-Verlag, pp. 320-323, 2010.

[10] S. Chu, S. Narayanan, C. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Transactions on Audio, Speech and Language Processing*, vol.17 pp. 1142-1158, 2009.

[11] H. Byeong-jun, H. Eenjun, "Environmental sound classification based on feature collaboration," *Proceedings of IEEE International Conference on Multimedia and Expo*, pp. 542-545, 2009.

[12] S. Ntalampiras, I. Potamitis, N. Fakotakis, "Automatic recognition of urban environmental sounds events," *Proceedings of International Association for Pattern Recognition Workshop on Cognitive Information Processing*, pp. 110-113, 2008.

[13] C. Kraetzer, A. Oermann, J. Dittmann, and A. Lang, "Digital audio forensics: a first practical evaluation on microphone and environment classification," *Proceedings of the 9th Workshop on Multimedia and Security*, pp. 63- 74, 2007.

[14] A. Eronen, V. Peltonen, J. Tuomi, A. Kalpuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, pp. 321-329, 2006.

[15] J.C. Wang, J.F. Wang, K. He, C. Hsu, "Environmental sound classification using hybrid svm/knn classifier and mpeg-7 audio lowlevel descriptor," *Proceedings of IEEE, International Joint Conference on Neural Networks*, pp. 1731-1735, 2006.

[16] R. Malkin, and A. Waibel, "Classifying user environment for mobile applications using linear autoencoding of ambient audio," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 509-512, 2005.

[17] Y. Toyoda, J. Huang, S. Ding, Y. Liu, "Environmental sound recognition by multilayered neural networks," *Proceedings of the 4th International Conference on Computer and Information Technology, CIT '04,* pp. 123-127, 2004.

[18] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5-32, 2001.

[19] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*, Wadsworth Int. Group, 1984.

[20] L. Lu and H. Zhang "Content analysis for audio classification and segmentation," *IEEE Transactions on Speech and Audio Processing*, vol. 10, pp. 504-516, 2002.

CHAPTER 3

SMARTPHONE-BASED REAL-TIME CLASSIFICATION OF NOISE SIGNALS USING SUBBAND FEATURES AND RANDOM FOREST CLASSIFIER*

Authors - Fatemeh Saki, Abhishek Sehgal, Issa Panahi, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents the real-time implementation and field testing of an app running on smartphones for classifying noise signals involving subband features and a random forest classifier. This app is compared to a previously developed app utilizing mel-frequency cepstral coefficients features and a Gaussian mixture model classifier. The real-time implementation has been carried out on both the Android and iOS smartphones. The field testing results indicate the superiority of this newly developed app over the previously developed app in terms of classification rates.

3.1 INTRODUCTION

The problem of environmental background noise classification has been previously examined in many papers for various applications. Some example applications include classifying environmental sound signals in robotics [1], in smart homes for elderly people [2], and in automatic tagging of sound files [3]. In addition, noise classification has been utilized as part of speech enhancement or noise suppression pipelines for hearing aid and cochlear implant devices [4-6], where the speech enhancement parameters are adjusted depending on the environmental background noise.

A typical environmental background noise classification algorithm consists of two major components: a feature extractor and a classifier. Signal features which have been previously considered for noise classification are many. The major ones include: mel-frequency cepstrum coefficients (MFCC), matching pursuit [7], zero crossing rate, centroid and roll-off point [8], spectral centroid, spectral spread, spectral flatness, spectral flux, change chirp rate spectrum, Hilbert envelope, local energy and discrete curvelet transform [9], harmonic ratio, upper limit of harmonicity, and audio fundamental frequency [10]. A combination of these features is often used to achieve a high classification rate [11].

As far as classifiers are concerned, Hidden Markov Model (HMM), Gaussian Mixture Model (GMM), Support Vector Machine (SVM), neural networks, deep belief network classifier, k-nearest neighbor have been utilized [7-12] for noise classification.

As discussed in [13], one issue that has not been adequately addressed is the real-time computation aspect of such features and classifiers. In [13], band-periodicity and band-entropy features and Random Forest (RF) classifier were used to achieve background noise classification for cochlear

implants applications. It was shown that computationally efficient subband features along with an RF classifier (subband+RF) outperformed a previously developed MFCC and GMM (MFCC+GMM) approach [4, 14].

In this chapter, a real-time implementation of the subband+RF noise classification is reported on both Android and iOS smartphones together with a performance comparison with the MFCC+GMM noise classification.

The rest of the chapter is organized as follows. An overview of our previously developed background noise classifier using subband features and random forest classifier is provided in section 3.2. The steps taken towards the smartphone implementation of this classification approach are then reported in section 3.3. Section 3.4 includes the results corresponding to both offline analysis as well as real-time field testing. Finally, the conclusion is stated in section 3.5.

3.2 OVERVIEW OF PREVIOUSLY DEVELOPED BACKGROUND NOISE CLASSIFICATION

Although MFCC features have been extensively used in the literature for noise signal classification, it is found that they have limitations in realistic noise environments. That is why additional features are often used in addition to MFCC features to gain high classification rates. However, a practical problem that arises as a result of utilizing many features is the computational complexity associated with running a classification signal processing pipeline in real-time on handheld devices, in particular on smartphones. In [13], subband features and a random forest (RF) classifier were used as an alternative to MFCC features and a GMM classifier that had been shown to be computationally suitable to achieve real-time throughputs compared to many other features [4].

As discussed in [13], subband features consist of band-periodicity and band-entropy features. Band-periodicity features capture the periodicity aspect of noise signals whose characteristics remain more or less stationary over time; whereas band-entropy features capture the non-stationary characteristics of noise signals. Band-periodicity and band-entropy features are computed from signal segments of duration S seconds. Each segment is divided into M overlapping frames of length N, with the m^{th} frame specified by $F_m \coloneqq \{x_n | x_n \in \mathbb{R}, n = 1, \dots N\}$, where $x_n$represents the n^{th} sample in the frame. Assuming the sampling rate of F_s, the frequency range $[0,\frac{F_s}{2}]$ is divided into B non-overlapping subbands. The cross-correlation between every two consecutive frames, that is $F_m$and F_{m-1} in each band, is computed and the peak value of the cross-correlation is denoted by $P_{b,m}$, where b and m represent the band and frame index, respectively. The band-periodicity feature in band b is then defined as [15]:

$$BP_b = \frac{1}{M}\sum_{m=1}^{M} P_{b,m} \ , b = 1, \dots, B \tag{3.1}$$

where M is the total number of frames over duration S.

The band-entropy feature in each band over duration S is defined as:

$$BE_b = \frac{1}{M}\sum_{m=1}^{M} H_{b,m} \ , b = 1, \dots, B \tag{3.2}$$

where $H_{b,m}$ represents the entropy of the m^{th} frame in band b. Considering B bands, a feature vector of $2\times B$ components is thus used to capture the signal characteristics over a duration of S seconds. The extracted feature vector is then fed into an RF classifier to find a matched class to the incoming signal frames. It is worth noting that band-periodicity and band-entropy features unlike the MFCC features are not sensitive to the sound loudness, thus they do not require any preprocessing normalization prior to their extraction.

An RF classifier [16] is an ensemble of T number of classification trees. Each tree is trained independently from other trees using a randomly selected (with replacement) subset of a training set. At the start of the training, or at the root node, the entropy is high since all training samples from all the classes are used at this stage. Then, the tree is built in such a way that the entropy is decreased as layers are added until the tree reaches its leaves with the lowest entropy allowing classification of all the training data.

MFCC features are widely used in speech processing. MFCC features attempt to capture the spectral information corresponding to the human auditory response. MFCC features are computed by grouping the short time Fourier transform coefficients of a frame into a set of L coefficients based on L mel-scale non-overlapping filters or filterbank, followed by a discrete cosine transform for decorrelation purposes. Normally, the first 13 coefficients are used to serve as MFCC features. Likewise, the Gaussian Mixture Model (GMM) classifier is extensively used for signal classification. In this classifier, the data or samples corresponding to a class is modeled by a mixture of several Gaussians in the feature space whose parameters are estimated using the iterative expectation-maximization algorithm.

3.3 REAL-TIME IMPLEMENTATION ON SMARTPHONES

The subband feature extraction and the random forest classifier were coded in C which were then integrated into the Android and iOS smartphones using the guidelines provided in the book "Smartphone-Based Real-Time Digital Signal Processing" [17]. The shell provided in the book was used for the microphone interfacing and the GUI. The software tools that were used to achieve the smartphone implementation are noted below: For Android smartphones, the IDE (Integrated Development Environment) of Android Studio was used together with the Android SDK (Software

Development Kit) [18]. To support C codes within Android smartphones, the Android NDK (Native Development Kit) [19] was used. For the iOS implementation, the IDE of Xcode [20] was used. C codes were interfaced with Objective-C of iOS by importing the header file. Interested readers are referred to the above book for the details of embedding and running C codes within the Android and iOS environments.

For feature extraction, signals were captured in frames of length 25msec with half a frame overlap, i.e., 12.5msec overlap. MFCC features were extracted from every frame and the extracted feature vector was fed into a GMM classifier. The implementation was done using 13 MFCC features with a mel-filter bank of 40 filters together with two Gaussians in the mixture model per class.

Band-periodicity and band-entropy features were computed per signal segment of duration S =1 second. Each incoming frame was divided into B = 8 non-overlapping bands of width 1kHz in the frequency domain. Thus, a feature vector of 16 subband features (8 band-periodicity and 8 band-entropy features) was obtained over every 1 second which was then fed into an RF classifier consisting of 20 trees.

Screen snapshots of the app on an Android smartphone are provided in Figure 3.1. The user has the option to perform online classification of sound signals that are captured by the smartphone microphone or to save captured sound signals for later examination. The app allows adjusting the sampling rate, frame length, frame overlap amount, and decision buffer length (in frame unit) for majority voting classification.

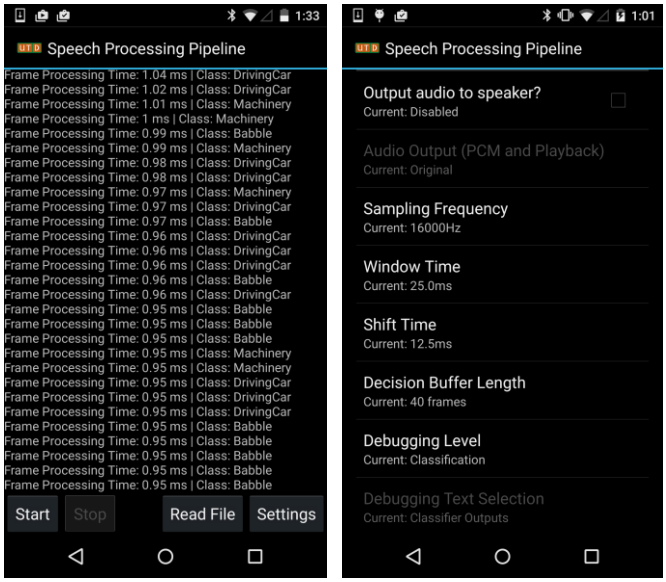

Figure 3.1. Snapshots of the developed noise classification smartphone app

3.4 EXPERIMENTAL RESULTS AND COMPARISON

The developed classification app was examined by considering three widely encountered noise types of babble, car driving and machinery. The examination was done in offline and field testing manners which are explained in more details in the subsection that follow.

3.4.1 Dataset

As part of the app development, a comprehensive dataset of 120 sound files for the three noise types of babble, car driving and machinery were put together which is accessible for public use at the website noted in [21]. The machinery class contains noise signals of home appliances. For each noise type, 40 sound files of duration 30 seconds were collected at different times at a sampling frequency of 16kHz using a Nexus 5 smartphone. For both the data collection and the real-time operation of the classifier, only one microphone of the smartphone is used.

3.4.2 Offline Evaluation and Comparison

The MFCC+GMM and subband+RF classification approaches were evaluated in an offline manner first as follows. The dataset was randomly divided into a training (80%) and a testing set (20%) with no overlap between them. This procedure was repeated 100 times. Each time the classifiers were trained using a different training and testing sets and the averaged results are indicated in Tables 3.1 and 3.2. As can be noted from these tables, the subband+RF approach provided a higher overall classification rate compared to the MFCC+GMM approach, in particular for babble type of noise. This is attributed to the discriminatory power of subband features as compared to MFCC features as evident by computing the Fisher discriminant measure [22]:

$$J = trace\ (S_w^{-1} S_b) \tag{3.3}$$

where S_w denotes the within-class scatter matrix and S_b the between-class scatter matrix. Higher J values indicate that samples in the multi-dimensional feature space are more separated. When using the subband features, this feature was found to be $J = 2350$, while when using the MFCC features, this measure was found to be $J = 38$, indicating a high level of spread or overlap between the babble class and the other two classes in the MFCC feature space.

Table 3.1. Offline evaluation of subband+RF, averaged over 100 different training and testing

Detected class / Actual class	Babble (%)	Car Driving (%)	Machinery (%)
Babble	**98.9**	0.1	1
Car Driving	0	**99.7**	0.3
Machinery	3.9	0.1	**96**

Table 3.2. Offline evaluation of MFCC+GMM, averaged over 100 different training and testing

Detected class / Actual class	Babble (%)	Car Driving (%)	Machinery (%)
Babble	**86.5**	11.5	2
Car Driving	3.3	**95.6**	1.1
Machinery	2.1	0.9	**97**

The next experimentation involved running the classifiers in real-time on actual smartphones in the field which is mentioned next.

3.4.3 Actual Filed Testing and Comparison

The developed classifier apps were run on smartphone platforms in the three noise environments to evaluate their actual performance in the field. The outcome of this experimentation appears in Tables 3.3 and 3.4. It is worth pointing out that since microphones on different smartphones have different frequency responses, the data collection and thus training were repeated for each device to remove any frequency dependency on the device microphone. As noted from these tables, the MFCC+ GMM app and the subband+RF app performed similarly in the noise environments of car driving and machinery. However, in the babble environment, the subband+RF app by far outperformed the MFCC+GMM app.

The reason for the poor performance of the MFCC+GMM app in the field was traced back to the sensitivity of MFCC features versus subband features. MFCC features were found to be quite sensitive to various variations that occur in babble type of noise environments in the field whereas

Table 3.3. Field testing of Subband +RF

Detected class / **Actual class**	**Babble** (%)	**Car Driving** (%)	**Machinery** (%)
Babble	**80.4**	0	19.6
Car Driving	0.4	**99**	0.6
Machinery	0	0	**100**

Table 3.4. Field testing of MFCC +GMM

Detected class / **Actual class**	**Babble** (%)	**Car Driving** (%)	**Machinery** (%)
Babble	**47.4**	1.1	51.4
Car Driving	1	**99**	0
Machinery	0	0	**100**

the subband features were found to be much less sensitive to various variations that occur in babble type of noise environments. As a percentage, it was found that MFCC features exhibited a large variation of 173% in the field testing performed whereas subband features only exhibited a variation of 2% when encountered with variations of babble type of noise for which the classifiers had not been trained.

Another study was conducted to assess the behavior of the apps in the presence of other noise types for which no training had been done. The outcome of this study appears in Table 3.5. As seen from this table, these other noise types got matched to the closest class with similar sound characteristics when using the subband+RF app, while the MFCC +GMM app could not distinguish between the babble and machinery noise types. For example, the crowded restaurant with music in the background, which was not part of the training data, was classified as machinery noise type and the loud indoor air conditioning (AC) noise, which was not part of the training data, was classified as babble noise type.

The average processing times per 25msec frames with a frame overlap of 12.5msec for the subband+RF model on an Android platform (Nexus 5) and on an iOS platform (iPad Mini 2) are shown in Table 3.6. This time incorporates the i/o delay time associated with these devices. To

Table 3.5. Treatment of other noise environments, subband+RF vs MFCC+GMM

	Subband + RF			MFCC + GMM		
Matched class / **Other classes**	**Babble** (%)	**Car Driving** (%)	**Machinery** (%)	**Babble** (%)	**Car Driving** (%)	**Machinery** (%)
Crowded Restaurant	**89.5**	3.4	7.1	1	**1**	98
Street	18.5	**13.2**	**68.3**	0	**17.5**	**82.5**
Loud Indoor AC	0	16.3	**83.7**	**98.9**	**1.1**	**0**
Washer	7.8	0.8	**91.4**	**51.3**	**8.7**	**40**
Dryer	0	8.3	**91.7**	**50.5**	**49.5**	**0**
Vacuum	0	0	**100**	0	**0**	**100**

Table 3.6. Averaged frame processing times of subband+RF model (25 msec frames with half frame overlap at 16 kHz sampling frequency)

Frame processing time in msec	Subband feature extraction + RF classifier
Android without using VFP	**3.1ms**
Android with using VFP	**1.5ms**
iOs without using VFP	**3.4ms**
iOS with using VFP	**3.1ms**

achieve real-time throughputs, the total processing time needed to remain below 12.5msec for no frame to get skipped. When using the Vector Floating-Point (VFP) coprocessor hardware on the smartphones, the timings naturally improved. The table lists the timings with and without using VFP. In all the cases, real-time throughputs were achieved. A video clip of the subband+RF classification app can be viewed at the link stated in [23].

3.5 CONCLUSION

This chapter has provided an app for carrying out background noise classification in real-time on smartphone platforms. Two classification approaches having low computational complexity which allowed them to be run in real-time on smartphone platforms, namely MFCC+GMM and subband+RF, were implemented and compared in the field. The extensive experimentations carried out have shown that the subband+RF approach provides both real-time throughputs and high classification performance for the three commonly encountered noise environments of babble, car driving and machinery.

3.6 ACKNOWLEDGMENT

This work was supported in part by the National Institute of the Deafness and Other Communication Disorders (NIDCD) of the National Institutes of Health under the award number

5R56DC014020-02. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

3.7 REFERENCES

[1] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun, "Towards robotic assistants in nursing homes: Challenges and results," *Special Issue Socially Interactive Robots, Autonomous Systems,* vol. 42, no. 3–4, pp. 271-281, 2003.

[2] J. Wang, H. Lee, J. Wang, and C. Lin, "Robust environmental sound recognition for home automation," *IEEE Transactions Automation Science and Engineering,* vol. 5, no. 1, pp. 25-31, 2008.

[3] S. Duan, J. Zhang, P. Roe, and M. Towsey, "A survey of tagging techniques for music, speech and environmental sound," *Artificial Intelligence Review*, pp. 1-25, 2012.

[4] V. Gopalakrishna, N. Kehtarnavaz, T. Mirzahasanloo, and P. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, pp. 1691-1700, 2012

[5] P. Loizou, A. Lobo, and Y. Hu, "Subspace algorithms for noise reduction in cochlear implants," *The Journal of the Acoustical Society of America*, pp. 2791-2793, 2005.

[6] F. Saki, T. Mirzahasanloo, and N. Kehtarnavaz, "A multi-band environment-adaptive approach to noise suppression for cochlear implants," *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'14)*, pp. 1699-1702, Chicago, August, 2014.

[7] S. Chu, S. Narayanan, C. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Transactions on Audio, Speech and Language Processing*, vol.17 pp. 1142-1158, 2009.

[8] H. Lozano, I. Hernaez, A. Picon, J. Camarena, and E. Navas, "Audio classification techniques in home environments for elderly/dependant people," *Proceedings of the 12th International Conference on Computers Helping People with Special Needs*: *Part I, ICCHP'10*, Springer-Verlag, pp. 320–323, 2010.

[9] H. Byeong-jun, H. Eenjun, "Environmental sound classification based on feature collaboration," *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, pp. 542-545, 2009.

[10] S. Ntalampiras, I. Potamitis, N. Fakotakis, "Automatic recognition of urban environmental sounds events," *Proceedings of International Association for Pattern Recognition Workshop on Cognitive Information Processing*, pp. 110-113, 2008.

[11] C. Kraetzer, A. Oermann, J. Dittmann, and A. Lang, "Digital audio forensics: a first practical evaluation on microphone and environment classification," *Proceedings of the 9th Workshop on Multimedia and Security*, pp. 63-74, Dallas, TX, 2007.

[12] P. Khunarsal, C. Lursinsap, and T. Raicharoen, "Very short time environmental sound classification based on spectrogram pattern matching," *Information Sciences*, vol. 243, pp. 57-74, 2013.

[13] F. Saki, N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

[14] S. Parris, M. Torlak, and N. Kehtarnavaz, "Real-time implementation of cochlear implant speech processing pipeline on smartphones," *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC),* pp. 886-889*,* August 2014.

[15] L. Lu and H. Zhang "Content analysis for audio classification and segmentation," *IEEE Transactions on Speech and Audio Processing,* vol. 10, pp. 504-516, 2002.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001

[17] N. Kehtarnavaz, S. Parris, A. Sehgal, *Smartphone-Based Real-Time Digital Signal Processing*, Morgan and Claypool Publishers, 2015.

[18] Android Studio, http://developer.android.com/sdk/index.html

[19] Android NDK, http://developer.android.com/tools/sdk/ndk/index.html

[20] Apple, https://developer.apple.com/xcode/

[21] The University of Texas at Dallas, http://www.utdallas.edu/~kehtar/NoiseData.rar

[22] R. Duda, and P. Hart, *Pattern Classification,* New York: Wiley, 2000.

[23] The University of Texas at Dallas, http://www.utdallas.edu/~kehtar/NoiseClassifierDemo.mp4

CHAPTER 4

AUTOMATIC SWITCHING BETWEEN NOISE CLASSIFICATION AND SPEECH ENHANCEMENT FOR HEARING AID DEVICES*

Authors - Fatemeh Saki, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents a voice activity detector (VAD) for automatic switching between a noise classifier and a speech enhancer as part of the signal processing pipeline of hearing aid devices. The developed VAD consists of a computationally efficient feature extractor and a random forest classifier. Previously used signal features as well as two newly introduced signal features are extracted and fed into the classifier to perform automatic switching. This switching approach is compared to two popular VADs. The results obtained indicate the introduced approach outperforms these existing approaches in terms of both detection rate and processing time.

4.1 INTRODUCTION

It is well known that the performance of the signal processing pipelines deployed in hearing aid devices degrade significantly in noisy environments. This in turn negatively impacts the hearing experience of users of these devices in noisy environments. Initial attempts to address this issue have involved adjusting the enhancement setting manually by users on these devices. More recently, attempts have been made to automatically adjust the enhancement setting depending on the noise types encountered by users. For example, in [1] a real-time speech processing pipeline for cochlear implants was developed that allowed an automatic on-the-fly classification of different background noise types for the purpose of tuning the speech enhancement parameters to the classified noise type.

Such noise adaptive solutions require a voice activity detector (VAD) to be used at the frontend of the pipeline in order to identify the input sound status as unvoiced or voiced, that is as pure noise or speech plus noise. As illustrated in Figure 4.1, the noise classifier is activated when the VAD identifies the presence of pure noise and the speech enhancement is activated when the VAD identifies the presence of speech, noting that speech may appear as clean speech with no noise or

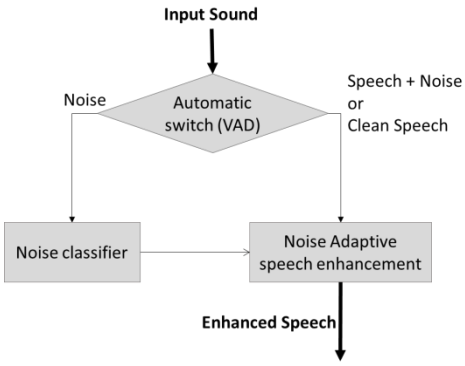

Figure 4.1. Noise adaptive speech enhancement pipeline

more realistically as speech plus noise. It is important to note that the VAD should be able to identify pure noise irrespective of its type. In this work, our objective has been to develop a computationally efficient VAD that can cope with different types of noise as part of a noise adaptive speech enhancement pipeline.

VAD constitutes an essential component in many speech processing applications such as speech enhancement and speech recognition for the purpose of distinguishing speech from noise. The literature includes many studies on VADs. A typical VAD consists of two modules: a feature extractor and a classifier. The first module extracts signal features that allow discrimination between voice and unvoiced signals. The second module is a decision making one to separate features of voiced signals from unvoiced signals. VADs can be categorized into two categories based on their decision making approach: statistical-based and machine learning-based. The statistical-based ones establish probability density functions for noise and speech classes and then a data driven decision rule is applied to classify speech signal segments or frames from noise signal segments or frames [2-4]. More recently, machine learning-based VADs have been developed generating improved performance [5-12]. Table 4.1 provides an overview of existing VAD approaches.

The existing approaches have mostly concentrated on one type of noise with often stationary statistical characteristics. In this work, we consider different types of noise (stationary, semi-stationary, and non-stationary). Furthermore, we have found many of the features used previously are computationally demanding relative to the features we have considered for separating voiced from unvoiced signals. The thrust of this work is thus on a computationally efficient VAD so that it can be deployed as an automatic switch as part of a real-time noise adaptive speech processing

Table 4.1. Overview of existing VAD approaches

Reference	Year	Features	Decision rule
Kim et al. [5]	2016	likelihood ratios	deep belief neural network
Hwang et al. [6]	2015	prior SNR, posterior SNR and statistical-based features named LR	ensemble of deep neural networks
Zou et al. [7]	2014	PCA- mel-frequency cepstral coefficients (MFCC)	support vector machine (SVM)
Zhang et al. [8]	2014	multi-resolution cochleagram (MRCG)	boosted deep neural network
Zhang et al. [9]	2013	pitch, discrete Fourier transform (DFT), MFCC, linear predictive coding (LPC), relative-spectral perceptual linear predictive analysis (RASTA-PLP) , and amplitude modulation spectrograms (AMS)	deep belief networks
Wu et al. [10]	2011	multiple-observation maximum probability (MO-MP) features/ multiple-observation SNR(MO-SNR) features	multiple kernel SVM
Jo et al. [11]	2009	likelihood ratio	SVM
Kinnunen et al. [12]	2007	MFCC, delta-MFCC and double delta-MFCC	SVM

pipeline in hearing devices. From a practical standpoint, this switch is designed in such a way that the switching between voiced and unvoiced situations only takes place in the presence of sustained type of sound environments, i.e. switching is not done frequently in response to noise or voiced frames rather in response to a large number of frames in a majority voting manner.

The rest of the chapter is organized as follows. In section 4.2, the modules of the developed automatic switch or VAD are discussed. This is followed by the experimental results in section 4.3. Finally, the conclusion is stated in section 4.4.

4.2 DEVELOPED AUTOMATIC SWITCH OR VAD

We have considered the following computationally efficient features to separate pure noise signals from speech signals that may also contain noise: band-periodicity, band-entropy, spectrum flux, subband short-time energy deviation ($STED$) and subband power spectral deviation ($SPSD$). These features are chosen here as they are found to exhibit good discriminatory power between pure noise signals and speech signals while at the same time they are computationally efficient to

compute. A feature vector consisting of the above feature components for a sound frame at time *t* is computed over the period [t-S, t].

Spectrum flux has been widely used to separate speech from noise signals. As defined in [13], spectrum flux denotes the averaged difference between spectra of two adjacent frames over a sound segment or frame of S-seconds duration, that is

$$SF = \frac{1}{K \times M} \sum_{k=1}^{K} \left[\log\big(F(m,k)\big) - \log\big(F(m-1,k)\big) \right]^2 \quad (4.1)$$

where $F(m,k)$ denotes the spectrum of the m^{th} frame in the period [t-S, t] at frequency k, K is the DFT length, and M is the number of frames in the period.

Band-periodicity and band-entropy features were introduced in [14] and later on were implemented to run in real-time on smartphone platforms [15]. These features along with a Random Forest (RF) classifier have been shown to provide effective discriminatory power for separating three different noise types: machinery, traffic and babble. These features are thus used here considering that they can be obtained in a computationally efficient manner. They are briefly explained next.

Assuming the sampling rate of the input signals is fs, the frequency range ([0,$\frac{fs}{2}$]) is divided into *B* non-overlapping subbands. For a frame at time *t*, the band-entropy and band-periodicity features are computed as follows:

$$BE_b = \frac{1}{M} \sum_{m=1}^{M} H_{b,m}\,, \quad b = 1, \ldots, B \quad (4.2)$$

where $H_{b,m}$ represents the entropy of the m^{th} frame during the period [t- S, t] in band b.

To compute the band-periodicity features, the cross-correlation between every two adjacent frames in each band is computed and then the peak value of the cross-correlation denoted by $\rho_{b,m}$is used to define the band-periodicity features in band *b* as follows [13]:

$$BP_b = \frac{1}{M}\sum_{m=1}^{M} \rho_{b,m}\,, \quad b = 1, \dots, B \tag{4.3}$$

Energy-based features have shown promising results for identifying the presence of speech. Hence, the energy level is also used here as a feature. Assuming that in a sustained noise environment, the level of background noise remains more or less constant, in the presence of speech, the energy level goes higher. In other words, on average, the deviation in the energy level between the highest (when a person talks) and the lowest energy level (gaps between speech frames) for speech and background noise is higher than that for pure noise as captured by a microphone. It is understood that there are exceptions to this assumption but in general this assumption holds in many practical situations. The deviation in the energy level of the input sound signal $STED$ is computed in different frequency bands as follows:

$$STED_b = \frac{\mu_b - \gamma_b}{\mu_b}\,, b = 1, \dots, B \tag{4.4}$$

where μ_b and γ_b are the average and the minimum energy of M frames during the period [t- S, t] in band *b*. Here, the average value is considered instead of the maximum value in order to capture noise which is sustained and to avoid capturing transient noise. The difference between the average and the minimum value for sustained noise is expected to be lower, while in the presence of noisy speech or clean speech is expected to be higher. Figure 4.2 presents the distributions of this feature for machinery noise and speech plus machinery noise at different SNRs. It can be seen that the distributions for the speech plus noise shift to the right of the pure noise distribution.

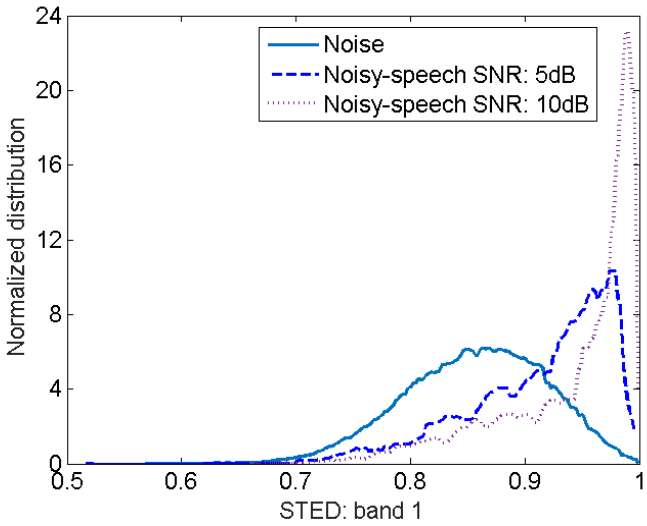

Figure 4.2. Normalized distributions of STED feature in band 1 for pure machinery noise and speech plus machinery noise at 5dB and 10dB SNRs.

Another feature which is introduced and used here is the difference between two adjacent bands in the average power spectral density of sound signals over a long duration. The shape of the average power spectral density over a long duration provides an indication of which frequency regions of the spectrum are more affected by noise distortion and which ones are least affected. Figure 4.3 shows an example of the average power spectral density of clean speech, noisy speech and noise. As shown in this figure, the averaged difference in the power spectrum between the first and the second frequency bands for the clean speech is more noticeable compared to the pure noise and noisy speech. This feature is computed as follows:

$$SPSD_b = \frac{1}{(K/B)}\left(\sum_{k=l_{b+1}}^{u_{b+1}} \widetilde{\omega}_{b+1}(k) - \sum_{k=l_b}^{u_b} \widetilde{\omega}_b(k)\right), b = 1, \dots, B-1 \quad (4.5)$$

$$\widetilde{\omega}_b = 10\log_{10}\sum_{m=1}^{M} \bar{P}_b, b = 1, \dots, B \quad (4.6)$$

where $\widetilde{\omega}_b$ is the sum of power spectral density of frames, denoted by $\bar{P}_b$, during the period [t-$\mathcal{S}$, t], andl_b& $u_b$represent the lower and upper frequencies of band b.

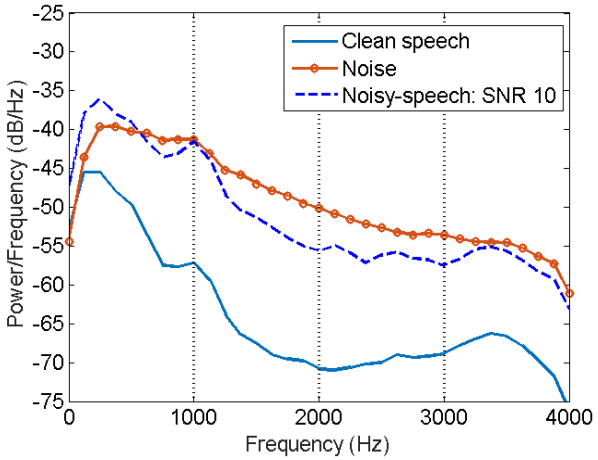

Figure 4.3. Average power spectral density of clean speech, noise and noisy-speech over long durations at 10dB SNR.

As stated earlier, for classification, an RF classifier is used since it is computationally efficient and provides good classification performance. An RF classifier is an ensemble of *T* number of decision trees. Each tree is trained independently using a randomly selected subset of the training set. Then, the tree is built in such a way that the entropy is decreased as tree levels are added until the tree reaches its leaves. In the recall mode, an input sound is assigned to the most voted class by all the trained trees in the RF. More details on RF classification can be found in [16].

4.3 EXPERIMENTAL RESULTS AND DISCUSSION

This section reports the experimental results of switching when using the developed VAD. HINT sentences [17] are widely used by audiologists to measure a person's ability to hear speech in noisy background environments. These sentences were thus used to serve as clean speech sounds. Noisy speech sounds were generated by adding three different noise types: machinery (stationary type), driving car (semi-stationary type) and babble (non-stationary type) to the HINT sentences at

different SNR levels. The noise files are made available for public use and can be downloaded from the link http://www.utdallas.edu/~kehtar/VAD-dataset.

Frames of 10 ms durations with 50% overlap were considered for feature extraction. Feature vectors were extracted for a majority voting period of S=200 ms. Four subbands, or B=4, were used for the subband features and 10 trees, or T=10, were used in the RF classifier. One half of the dataset was randomly chosen for training and the remaining half with no overlap was used for testing. Among the extracted subband features, the four band-periodicity features, the first two band-entropy features and the first band $STED$ feature were found to be the most effective features by carrying out a minimum redundancy maximum relevance (MRMR) feature selection analysis as discussed in [18]. Thus, together with the spectrum flux and $SPSD$ features, 9 features were used in total.

The developed VAD was evaluated in terms of speech hit rate or true positive rate (TPR) and false alarm rate (FAR). True positive rate here means the ratio of number of correctly detected speech frames to the number of true speech frames and false alarm rate is defined based on non-speech detection hit rate (NHR) (FAR = 100-NHR), where non-speech hit rate denotes the ratio of number of correctly detected noise frames to number of true noise frames.

Table 4.2 provides a comparison of the performance of our developed VAD with the standard G.729 annex B [19] and Sohn's VAD [3] in terms of TPR and NHR. These results reflect the outcome after applying majority voting over 200 ms. The same majority voting was also considered for the other two VADs. As can be seen from this table, our approach outperformed these approaches by more than 25% in terms of TPR. The FAR for our VAD was about 5% while for G.729 and Sohn, it was much higher; 40.8% and 57.8%, respectively. Figure 4.4 shows an

Table 4.2. Comparison between the developed VAD, G.729 VAD, and Sohn's VAD in terms of speech hit rate and noise hit rate in percentage %

Environments	SNR (dB)	Developed VAD	G.729	Sohn
Real noisy speech	-	98.8	79.1	77.6
Simulated noisy speech (machinery, driving car and babble noises)	0	94.1	55.0	69.0
	5	95.9	74.1	70.0
	10	96.9	68.8	70.6
	15	98.4	70.6	73.5
Clean speech	-	99.5	56.2	67.8
Pure noise (machinery, driving car, babble noises)	-	95.0	59.2	42.2

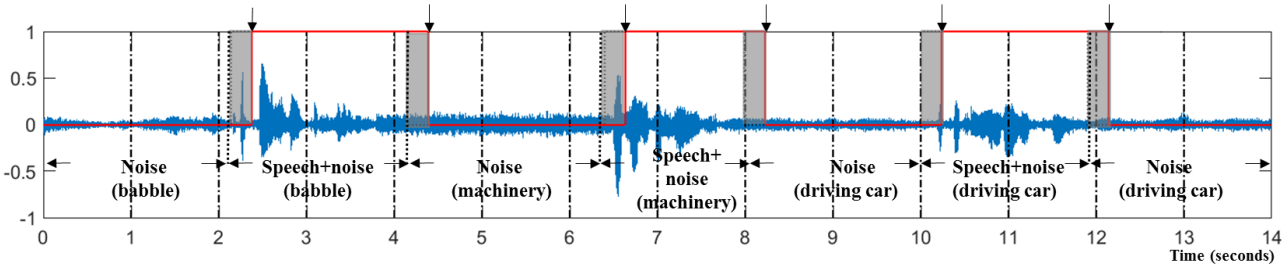

Figure 4.4. Illustration of switching between sustained noise and speech presence, small vertical arrows on top indicate the occurrence of switching, the grey area shows the latency associated with switching after transitioning to a new sound environment (for a 200 ms majority voting decision buffer).

example of sound segments passed through the developed VAD. This example consists of the concatenation of seven files consisting of pure noise and speech plus noise. As shown in this figure, it is important to note that the switching, denoted by vertical arrows, only occurred when the noise was sustained for more than 200 ms and the silent gaps between speech sounds did not get detected as noise. The gray area indicates the 200 ms latency associated with switching after the sound environment was changed. In terms of computation time, it took 7.5 ms to compute our features for 10 ms frames using a PC with 2.67 GHz clock compared to 13.4 ms and 11.1 ms for the features used in the G.729 and Sohn VADs.

4.4 CONCLUSION

This chapter has presented an automatic switching mechanism between speech enhancement and noise classification for deployment in hearing devices. A total of nine computationally efficient features have been extracted and fed into a random forest classifier to identify the presence of speech in different types of noise. The results obtained have indicated that the developed automatic switch or voice activity detector outperforms two other popular voice activity detectors in terms of both detection rate and processing time.

4.5 ACKNOWLEDGMENTS

This work was supported in part by the National Institute of the Deafness and Other Communication Disorders (NIDCD) of the National Institutes of Health under the award number 5R56DC014020-02. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

4.6 REFERENCES

[1] V. Gopalakrishna, N. Kehtarnavaz, T. Mirzahasanloo, and P. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 6, pp. 1691-700, Jul. 2012.

[2] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109-1121, Dec. 1984.

[3] J. Sohn, N. Kim, and W. Sung, "A statistical model-based voice activity detection," *IEEE Signal Processing Letters*, vol. 6, no. 1, pp. 1-3, Jan. 1999.

[4] J. Chang, N. Kim, and S. Mitra, "Voice activity detection based on multiple statistical models," *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 1965-1976, 2006.

[5] S. Kim, Y. Park, and S. Lee, "Voice activity detection based on deep belief networks using likelihood ratio," *Journal of Central South University*, vol. 23, no. 1, pp. 145-149, 2016.

[6] I. Hwang, H. Park, and J. Chang. "Ensemble of deep neural networks using acoustic environment classification for statistical model-based voice activity detection," *Computer Speech & Language*, vol.38, pp. 1-12, 2015.

[7] Y. Zou, W. Zheng, W. Shi, H. Liu, "Improved voice activity detection based on support vector machine with high separable speech feature vectors," *Proceedings of the 19th International Conference on Digital Signal Processing*, Hong Kong, China, pp. 763-767, 2014.

[8] X. Zhang and D. Wang, "Boosted deep neural networks and multi-resolution cochleagram features for voice activity detection," *Proceedings of Interspeech*, pp. 1534-1538, 2014.

[9] X. Zhang and J. Wu, "Deep belief networks based voice activity detection," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 4, pp. 697-710, 2013.

[10] J. Wu and X. Zhang, "Efficient multiple kernel support vector machine based voice activity detection," *IEEE Signal Processing Letters*, vol. 18, no. 8, pp. 466-469, 2011.

[11] Q. Jo, J. Chang, J. Shin, and N. Kim, "Statistical model- based voice activity detection using support vector machine," *IET Signal Processing*, vol. 3, no.3, pp. 205-210, 2009.

[12] T. Kinnunen, E. Chernenko, M. Tuononen, P. Fränti, and H. Li, "Voice activity detection using MFCC features and support vector machine," *International Conference on Speech and Computer*, vol. 2, no. 2, pp. 556-561, 2007.

[13] L. Lu and H. Zhang "Content analysis for audio classification and segmentation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 10, no.7, pp. 504-516, 2002.

[14] F. Saki and N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

[15] F. Saki, A. Sehgal, I. Panahi and N. Kehtarnavaz, "Smartphone-based real-time classification of noise signals using subband features and random forest classifier," *Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2204-2208, Shanghai, China, March 2016.

[16] L. Breiman, Random Forests, *Machine Learning*, doi:10.1023/A:1010933404324, 2001.

[17] California Ear Institute, http://www.californiaearinstitute.com/audiology-services-hint-bay-area-ca.php

[18] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226-1238, 2005.

[19] ITU-T Recommendation G.729-Annex B: A Silence Compression Scheme for G.729 Optimized for Terminals Conforming to Recommendation V.70, 1996.

CHAPTER 5

HIERARCHICAL CLASSIFICATION OF SOUND SIGNALS FOR HEARING IMPROVEMENT DEVICES

Authors - Fatemeh Saki, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents a hierarchical approach to sound signal classification for utilization in hearing improvement devices. The developed classification hierarchy consists of three levels to classify speech, music and different noise types. A distinguishing attribute of this hierarchical approach is that effective features are computed as needed at different levels of the hierarchy making the classification process computationally efficient. This approach is compared to the conventional one-step classification approach by examining both trained and non-trained sound signals. The results obtained show higher classification rates as well as higher computational efficiency of this hierarchical approach compared to the conventional one-step approach.

5.1 INTRODUCTION

Classification of sound signals plays a major role in hearing improvement devices. Examples of such devices that benefit from a sound classification component or subsystem include hearing aids, cochlear implants, and smart headphones. It is well established that the hearing sensation of hearing aid users degrades considerably in noisy environments. Thus, there have been attempts at developing speech enhancement/noise reduction algorithms that are adaptive to different sound environments, e.g. [1-5]. Examples of commercially available hearing improvement devices that include a sound classification component are Phonak Bolero V hearing aid [6] and Cochlear Limited Necleus6 cochlear implant [7]. Classification of sound signals enables adapting the speech enhancement/noise reduction algorithm in such devices to different sound environments in an automatic manner.

This work is aimed at introducing a computationally efficient classification component for hearing improvement devices in order to cope with three major categories of environmental sounds that are commonly encountered on a daily basis. These categories include speech in the presence or absence of background noise, music, and various types of background noise.

A typical sound signal classification component or subsystem possesses two major modules: feature extraction and classifier. Different types of environmental sound signals have been considered in the literature for different applications. As the number of features is increased, the computational complexity of the classification is increased. It is well-known that the combination of many individual features does not necessarily lead to higher classification rates and often causes a limitation as far as the real-time implementation aspect is concerned. Thus, from a practical implementation standpoint, it would be helpful to break down the above multi-class classification

problem into several two-class classifiers, similar to the modulation signal classification discussed in [8]. This way, the classification process can be made computationally efficient by using a small number of effective features for each two-class classifier. This is achieved by performing the classification in a hierarchical manner. In this chapter, a hierarchical classification approach is thus developed to gain computational efficiency for utilization in hearing improvement devices noting that a hierarchical approach avoids unnecessary computation of features and classifiers.

The rest of the chapter is organized as follows. Section 5.2 discusses the developed hierarchical classification approach. Then, in section 5.3, the details of the features used at each level of the hierarchy are mentioned. The experimental results appear in section 5.4. Finally, the conclusion is stated in section 5.5.

5.2 HIERARCHICAL SOUND SIGNALS CLASSIFICATION

The hierarchical approach to classification of sound signals has been limited to a few studies in the literature. In [9], a hierarchical rule-based approach was developed to classify audio signals from movies or TV programs. In [10], a hierarchical approach was proposed for recognition of environmental noise events, where input sound signals were initially classified into road vehicle/non-road vehicle noise. This was then followed by additional classifiers to separate the road vehicle noise into car, truck and motorbike noise classes, and the non-road vehicle noise into aircraft, train and industrial machinery noise classes. The features considered were MPEG-7, mel-frequency cepstral coefficients (MFCC) and the classifiers used consisted of a k-nearest neighbor, a neural network, and a Gaussian mixture model (GMM) classifier. In [11], audio signal classification for a movie video abstraction scheme was presented in three stages: (i) silence or environmental noise detection, (ii) speech and non-speech classification, and (iii) pure music/songs

and speech with background music classification. In [12], a hierarchical algorithm for classifying urban mechanical sound signals consisting of aircraft, motorcycle, car, crowd, thunder, wind, train, horn was covered where at the top level, sound signals were classified into two categories of mechanical and non-mechanical sounds, then at a lower level, the mechanical sound signals were classified into aircraft, motorcycle, car and train, and the non-mechanical sounds were classified into crowd, thunder, wind, and horn via GMM and HMM classifiers.

The focus of this work is on the development of a computationally efficient hierarchical classification approach for classifying non-quiet sound environments into music, speech, and background noise, where background noise signals are also classified into three types of noise: stationary (such as machinery), semi-stationary (such as driving car) and non-stationary (such as babble). Hence, basically a five class classification problem is addressed here in a hierarchical manner for the purpose of utilizing it in hearing improvement devices including hearing aids and cochlear implants.

Figure 5.1 shows a block diagram of the developed hierarchical classifier. An incoming sound signal is first checked to see whether the environmental condition is quiet or not. No further processing is done for the quiet condition. Non-quiet sound signals are passed through a classifier to separate speech activity from absence of speech. If speech activity is not detected, the signal is passed to the second level of the hierarchy to see whether it is music or background noise. If the signal is detected to be noise, at the third level of the hierarchy, it is classified into three different noise types having stationary, semi-stationary and non-stationary statistical characteristics.

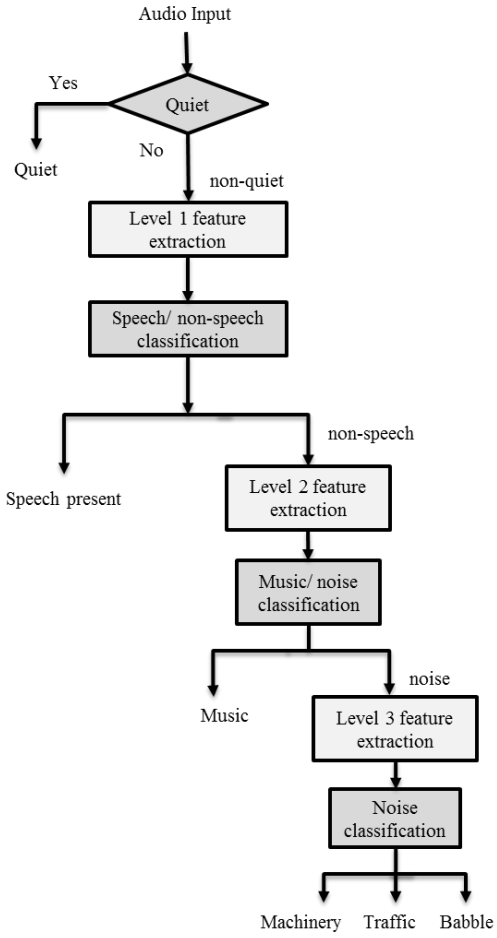

Figure 5.1. Hierarchical classification of sound signals for hearing improvement devices

It is important to note that despite the existing hierarchical approaches that compute all the features at the start of the classification, in this hierarchical approach, effective features are extracted as needed at the appropriate level of the hierarchy. This approach improves the computational efficiency of the classification. It should be noted that the features that are already computed at higher levels are also utilized at lower levels. In the next section, the features that have been found effective at different levels of the hierarchy are described.

5.3 FEATURES AT DIFFERENT LEVELS OF HIERARCHY

A list of the effective features used at different levels of the hierarchy is provided in Table 5.1.

5.3.1 Quiet/ Non-quiet Condition

As shown in Figure 5.1, an incoming sound signal is first seen to be quiet or non-quiet. This is achieved based on Sound Pressure Level (SPL). The louder a sound signal becomes, the greater the change in air pressure gets. SPL is computed as follows:

$$SPL = 20\log_{10}(\lambda/\lambda_{ref}) \tag{5.1}$$

where λ denotes the root mean square value of sound pressure and λ_{ref} the pressure of the lowest sound level that a user of a hearing improvement device can hear. Usually the SPL of quiet environments is less than 40dB. This value can be adjusted by the user depending on his/her perception level of a quiet environment. When the SPL level exceeds such a set amount, the sound

Table 5.1. List of features at different levels of the hierarchical classification approach

Classification levels	**Features**
Level 0: quiet/non-quiet	Sound Pressure Level (SPL)
Level 1: speech present/ speech absent	Band-Periodicity $BP_b, b = 1, \dots, B$, Subband Power Spectral Deviation $\text{SPSD}_b, b = 1$, Spectral Centroid SC, High Zero-Crossing Rate Ratio ($HZCRR$), Low Short-Time Energy Ratio ($LSTER$), and Spectrum Flux
Level 2: music/noise	Spectral Centroid SC, High Zero-Crossing Rate Ratio ($HZCRR$), Low Short-Time Energy Ratio ($LSTER$), Spectrum Flux Band-Periodicity $BP_b, b = 1$, Band-Periodicity Deviation $BPD_b, b = 1, \dots, B-1$ Subband Short-Time Energy Deviation $\text{STED}_b, b = 1$
Level 3: noise classifier	Band-Periodicity $BP_b, b = 1, \dots, B$ Band-Entropy $BE_b, b = 1, \dots, B$

signal is considered to be non-quiet and is passed to the first level of the hierarchy to detect any speech activity in it.

5.3.2 Speech/ Non-speech Activity Detection

Many works have been carried out in the literature to separate speech from non-speech signals, e.g. [13-16]. In [11], normalized root mean square (RMS) amplitude, normalized RMS variance, low short-time energy ratio, minimum value of RMS amplitude, mean value of autocorrelation coefficient, variance of log-energy, variance of differential-log energy, variance of spectral entropy, and the variance of second and third MFCCs were used for separating speech activity signals from other sound signals. In [17], spectral centroid, spectral flux, zero-crossing rate, 4Hz modulation energy (related to the syllable rate of speech), and the percentage of low-energy frames, were used to discriminate speech signals from various types of music. In [18], low short-time energy ratio, high zero-crossing rate ratio, spectrum flux were used to separate speech from non-speech sound signals. It is observed that zero-crossing rate (ZCR), short time energy, spectrum flux, root mean square (RMS), MFCC, and spectral centroid have been widely used for the purpose of detecting speech activity.

In this chapter, the features reported in the previous works were examined in terms of their discriminatory power towards detecting the presence of speech via the Fisher discriminant score:

$$J = trace({S_w}^{-1} S_I) \tag{5.2}$$

where S_w denotes the within-class scatter matrix and S_I the between-class scatter matrix. Higher J values indicate more separation in the multi-dimensional feature space. The features identified to have high J values for separating the presence and absence of speech are mentioned next.

In [18], the high zero-crossing rate ratio ($HZCRR$) feature was proposed for separating speech from non-speech sound signals. This feature captures the ratio of number of frames whose zero-crossing rate ratio is above the average value across a segment of the signal,

$$HZCRR = \frac{1}{2M}\sum_{m=1}^{M}[sgn(zcr(m) - 1.5\mu_{zcr}) + 1] \quad (5.3)$$

$$\mu_{zcr} = \frac{1}{M}\sum_{m=1}^{M} zcr(m) \quad (5.4)$$

where $zcr(m)$ denotes the zero-crossing rate of the m^{th} frame, M is the number of frames and μ_{zcr} is the average zcr in a duration of $\mathcal{S}$-seconds. It is worth noting that all features in this work are computed from signal segments of duration $\mathcal{S}$-seconds of M overlapping signal frames, with the m^{th} frame expressed as $F_m \coloneqq \{x_n | x_n \in \mathbb{R}, n = 1, \dots N\}$, where x_n represents the n^{th} sample in this frame of size N.

The low short-time energy ratio ($LSTER$) feature was discussed in [18] for separating speech from music signals. This feature reflects the ratio of the number of frames whose energy level is one half below the average energy of the signal over a $\mathcal{S}$-seconds signal segment, that is

$$LSTER = \frac{1}{2M}\sum_{m=1}^{M}[sgn\big(0.5\mu_{ER} - ER(m)\big) + 1] \quad (5.5)$$

where $ER(m)$ and μ_{ER} denote the energy of the m^{th} frame and the average energy in a $\mathcal{S}$-seconds signal segment, respectively. Figures 5.2 and 5.3 illustrate the distributions associated with the high zero-crossing rate ratio and the low short-time energy ratio features for the presence and absence of speech of a typical dataset. Figure 5.4(a) shows a scatter plot of the $HZCRR$ and $LSTER$ features for speech and noise signals of a typical dataset.

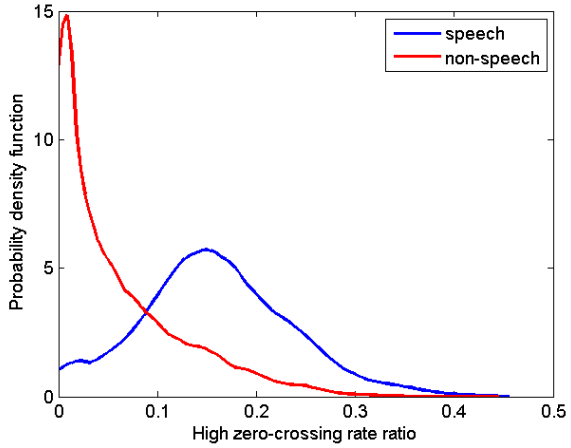

Figure 5.2. Distributions of the high zero-crossing rate ratio ($HZCRR$)feature for speech and non-speech sound signals of a typical dataset

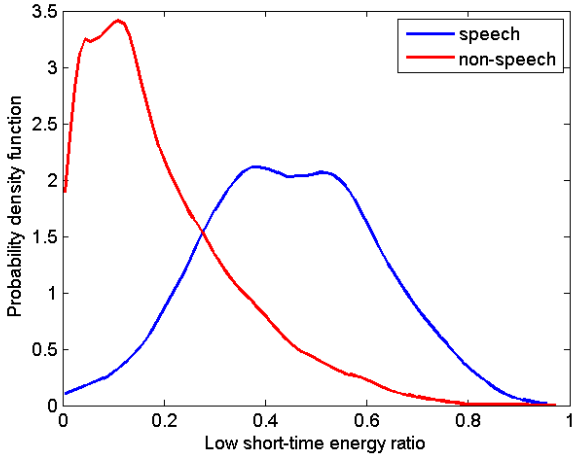

Figure 5.3. Distributions of the low short-time energy ratio ($LSTER$) feature for speech and non-speech sound signals of a typical dataset

Another feature which is used here is the difference between two adjacent bands in the average power spectral density of sound signals named subband power spectral deviation ($SPSD$). This feature was introduced in [19] for speech activity detection, and is computed as follows:

$$SPSD_b = \frac{1}{(K/B)}\left(\sum_{k=l_{b+1}}^{u_{b+1}} \widetilde{\omega}_{b+1}(k) - \sum_{k=l_b}^{u_b} \widetilde{\omega}_b(k)\right), b = 1, \dots, B-1 \quad (5.6)$$

$$\widetilde{\omega}_b = 10\log_{10}\sum_{m=1}^{M} \bar{P}_b, b = 1, \dots, B \quad (5.7)$$

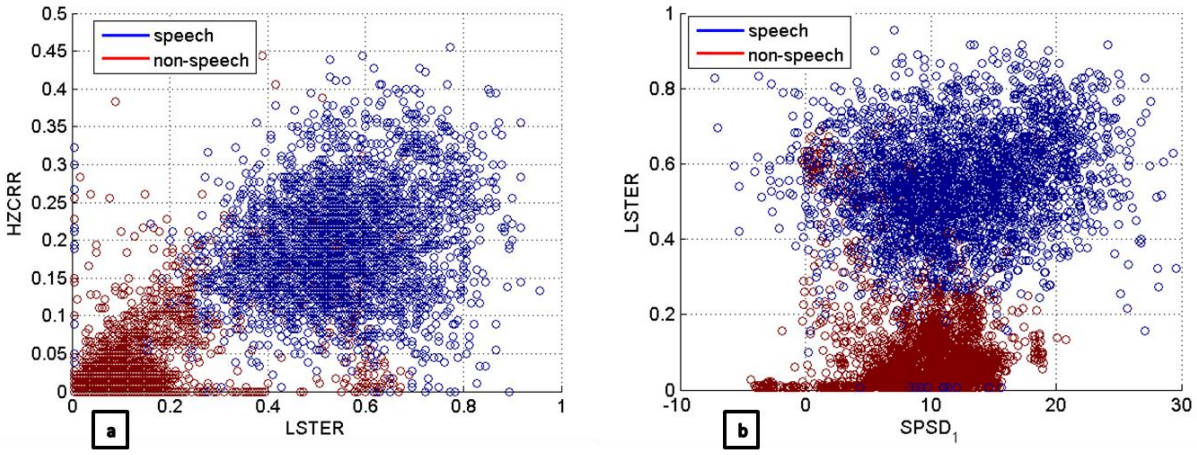

Figure 5.4. Scatter plots of (a) high zero-crossing rate ratio ($HZCRR$) and low short-time energy ratio ($LSTER$) features, (b) low short-time energy ratio ($LSTER$) and subband power spectral deviation1 ($SPSD_1$) features for speech and non-speech sound signals of a typical dataset

where $\widetilde{\omega}_b$ is the sum of power spectral density of frames, denoted by $\bar{P}_b$ over [t-$\mathcal{S}$,t], and l_b& u_b represent the lower and upper frequencies of band b. As mentioned in [19], the average difference in the power spectrum between the first and the second frequency bands $SPSD_1$ for speech signals is more noticeable compared to the other sounds. Thus, in this work, only $SPSD_1$ is used. Adding this feature increases the Fisher discriminant score. Figure 5.4(b) shows the scatter plot of the $SPSD_1$ and $LSTER$ features of a typical dataset.

In addition to these features, spectral centroid, spectrum flux, and band-periodicity are considered in this work to provide more discriminatory power between the presence and absence of speech. By adding these features to the aforementioned ones, a feature vector with higher Fisher discriminant score is achieved.

Spectrum flux defines the difference between spectra of adjacent frames and is widely used in speech activity detection applications. This feature is expressed as

$$SF = \frac{1}{K \times M} \times \sum_{m=1}^{M} \sum_{k=1}^{K} [A(m,k) - A(m-1,k)]^2 \quad (5.8)$$

where $A(m,k)$ is the spectrum of the m^{th} frame at bin k. In this work, similar to [18], the average value of spectrum flux over $\mathcal{S}$-seconds of sound signals is considered.

Spectral centroid is also a widely used feature for classifying different sound activities. This feature is expressed as

$$SC(m) = \frac{\sum_{k=1}^{K} kA(m,k)}{\sum_{k=1}^{K} A(m,k)} \tag{5.9}$$

where $A(m,k)$ is the spectrum of the m^{th} frame at bin k. Here, the average spectrum centroid over $\mathcal{S}$-seconds of sound signals is considered.

In [18], so called band-periodicity features were used to separate music from other environmental sounds. These features along with a Random Forest (RF) classifier were shown to provide high discriminatory power towards distinguishing speech sound signals from noise sound signals [19]. These features are briefly explained below.

Assuming the sampling rate is fs, the frequency range $[0,\frac{fs}{2}]$ is divided into a number of *B* linear non-overlapping subbands. To compute the band-periodicity features, the cross-correlation between every two adjacent frames in each band is computed and then the peak value of the cross-correlation denoted by $\rho_{b,m}$ is used to define the band-periodicity features in band *b* as follows [20-21]:

$$BP_b = \frac{1}{M}\sum_{m=1}^{M} \rho_{b,m}\,, b = 1 \ldots, B \tag{5.10}$$

5.3.3 Music/ Noise Separation

Classifying or separating music from noise signals such as machinery noise is a challenging task since such signals exhibit similar periodicity characteristics. Although there are many papers in

the literature that have addressed the separation of speech from music, there are relatively limited works on separating music from noise [22]. In [23], short-time energy (STE), spectral flatness and MFCC were used for this purpose. In [18], the band-periodicity, spectrum flux, and noise frame ratio features were used to distinguish between music and environmental noise signals.

In the developed hierarchical approach, the music/noise classification is activated only if the sound signal at the previous level is detected as a non-speech sound signal by using these features: SF, SC, $HZCRR$, $LSTER$, $SPSD_1$ and $BP_b, b = 1, \dots, B$. As noted earlier, the features which are extracted at higher levels are also used at lower levels. Our analysis has shown that $SPSD_1$, which is extracted in the speech/non-speech classification level, does not have any noticeable impact on the music/noise classification outcome. Also, our analysis has revealed that the band-periodicity of the first band provides the highest discriminatory power between the two classes compared to the other bands. Figure 5.5 shows the distributions of the band-periodicity feature in the first band for music and noise signals of a typical dataset.

The difference between the first two band-periodicity features, that is

$$BPD_b = BP_{b+1} - BP_b, b = 1, \dots, B - 1 \tag{5.11}$$

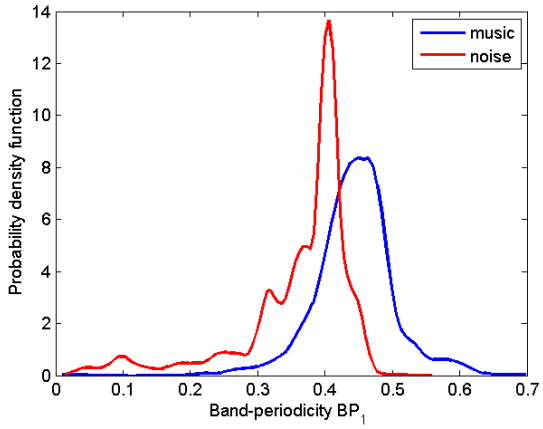

Figure 5.5. Distributions of the band-periodicity BP_1 feature for music and noise sound signals of a typical dataset.

is also used at this level. Another feature used at this level is the subband short-time energy deviation $STED$. In [19], this feature was used to separate speech from noise signals. Assuming that in a sustained noise environment, the level of background noise remains more or less constant, the energy level fluctuates considerably for music signals in particular when the music involves singing. In other words, on average, the deviation in the energy level between the highest and the lowest energy level for music is higher than that for pure noise. It is understood that there are exceptions to this assumption but in general this assumption holds in many practical situations. The deviation in the energy level of the input sound signal $STED$ is computed in different frequency bands as follows:

$$STED_b = \frac{\mu_b - \gamma_b}{\mu_b}, b = 1, \dots, B \quad (5.12)$$

where μ_b and γ_b denote the average and the minimum energy of M frames over [t-$\mathcal{S}$,t] in band b. Here, the average value is considered instead of the maximum value in order to capture noise which is considered to be of a sustained nature and to avoid capturing any transient noise. The difference between the average and the minimum value for sustained noise is expected to be lower, while in music it is expected to be higher. Figure 5.6 shows the distributions of this feature for machinery noise and music of a typical dataset. From this figure, one can see that the distribution for music signals appears to the right of the distribution for pure noise signals.

5.3.4 Noise Type Classification

Noise classifier is activated at the lowest level of the hierarchy if noise activity is detected at the music/noise level. At this level, the noise signal is classified into three different classes labeled stationary, semi-stationary and non-stationary based on their statistical characteristics. Here, the

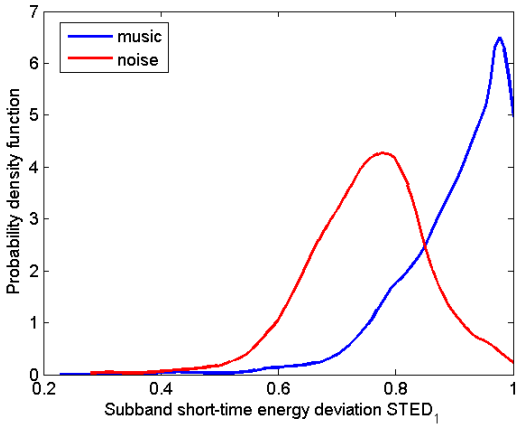

Figure 5.6. Distributions of the subband short-time energy deviation $STED_1$ for music and noise sound signals of a typical dataset.

noise classifier developed in [20] is used based on the band-periodicity and band-entropy features. Similar to the band-periodicity features, the band-entropy features are computed as follows:

$$BE_b = \frac{1}{M}\sum_{m=1}^{M} H_{b,m}\,, \ldots\ldots, B \tag{5.13}$$

where $H_{b,m}$ denotes the entropy of the m^{th} frame in band b.

5.3.5 Random Forest Tree Classifier

At each level of the hierarchy, a random forest tree classifier is used to perform the classification. Tree classifiers are widely used in machine learning applications and are suited for real-time implementation due to their computational efficiency during testing or operation. In [24], the method of random forest (RF), which is an ensemble of randomly trained decision trees, was shown to provide higher accuracy than other commonly used classifiers. An RF classifier is a combination of a number of classification trees. Each tree is trained independently from other trees via a randomly selected (with replacement) subset of training data. A brief overview of training and testing a tree is stated below. The interested reader is referred to [24] for more details on RF classifiers.

A tree is a set of nodes and branches which is structured in a hierarchical manner. Nodes are either internal nodes or terminal nodes. All nodes have one input (incoming branch) and only two outgoing branches. Training a tree starts at the root node, where all the training samples from all the classes are included at this level which correspond to the situation with the information content being low or entropy being high. Then, the tree is grown in such a way that the amount of entropy is decreased at each level by maximizing an information gain objective function at each split node, where entropy 0 at a node means that all the data at that node are from one class. After training, classification decision is made based on the most voted class over all the trees. A test sample is pushed through all the trees simultaneously until it reaches the leaves.

5.4 EXPERIMENTAL RESULTS AND DISCUSSION

To examine the effectiveness of the developed hierarchical classification approach, a dataset consisting of five classes of music, speech, machinery noise, driving car noise, and babble noise was considered. Sound files were collected at different RMS levels from these sound environments. Each sound class contained 120 sound files of duration 30 seconds that were collected at a sampling rate of 16kHz. The speech class contained clean speech and noisy speech that was created by adding the background noise signals at different signal to noise ratios (SNRs). One half of the dataset was chosen randomly for training and the other half for testing. Input sound signals were captured using a frame size of 11ms with 5.5ms overlap between consecutive frames. The segment length $\mathcal{S}$ for feature extraction was considered to be 1 second. Sample sound files of the dataset examined can be downloaded and listened to from this link http://www.utdallas.edu/~kehtar/SampleSoundFiles.rar.

For the subband feature extraction, the use of more than 4 bands or $B = 4$ did not make much difference in the outcome. Thus, to gain computational efficiency, only 4 bands were considered. For the RF classification, different numbers of trees were considered and it was found that a random forest of size more than 10 did not make much difference in the classification outcome. Thus, the number of trees of the RF classification was set to 10.

In our first study, the developed hierarchical approach was compared with the one-step classification approach, that is by extracting all the features together and the classification was conducted by separating the classes in one step. The results of this comparison averaged over 100 different training and testing cases (no overlap between training and testing samples in each case) and also over different RMS levels are provided in Tables 5.2 and 5.3. The classification results at each level of the hierarchy are also shown in these tables. As seen from these tables, the hierarchical classification outperformed the one-step classification by 13% over all the classes; more specifically, by 10% in the classification of music, by 27% in the classification of stationary noise type, and by 30% in the classification of non-stationary noise type. It can also be observed that at the first level of the hierarchy, the presence of speech was achieved 97% of the time. At the second level of the hierarchy, music and noise were separated 88% of the time and at the third level of the hierarchy, the noise type was identified 99% of the time.

Another study was conducted to assess the behavior of the developed hierarchical classification in the presence of other sound signals for which no training had been done. A dataset consisting of non-trained sound files were collected and used for testing. Sample sound files of this testing dataset can be downloaded and listened to from this link http://www.utdallas.edu/~kehtar/SampleSoundFiles.rar.

Table 5.2. Confusion matrix (percentages) of one-step classification averaged over 100 training/testing cases with an overall classification rate of 79%

	Speech present	Music	Stationary noise type	Semi-stationary noise type	Non-stationary noise type
Speech present	**95**	1	0.2	0.2	3.6
Music	18.9	**70**	1.7	1.7	7.7
Stationary noise type	0.8	0.3	**73.5**	9	16.4
Semi-stationary noise type	1.8	0.5	1.4	**95.3**	1
Non-stationary noise type	16.4	2.2	11.4	8.4	**61.4**

Table 5.3. Confusion matrix (percentages) of hierarchical classification averaged over 100 training/testing cases with an overall classification rate of 92.6%

	Speech present	Music	Stationary noise type	Semi-stationary noise type	Non-stationary noise type
Speech present	**96**	1.3	0.6	0	2.1
Music	6	**80**	4	4.7	5.3
Stationary noise type	0.3	0.5	**99**	0.1	0.1
Semi-stationary noise type	0.1	2.5	0.1	**97.3**	0
Non-stationary noise type	3.5	4.5	0	1	**91**

Level 1	Speech present	Non-speech
Speech present	**96**	4
Non-speech	2	**98**

Level 2	Music	Noise
Music	**81**	19
Noise	4	**96**

Level 3	Stationary noise type	Semi-stationary noise type	Non-stationary noise type
Stationary noise type	**100**	0	0
Semi-stationary noise type	0.3	**99.5**	0.2
Non-stationary noise type	0	0.9	**99.1**

The confusion matrices of this study appear in Tables 5.4 and 5.5. As can be seen from these tables, nearly 50% improvement in the overall classification rate was obtained when using the hierarchical approach as compared to the conventional one-step approach.

As far as the computational efficiency aspect is concerned, the processing time of the one-step classification was found to be 2.3ms on a laptop with a 2.4GHz processor for 11ms signal frames

Table 5.4. Classification percentages of non-trained sound environments when using the one-step approach

	Music	Speech present	Stationary noise type	Semi-stationary noise type	Non-stationary noise type
live bar music	64.9	16.7	2.3	15.7	0.4
loud car engine	1.1	1.5	1.3	96.2	0
driving car	0	0	0	99	1
outdoor ac	0	0	100	0	0
train	9.8	50.8	4.9	31.1	3.3
vacuum	4.4	22.7	62.5	1.9	8.4
hair dryer	3.1	17.8	68.3	8.5	0.7
mall	4.8	92.9	0	1.8	0.5
church	2.4	1.2	0	96.4	0
Speech (HINT sentences)	9.1	81.8	0	0	9.1
restaurant	0.4	3.3	0	35	61.3

Table 5.5. Classification percentages of non-trained sound environments when using the hierarchical approach

	Music	Speech present	Stationary noise type	Semi-stationary noise type	Non-stationary noise type
live bar music	98.5	0	1.5	0	0
loud car engine	1.1	0	0.4	98.5	0
driving car	0	0	0	100	0
outdoor ac	0	0	97.7	2.3	0
train	57.4	0	8.2	26.2	8.2
vacuum	2.7	0	97.3	0	0
hair dryer	0	1.1	79.8	15.2	2.8
mall	3.6	91.6	0	0	4.8
church	84.3	15.7	0	0	0
Speech (HINT sentences)	0	100	0	0	0
restaurant	2.1	1.7	3.8	13.3	79.2

captured by its sound card. However, when using the hierarchical classification, the average processing time across the three levels was reduced to 1.8ms.

5.5 CONCLUSION

A hierarchical classification approach to distinguish environmental sound signals has been developed in this chapter. The computational efficiency of this classification approach makes it suited for deployment in hearing improvement devices. Furthermore, it has been shown that by processing input signal frames via the hierarchical classification developed in this work, higher detection rates with higher robustness to non-trained sound signals are acquired compared with the conventional one-step classification.

5.6 REFERENCES

[1] V. Gopalakrishna, N. Kehtarnavaz, T. Mirzahasanloo, and P. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 6, pp. 1691-700, 2012.

[2] I. Panahi, N. Kehtarnavaz, and L. Thibodeau, "Smartphone-based noise adaptive speech enhancement for hearing aid applications," *Proceedings of the 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 85-88, Orlando, August 2016.

[3] T. Mirzahasanloo, N. Kehtarnavaz, V. Gopalakrishna, and P.Loizou. "Environment-adaptive speech enhancement for bilateral cochlear implants using a single processor," *Speech Communication*, vol. 55, no. 4, pp. 523-534, 2013.

[4] F. Saki, T. Mirzahasanloo, and N. Kehtarnavaz, "A multi-band environment-adaptive approach to noise suppression for cochlear implants," *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'14)*, pp. 1699-1702, Chicago, August, 2014.

[5] R. Yu, Y. Hao, I. Panahi, and N. Kehtarnavaz, "Smartphone-based real-time speech enhancement for improving hearing aids speech perception," *Proceedings of the 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 5885-5888, Orlando, August 2016.

[6] Cochlear Ltd- http://www.cochlear.com/wps/wcm/connect/us/home/treatment-options-for-hearing-loss/cochlear-implants/nucleus-6-features#smartsound.

[7] Phonak- http://www.phonak.com/com/b2c/en/products/hearing_instruments/bolero-v/hearing-aid.html

[8] N. Kim, N. Kehtarnavaz, M.B. Yeary, and S. Thornton, "DSP-based hierarchical neural network modulation signal classification," *IEEE Transactions on Neural Network*s, vol. 14, no. 5, pp.1065-1071, 2003.

[9] T. Zhang , C. J. Kuo "Audio content analysis for online audiovisual data segmentation and classification," *IEEE Transactions on speech and audio processing*, vol. 9, no. 4 pp. 441-457, 2001.

[10] X. Valero and F. Alías, "Hierarchical Classification of environmental noise sources considering the acoustic signature of vehicle pass-bys," *Archives of Acoustics*, vol. 37, no. 4, pp. 423–434, 2012.

[11] P. Krishnamoorthy and S. Kumar, "Hierarchical audio content classification system using an optimal feature selection algorithm," *Multimedia Tools and Applications,* vol. 54, no. 2, pp. 415-444, 2011.

[12] S. Ntalampiras, I. Potamitis, and N. Fakotakis, "Automatic recognition of urban environmental sound events," *Proceedings of International Association for Pattern Recognition Workshop on Cognitive Information Processing,* pp. 110–113, 2008.

[13] N. Mesgarani, M. Slaney, and S. a. Shamma, "Discrimination of speech from nonspeech based on multiscale spectro-temporal modulations," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 14, no. 3, pp. 920–930, 2006.

[14] E. Alexandre, L. Cuadra, L. Alvarez, M. Zurera, and F. Ferreras, "Automatic sound classification for improving speech intelligibility in hearing aids using a layered structure," *International Conference on Intelligent Data Engineering and Automated Learning,* pp. 306-313, Springer Berlin Heidelberg, 2006.

[15] P. Ruvolo, I. Fasel, and J. Movellan, "A learning approach to hierarchical feature selection and aggregation for audio classification," *Pattern Recognition Letter,* vol. 31, pp. 1535-1542, 2010.

[16] J. Burred and A. Learch, "Hierarchical automatic audio signal classification," *Journal of Audio Engineering Society,* vol. 52, pp. 724-739, 2004.

[17] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing,* pp.1331-1334, 1997.

[18] L. Lu, H. Zhang and H. Jiang, "Content analysis for audio classification and segmentation," *IEEE Transactions on Speech and Audio Processing,* vol. 10, no. 7, pp. 504-516, 2002.

[19] F. Saki, N. Kehtarnavaz, "Automatic switching between noise classification and speech enhancement for hearing aid devices," *Proceedings of the 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC),* pp. 736-739,Orlando, August 2016.

[20] F. Saki and N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

[21] F. Saki, A. Sehgal, I. Panahi and N. Kehtarnavaz, "Smartphone-based real-time classification of noise signals using subband features and random forest classifier," *Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 2204-2208, Shanghai, China, March 2016.

[22] F. Alías, J. Socoró, and X. Sevillano, "A review of physical and perceptual feature extraction techniques for speech, music and environmental sounds*," Applied Sciences,* vol. 6, no. 5, p. 143, 2016.

[23] E. Alexandre, R. Gil-Pita, L. Cuadra, L. Álvarez, and M. Rosa-Zurera, "Speech/music/noise classification in hearing aids using a two-layer classification system with MSE linear discriminants," *Proceedings of the 16th European Signal Processing* Conference, pp. 1-5, 2008.

[24] L. Breiman, "Random forest" *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001.

CHAPTER 6

A MULTI-BAND ENVIRONMENT-ADAPTIVE APPROACH TO NOISE SUPPRESSION FOR COCHLEAR IMPLANTS*

Authors - Fatemeh Saki, Taher Mirzahasanloo, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents an improved environment-adaptive noise suppression solution for the cochlear implants speech processing pipeline. This improvement is achieved by using a multi-band data-driven approach in place of a previously developed single-band data-driven approach. Seven commonly encountered noisy environments of street, car, restaurant, mall, bus, pub and train are considered to quantify the improvement. The results obtained indicate about 10% improvement in speech quality measures.

6.1 INTRODUCTION

Cochlear Implants (CIs) are surgically implanted devices that enable hearing sensation in profoundly deaf people. It is known that speech understanding by CI patients drops significantly in noisy environments. The literature includes many studies, e.g. [1, 2], where noise suppression is achieved by treating all noise types as noise with no distinction in the characteristics of the noise in a particular environment.

In the previous works conducted by our research team [3-6], a more effective noise suppression in terms of speech quality was developed by automatically adapting to different noise types. In addition, the real-time implementation of our environment-adaptive speech enhancement was provided as part of the CI speech processing pipeline on the FDA-approved PDA (Personal Digital Assistant) research platform. In these works, the adaptive-environment aspect was achieved by utilizing a number of gain tables for different noise environments based on the data-driven approach in [7]. In other words, for each noisy environment, a gain table discretized over a range of priori and posteriori SNRs was obtained. This table was built without distinguishing among different frequency bands.

Noting that the spectrum of real-world noise signals varies depending on different frequency bands, this chapter provides a multi-band environment-adaptive speech enhancement approach. In this approach, a number of gain tables were trained for different frequency bands. It is shown that this multi-band approach generates improved results over the previously developed single-band approach.

The rest of the chapter is organized as follows: section 6.2 provides an overview of the previously developed environment-adaptive speech processing pipeline of CIs. The new multi-band approach

is then presented in section 6.3 followed by the experimental results in section 6.4. Finally, the conclusion is stated in section 6.5.

6.2 OVERVIEW OF PREVIOUSLY DEVELOPED ENVIRONMENT-ADAPTIVE NOISE SUPPRESSION PIPELINE

Figure 6.1 shows a block diagram of the environment-adaptive pipeline for cochlear implants that was previously developed in [3]. This environment-adaptive CI speech processing pipeline is briefly mentioned here to set the stage for the understanding of the multi-band approach. The pipeline consists of two parallel paths running in real-time: speech processing path, and noise detection/classification path. The noise detection/classification path uses a Voice Activity Detector (VAD) to determine if a current signal frame is speech+noise or pure noise. If it is found to be pure noise, mel-frequency cepstrum (MFCC) or sub-band features are extracted and fed into a trained Gaussian Mixture Model (GMM) or Random Forest (RF) classifier to determine the noise type [8]. The speech processing path includes a parameterized noise suppression component whose parameters get automatically used based on the noise class determined by the classification path.

6.2.1 Data-driven Noise Suppression

To achieve speech enhancement by noise suppression, a gain function is used to map the magnitude spectrum of the input noisy speech signal to an estimate of the associated clean spectrum according to

$$\hat{A}_k(n) = \tilde{G}\big(\xi_k(n), \gamma_k(n)\big) R_k(n) \tag{6.1}$$

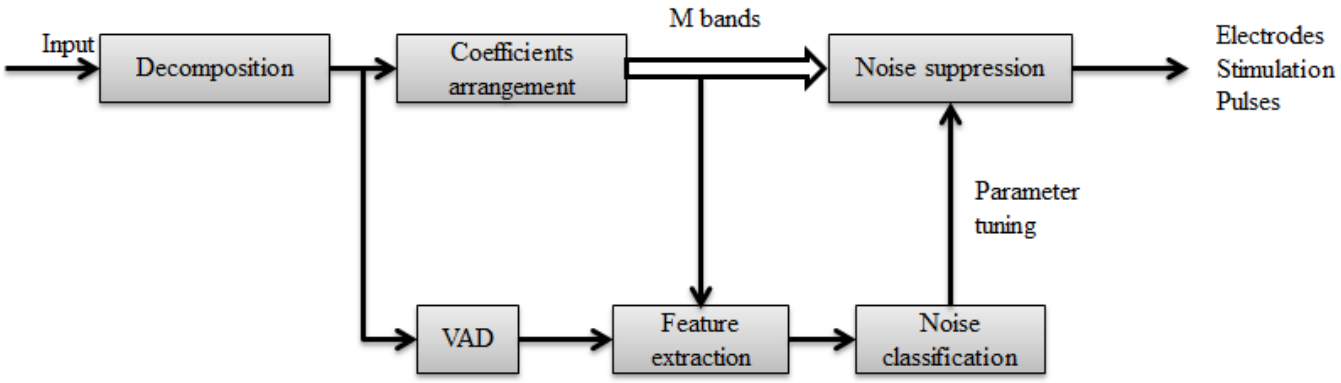

Figure 6.1. Cochlear implant speech processing pipeline implemented in real-time [3]

$$\xi_k(n) = \frac{\lambda_x(k,n)}{\lambda_d(k,n)} \tag{6.2}$$

$$\gamma_k(n) = \frac{R_k^2(k,n)}{\lambda_d(k,n)} \tag{6.3}$$

where $\hat{A}_k(n)$and $R_k(n)$ are the estimated clean spectral and noisy amplitudes in the frequency bin k for the time frame n, respectively, $\tilde{G}$ denotes the optimized gain function, and ξ_k& $\gamma_k$represent the priori and posteriori SNRs, respectively. To compute these SNRs, estimations of the clean spectral variance $\lambda_x(k)$and noise spectral variance $\lambda_d(k)$ are needed. The so called decision-directed estimator involves the use of the following rule to update the priori SNR for each frame n [9]:

$$\hat{\xi}_k(n) = \max\left[\alpha \frac{\hat{A}_k^2(n-1)}{\lambda_d(k,n)} + (1-\alpha)[\gamma_k(n) - 1], \xi_{min}\right] \tag{6.4}$$

where α is a weight close to one and ξ_{min} is a lower bound on the estimated value of $\hat{\xi}_k(n)$. In this work, the estimator and noise tracking discussed in [7] are utilized.

According to the estimated priori and posteriori SNRs, the spectral amplitude of the enhanced (clean) signal is estimated from the noisy signal based on an assumed probability density function and the optimization of an objective function. The objective function can involve MMSE, log

MMSE, maximum a posteriori (MAP) estimation methods [10] or involve more recent data-driven methods [7]. In the data-driven methods, no estimation of the spectral variance is required. A brief explanation of the data-driven approach is provided next.

Let $X$and $\hat{X}$ be the clean and enhanced signals. In the data-driven approach, the aim is to find the function $\tilde{G}(\xi_k, \gamma_k)$ so that by applying it to the noisy signal, the estimated clean signal gets close to the clean signal. In other words, the average distortion $D(X, \hat{X})$ between clean and enhanced signals for (ξ_k, γ_k) pairs is minimized. This distortion can be any of the following: Weighted-Euclidean (WE), Log-Euclidean (LE), Weighted-Cosh (WC) or simple mean-square error (MSE) [7, 11]. Mathematically, the following equations describe the data-driven approach:

$$\tilde{G} = \{\tilde{G}_{ij}, \forall_i = 1, \dots, I, \forall_j = 1, \dots, J\} \tag{6.5}$$

$$\tilde{G}_{i,j} = arg \min_{G_{i,j}} D(X, \hat{X}) \tag{6.6}$$

where $\tilde{G}$ is a look-up table discretized over a grid of priori and posteriori SNRs. A parameter cell contains the closest values of $\hat{\xi}$ and γ to a grid point with values $\tilde{G}_{ij}$ stored in matrix $\tilde{G}$. Thus, for a total of I and J priori and posteriori SNRs, respectively, the gain table consists of an $I \times J$ matrix containing the noise suppression parameters.

6.3 MULTI-BAND DATA-DRIVEN NOISE SUPPRESSION

In the data-driven method discussed in [7], for each frame and each frequency bin, there is an $(\hat{\xi}_k, \gamma_k)$pair that falls into one of the parameter cells of the gain table. As a result, an$(\hat{\xi}_k, \gamma_k)$pair from different frequency bins and different frames may fall into the same parameter cell during the training of $\tilde{G}(\hat{\xi}_k, \gamma_k)$involving a clean amplitude A_k and a noisy amplitudeR_k.

In the multi-band data-driven approach introduced here, the signal is divided into M different non-overlapping frequency bands. Then, M different gain tables corresponding to M frequency bands are trained. The frequency band decomposition can be done in Fourier domain or by using a filter bank. In each frame for a frequency band, the priori and posteriori SNRs $(\hat{\xi}_{bk}, \gamma_{bk})$, with b denoting the band index, are computed. Therefore, the parameterized suppression values for each frequency band get trained separately. It is worth mentioning that the size of the gain tables is kept the same considering that each gain table covers the same prior and posterior SNR ranges. Hence, there would be M gain tables for each environment, that is:

$$\tilde{G}_b = \{\tilde{G}_{bij}, \forall_i = 1, \dots, I, \forall_j = 1, \dots, J\}, \mathrm{b} = 1, \dots, M \quad (6.7)$$

As mentioned earlier, in the single-band noise suppression, $(\hat{\xi}_k, \gamma_k)$pairs from different frequency bands and different frames might fall in the same cell of the gain table. This means that the corresponding suppression value $\tilde{G}(\hat{\xi}_k, \gamma_k)$ for an input frame is only a function of the estimated priori and posteriori SNRs, and thus the frames from different frequencies are treated the same. This causes some distortion in the signal. By separating the gain tables based on the frequency bands, any such distortion can be avoided. Here it is worth pointing out that the data driven suppression is performed independently in each band. As reported in [3], the suppression processing time takes only 2.4 ms out of a total processing time of 8.41 ms on the PDA platform for 11.6 ms frames. Hence, the two-band suppression processing time is still expected to run in real-time on the PDA platform.

6.4 EXPERIMENTAL RESULTS AND DISCUSSION

The introduced multi-band noise suppression was evaluated by using seven commonly encountered noise types of street, car, restaurant, mall, bus, pub and train. Noise samples were collected using the same BTE (Behind-The-Ear) microphone worn by Nucleus ESPrit cochlear implant users at a sampling frequency of 8000 Hz. For training, the first 50 IEEE sentences provided in [12] (approximately 2-3s long) were used to serve as clean speech files. For each noisy environment, 50% of the noise files were added to each speech signal at several SNRs from -12.5 to 27.5 dB in steps of 5 dB to generate the training dataset. The signals were windowed into 25-ms frames via a Hamming window with 50% overlap across two non-overlapping low and high frequency bands. In the experiments reported in this work, $\alpha$and ξ_{min}were set to 0.98 and -19 dB, respectively, the prior SNR was discretized from -19 dB to 40 dB and the posterior SNR from -30dB to 40 dB in steps of 1 dB with a grid size of 60×71. It was found that the use of two bands maintained the real-time throughput of the pipeline.

The speech quality measures of Perceptual Evaluation of Speech Quality (PESQ) and Log-Likelihood Ratio (LLR) [10] were computed to provide a quantification of the improvement in the noise suppressed output signals. Figure 6.2 shows the comparison of the PESQ and the LLR measures for the multi-band and single-band approaches for 0 dB SNR. The non-suppressed noisy signals are shown to serve as the baseline. The results reflect the averages on the second half of the 50 IEEE sentences which had not been used in the training dataset. This figure illustrates that the multi-band approach provides an improvement of nearly 10% in speech quality measures averaged across the noisy environments considered compared to the single-band approach. An Analysis of Variance (ANOVA) was conducted to show the statistical significance of the

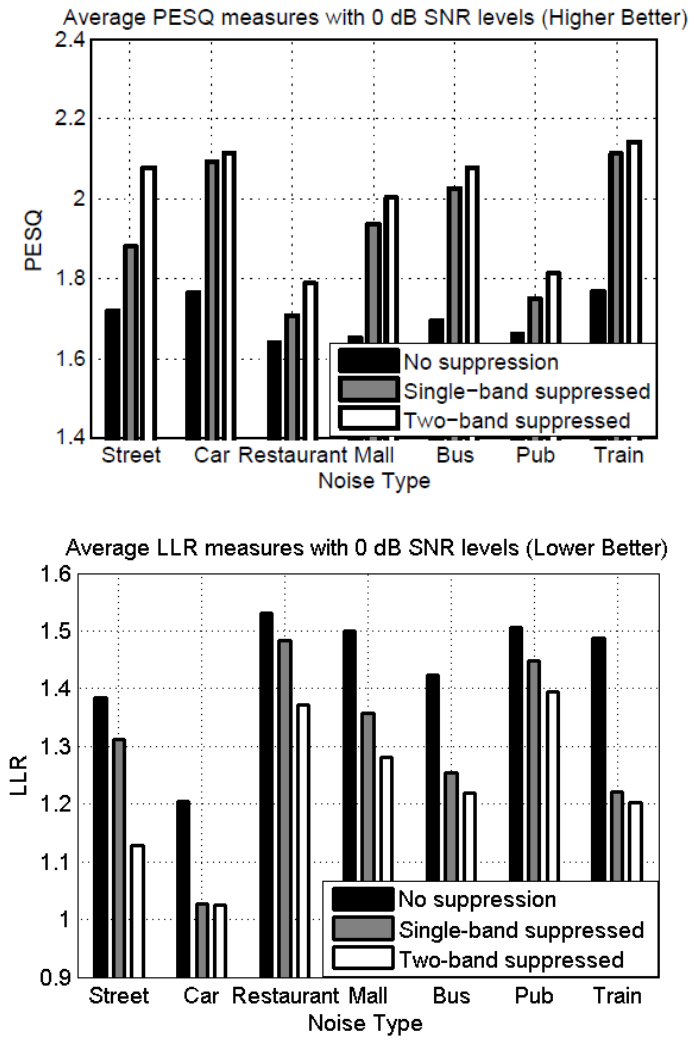

Figure 6.2. Bar charts showing the performance of the single-band data-driven adaptive noise suppression, two-band data-driven adaptive noise suppression and no-noise suppression in terms of the speech quality measures of Perceptual Evaluation of Speech Quality (PESQ) and Log-Likelihood Ratio (LLR)

improvement ($p < 0.001$). In our noise dataset, the files for train and car noises had approximately uniform spectrum over all the frequency bands. That is why the improvement did not generate statistically significant improvement over the single-band approach for these two noise types while for the other noise types the improvement was found to be statistically significant.

Another experiment was carried out to examine the performance of the multi-band approach in the presence of other noise types which had not been considered in the original set of environments.

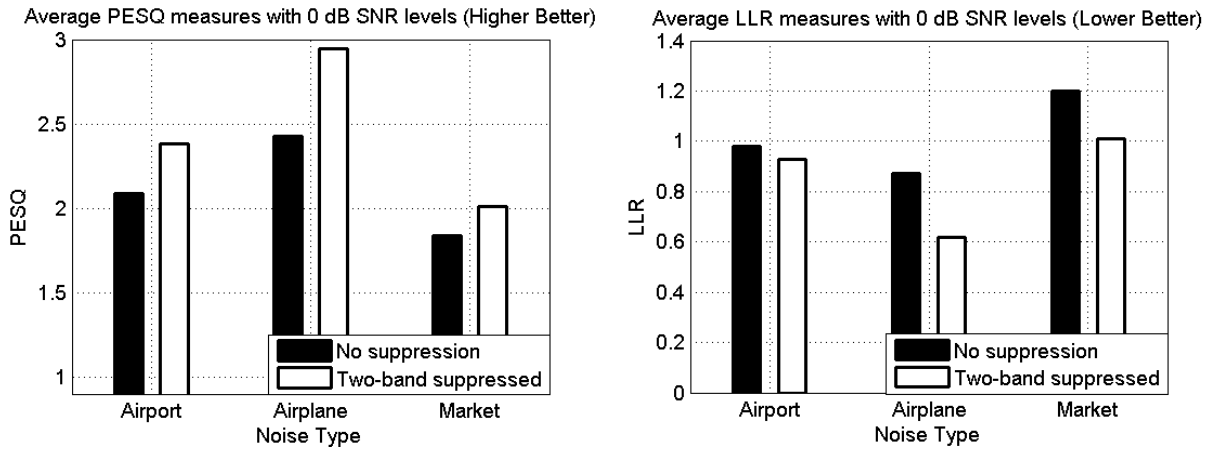

Figure 6.3. Comparison of PESQ and LLR quality measures when encountering unknown noise.

Figure 6.3 shows a comparison of the PESQ and the LLR measures exhibiting the outcome for the multi-band data-driven approach versus the noisy non-processed signals for three noise environments of airport, airplane and market. These three environments in the classification path were placed into the closest class, namely street, bus and restaurant, respectively. Consequently, the suppression parameters of these detected classes were used for the noise suppression. A visual comparison can be made in Figure 6.4 where the spectrogram of the clean, noisy, single-band data-driven suppressed and multi-band data-driven suppressed signals at 0 dB SNR are shown. This figure shows the background noise was suppressed by the developed multi-band method more than the single band method, thus retrieving the speech signal more accurately.

6.5 CONCLUSION

A modification to the previously developed noise suppression path of the environment-adaptive speech processing pipeline of cochlear implants was introduced in this chapter to improve speech enhancement via noise suppression. This modification involved the use of multiple frequency bands instead of a single band to achieve data-driven environment-adaptive noise suppression. The

experimental results showed 10% improvement in speech quality measures for seven noisy environments considered while at the same time maintaining the real-time throughput of the entire speech processing pipeline.

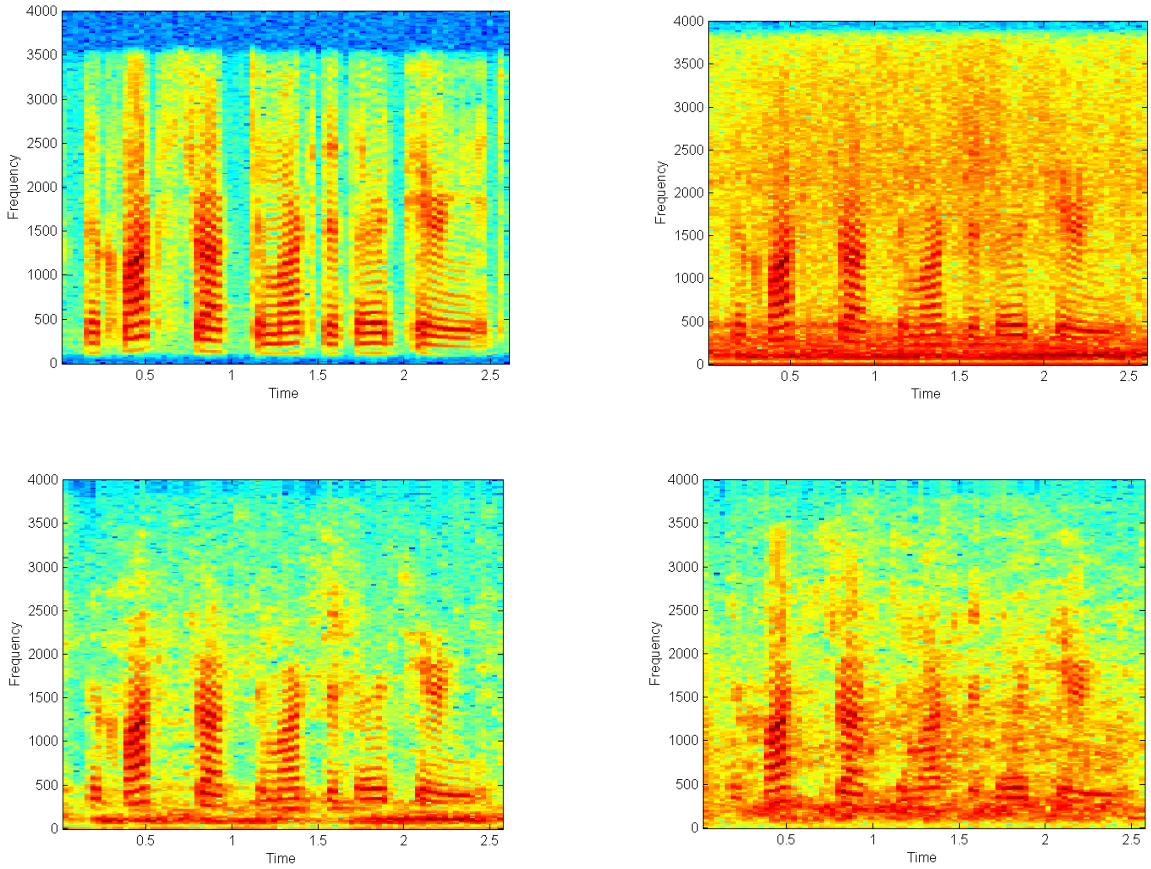

Figure 6.4. Spectrograms of the clean speech (top left) and noisy signals (top right) (SNR = 0 dB). Bottom left figure shows enhanced signals by the introduced two-band noise suppression approach and the bottom right one shows the single-band noise suppression approach; IEEE sentence: "*The clock struck to mark the third period*".

6.6 REFERENCES

[1] Y. Hu, P. Loizou, N. Li, and K. Kasturi, "Use of a sigmoidal-shaped function for noise attenuation in cochlear implants," *The Journal of the Acoustical Society of America*, 122, pp. 128-134, 2007.

[2] P. Loizou, A. Lobo, and Y. Hu, "Subspace algorithms for noise reduction in cochlear implants," *The Journal of the Acoustical Society of America*, 118, pp. 2791-2793, 2005.

[3] V. Gopalakrishna, N. Kehtarnavaz, T. Mirzahasanloo, and P. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, pp. 1691-1700, 2012.

[4] T. Mirzahasanloo, N. Kehtarnavaz, V. Gopalakrishna, P. Loizou, "Environment-adaptive speech enhancement for bilateral cochlear implants using a single processor," *Speech Communication*, vol. 55, no. 4, pp. 523-534, 2013.

[5] T. Mirzahasanloo, V. Gopalakrishna, N. Kehtarnavaz, and P. Loizou, "Adding real-time noise suppression capability to the cochlear implant PDA research platform," *Proceedings of the 34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2271-2274, 2012.

[6] V. Gopalakrishna, N. Kehtarnavaz, P. Loizou, and I. Panahi, "Real-time automatic switching between noise suppression algorithms for deployment in cochlear implants," *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 863- 866, 2010.

[7] J. Erkelens, J. Jensen, and R. Heusdens, "A data-driven approach to optimizing spectral speech enhancement methods for various error criteria," *Speech Communication*, vol. 49, pp. 530-541, 2007.

[8] F. Saki and N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

[9] Y. Ephraim, and D. Malah, "Speech enhancement using a minimum mean-square error-log-spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 33, no. 2, pp. 443-445, 1985.

[10] P. Loizou, *Speech Enhancement: Theory and Practice.* Boca Rotan, FL: CRC, Taylor and Francis, 2007.

[11] T. Mirzahasanloo, N. Kehtarnavaz, "A generalized data-driven speech enhancement framework for bilateral cochlear implants," *Proceedings of 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 7269-7273, 2013.

[12] IEEE Subcommittee, "IEEE recommended practice for speech quality measurements," *IEEE Transactions on Audio and Electroacoustics, AU-17*, pp. 225-246, 1969.

CHAPTER 7

ONLINE FRAME-BASED CLUSTERING WITH UNKNOWN NUMBER OF CLUSTERS[*]

Authors - Fatemeh Saki, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents an online frame-based clustering algorithm (OFC) for unsupervised classification applications in which data are received in a streaming manner as time passes by with the number of clusters being unknown. This algorithm consists of a number of steps including density-based outlier removal, new cluster generation, and cluster update. It is designed for applications when data samples are received in an online manner in frames. Such frames are first passed through an outlier removal step to generate denoised frames with consistent data samples during transitions times between clusters. A classification step is then applied to find whether frames belong to any of existing clusters. When frames do not get matched to any of existing clusters and certain criteria are met, a new cluster is created in real time and in an on-the-fly manner by using support vector domain descriptors. Experiments involving four synthetic and two real datasets are conducted to show the performance of the introduced clustering algorithm in terms of cluster purity and normalized mutual information. Comparison results with similar clustering algorithms designed for streaming data are also reported exhibiting the effectiveness of the introduced online frame-based clustering algorithm.

7.1 INTRODUCTION

Clustering algorithms normally assume that data samples are available as a collection or in their entirety [1, 2]. However, there are applications that demand clustering to be performed on-the-fly or online as new data samples become available in a streaming manner with passing time without having any prior knowledge of the number of clusters or classes. Clustering algorithms, such as k-means and k-medians [3-6], require that the number of clusters or classes to be specified beforehand and operate on all the samples of a dataset that exist in one place without considering the element of time. In general, clustering algorithms are not designed to cope with data samples that are made available in a time gradual or streaming manner without knowing the number of clusters or classes.

Although several online clustering algorithms have been reported in the literature, e.g., [7-15], these algorithms are primarily designed for sample-based clustering and some of them require the number of clusters or classes to be known. A single data sample often does not carry much information and it is more effective to consider a frame of data samples. The clustering algorithm introduced in this chapter is based on frames of data samples that become available in an online streaming manner without having any prior knowledge of the number of clusters or classes.

The rest of the chapter is organized as follows: In section 7.2, an overview of similar clustering algorithms designed for streaming data is provided. Section 7.3 describes the details of the introduced online frame-based clustering algorithm. The experimental results are then reported in section 7.4 together with a comparison to three existing clustering algorithms. Finally, the conclusion is stated in section 7.5.

7.2 OVERVIEW OF EXISTING ONLINE CLUSTERING ALGORITHMS

In this section, an overview of existing online clustering algorithms is presented. One of the earliest and well-known clustering algorithms is the STREAM algorithm proposed by Guha et al. [11, 12]. This algorithm utilizes a divide-and-conquer strategy to segment streaming data into segments followed by k-means clustering. An extension to this algorithm using a sliding window appeared in [16].

A clustering algorithm named CluStream was covered in [7]. This algorithm includes two parts: an online micro-clustering part and an offline macro-clustering part. It uses a predefined number of micro-clusters to store a summary of streaming data. In its offline part, the k-means clustering algorithm is used to form larger clusters out of micro-clusters. In [17, 18], a clustering algorithm named HPStream was discussed for clustering of high-dimensional streaming data. HPStream applies a data projection module before clustering to reduce the high dimensionality of streaming data and then uses a so called Fading Cluster Structure (FCS) to maintain a summary of data samples while attaching more importance to recent data samples. The number of clusters and average number of projected dimensions are considered known in this clustering algorithm.

The aforementioned algorithms are similar to the k-means clustering algorithm and have the limitation of not being able to cope with clusters of non-spherical shapes. In many applications, clusters may have arbitrary shapes and thus are not easily separable by k-means type algorithms [7], [19-21]. Cao et al. [8] presented an extension of the clustering algorithm DBSCAN [20] called DenStream to cope with clusters of arbitrary shapes. This algorithm incorporates an online and an offline part. In its online part, micro-clusters are obtained and in its offline part, DBSCAN is applied to generate the final clusters from micro-clusters.

The clustering algorithm named D-Stream [19], [22] is a density-based clustering which divides the data space into a grid for density estimation instead of using micro-clusters. It involves two parts. In its online part, it maps data samples onto a grid. In its offline part, it obtains the grid density. The final clusters are then created based on the grid density. A fading function is used to decrease the grid density over time if it falls below a predefined threshold. In this algorithm, the number of grids increases exponentially with the number of dimensions. In [23], another density-based clustering algorithm was discussed which is capable of detecting clusters of arbitrary shapes. It uses a sparse-graph approach to incrementally cluster incoming data samples by modeling their spatiotemporal relationships.

More recently, the clustering algorithm named SVStream was introduced in [24], which is a modification of the Support Vector Clustering method in [25]. SVStream utilizes support vector descriptors and a complete graph (CG) labeling method [25] to label clusters. In this algorithm, the complexity of the CG labeling is dependent on the data dimensionality. Furthermore, if more than one half of incoming data samples fall outside existing sphere-shaped areas, a new sphere gets added and the cluster labeling is updated. Thus, when data are noisy, unnecessary spheres get added.

It is worth noting that in all of the aforementioned clustering algorithms, there are many user-specified parameters that highly influence the clustering outcome. The computationally efficient clustering algorithm introduced in this chapter is capable of creating clusters of arbitrary shapes on-the-fly in real time without having access to all the data samples in one place and more notably without knowing the number of clusters.

7.3 ONLINE FRAME-BASED CLUSTERING

The Online Frame-based Clustering algorithm introduced in this chapter is named OFC. This clustering algorithm is meant to be used for applications where streaming data associated with a cluster occurs for some time duration and each sample by itself does not carry enough information to be assigned to a cluster or class. An example application includes background noise classification, where audio data frames get captured in a streaming manner from a noisy environment and such frames need to be classified on-the-fly and in real time with no prior knowledge of the number of noise types or classes.

OFC comprises the following steps: denoising or outlier removal, density connection check, classification, new cluster creation, and cluster update. The algorithm begins by considering a frame of samples for making a decision. This is done by collecting data samples in a buffer of size N, and when the buffer becomes full, its data are referred to as a frame. Each frame is passed onto the outlier removal step to generate denoised frames with consistent data samples. Denoised frames are then passed onto the classification step. When no cluster has formed, denoised frames are moved to a collection of frames named *Chunk*. Other than the very beginning, frames are passed onto the classification step to find whether a match exists to any of existing clusters. If a frame does not get matched to any of existing clusters, it is moved to *Chunk*. In other words, *Chunk* keeps track of those frames identified to be new or novel. Figures 7.1 and 7.2 provide a flowchart of the introduced clustering algorithm OFC. In what follows, a detailed explanation of the steps appearing in this flowchart is provided.

7.3.1 Initialization

The clustering process begins by placing data samples into a temporary buffer, labeled *Buf*, which denotes a frame. When *Buf* gets full (i.e., a frame of size *N* data samples is collected), it is passed to the outlier removal step, where possible noisy samples are removed from a frame. A denoised frame is then passed onto the classification step. In the classification step, if no match to a denoised

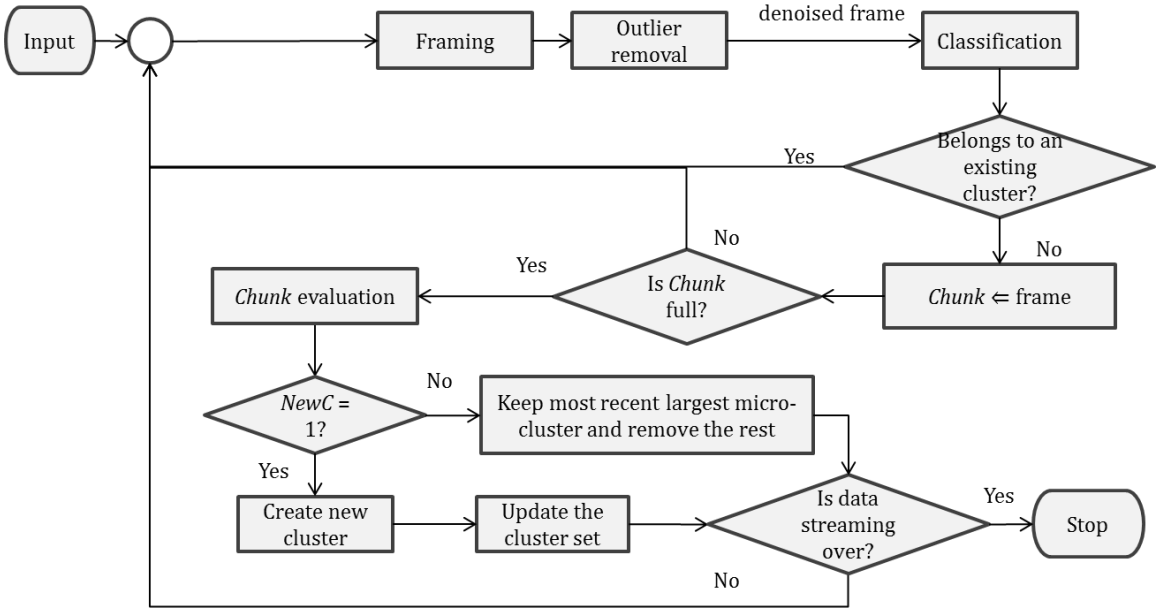

Figure 7.1. Flowchart of the introduced OFC clustering algorithm

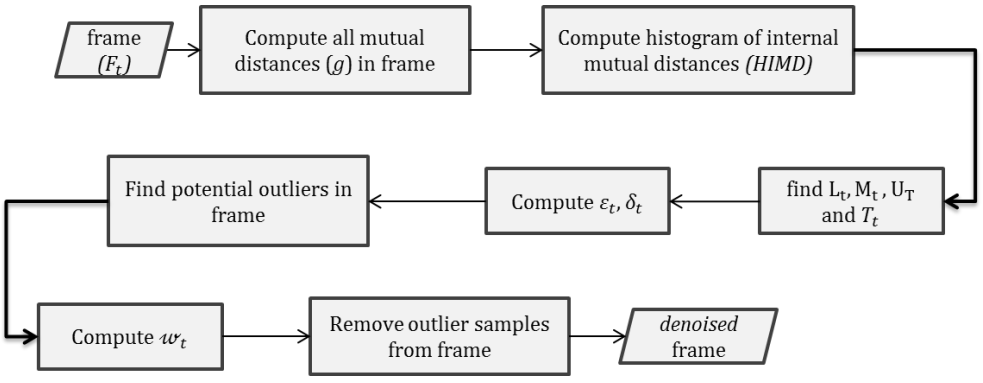

Figure 7.2. Outlier removal flowchart

frame is found, that frame is considered to be from a potential new cluster, a flag named *NewProb* is set to one and the frame is moved to *Chunk*. When *Chunk* gets full, the new cluster creation step is activated. Here it is worth mentioning that the initial streaming data time duration denoted by *InitT* is assumed to be longer than *Chunk* size *L*, i.e., *InitT* >>*L* or *InitT* ≥ *L*×*N* (*N* is frame size).

Framing- Each new incoming sample is first placed into a buffer *Buf*, and its mutual Euclidian distances to all the previous samples in a frame are computed. Euclidian distance is considered here due to its computational simplicity noting that it is also possible to use other distances. When *Buf* becomes full, it is labeled to be a frame F. Let the t^{th} frame be denoted by $F_t \coloneqq \{\mathcal{Y}_{i,t} | \mathcal{Y}_{i,t} \in \mathbb{R}^d, i = 1, \dots N\}$, $t = 1,2, \dots$, where $\mathcal{Y}_{i,t}$ indicates the i^{th} d-dimensional feature vector in the t^{th} frame. For a frame of size N, $\binom{N}{2} = \frac{N(N-1)}{2}$ so called internal mutual distances (denoted by g) between sample pairs are computed, where $g_{i,j}(\mathcal{Y}_{i,t}, \mathcal{Y}_{j,t}) = \|\mathcal{Y}_{i,t} - \mathcal{Y}_{j,t}\|_2$, $i, j = 1, \dots, N$, . Alternatively, mutual distances can be computed after a frame is created rather than at each time instance.

7.3.2 Outlier Removal

Outlier removal is a pre-clustering step that is performed in many density-based clustering algorithms. Noting that it is desired to find clusters of arbitrary shapes, a density-based approach is also adopted here. The density-based outlier detection approaches of DBSCAN [20] and OPTICS [26] involve several parameters which need to be specified by users. Therefore, to avoid having such user-specified parameters, a data-driven density-based approach is considered here to find outliers in a frame. The objective of the outlier removal step in our algorithm is to remove

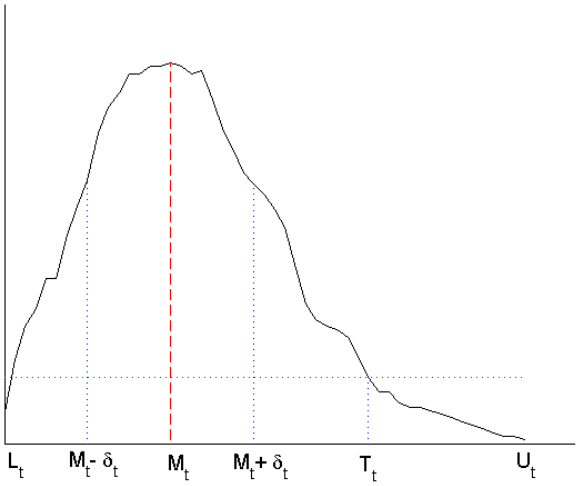

Figure 7.3. Histogram of internal mutual distances (HIMD) of a typical frame; $L_t, M_t,$ and U_t denote the minimum, peak, maximum of HIMD, respectively; T_t and δ_t are the threshold values for outlier detection and standard deviation of internal mutual distances, respectively.

inconsistent samples from a frame that do not belong to the same cluster. Basically, a frame is purified before getting passed onto the classification step.

The outlier removal is performed based on the histogram of internal mutual distances (*HIMD*) of a frame. The assumption commonly made in the existing density-based outlier detection methods is that the density of the outliers is less than the density of the main object and outliers usually are located far from the main object. In other words, instances of the main object are closer to each other in a more dense area. Likewise, samples in low density areas with far mutual distances from other samples (more than a threshold value T_t) are considered to be potential outliers. To find potential outliers, the histogram *HIMD* is used. Figure 7.3 represents the *HIMD* corresponding to a typical frame. The minimum, maximum and peak values of this histogram are denoted by L_t , U_t and M_t , respectively.

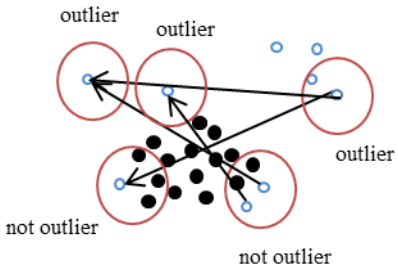

Figure 7.4. Potential outliers (hollow dots) and main object (black dots)

The threshold T_t is set automatically as follows: Let $Q_t := \{\mathscr{b} | \mathscr{b} > M_t, f_t(\mathscr{b}) = \varrho f_t(M_t)\}$ denote a set corresponding to $t = 1,2, ...,$ with weight $\varrho \in [0,1]$ and $f_t: \mathbb{R} \to [0,1]$ indicating the normalized histogram. The threshold T_t is then set to be the smallest member of Q_t, i.e., $T_t = \min(Q_t)$. Samples with mutual distances greater than T_t are then regarded as potential outliers considering that they have distances far from other samples and their density is low (ϱ times less than $f_t(M_t)$). It is important to note that non-outlier samples having mutual distances greater than T_t may get detected as potential outliers. Figure 7.4 provides an illustration of such samples to make this point more clear. These samples should not be treated as outliers, and they need to be removed from the list of potential outliers. This is achieved by examining the closeness of these samples, where closeness is defined by a neighborhood ε_t around $\mathscr{w}_t$ samples. Thus, if any of the potential outlier samples appears close, it is excluded as a non-outlier sample.

For finding ε of a frame, a density approach is adopted here by using the following equation:

$$D_t = \frac{M_t - L_t}{U_t - M_t}, \ t = 1,2, ... \tag{7.1}$$

where D_t denotes a measure of the t^{th} frame density. It provides a representation of the histogram shape and its peak location, see Figure 7.5. When D_t becomes equal to one, it indicates M_t is located at equal distances from the minimum (L_t) and maximum (U_t) values of the *HIMD* of the

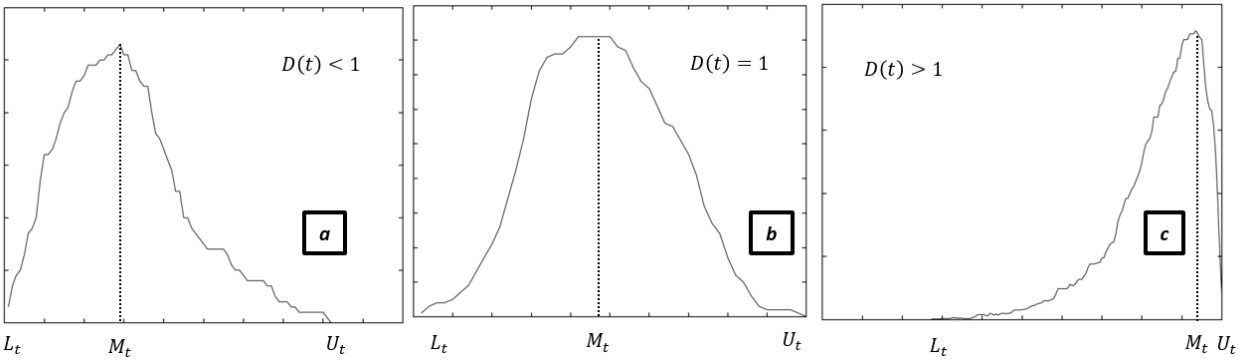

Figure 7.5. Different possible values and distributions of frame internal distances

t^{th} frame (see Figure 7.5b). A frame with D_t less than one is considered to be a dense frame. M_t getting closer to L_t indicates samples generate close distances to each other and vice versa. The neighborhood ε_t for a dense frame is defined based on M_t. However, to allow for some tolerance margin, the following neighborhood is considered:

$$\varepsilon_t = M_t + \delta_t, t = 1,2, \ldots \tag{7.2}$$

$$\delta_t = \left(\frac{1}{N^2}\sum_{i,j=1}^{N}\left(g_{i,j} - \overline{g_t}\right)^2\right)^{\frac{1}{2}} \tag{7.3}$$

$$\overline{g_t} = \frac{1}{N^2}\sum_{i,j=1}^{N} g_{i,j} \tag{7.4}$$

where δ_t denotes the standard deviation of $g_{i,j}$ values, $i,j = 1, \ldots, N$, found via (7.3). A frame with D_t greater than one translates into three possibilities: (i) M_t is located close to U_t due to $f_t(M_t)$ not being significantly larger than other values, (ii) L_t and U_t are close to each other, or (iii) density of samples of interest and density of samples of noninterest are the same (mostly when a frame has data from two different clusters). In such cases, $g_{i,j}s$ range is divided into two intervals: $[L_t, M_t]$ and $[M_t, U_t]$. For D_t greater than one, the first interval is used to find M_t and ε_t among the smaller mutual distances. w_t can be specified to be any number up to *N*-|potential outliers|, where bars denote the number of potential outliers.

Based on $\wp_t$ and ε_t, outliers get identified among potential outliers. This is done by examining the neighborhood ε_t in (7.2) of each potential outlier sample. If there are at least $\wp_t$ non-outliers in this neighborhood, the sample is considered to be a non-outlier sample; otherwise it is assigned to be an outlier sample. After excluding all the outliers from a frame, its purified version is named a dense or a denoised frame. The t^{th} denoised frame can be specified as $\tilde{F}_t := \{(\mathcal{Y}_{j,t}, \mu_t, \varepsilon_t) | \mathcal{Y}_{j,t} \in \mathbb{R}^d, \varepsilon_t = M_t + \delta_t,\ j = 1, \dots, \mathcal{N}_t\}$, $t = 1,2, \dots$, where M_t and δ_t are found as explained earlier and μ_t is obtained using (7.5). The centroid of this frame is given by:

$$\mu_t = \frac{1}{\mathcal{N}_t} \sum_{j=1}^{\mathcal{N}_t} \mathcal{Y}_{j,t}\ , t = 1,2, \dots \quad (7.5)$$

where $\mathcal{N}_t$ is the size of a denoised frame, which is smaller than the size of an original frame $\mathcal{N}_t \leq N$.

7.3.3 Chunk Evaluation

A new denoised frame is moved to *Chunk* having a size *L* when it does not get matched to any of the existing clusters by the classification step. Any such frame that gets to *Chunk* is analyzed to find its connection to the last frame in *Chunk* as described below.

Frame Connection - This step is for the purpose of seeing whether two frames are connected/similar to each other (**Definition 3**) or not (the expectation is that frames from the same cluster or class are close to each other). After the outlier detection, all the samples in the t^{th} frame either have mutual distances less than T_t (not selected as potential outliers; main object samples) or have $\wp_t$ samples in their ε_t-neighborhoods (potential outliers kept as non-outliers). Often such samples are transitioning samples in a denoised frame as illustrated in Figure 7.4. Two consecutive frames F_t and $F_{t'}$ can be either similar/connected to each other or disconnected. Connected frames

are required to be from the same class. Therefore, their distribution and their most common mutual distances (and thus ε_t and $\varepsilon_{t'}$) appear close to each other.

Definition 1: Two objects $O_{\mathfrak{x}}$ and $O_{\mathfrak{y}}$ from two different frames $F_{\mathfrak{x}}$ and $F_{\mathfrak{y}}$ are close enough/connected to each other if at least one of the objects is in the ε-neighborhood of the other object, that is

$$\begin{cases} g_{\mathfrak{x},\mathfrak{y}}(O_{\mathfrak{x}}, O_{\mathfrak{y}}) < \varepsilon_{\mathfrak{x}} + \varepsilon_{\mathfrak{y}}\,, & connected \\ otherwise, & not\ connected \end{cases} \tag{7.6}$$

Two frames F_t (of size $\mathcal{N}_t$) and $F_{t'}$ (of size $\mathcal{N}_{t'}$) are defined as connected frames if there exists a connection between their samples. To find connections between all the samples of a frame, $\mathcal{N}_t \times \mathcal{N}_{t'}$ Euclidian distances need to be computed. To reduce this computational burden, instead of computing the mutual distances between all the samples of two frames, only the distances between their centroids is computed.

Definition 2: Two frames F_t and $F_{t'}$ are directly-connected to each other if their centroids are close (as per **Definition 1**), that is

$$g_{t,t'}\,(\mu_t, \mu_{t'}) < \varepsilon_t + \varepsilon_{t'} \tag{7.7}$$

Definition 3: A frame F_t is connected to a frame $F_{t''}$ if there is a chain of frames $F_{t_1}, F_{t_2} \ldots, F_{t_n}, F_{t_1} = F_t, F_{t_n} = F_{t''}$ such that $F_{t_{i+1}}$ is directly-connected (as per **Definition 2**) to F_{t_i}.

Definition 4: A micro-cluster $\mathcal{C}$ is defined as an ensemble of connected frames (as per **Definition 3**), that is

$$\mathcal{C} \coloneqq \{F_{t_i}, \forall i = 1, \ldots, I, g_{t_i,t_{i+1}}(\mu_{t_i}, \mu_{t_{i+1}}) < \varepsilon_{\mathrm{t_i}} + \varepsilon_{\mathrm{t_{i+1}}}\} \tag{7.8}$$

where *I* denotes the number of frames in the micro-cluster.

In an ideal case, *Chunk* should contain only one micro-cluster of connected samples from one new cluster. However, in practice, because of the presence of noisy data samples, some disconnections between adjacent frames occur, causing an increase in the number of micro-clusters in *Chunk.* When *Chunk* gets full, an evaluation of micro-clusters is conducted as follows:

Number of micro-clusters and their sizes (number of frames in each micro-cluster) are first found. Sporadic micro-clusters are removed. That is if the size of a micro-cluster is less than two, it is considered to be a sporadic micro-cluster. This means if a frame is disconnected from both directions, it is not connected to previous frames and neither to proceeding ones. Close micro-clusters are then merged.

As mentioned before, disconnection between frames usually occur because of the presence of noisy data samples. Thus, if any connection between frames of different micro-clusters exists, these micro-clusters are considered to be connected micro-clusters (**Definition 5**).

Definition 5: Two micro-clusters are connected to each other if both micro-clusters are directly-connected to each other through at least one frame (as per **Definition 2**).

After removing sporadic micro-clusters and connecting similar ones, if there is a micro-cluster with a size greater than one half of the size of *Chunk*, the algorithm declares that a new cluster or class needs to be created, the flag *NewC* gets set to one and the micro-cluster is moved to the new cluster creation step. By conducting such a chunk evaluation process, it is made sure that only consistent and most homogeneous data are used for creating a new cluster.

7.3.4 New Cluster Creation

In this step, data associated with *Chunk* are used to create a new cluster based on a data description method. Different data description methods appear in the literature. Most involve estimating a

probability density using Parzen window [27] or Gaussian distribution [28]. The drawbacks of these methods are that in general a large number of samples are required (in particular, in higher dimensional feature spaces) and also the focus is placed on modeling high density areas and not low density areas. In [29], Tax proposed a simple nearest neighbor data description method and its improved version, called k-nearest neighbor data description. It was shown that these two methods did not work well for low-dimensional data. In [30], Vapnik presented an improved solution by computing the boundary around a dataset rather than estimating the density. An attempt to use just the boundary of a dataset was made based on neural networks in [31], which required one to specify many parameters such as network size, weight initialization, stopping criterion, etc. In [32], Tax utilized a Support Vector Data Description (SVDD) to obtain a spherically shaped boundary around a dataset. It was shown that SVDD was quite effective when a proper kernel was used [29]. Hence, SVDD is also used here.

In our clustering algorithm, one class gets created at a time. Thus, when *Chunk* is moved to the create-new-cluster step, only one cluster is created. The task of this step is to provide a cluster representative that can be used for the classification step instead of keeping all the samples of a cluster. SVDD is used for this purpose.

Support Vector Domain Description (SVDD) - SVDD is a sphere-shaped data description involving nonlinear transformation (kernel functions). SVDD provides an effective data description relying on only a small number of support vectors (*SVs*) [32]. Let $X \coloneqq \{x_j | x_j \in \mathbb{R}^d, j = 1, \dots, \mathcal{J}\}$ be a dataset of $\mathcal{J}$ points. Using a nonlinear transformation φ from X to a high-dimensional kernel feature space, the smallest enclosing hypersphere of radius R and center a can be stated as:

$$H(R, a, \xi_j) = R^2 + \gamma \sum_{j=1}^{J} \xi_j \tag{7.9}$$

with the constraints

$$\left\|\varphi(x_j) - a\right\|^2 \leq R^2 + \xi_j, \ \forall j = 1, \ldots, J \tag{7.10}$$

The weight γ establishes a trade-off between volume and error (accuracy of data description), $\xi_j \geq 0$, denotes variables that punish samples whose distances from the center a are farther than R. By using the Lagrange multiplier method, the constraints (7.10) is incorporated into (7.9) generating the Lagrangian function:

$$\mathcal{L}(R, a, \xi_j, \beta_j, \alpha_j) = R^2 + \gamma \sum_{j=1}^{J} \xi_j - \sum_{j=1}^{J} \beta_j \left(R^2 + \xi_j - \left\|\varphi(x_j) - a\right\|^2\right) - \sum_{j=1}^{J} \alpha_j \xi_j \tag{7.11}$$

where $\beta_j \geq 0, \ \alpha_j \geq 0$ represent Lagrange multipliers. The function (7.11) is minimized with respect to R, a, ξ_j and maximized with respect to β_j and α_j. As shown in [30], (7.11) can be written as:

$$\mathcal{L} = \sum_{j=1}^{J} \beta_j \varphi(x_j).\varphi(x_j) - \sum_{j,k=1}^{J} \beta_j \beta_k \varphi(x_j).\varphi(x_k) \tag{7.12}$$

The inner product $\varphi(x_j).\varphi(x_k)$ can be replaced by an appropriate kernel function:

$$\mathcal{L} = \sum_{j=1}^{J} \beta_j \mathcal{K}(x_j, x_j) - \sum_{j,k=1}^{J} \beta_j \beta_k \mathcal{K}(x_j, x_k) \tag{7.13}$$

In this paper, the Gaussian kernel $\mathcal{K}(x_j, x_k) = \exp(\frac{1}{\sigma}\left\|x_j - x_k\right\|^2)$ is used, where σ denotes a width parameter.

Samples with $\beta_j = 0$ are inner samples meaning that they either lie inside or on the sphere surface. Samples with $0 < \beta_j < \gamma$ are called support vectors (SVs). These samples lie on the sphere surface. Samples with $\beta_j = \gamma$ fall outside the sphere boundary and are excluded.

To test a new sample z, its distance to the center of the sphere is computed and if this distance is smaller than *R*, it means that the sample belongs to the sphere and it is accepted,

$$\mathrm{R}_{\mathrm{z}}^2 = \|z - a\|^2 \tag{7.14}$$

Expressing the center of the sphere in terms of the support vectors, one gets

$$\mathrm{R}_{\mathrm{z}}^2 = \mathcal{K}(z,z) - 2\sum_{j=1}^{\mathcal{J}} \beta_j \mathcal{K}(z, x_j) + \sum_{j,k=1}^{\mathcal{J}} \beta_j \beta_k \mathcal{K}(x_j, x_k) \tag{7.15}$$

The radius of the sphere is given by:

$$R = \max\{R_{x_j} | x_j \; is \; a \; Support \; Vector\} \tag{7.16}$$

A practical way for defining *R* is to use the maximum or the average value over all the support vectors. Next, a cluster representative is defined.

Definition 6: Given a set of $\mathcal{J}$ data samples, the sphere structure ψ is defined as:

$$\psi = \{\bar{\mathcal{S}}, \|a\|^2, R, \ell\} \tag{7.17}$$

where $\bar{\mathcal{S}}$ denote the support vectors and their Lagrange multipliers, that is:

$$\bar{\mathcal{S}} = \{(x_j, \beta_j) | \; 0 < \beta_j < \gamma\} \tag{7.18}$$

with this squared length of the sphere center

$$\|a\|^2 = \sum \beta_j \beta_k \mathcal{K}(x_j, x_k), \; \forall (x_j, \beta_j), (x_k, \beta_k) \in \bar{\mathcal{S}} \tag{7.19}$$

When *Chunk* is moved to the new cluster creation step, the data associated with the frames are used to solve the optimization problem in (7.13) towards creating a new sphere or cluster and obtaining its parameters (7.17)-(7.19). The cluster label ℓ for the newly created cluster is specified to be the current cluster label plus one.

7.3.5 Classification

In this step, it is seen whether incoming samples belong to any existing clusters and whether there are changes occurring in the streaming data. There exist other methods for detecting changes in the streaming data, e.g., [33], [34], which can also be used here. After creating the first cluster, each new frame gets fed into the classification step to find whether it belongs to any of the existing clusters or not.

By using SVDD to characterize the data samples of a cluster, the boundary of its samples, its kernel centroid and radius are obtained. Such boundaries are used to assign a new cluster. To find a cluster getting matched to a new frame $\tilde{\mathrm{F}}_{\mathrm{t}} \coloneqq \{Z_{i,t} \big|\, Z_{i,t} \in \mathbb{R}^d, i = 1, \dots, \mathcal{N}_t, \}$, $t = 1,2, \dots$, the distance of each sample $R^2_{Z_{i,t}}$ in a new frame to the center of each sphere is computed via (7.15). Samples with distances smaller than the sphere radius lie inside the cluster sphere and samples with larger distances lie outside or on the cluster sphere surface. First, the closest sphere to a frame (to its centroid) is obtained. Then, if the number of the inside samples is greater than the outside samples, that frame is assigned to the corresponding sphere. If the frame is outside all the cluster spheres, a flag named *NewProb* is set to one which means the frame might be from an unseen cluster, therefore it is moved to *Chunk.*

7.3.6 Cluster Update

If two clusters are too close to each other, they need to be merged. Based on **Definition 6**, a cluster is defined as $\psi = \{\bar{\mathcal{S}}, \|a\|^2, R, \ell\}$. If two spheres ψ and $\tilde{\psi}$ have the center distance $\|a - \tilde{a}\|$ less than the summation of their radii (7.20)-(7.21), the two spheres are considered to be too close and thus one cluster is used to represent them, that is [24]:

$$\|a-\tilde{a}\|^2 = \|a\|^2 + \|\tilde{a}\|^2 - 2\sum\beta_j\,\tilde{\beta}_k\,\mathcal{K}(x_j,\tilde{x}_k), \quad \forall(x_j,\beta_j)\in\psi, \forall\,(\tilde{x}_k,\tilde{\beta}_k)\in\tilde{\psi} \tag{7.20}$$

$$\begin{cases} \|a-\tilde{a}\| \le R+\tilde{R} & two\ connected\ spheres \\ otherwise & two\ separate\ spheres \end{cases} \tag{7.21}$$

7.4 EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we provide extensive experimentations to evaluate the introduced clustering algorithm as well as comparisons with three clustering algorithms that are designed for streaming data, namely CluStream [7], DenStream [8], and SVStream [24]. It is worth mentioning that in [24], it was shown that SVStream outperformed the clustering algorithms StreamKM [11], RepStream [23] and StrAP [35]. At the beginning of the clustering process, when there was no cluster, all frames were stored in *Chunk.* Once the first cluster was built, frames were then fed into the classification part and if they were not labeled with any of the existing clusters or classes, they got moved to *Chunk.*

7.4.1 Datasets

To evaluate the developed algorithm OFC for clustering streaming data arriving in an online manner, both synthetic and real datasets were examined. The four synthetic datasets, labeled DS1, DS2, DS3, and DS4, were generated using the code in [36]. These datasets are shown in Figure 7.6. DS1 consists of 14,000 data samples forming four classes; DS2 consists of 10,000 samples forming two classes; DS3 consists of 12,000 samples belonging to 7 classes; and DS4 dataset consists of 4,000 samples of two classes. Time was simulated by feeding these samples into the clustering algorithms in a streaming manner.

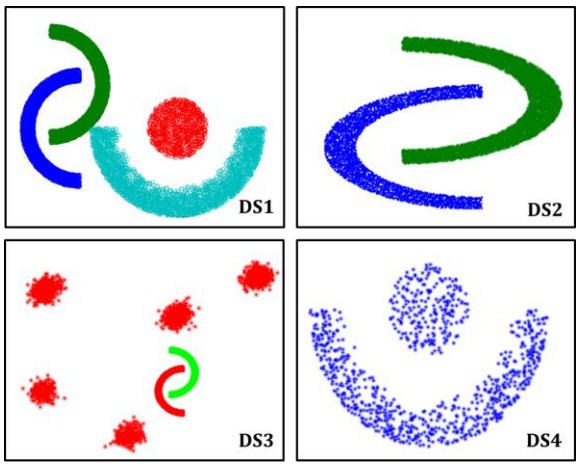

Figure 7.6. Four synthetic datasets used in the experiments.

Two real-world datasets from the UCI Machine Learning Archive [37] were also examined. These datasets are known as KDD-CUP99 Network Intrusion Detection and Forest-CoverType, respectively. The KDD-CUP99 dataset is a real dataset that has evolved over years and has been widely used to evaluate clustering algorithms for streaming data [7], [8], [17], [23]. This dataset consists of a stream of TCP connection records from two weeks of LAN network traffic managed by the MIT Lincoln Lab. The complete dataset contains approximately 4.9 million records. 42 features were collected for each record which included the duration of connections, the number of transmitted bytes from source to destination (and vice versa) and the number of failed login attempts. As done in [7], [17], [19], here a subset of length 494,020 was examined. Each connection was classified into either a normal connection or an intrusion (attack). The attacks were categorized into four main categories: denial-of-service (DOS), R2L (unauthorized access from a remote machine), U2R (unauthorized access to local super user privileges), and PROBING (surveillance and other probing).

The dataset Forest-CoverType contains a total of 581,012 observations from seven types of forest cover. Each observation consisted of 54 geological and geographical features that described environments in which trees were observed, including 10 quantitative variables, 4 binary wilderness areas and 40 binary soil type variables. As done in [17], all 10 quantitative variables were used here. To be able to use these datasets, they were converted into data streams by taking the data order as the order of streaming. Based on the consideration that the streaming data of each cluster was greater than at least one half the *Chunk* size, the samples of each cluster were arranged such that they appeared for some time duration and the order of the clusters was randomly shuffled in time.

To evaluate the performance of the OFC algorithm, initially the cluster purity measure was considered. This measure is defined as follows [8]:

$$Purity = \frac{\sum_{\ell=1}^{U} \frac{|\widehat{V_\ell}|}{|V_\ell|}}{U} \times 100 \tag{7.22}$$

where U denotes the number of clusters, $\left|\widehat{V_\ell}\right|$ indicates the number of samples with the dominant cluster label in cluster ℓ, and $|V_\ell|$ indicates the total number of samples in cluster ℓ. In other words, this measure indicates the purity of the identified clusters with respect to the groundtruth or the true clusters. It is important to note that of course in practice the number of clusters is unknown or there exists no groundtruth.

Another measure that was examined to evaluate the performance of our clustering algorithm was normalized mutual information (NMI). This measure indicates the similarity of created clusters with respect to groundtruth clusters keeping in mind that in practice groundtruth clusters are not

known. Let Q and Q' denote the cluster sets corresponding to the groundtruth and the algorithm, respectively. The normalized mutual information $NMI(Q, Q')$ is given by

$$NMI(Q, Q') = \frac{MI(Q,Q')}{\sqrt{\mathrm{E(Q).E(Q')}}} \tag{7.23}$$

$$MI(Q, Q') = \sum_{q_i \in Q, q'_\ell \in Q'} p(q_i, q'_\ell). \log \frac{p(q_i, q'_\ell)}{p(q_i).p(q'_\ell)} \tag{7.24}$$

where $p(q_i)$, $p(q'_\ell)$ and $p(q_i, q'_\ell)$ are the probabilities of samples being in the clusters q_i, q'_ℓ, and the intersection of q_i and q'_ℓ, respectively. $\mathrm{E(Q)}$ and $\mathrm{E(Q')}$ denote the entropies of the clusters. Next, before showing the performance results, let us first state how the parameter values were set.

7.4.2 Parameter Setting

This section provides the OFC outcome for the four datasets DS1, DS2, DS3 and DS4 when using different buffer size (frame sizes) N, *Chunk* size L (units expressed in terms of frames), and Gaussian kernel width σ. The default values of the frame size N, *Chunk* size L, and smoothing kernel parameter σ in our experimentations were 10, 10, and 2.5, respectively. While altering one of the parameters, the other two parameters were kept constant.

Gaussian Kernel Width

The parameter σ controls the smoothness of the contour generated by SVDD. Small σ values for Gaussian kernel cause rough boundaries and involve a large number of support vectors and as σ is increased, cluster boundaries become smoother and the number of support vectors decreases [25]. Table 7.1 and 7.2 show the average purity and the average NMI of the OFC algorithm, respectively, for the DS1, DS2, DS3 and DS4 datasets across different values of σ. As seen from these tables, $\sigma = 2.5$ provided a balance between performance and number of support vectors while setting N and L to 10. As mentioned in [32], the created boundary by SVDD is controlled

Table 7.1. Average cluster purity in percentages versus Gaussian kernel width σ for synthetic dataset DS1, DS2, DS3 and DS4.

Datasets \ σ	0.5	1	1.5	2	2.5	3 - 6.5	7	7.5	8	8.5	9	9.5	10
DS1	36	41	61	90	99.5	98.5	99.8	99.4	99.1	99	98.6	97.5	97.1
DS2	50	50	100	100	100	100	100	100	100	99.9	99.8	99.6	98.6
DS3	42	41	98.5	98.5	98.5	98.5	98.5	98.4	98	97.9	97.7	97.1	96.5
DS4	77	76	96	99	99	100	100	100	100	100	100	100	100

Table 7.2. Average NMI value versus Gaussian kernel width σ for synthetic dataset DS1, DS2, DS3 and DS4

Datasets \ σ	0.5	1	1.5	2	2.5	3 - 6.5	7	7.5	8	8.5	9	9.5	10
DS1	0.1	0.1	0.4	0.8	0.97	0.99	0.99	0.99	0.97	0.95	0.93	0.89	0.88
DS2	0.4	0.4	1	1	1	1	1	1	1	0.98	0.98	0.96	0.9
DS3	0.03	0.03	0.99	0.99	0.99	0.99	0.99	0.97	0.93	0.92	0.91	0.86	0.82
DS4	0.14	0.14	0.97	1	1	1	1	1	1	1	1	1	1

by the kernel width parameter, thus for different numbers of data samples, the kernel parameters need to get adjusted so that the created boundary is made close to the actual boundary. As the number of data samples is increased for a fixed kernel parameter, the created boundary by SVDD grows and would cover part of the domains of other close clusters leading to a drop in performance [38].

Frame Size

The average cluster purity and NMI were examined when using different buffer sizes to create frames for the four synthetic datasets. Tables 7.3 and Table 7.4 show the average purity and NMI for different N values for the DS1, DS2, DS3 and DS4 datasets, respectively. Using large N values with $L=10$ dropped the performance because some of the clusters did not get created at the time they were supposed to get created. Therefore, as stated earlier, it is important to set N and L values depending on the application in such a way that they are consistent with the streaming duration of a cluster at its first appearance.

Chunk Size

The effect of changing *Chunk* size was also examined for *N=10* and $\sigma = 2.5$. Table 7.5 and 7.6 provide the outcome for different *L* values on the DS1 dataset. Likewise, for the other datasets *L=10* was found to provide the highest purity and *NMI* value.

It is worth pointing out that for the datasets DS1, DS3 and DS4, the purity and *NMI* values did not always increase by increasing the frame size or the *Chunk* size. As stated earlier, the initial number of data samples received from each cluster was considered to be longer than the *Chunk* size, i.e., longer than $L \times N$. Increasing any of these two parameters or both of them up to the point that exceeded the number of the initial streaming data samples of a cluster caused errors since *Chunk* got filled with the data from more than one cluster. As explained in the chunk evaluation step, when dissimilarities were discovered, only the consistent data were used for cluster creation.

A study was also conducted by changing the frame size and the *Chunk* size for all the datasets in terms of the correct creation of clusters. The parameter area that generated the correct outcome for all the six datasets is displayed in Figure 7.7. This figure shows that regardless of the shape and

Table 7.3. Average cluster purity in percentages versus frame size N for synthetic dataset DS1, DS2, DS3 and DS4

Datasets \ *N*	1-3	4	6	8	10	12	14	16	18	20	22	24	26	28	30
DS1	98	99	99	99	99.5	97.2	94.3	91.9	93.8	88.5	93	91.6	84.5	78.6	78.7
DS2	98	98	98	98	100	100	100	100	100	100	100	100	100	100	100
DS3	96	99	98.1	97.8	98.5	98.5	97	98.5	99.1	100	99.9	96.55	98.5	98.5	93.68
DS4	99	99	99	99	99	99	99	99	99	99	78.95	78.9	78.3	77.3	77.2

Table 7.4. Average *NMI* value versus frame size *N* for synthetic dataset DS1, DS2, DS3 and DS4

Datasets \ *N*	1-3	4	6	8	10	12	14	16	18	20	22	24	26	28	30
DS1	0.5	1	0.99	0.99	0.97	0.92	0.87	0.82	0.85	0.76	0.84	0.79	0.53	0.1	0.1
DS2	0.4	1	1	1	1	1	1	1	1	1	1	1	1	1	1
DS3	0.76	1	0.88	0.85	0.99	0.99	0.81	0.83	0.99	1	0.7	0.8	0.75	0.74	0.72
DS4	0.67	1	1	1	1	1	1	1	1	1	0.16	0.12	0.12	0.1	0.1

distribution of the datasets, by keeping *N* and *L* values in the mid-range (white area in the figure), the correct clustering outcome was obtained.

Table 7.5. Average cluster purity in percentages versus Chunk size *L* for synthetic dataset DS1, DS2, DS3 and DS4

Datasets \ *L*	1-3	4	6	8	10	12	14	16	18	20	22	24	26	28	30
DS1	98	99.8	99.8	99.8	99.5	99.5	97.5	97	96.3	94.9	92.8	92.4	85.3	78.6	78.6
DS2	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
DS3	98.7	98.8	98.7	98.6	98.5	98.4	98.2	98.1	97.8	97.5	97.1	95.8	94.24	89.9	97.7
DS4	99	99	99	99	99	99	99	99	99	99	78.6	78.7	78.6	78.5	78.4

Table 7.6. Average *NMI* value versus Chunk size L for synthetic dataset DS1, DS2, DS3 and DS4

Datasets \ *L*	1-3	4	6	8	10	12	14	16	18	20	22	24	26	28	30
DS1	0.7	0.99	0.99	0.99	0.97	0.97	0.93	0.92	0.91	0.88	0.85	0.82	0.6	0.1	0.1
DS2	0.75	1	1	1	1	1	1	1	1	1	1	1	1	1	1
DS3	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.79	0.78	0.77	0.77	0.77
DS4	0.8	1	1	1	1	1	1	1	1	1	0.11	0.13	0.1	0.1	0.1

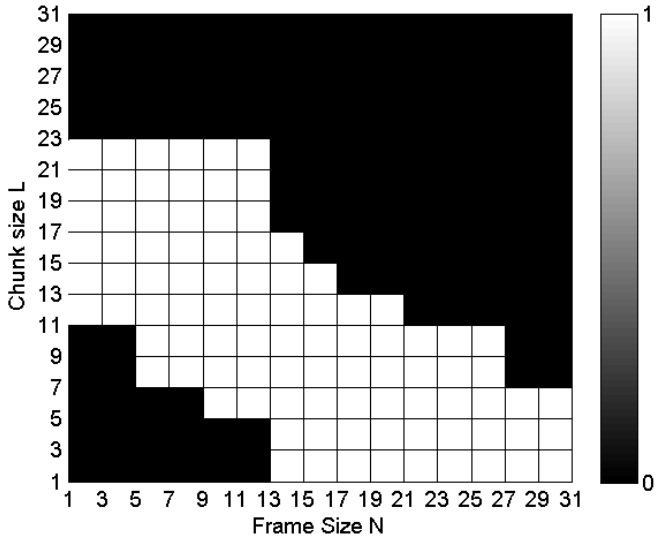

Figure 7.7. Parameter setting area (white area) for frame size and *Chunk* size generating correct clusters for all six datasets

7.4.3 Performance Evaluation

The clustering outcome and new cluster detection at different elapsed times are reported in this subsection. Figures 7.8 through 7.11 represent the clustering outcome across different elapsed times or frames for the DS1, DS2, DS3, and DS4 datasets, respectively. The true cluster labels at a number of elapsed times in frame units are provided in parts (a) of the figures while parts (b) show the generated cluster labels with parts (c) showing the flag *NewC*. This flag exhibits the new

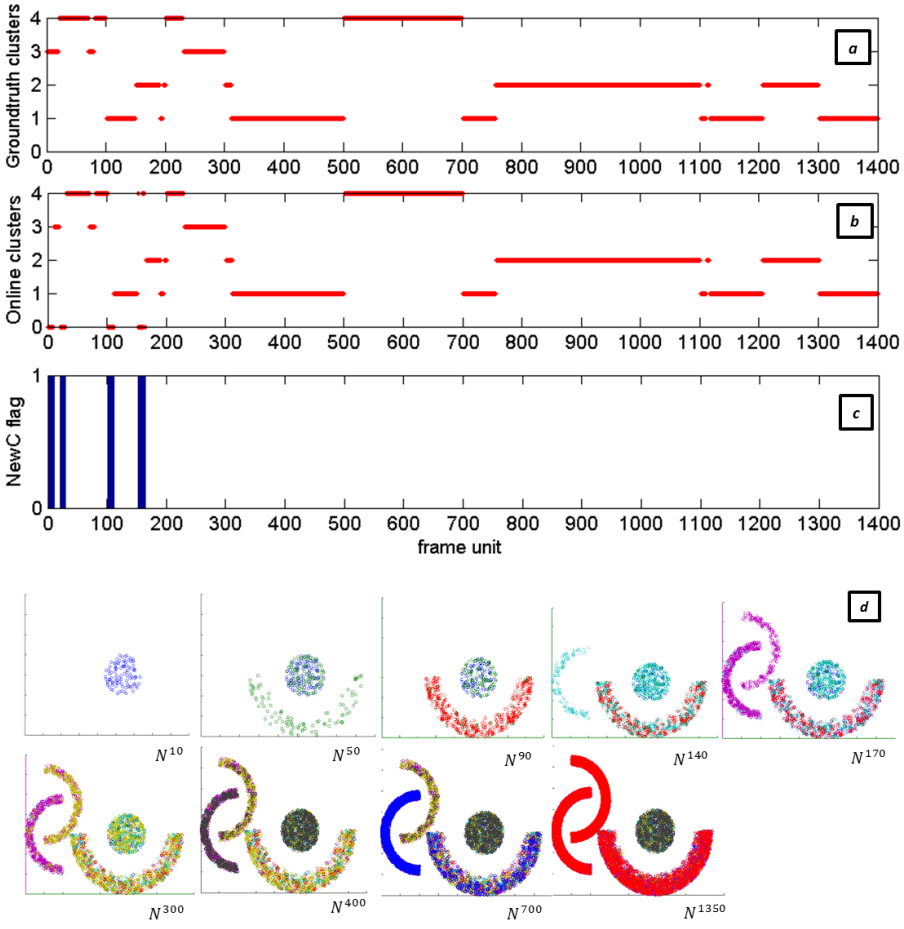

Figure 7.8. DS1 results: (a) groundtruth clusters, (b) OFC clustering outcome, (c) new cluster detection, (d) clusters versus elapsed time

cluster creation time. These results are provided using the default values of *N=10*, *L=10*, and $\sigma = 2.5$. The clusters at different elapsed times are shown in parts (d) of the figures.

7.4.4 Comparison

The OFC algorithm was compared in terms of purity and normalized mutual information with the three clustering algorithms CluStream, DenStream and SVStream. For CluStream [7] which involves an online micro-clustering part and an offline macro-clustering part, the first initial number *(IntNo)* of data samples were collected and clustered by k-means to create *W* micro-clusters. Each new data sample was added to the nearest micro-cluster. If it was not in the

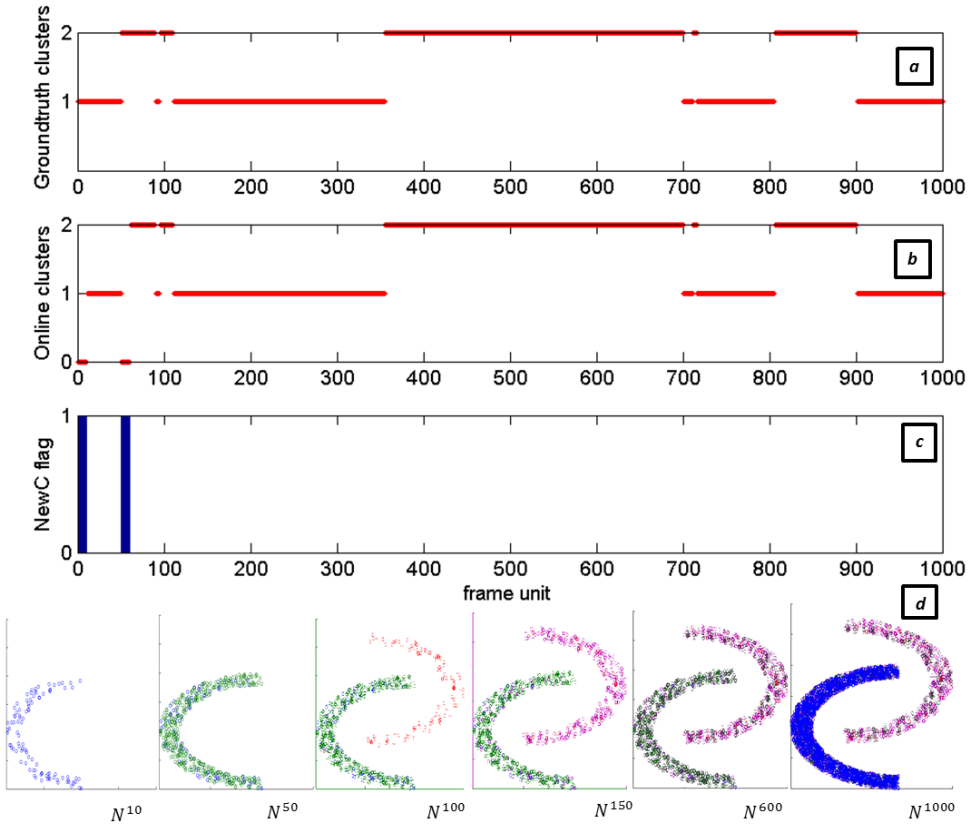

Figure 7.9. DS2 results: (a) groundtruth clusters, (b) OFC clustering outcome, (c) new cluster detection, (d) clusters versus elapsed time

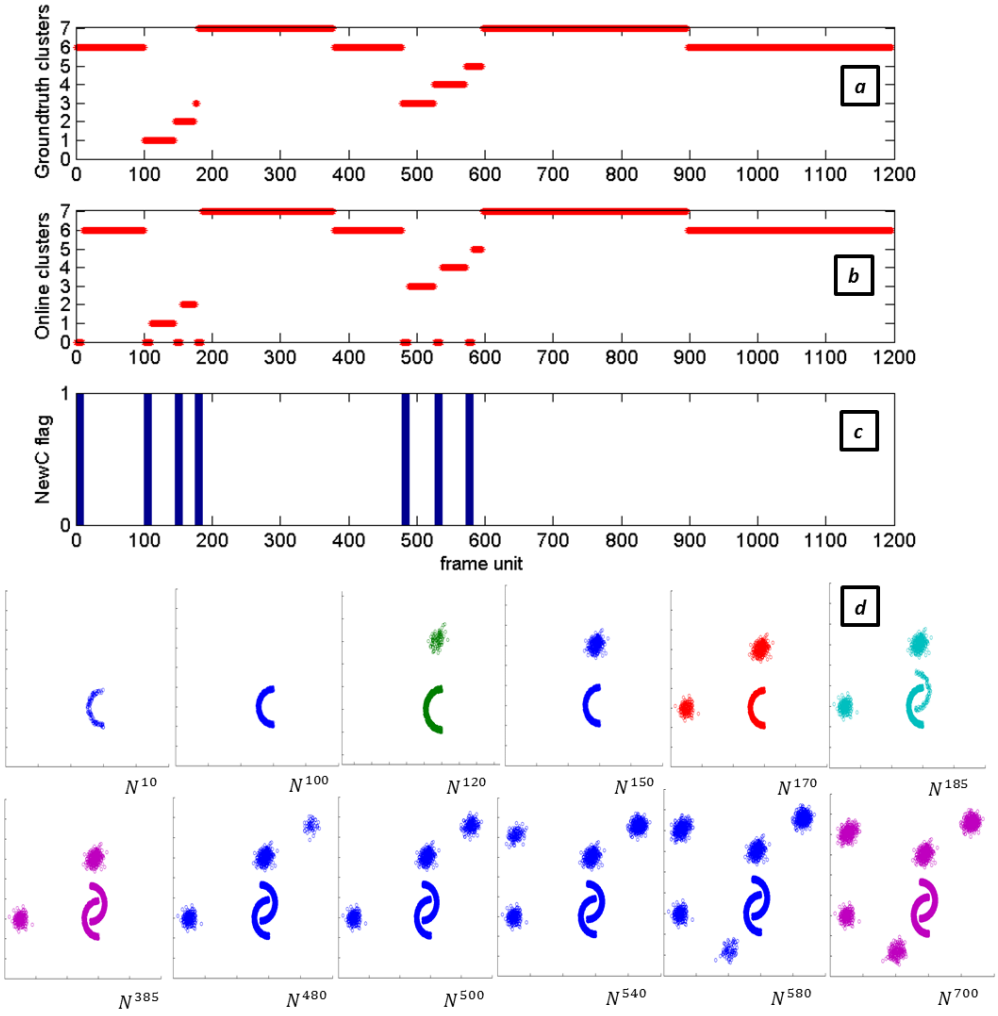

Figure 7.10. DS3 results: (a) groundtruth clusters, (b) OFC clustering outcome, (c) new cluster detection, (d) clusters versus elapsed time

maximum boundary of the existing micro-clusters, a new micro-cluster was created and one of the old micro-clusters was deleted or two close micro-clusters were merged. In the offline part, *K* macro-clusters were generated at each time by merging the micro-clusters within a time duration *h*. The parameters of CluStream were set based on the ones that generated the best results as recommended in [7], i.e., *IntNo* = 1000, *W=10*× *K* and *h=100*.

DenStream [8] also involves an online and an offline part. In this algorithm, there is a fading function $h(\tau) = 1/2^{\lambda\tau}$ to weight data samples; τ denotes the elapsed time and λ a fading factor. The first initial number *(IntNo)* of data samples were collected and clustered using DBSCAN to create the initial micro-clusters; a new data sample was merged to the nearest micro-cluster or a new micro-cluster was created. In the offline part, different DBSCANs [20] were performed to create the macro-clusters. The parameters of DenStream were set based on the ones recommended in [8], i.e., *IntNo=1000*, λ =0.25.

In addition, a comparison was made to the algorithm SVStream [24]. SVStream processes data chunk by chunk. After creating an initial sphere, new chunk data are assigned to outside the sphere

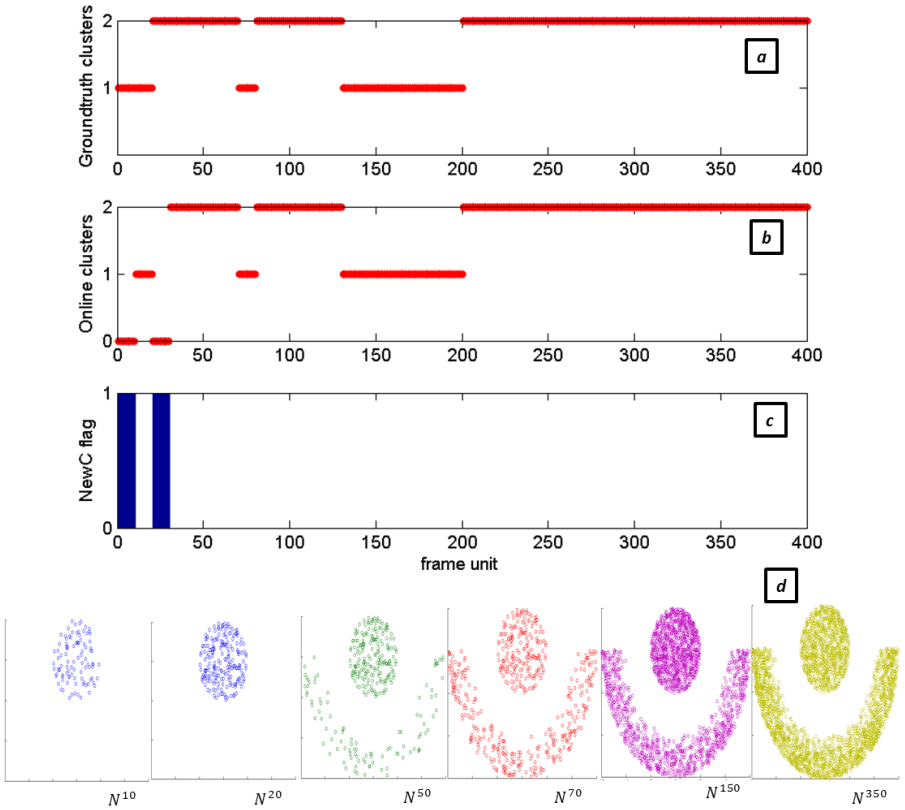

Figure 7.11. DS4 results: (a) groundtruth clusters, (b) OFC clustering outcome, (c) new cluster detection, (d) clusters with elapsed time

if more than δ times of the chunk size are outside the sphere, and then a new sphere is created. After creating a sphere, if all the spheres have distances less than η, they are merged together and the cluster labels are updated. Boundary samples with an age parameter more than ζ are removed from data. For comparison purposes, these parameters were set as the ones recommended in [24], that is chunk size = 100, $\eta = 1$, $\delta = 0.6$, $\zeta = 2$, trade-off parameter C =0.25, and Gaussian kernel width q = 0.011 and q = 16 for the KDDCUP and Forest-CoverType datasets, respectively. For the four synthetic datasets, SVStream was examined using different parameters. It was noticed that the parameter q highly influenced its outcome. SVStream generated many clusters (up to 100 clusters) or all the samples during the streaming were removed by labeling them as outside boundary samples and sometimes only part of the last chunk was kept. It was obtained that for SVStream, q =10, 12, 0.1, and 0.2 provided the best results for the DS1, DS2, DS3 and DS4 datasets, respectively.

Figures 7.12 and 7.13 summarize the comparison between our OFC clustering algorithm and SVStream, CluStream and DenStream for the four synthetic datasets and the two real datasets KDDCUP and Forest-CoverType, averaged over different time arrangements of the streaming data. In general, one can see that OFC provided higher average purities and NMIs across all the datasets compared to SVStream, DenStream and CluStream. In terms of purity, the improvements over SVStream were 43%, 48%, 24%, 30%, 70% and 15% for the DS1, DS2, DS3, DS4, KDD and Forest-CoverType datasets, respectively. For the DS2, DS3, DS4, KDD and Forest-CoverType datasets, OFC outperformed DenStream by 36%, 44%, 51%, 17% and 18%, respectively. The improvements over CluStream were 14%, 2%, 8%, 17% and 13%, respectively. For the DS1 dataset, OFC performed the same as CluStream and DenStream in terms of purity. A statistical

analysis of variance (ANOVA) based on the t-test was conducted to show the statistical significance of the improvements provided by OFC. This statistical analysis indicated that in terms of purity, OFC generated 9% and 38% improvements at 95% confidence interval over CluStream and SVStream, respectively, and 28% improvement over DenStream at 95% confidence interval.

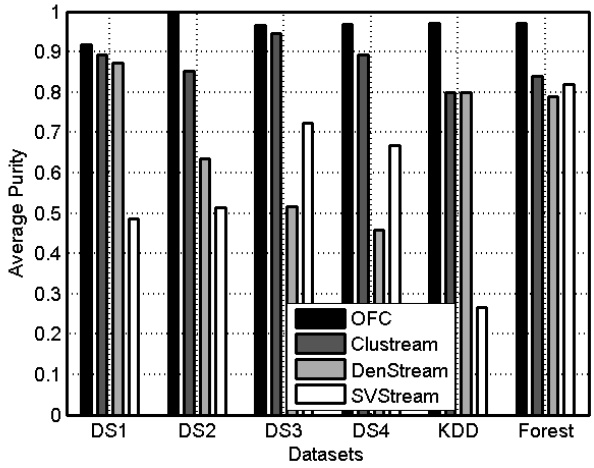

Figure 7.12. Cluster purity (1 represents 100%) comparison between OFC, SVStream, CluStream and DenStream over the four synthetic and two real datasets.

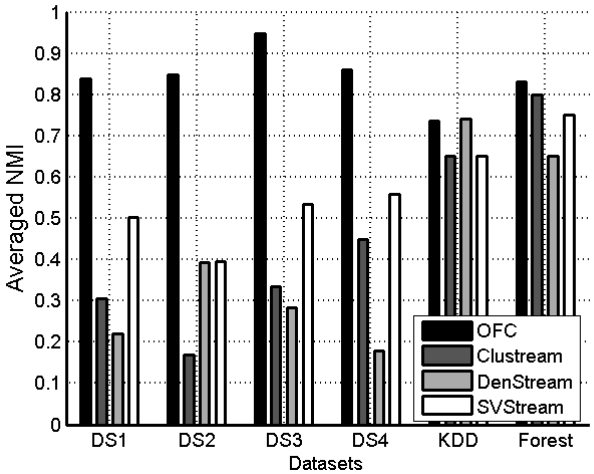

Figure 7.13. Cluster *NMI* comparison between OFC, SVStream, CluStream and DenStream over the four synthetic and two real datasets

The t-test showed the statistical significance of the improvement with the p value < 0.003 across all the six datasets. In terms of NMI, the average improvement over SVStream, DenStream and CluStresm was around 32%, 51% and 46.47%, respectively.

OFC was also compared with SVStream in terms of processing time and memory usage noting that SVStream was stated in [24] to be more computationally efficient than the other clustering algorithms. The algorithm was written in MATLAB running on a Windows operating system PC equipped with a 2.67GHz clock processor. The same machine was used to run all the algorithms. The results obtained are provided in Tables 7.7 and 7.8, exhibiting the computational efficiency of OFC over SVStream. It was found that although SVStream was more efficient in memory usage by 12% on average, the processing time of OFC was at least one-third of the processing time of SVStream. It is worth mentioning that the processing time of SVStream for a typical chunk increased over time during the streaming process as the number of created spheres grew or the dimensionality of the data increased.

Table 7.7. Average processing time for a frame of length N=10 considering the lowest processing time for SVStream

Datasets	OFC	SVStream
	Time (msec)	Time (msec)
DS1	0.043	0.12
DS2	0.040	0.14
DS3	0.048	0.17
DS4	0.047	0.17
KDDCUP	0.060	0.19
Forest-CoverType	0.052	0.082

Table 7.8. Average memory usage of OFC and SVStream algorithms

Datasets	OFC	SVStream
	Memory(kB)	Memory(kB)
DS1	384	316
DS2	384	316
DS3	384	316
DS4	152	72
KDDCUP	1140	1400
Forest-CoverType	500	660

7.5 CONCLUSION

An online clustering algorithm operating on frames of data samples has been introduced in this chapter which allows streaming data to be clustered without knowing the number of clusters or classes. This algorithm provides a major advantage over the existing clustering algorithms designed for streaming data as it does not require the number of clusters to be known. This algorithm allows data samples to be processed and grouped into clusters of arbitrary shapes as the data are received in a streaming fashion in real time in an on-the-fly manner. Experimental results involving four synthetic datasets and two real datasets were carried out which indicated that this new clustering algorithm outperformed the existing clustering algorithms in terms of the cluster purity and normalized mutual information measures. In our future works, we plan to apply this clustering algorithm to specific applications in signal and image processing where it is required to perform clustering in an online and frame-based manner while not knowing the number of clusters.

7.6 APPENDIX A

Table 7.9 provides a list of the notations in the chapter.

Table 7.9. Listing of notations

Notation	Type	Definition
N	$\mathbb{N}$	frame size
L	$\mathbb{N}$	*Chunk* size
t		time
d	$\mathbb{N}$	dimension of input samples
$\mathcal{Y}_{i,t}$	$\mathbb{R}^d$	input sample
F_t	$\mathbb{R}^{d\times N}$	Frame; $F_t = \{\mathcal{Y}_{i,t} \mid \mathcal{Y}_{i,t} \in \mathbb{R}^d, i = 1, \dots N\}, t = 1,2, \dots$
$g_{i,j}(\mathcal{Y}_{i,t}, \mathcal{Y}_{j,}$	$\mathbb{R}^{N\times N} \to \mathbb{R}_0^+$	mutual distance, $g_{i,j}(\mathcal{Y}_{i,t}, \mathcal{Y}_{j,t}) = \left\|\mathcal{Y}_i - \mathcal{Y}_j\right\|_2, i,j = 1, \dots, N$
L_t	$\mathbb{R}_0^+$	minimum value of the *HIMD* of F_t
U_t	$\mathbb{R}_0^+$	maximum value of the *HIMD* of F_t
M_t	$\mathbb{R}_0^+$	peak value of the *HIMD* of F_t
$\overline{g_t}$	$\mathbb{R}_0^+$	average of mutual distances of F_t; $\overline{g_t} = \frac{1}{N^2}\sum_{i,j=1}^{N} g_{i,j}$

Table 7.9. Listing of notations (continued)

$\mathcal{Q}_t$	$\subseteq \mathbb{R}^{N^2}$	$\mathcal{Q}_t \coloneqq \{\mathcal{b} \mid \mathcal{b} > M_t, f_t(\mathcal{b}) = f_t(M_t)\}$,
δ_t	$\mathbb{R}_0^+$	standard deviation of the mutual distances of F_t; $\delta_t = \left(\frac{1}{N^2}\sum_{i,j=1}^{N}\left(g_{i,j} - \overline{g_t}\right)^2\right)^{\frac{1}{2}}$
ϱ	[0,1]	weight value
$f_t(.)$	$\mathbb{R} \to [0,1]$	probability density function of F_t
T_t	$\mathbb{R}_0^+$	threshold for finding potential outliers in F_t
D_t	$\mathbb{R}_0^+$	density of a of F_t; $D_t = \frac{M_t - L_t}{U_t - M_t}$, $t = 1,2, \ldots$
ε_t	$\mathbb{R}_0^+$	radius of neighborhood around potential outliers to search for non-outliers samples
w_t	$\mathbb{N}$	minimum required non-outliers in the ε_t neighborhood of the potential outliers to consider
$\mathcal{N}_t$	$\mathbb{N}$	frame size after outlier removal
$\tilde{\mathrm{F}}_\mathrm{t}$	$\mathbb{R}^{d \times \mathcal{N}_\mathrm{t}}$	$\tilde{\mathrm{F}}_\mathrm{t} \coloneqq \{(\mathcal{Y}_{j,t}, \mu_t, \varepsilon_\mathrm{t}) \mid \mathcal{Y}_{j,t} \in \mathbb{R}^d, \varepsilon_t = M_t + \delta_t, \quad j = 1, \ldots, \mathcal{N}_t\}\}$
μ_t	$\mathbb{R}$	$\mu_t = \frac{1}{\mathcal{N}_t}\sum_{j=1}^{\mathcal{N}_t} \mathcal{Y}_{j,t}$, $t = 1,2, \ldots$
X	$\mathbb{R}^{d \times \mathcal{J}}$	$X \coloneqq \{x_\mathcal{j} \mid x_\mathcal{j} \in \mathbb{R}^d, \mathcal{j} = 1, \ldots, \mathcal{J}\}$, $\mathcal{J} \in \mathbb{N}$
$H(R, a, \xi_\mathcal{j})$	$(\mathbb{R}^+)^3 \to \mathbb{R}$	smallest enclosing hypersphere representing a dataset
R	$\mathbb{R}^+$	radius of the hypersphere
a	$\mathbb{R}^+$	center of the hypersphere
$\xi_\mathcal{j}$	$\mathbb{R}_0^+$	slack variables to punish samples whose distances from the center a are farther than R
$\beta_\mathcal{j}$ and $\alpha_\mathcal{j}$	$\mathbb{R}_0^+$	Lagrange multipliers $\beta_\mathcal{j} \geq 0$, $\alpha_\mathcal{j} \geq 0$
γ	$\mathbb{R}_0^+$	establishes a trade-off between volume and error (accuracy of data description)
$\varphi(x_\mathcal{j})$	$\mathbb{R}^d \to \mathbb{R}^{d'}$	transfer function, d' kernel domain
$\mathcal{K}(x_\mathcal{j}, x_\mathcal{k})$	$\mathbb{R}^{d \times d} \to \mathbb{R}$	kernel function
σ	$\mathbb{R}^+$	Gaussian kernel width
ℓ	$\mathbb{N}$	cluster label
$\bar{\mathcal{S}}$	-	support vectors and their Lagrange multipliers, $\bar{\mathcal{S}} = \{(x_\mathcal{j}, \beta_\mathcal{j}) \mid 0 < \beta_\mathcal{j} < \gamma\}$
ψ	-	sphere structure $\psi = \{\bar{\mathcal{S}}, \|a\|^2, R, \ell\}$
I	$\mathbb{N}$	Number of frames in a micro-cluster
$\mathcal{C}$	-	micro-cluster, $\mathcal{C} \coloneqq \{F_{t_i}, \forall i = 1, \ldots, \mathrm{I}, g_{t_i,t_{i+1}}(\mu_{t_i}, \mu_{t_{i+1}}) < \varepsilon_{\mathrm{t_i}} + \varepsilon_{\mathrm{t_{i+1}}}\}$
U	$\mathbb{N}_0$	number of clusters created by the algorithm
$Purity$	[0,100]	averaged purity of a cluster set; $Purity = \frac{\sum_{\ell=1}^{U}\frac{\lvert\bar{V}_\ell\rvert}{\lvert V_\ell\rvert}}{U} \times 100$
$MI(Q, Q')$	$\mathbb{R}^+$	mutual information of clusters sets Q and Q' ; $MI(Q, Q') = \sum_{q_i \in Q, q'_\ell \in Q'} p(q_i, q'_\ell).\log\frac{p(q_i, q'_\ell)}{p(q_i).p(q'_\ell)}$
$NMI(Q, Q')$	[0,1]	normalized mutual information $NMI(Q, Q') = \frac{MI(Q, Q')}{\sqrt{\mathrm{E(Q)}.\mathrm{E(Q')}}}$
$p(q_i)$	$\mathbb{R} \to [0,1]$	probabilities of samples being in cluster q_i
$p(q'_\ell)$	$\mathbb{R} \to [0,1]$	probabilities of samples being in cluster q'_ℓ
$p(q_i), p(q'_\ell)$	$\mathbb{R} \to [0,1]$	probabilities of samples being in the intersection of q_i and q'_ℓ
E(.)	$\mathbb{R}^+$	entropy function
W	$\mathbb{N}$	number of micro-clusters in CluStream method
IntNo	$\mathbb{N}$	first initial number of data in CluStream and DenStream methods
K	$\mathbb{N}$	number of macro-clusters in CluStream method
h	-	time duration in CluStream method
τ	$\mathbb{R}^+$	elapsed time in DenStream method
λ	$\mathbb{R}^+$	fading factor in DenStream method
$h(.)$	$\mathbb{R}^+ \to \mathbb{R}^+$	fading function in DenStream method, $h(\tau) = 1/2^{\lambda\tau}$
δ	[0,1]	number of outside sphere samples in terms of chunk size, for creating new sphere in SVStream method
C	$\mathbb{R}^+$	Trade-off parameter in SVStream method
η	$\mathbb{R}^+$	merging spheres distance threshold
q	$\mathbb{R}^+$	Gaussian kernel width in SVStream method
ζ	$\mathbb{R}^+$	data age threshold in SVStream method

7.7 REFERENCES

[1] C. D. Wang, J.- H. Lai, "Energy based competitive learning," *Neurocomputing*, vol. 74, no.12-13, pp. 2265-2275, 2011.

[2] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492-1496, Jun. 2014.

[3] J. A. Hartigan and M. A. Wong, "Algorithm AS136: A K-means clustering algorithm," *Applied Statistics*, vol. 28, no. 1, pp. 100-108, 1979.

[4] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recognition*, vol. 41, no.1, pp.176-190, 2008.

[5] G. Tzortzis, A. Likas, "The MinMax k-Means clustering algorithm," *Pattern Recognition*, vol. 47, no. 7, pp. 2505–2516, 2014.

[6] A. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651-666, 2010.

[7] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, "A framework for clustering evolving data streams," *Proceedings of 29th International Conference on Very Large Data Bases*, vol. 29, pp. 81-92, VLDB Endowment, 2003.

[8] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," *Proceedings of 6th SIAM International Conference on Data Mining*, pp. 328-339, 2006.

[9] P. Zhang, X. Zhu, J. Tan, and L. Guo, "Classifier and cluster ensembles for mining concept drifting data streams," *Proceedings of 10th International Conference on Data Mining*, pp. 1175-1180, 2010.

[10] Z. Zhou, W. Zheng, J. Hu, Y. Xu, J. You, "One-pass online learning: a local approach," *Pattern Recognitio*n, vol. 51, pp. 346-357, 2016.

[11] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: theory and practice," *IEEE Transaction on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 515-528, 2003.

[12] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams," *Proceedings of 41st Annual Symposium on Foundations of Computer Science, FOCS*, pp. 359-366, 2000.

[13] C.D. Wang, J.H. Lai, and J.Y. Zhu, "A conscience on-line learning approach for kernel-based clustering," *Proceedings of the 10th International Conference on Data Mining*, pp. 531-540, 2010.

[14] P. Patil, Y. Fatangare and P. Kulkarni, "Semi-supervised learning algorithm for online electricity data streams", *Advances in Intelligent Systems and Computing*, vol. 324, pp. 349-358, 2015.

[15] V. Bhatnagar, S. Kaur, S. and S. Chakravarthy, "Clustering data streams using grid-based synopsis," *Knowledge and Information Systems*, vol. 41, no. 1, pp. 127–152, 2014.

[16] B. Babcock, M. Datar, and R.M.L. O'Callaghan, "Maintaining variance and k-medians over data stream windows," *Proceedings of the 22nd ACM symposium on Principles of database systems*, pp. 234-243, 2003.

[17] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, "On high dimensional projected clustering of data streams," *Data Mining and Knowledge Discovery*, vol. 10, no.3, pp. 251-273, 2005.

[18] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, "A framework for projected clustering of high dimensional data streams," *Proceedings of the 30th International Conference on. Very Large Data Bases*, vol. 30, pp. 852-863, VLDB Endowment, 2004.

[19] Y. Chen and L. Tu, "Density-based clustering for real-time stream data," *Proceedings of the 13th ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining*, pp. 133-142, 2007.

[20] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226-231, 1996.

[21] C.D. Wang and J.H. Lai, "Position regularized support vector domain description," *Pattern Recognition*, vol. 46, no.3, pp. 875–884, 2013.

[22] L. Tu and Y. Chen, "Stream data clustering based on grid density and attraction," *ACM Transaction on Knowledge Discovery from Data*, vol. 3, no. 3, pp. 1-27, July 2009.

[23] S. Luhr and M. Lazarescu, "Incremental clustering of dynamic data streams using connectivity based representative points," *Data and Knowledge Engineering*, vol. 68, no.1, pp. 1-27, 2009.

[24] C. D. Wang, J. H. Lai, D. Huang and W.-S.i Zheng, "SVStream: a support vector based algorithm for clustering data streams," *IEEE Transaction on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1410-1424, 2013.

[25] A. Ben-Hur, D. Horn, H.T. Siegelmann, and V. Vapnik, "Support vector clustering," *The Journal of Machine Learning Research,* vol. 2, pp. 125- 137, 2002.

[26] M. Ankerst, M. Breunig, H. P. Kriegel, and J. Sander. "Optics: ordering points to identify the clustering structure," *Proceedings of International Conference on Management of Data,* pp. 49-60, 1999.

[27] L. Tarassenko, P. Hayton and M. Brady, "Novelty detection for the identification of masses in mammograms," *Proceedings of the 4th International Conference on Artificial Neural Networks,* vol. 4, pp. 442–447, 1995.

[28] L. Parra, G. Deco and S. Miesbach, "Statistical independence and novelty detection with information preserving nonlinear maps," *Neural Computation.*, vol. 8, no. 2, pp. 260–269, 1996.

[29] D. M. J. Tax. "One-class classification," PhD thesis, Delft University of Technology, http://ict.ewi.tudelft.nl/~davidt/thesis.pdf, June 2001.

[30] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.

[31] M. Moya, and D. Hush, "Network constraints and multi-objective optimization for one-class classification," *Neural Networks*, vol. 9, no. 3, pp.463-474, 1996.

[32] D. M. Tax and R.P. Duin, "Support vector domain description," *Pattern Recognition Letters,* vol. 20, no.11, pp. 1191-1199, 1999.

[33] D.Y. Yeung, Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models," *Pattern Recognition,* vol. 36, no.1, pp. 229-243, 2003.

[34] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters,* vol. 33, no. 2, pp. 191-198, Jan. 2012.

[35] X. Zhang, C. Furtlehner, J. Perez, C. Germain-Renaud, and M. Sebag, "Toward autonomic grids: analyzing the job flow with affinity streaming," *Proceedings of the 15th International Conference on Knowledge Discovery and Data Mining*, pp. 987-996, 2009.

[36] Mathworks- http://mathworks.com/matlabcentral/fileexchange/41459 .

[37] S. Hettich and S.D. Bay, "The UCI KDD Archive," Department of Information and Computer Science, University of California, Irvine, CA, http://kdd.ics.uci.edu, 1999.

[38] F. van der Heijden, R.P.W. Duin, D. de Ridder, and D.M.J. Tax, *Classification, Parameter Estimation and State Estimation*. Wiley, 20.

CHAPTER 8

REAL-TIME UNSUPERVISED CLASSIFICATION OF ENVIRONMENTAL NOISE SIGNALS

Authors - Fatemeh Saki, Nasser Kehtarnavaz

The Department of Electrical and Computer Engineering, EC33

The University of Texas at Dallas

800 West Campbell Road

Richardson, Texas 75080-3021

ABSTRACT

This chapter presents a real-time unsupervised classification of environmental noise signals without knowing the number of noise classes or clusters. A previously developed online frame-based clustering algorithm is modified by adding feature extraction, a smoothing step and a fading step. The developed unsupervised classification or clustering is examined in terms of purity of clusters and normalized mutual information. The results obtained for actual noise signals exhibit the effectiveness of the introduced unsupervised classification in terms of both classification outcome and computational efficiency.

8.1 INTRODUCTION

Many studies have been conducted on environmental noise classification, for example [1-6]. A typical noise classifier incorporates two major components: a feature extractor and a classifier. Table 8.1 provides a listing of some representative noise signal classification approaches that have been reported in the literature.

The attribute which is common among all these approaches is the supervised nature of the classification, meaning that a training dataset for each noise environment is first collected in order to train a classifier. Often, a frontend Voice Activity Detector (VAD) is used to separate pure noise signals from speech or speech in noise signals. During testing or operation, the trained classifier is then used to assign an unknown noise signal to a trained noise class based on the features extracted from the unknown noise signal. In these approaches, noise signals for which no training is done get assigned to the closest trained noise class although their noise characteristics may be different than the trained noise classes. This issue becomes of importance in applications, e.g. in hearing devices [2, 5-6], where the detected noise class is used to perform further signal processing such as noise suppression or speech enhancement. For example, in [17], a tunable speech enhancement algorithm was discussed whose parameters were adjusted depending on a number of noise environments to improve both the quality and intelligibility of noisy speech. As another example, in [2], a classifier path was added to the speech processing pipeline of cochlear implants to achieve noise adaptive suppression depending on a number of predefined noise environments.

Table 8.1. Representative previous works on background noise classification

References	Year	Features	Classifier	Dataset
Wang et al. [7]	2014	Non-uniform frequency map	SVM	cat meows, clapping, coughing, double clapping, dogs barking, doorbell ringing, female speeches, frogs croaking, glass breaking, gunshot firing, door knocking, laughing, male speeches, motorcycles revving their engines, pianos playing, screaming, and telephones ringing
Saki et al. [8]	2013	Band-periodicity and band-entropy features	Random forest tree	babble, machinery , street
Khunarsal et al. [9]	2013	Spectrogram pattern matching	NN and k-NN	car engine, construction, crowd applause, crowd clamor, fire, helicopter, office, outdoor sounds- forest, road, restaurant, transportation-motorcycle, transportation -train, water, weather-rain, weather-thunder, household, airplane, water (Ocean), chicken farm, and auto racing
Chu et al. [10]	2012	Mel-frequency cepstral coefficients (MFCC) and matching pursuit (MP)	deep belief network classifier	inside casino, playground, nature-nighttime,nature-daytime, inside restaurants,next to rivers/streams,train passing, inside vehicles, raining, street with traffic, ocean waves, and thundering.
Li et al. [11]	2010	MFCC and matching pursuit (MP)	SVM	sounds of water, sounds of birds, chirpings of insects, roars of mammal, the sounds emitted in certain weather condition, sounds on the street or road with traffic, clamors in shopping centers or supermarkets.
Lozano et al. [12]	2010	MFCC, zero crossing rate (ZCR), centroid and roll-off point with multi-resolution window size	GMM	pans, cups, bottles, china, sprays, phones, clocks, rattles(kara), door locks, shavers and dryers
Chu et al. [13]	2009	MP and MFCC	GMM and K-NN	inside restaurants, playground, street with traffic and pedestrians, train passing, inside moving vehicles, inside casinos, street with police car siren, Street with ambulance siren, nature-daytime, nature-nighttime, ocean waves, running water/stream/river, raining/shower, and thundering
Byeong et al. [14]	2009	MFCC, ZCR, spectral centroid, spectral spread, spectral flatness, spectral flux, change chirp rate spectrum, Hilbert envelope of the analytic signal, and the local energy and discrete curvelet transform	SVM	on the street , on the road, talking, raining, pub/bar, in car
Ntalampiras et al. [15]	2008	MFCC and audio waveform, power, spectrum envelope, spectrum centroid, spectrum spread, spectrum flatness, harmonic ratio, upper limit of harmonicity, audio fundamental frequency	GMM and hidden Markov model (HMM)	aircraft , motorcycle, car , crowd , thunder, wind, train and horns
Kraetzer et al. [16]	2007	63 statistical features computed by AAST	Bayes classifier, K-means clustering	large office, small office, bathroom, laboratory, lecture hall, anechoic chamber, quiet outside environment, busy parking lot, long and narrow corridor, stone stairwell, strong echo

A challenging issue that has not been adequately addressed in the literature is which noise classes or types to consider for the purpose of achieving optimum noise suppression in response to different noise types. In practice, no matter how many noise types or classes are considered, different users of these devices experience different noise environments that vary from user to user. A hearing device can be made more useful if it learns the noise environments that a specific user encounters in his/her daily life in an unsupervised manner. In other words, a hearing device can be made more effective and usable by making it *user-specific*. This means that the hearing device can be designed to automatically learn those noise environments that a specific user encounters.

In [18], we developed an online clustering algorithm, which is capable of defining different clusters on-the-fly with no knowledge about the number of clusters or classes. In this chapter, this clustering algorithm is modified and applied to address the problem of background noise classification in an unsupervised manner. Such an approach enables the development of more advanced hearing devices that learn on their own. In addition, the real-time aspect of deploying this clustering algorithm for online background noise classification is presented. To the best of our knowledge, there exists no unsupervised classification algorithm that is capable of performing real-time background noise classification without knowing the number of classes.

It should be noted that although there exist a number of unsupervised classification or clustering algorithms that are designed for streaming data such as noise signals [19-23], they all require the number of clusters to be specified. Thus, the key element that differentiates the clustering algorithm in [18], named Online Frame-based Clustering (OFC), from the existing clustering algorithms, is that OFC does not require the number of classes or clusters to be specified. In [18],

it was shown that OFC outperformed a recent clustering algorithm SVStream [24] designed for streaming data in terms of both accuracy and computational efficiency. In this chapter, OFC has been modified for the background noise classification application. Therefore, unlike the existing background noise classifiers, it is important to note that the introduced approach does not require any data collection and training phase, and the classification is achieved in an on-the-fly and unsupervised manner without knowing the number of classes.

The rest of the chapter is organized as follows: section 8.2 provides an overview of the previously developed OFC clustering algorithm. section 8.3 covers the modifications made to this algorithm to perform online unsupervised background noise classification. The real-time implementation aspect of the developed unsupervised classification is then reported in section 8.4. Finally, the conclusion is stated in section 8.5.

8.2 OVERVIEW OF ONLINE FRAME-BASED CLUSTERING WITH UNKNOWN NUMBER OF CLUSTERS

In this section, an overview of the OFC clustering algorithm is provided to set the stage for the modifications made to it in section 8.3 to achieve unsupervised background noise classification. A detailed explanation of the OFC algorithm appears in [18].

Figure 8.1 shows the flowchart of the components of the OFC clustering algorithm. In this algorithm, incoming samples are processed frame by frame. Let $F_t \coloneqq \{\mathcal{Y}_{i,t} | \mathcal{Y}_{i,t} \in \mathbb{R}^d, i = 1, \dots N\}$, $t = 1,2, \dots$ represent a frame at time instance *t*, where $\mathcal{Y}_{i,t}$ denotes a *d*-dimensional feature vector at the i^{th} instance of extracting a feature vector. Each frame is first passed through the classifier to see whether it belongs to any of the existing clusters or not. Frames which do not

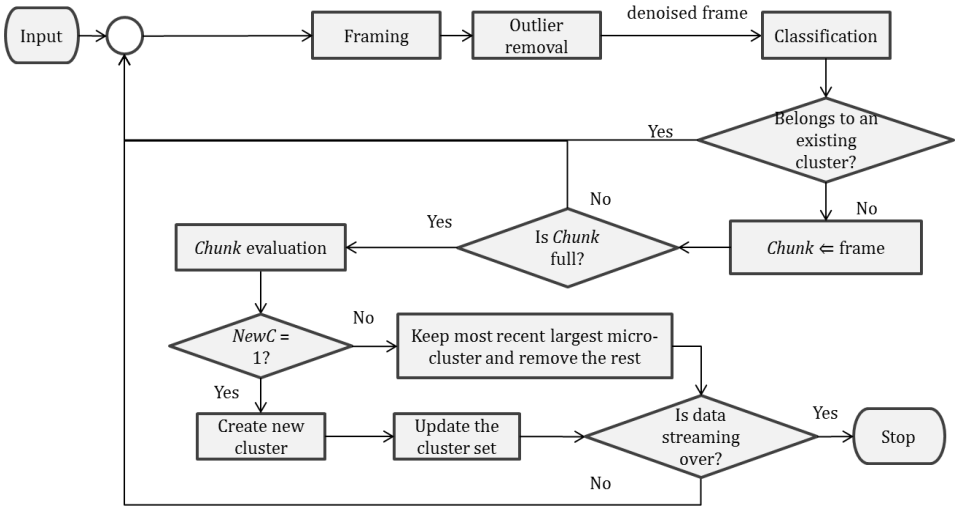

Figure 8.1. Flowchart of the OFC clustering algorithm introduced in [18]

match any existing cluster or class are examined to see whether they belong to a possible new cluster by moving them to another buffer called Chunk. Of course, at the beginning when there is no cluster, all frames go to Chunk. The buffer Chunk is used to keep track of data which belong to a potential new cluster or class. In the ideal case, all frames in Chunk should be from one cluster or class and similar to each other.

Frames in Chunk are checked for similarity. Similarity is defined using the statistical attributes of mean or centroid μ and radius ε of frames, see Figure 8.2. The radius ε of a frame is defined based on the density method described in [18]. This method involves first finding the histogram of mutual distances of feature vectors. Then the radius is set as $\varepsilon_t = P_t + \delta_t$, where P_t denotes the histogram peak value and δ_t the histogram standard deviation. As illustrated in Figure 8.3, similar frames in Chunk can be represented by connected nodes in a graph. Each node would have two direct connections (one to a previous frame and one to a next frame). A collection of connected frames is defined to be a micro-cluster. Note that connections between frames denote similarity and two

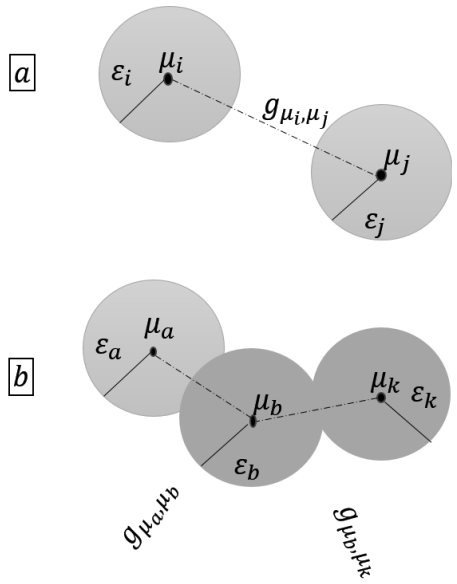

Figure 8.2. (a) Frames *i* and *j* are directly connected, (b), frames *a* and *b*, *b* and *k* are directly connected, while frames *a* and *k* are connected through frame *b*; μ denotes frame center, g_{μ_i,μ_j} distance between frame centers *i &j*, and ε frame radius as defined in [18].

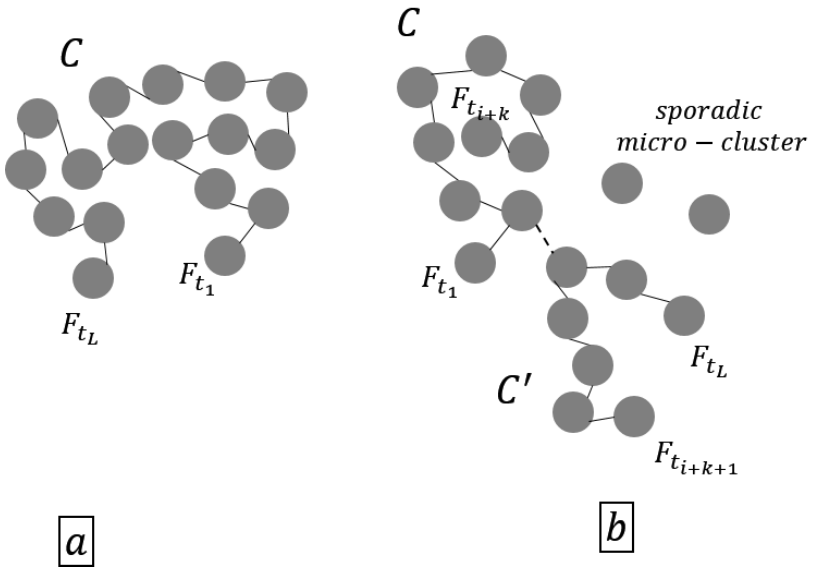

Figure 8.3. (a) A grid of frames in Chunk of length L in an ideal case when all frames are connected and no disconnection exists in Chunk; F_{t_1} corresponds to the first frame and F_{t_L} corresponds to the time that the latest frame gets to Chunk, (b) four micro-clusters in Chunk, two are sporadic and two are connected through a frame.

connected frames are not necessarily the same as two frames that occur consecutively in time. Of

course, in the ideal case, connected frames would occur consecutively in time. However, in practice, due to the presence of data noise, there may be disconnections between frames in Chunk, which means that there could be more than one micro-cluster in Chunk, as illustrated in Figure 8.3(b).

The most prominent micro-cluster in Chunk is then identified and used to create or establish a new cluster. This is achieved by using the Support Vector Domain Descriptors (SVDD) method [25, 26]. This method involves a sphere-shaped data description using nonlinear transformations (kernel functions). SVDD provides the smallest closed boundary or a hypersphere based on a small number of support vectors. As explained in [18], a cluster ψ is defined in terms of the support vectors, sphere center, radius and class label as follows:

$$\psi = \{\bar{\mathcal{S}}, \|a\|^2, R, \ell\} \tag{8.1}$$

where $\bar{\mathcal{S}}$ and a denote support vectors and sphere center, respectively, R sphere radius and ℓ cluster label for a newly created cluster which is specified to be the current number of clusters plus one.

After creating the first cluster or class, each new frame is evaluated to see whether it is inside the boundary of any existing clusters. Frames with their distances to the sphere center smaller than the sphere radius are considered to be inside frames and those larger than the sphere radius are considered to be outside frames. The sphere class label is then assigned to inside frames and the outside frames are moved to Chunk.

8.3 REAL-TIME UNSUPERVISED BACKGROUND NOISE CLASSIFICATION

In this section, the steps and modifications made to the OFC algorithm to perform real-time unsupervised background noise classification are discussed. The intent here is to identify different

background noises or sounds with no training or no prior knowledge of these noises. For this purpose, the following additions or modifications are made to the original OFC algorithm: feature extraction, fading function and classification smoothing. The subsections that follow discuss these additions and modifications.

8.3.1 Feature Extraction

Any noise classification approach includes a feature extraction component which is critical to provide distinguishing characteristics of different environmental noises. As discussed in [27], basically two processing schemes are utilized: sample-based processing and frame-based processing. Frame-based processing is more widely used since often there is not enough information in a single sample to perform clustering. Furthermore, the i/o of commonly used mobile devices is designed to read and write data one frame at a time and not one sample at a time. For unsupervised noise classification discussed in this work, feature vectors are extracted from a number of captured signal frames and then they are combined to form a feature vector frame or an OFC frame that is used by the OFC algorithm. It is worth pointing out that the captured signal frame is different than the feature vector frame. The captured signal frame denotes a frame of input audio signal and the feature vector frame denotes a number of feature vectors extracted from signal frames. Note that the classification decision is made based on the feature vector frame and not the captured signal frame.

The first step in the OFC algorithm is the framing step. In this step, input data samples are buffered to form an OFC or feature vector frame. For the application of environmental noise classification, input data samples correspond to feature vectors which are extracted from captured signal frames. Let γ denote the duration of captured signal frames. Each signal frame is first passed through a

feature extraction step, and then the extracted features are stored/concatenated in a buffer to form a feature vector frame that corresponds to the OFC frame. In other words an ensemble of N feature vectors, $\mathcal{Y}_{i,t}$, are extracted over a time period $[t-\lambda, t]$; $F_t \coloneqq \{\mathcal{Y}_{i,t} | \mathcal{Y}_{i,t} \in \mathbb{R}^d, i = 1, \dots N\}$, where $\mathcal{Y}_{i,t}$ indicates i^{th} feature vector and N denotes the number of concatenated feature vectors over this time duration. The extracted features are briefly described below.

As discussed in [8], the feature vectors of band-periodicity and band-entropy, named subband features, have been found to be effective as noise features that can be computed in a computationally efficient manner. These features are thus used here due to their effectiveness as well as their low computational complexity for the purpose of achieving real-time processing rates on a typical laptop or PC computer.

Based on the sampling rate f_s for captured signal frames of duration γ seconds, the frequency range of frames, that is $[0, \frac{f_s}{2}]$, is divided into a number of B linear non-overlapping subbands. It is worth noting that both Mel filter bank and bark filter bank were also considered but there was no impact on the classification outcome. To compute the band periodicity features, the cross-correlation between every two adjacent frames in each band is computed and then the maximum peak of the cross-correlation denoted by $r_{b,m}$ is used to define the band-periodicity features in band b for duration of $\mathbb{S}$ seconds as follows [28]:

$$BP_b = \frac{1}{M} \sum_{m=1}^{M} r_{b,\mathrm{m}} \ , b = 1, \dots, B \tag{8.2}$$

where $r_{b,\mathrm{m}}$ is the maximum peak of the correlation between two consecutive signal frames at band b and frame m with M denoting the number of captured signal frames in $\mathbb{S}$ seconds. Band-

periodicity features essentially reflect the correlation of each incoming signal frame with a previous signal frame in different bands.

The band-entropy features over $\mathbb{S}$ seconds are computed as follows:

$$BE_b = \frac{1}{\mathrm{M}} \sum_{m=1}^{M} H_{b,m} \tag{8.3}$$

where $H_{b,m}$ denotes the entropy of the Fourier transform of m^{th} frame in band *b*. Note that the computed feature vector at the instance *t* denotes the signal information over the time duration [*t*-$\mathbb{S}$, *t*]. An illustration of the subband feature extraction process is provided in Figure 8.4.

The computed subband feature vectors across a longer time duration $[t-\lambda, t]$; ($\mathbb{S} < \lambda$) are concatenated to form the feature vector frame or OFC frame $F_t := \{\mathcal{Y}_{i,t} | \mathcal{Y}_{i,t} \in \mathbb{R}^d, i = 1, \dots N\}$ (see Figure 8.4) that is used in the OFC algorithm, where $\mathcal{Y}_{i,t}$ indicates i^{th} feature vector with $d = 2 \times B$ (for each band, two features of band-periodicity and band-entropy are computed) and N denotes the number of concatenated subband feature vectors over the last λ seconds. Naturally, N depends on $\mathbb{S}$ for a fixed λ. The feature vector frame F_t is then fed into the OFC algorithm to establish the background noise class at time *t*. It is worth mentioning that the developed unsupervised classification solution is general purpose in the sense that it allows any other set of features to be used in place of subband features.

In the classification step, the distances of the OFC frame F_t at time *t* to the existing sphere centers is computed. For the cluster label at time *t*, first the closest sphere to F_t is identified based on this distance

$$d_{t,\ell} = \frac{\|\mu_{F_t} - a_\ell\|_2}{R_\ell^2} \tag{8.4}$$

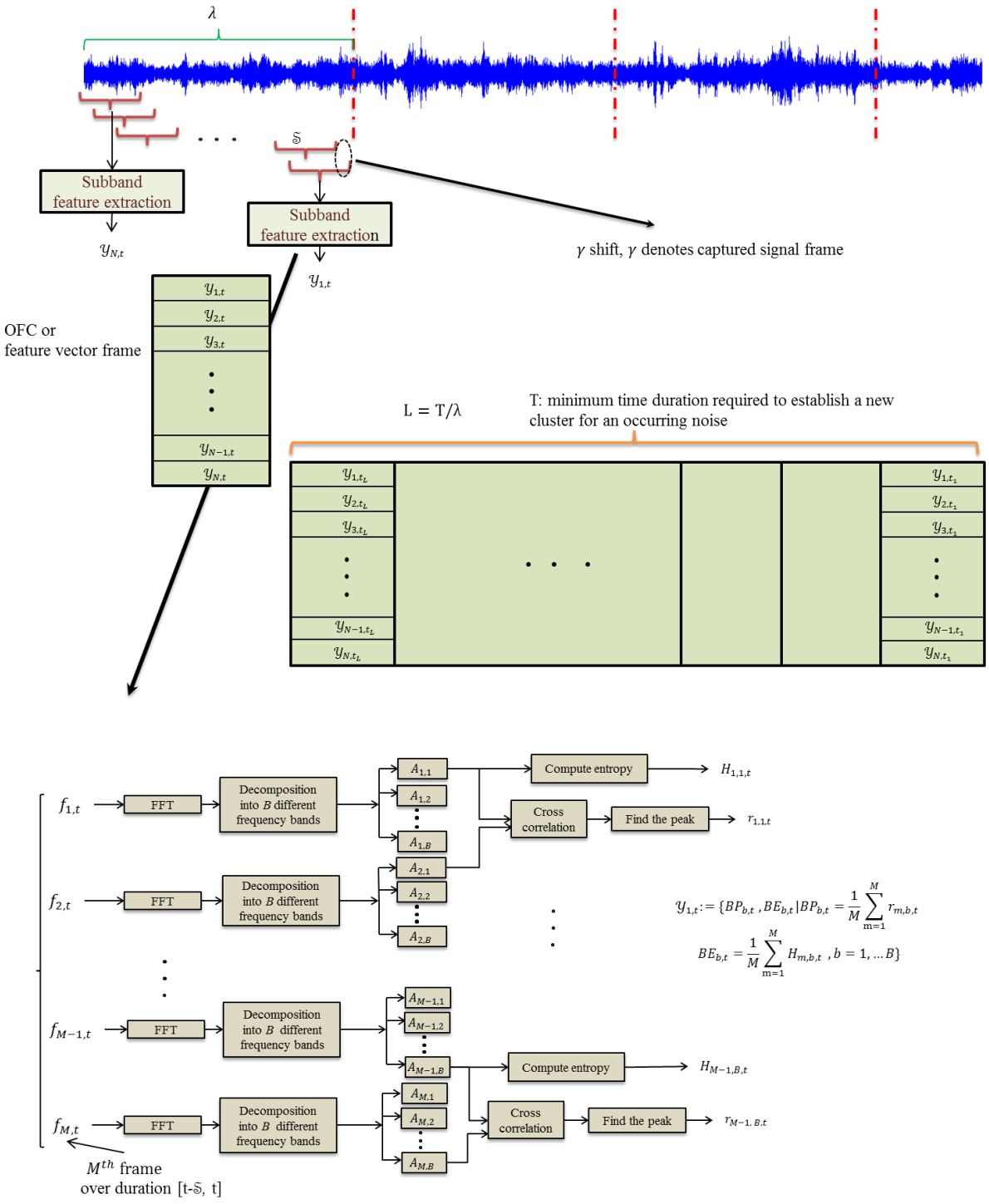

Figure 8.4. Subband feature extraction process

This distance denotes the normalized distance of frame F_t and cluster ℓ, μ_{F_t} represents the center

of frame F_t and a_ℓ & R_ℓ represent the sphere center and radius of a cluster or class ℓ, respectively. If the number of feature vectors falling inside the closest sphere is greater than the number of feature vectors falling outside, that F_t is assigned to the corresponding sphere. Otherwise, a new cluster flag is activated indicating that data from a new cluster or class is coming in and the feature vector frames are moved to Chunk.

8.3.2 Fading Function

Feature vector frames which appear in Chunk indicate data from a potential unseen cluster or class. When Chunk gets full, a new cluster or class gets created. Before adding a new cluster, the data in Chunk need to be evaluated in terms of homogeneity and similarity. In practice, due to data noise, some dissimilarity and disconnections between parts of the data in Chunk may occur. To address this issue, only the homogenous and connected data in Chunk are used to form a micro-cluster. Assuming that the size of a micro-cluster meets the size specified for creating a new cluster, a flag named *NewC* is used to create a new cluster and Chunk is emptied. Otherwise, a micro-cluster stays linked to Chunk and later when Chunk gets filled, it is examined again.

In practice, sound data occur in a streaming manner or continuously. Thus, it may happen that data from one unseen noise environment get to Chunk, but the data stream is not long enough to fill Chunk and the rest of Chunk may get filled with another unseen cluster data. Such data are not regarded as valid data for cluster creation. As a result, it is more effective to keep only useful data before running the Chunk evaluation. A simple computation is thus performed to remove the data that stay in Chunk for a long time. To this end, before checking the similarity between the frames

in Chunk, the age of the feature vector frames in Chunk are examined by using a fading function mentioned next.

To check the age of the feature vector frames in Chunk, a time-sensitive weight for a frame is defined by considering a time fading function $h(\tau)$, which assigns lower weights to older frames and higher weights to newer frames:

$$h(\tau) = \exp(-\log_2(\tau + 1)) \tag{8.5}$$

where τ denotes the age of the feature vector frames in Chunk. The age of a current frame is $\tau = 0$, or the corresponding fading weight is one. Over time, the frame's age increases. The maximum accepted elapsed time for frames can be adjusted based on the application at hand by setting a threshold $\tau = \Theta$, where Θ is defined in terms of Chunk size. As a result, frames that are older than $\exp(-\log_2(\Theta + 1))$ get removed from Chunk.

8.3.3 Classification Smoothing

Another modification that is made here is classification smoothing. Because of the presence of noisy data, in practice, fluctuations in the classification outcome might occur which cause noisy feature vector frames to be placed into Chunk. To avoid or reduce this possibility, a smoothing step is added to the clustering decision outcome. This smoothing reflects the situation that if frame F_t is from class ℓ and frame F_{t+2} is from class ℓ as well, then frame F_{t+1} is expected to be from the same class, noting that signal data occur in a streaming manner and the interest is primarily on sustained type of noise. The smoothing step consists of first applying a median filter of length 3 for removing spikes in classification and then applying a majority voting across a window of size W. Majority vote reflects the detected class at time *t* is the most detected class over the time

duration of $W \times \lambda$ seconds. This step copes with noisy data getting into Chunk, and prevents filling Chunk with noisy data and running unnecessary Chunk evaluations.

8.3.4 Parameter Setting

In this subsection, a set of guidelines are provided as how the parameters of the algorithm can be set when running the unsupervised noise classification in real-time. These parameters include the captured signal frame duration γ, the segment length $\mathbb{S}$ for computing the subband features, the Chunk size L, the smoothing window size W, the age threshold Θ for frames in Chunk, and finally the number of bands B for feature extraction.

In many audio applications, frames durations of 10-40 msec are considered since over this time duration, sound characteristics remain mostly statistically stationary. On the other hand, such short sound segments may not carry adequate information for classification purposes. Often, the extracted features from a noise signal over at least 100-500 msec would be needed to indicate the noise type. In the experimentations reported in this work, the captured signal frames was considered to be $\gamma = 25$ msec and the segment length was considered to be $\mathbb{S}$=100 msec. Then, the extracted subband feature vectors over duration λ =500 msec were concatenated to form an OFC or a feature vector frame at which rate a decision was made.

To set the length of the smoothing window W for the majority voting decision, one needs to take into consideration how frequent the noise environment changes or how fast the noise environment is to be updated. For a user moving between two different noise environments, a reasonable update rate would be every two to three seconds, that is [t-$W \times \lambda$,t] with at least $W \times \lambda$= 2~3 seconds, i.e. W=4 to 6 when λ duration is 500 msec.

To set Chunk size, one needs to specify how long an occurring noise is to be sustained in order to establish a new cluster. In other words, if a user gets to a new noise environment, how long the noise needs to last for a new noise cluster to get established. For our experimentations reported in the next section, this time was assigned to be at least 5 seconds. Note that this is a user-specified parameter. Expressing the Chunk size in terms of the number of feature vector or OFC frames for the duration λ =500 msec, one gets L= 10. The longer the Chunk size, the more information about a new cluster is made available. However, to accommodate for the real-time implementation aspect of creating a cluster in an on-the-fly manner, one needs to keep in mind the tradeoff between Chunk size and computational complexity. This is due to the fact that a longer Chunk size causes more delay in the creation of a new cluster.

For subband feature extraction, the number of bands used in [8] was 8 and thus 16 band-periodicity and band-entropy features were used for background noise classification. Here 2, 4 and 8 bands were considered and it was found that 4 and 8 bands were adequate for the application under consideration. That is why only 4 bands, corresponding to 8 band-periodicity and entropy features, were used for computational efficiency. The interested reader is referred to [8] for more details on the feature extraction parameters.

Finally, the age threshold for each feature vector frame in Chunk depends on the Chunk size. For a new cluster to get added when data from a noisy data are received, the maximum elapsed time between the oldest feature vector frame in Chunk and a current incoming feature vector frame in Chunk was set here to $\Theta = 2 \times L$ to accommodate for interruptions when placing new data into Chunk.

8.4 EXPERIMENTAL RESULTS AND REAL-TIME OUTCOME

In this section, the experimental results of the developed background noise unsupervised classification algorithm as well as its comparison with the SVStream algorithm are presented. In the first experiment, the dataset in [2] was used to carry out the classification without any training. This dataset consisted of five different background noise classes of babble, driving car, machinery, train, and street. These signals were then fed into the unsupervised classifier in a streaming manner and the following cluster purity measure described in [21] was computed:

$$Purity = \frac{\sum_{\ell=1}^{U} \frac{|\widehat{V_\ell}|}{|V_\ell|}}{U} \times 100 \tag{8.6}$$

where U denotes the number of clusters, $\left|\widehat{V_\ell}\right|$ indicates the number of samples with the dominant class label in cluster ℓ, and $|V_\ell|$ indicates the total number of samples in cluster ℓ. Basically, this measure indicates the purity of the identified clusters with respect to the groundtruth or the true clusters. It is important to bear in mind that of course in actual operation or in practice, the number of clusters is unknown or the groundtruth clusters are not known.

Another measure that was used here to evaluate the performance of the developed unsupervised classification algorithm was normalized mutual information (NMI). NMI uses the mutual information between two cluster sets, i.e. the groundtruth clusters and the created clusters, by comparing the clusters of each set one by one. This measure indicates the similarity of the created clusters with respect to the groundtruth clusters. Note that in practice the groundtruth clusters are not known. Let Q and Q' denote the cluster sets corresponding to the groundtruth clusters and the clusters generated by the developed algorithm, respectively. The normalized mutual information $NMI(Q, Q')$ is given by

$$NMI(Q,Q') = \frac{MI(Q,Q')}{\sqrt{\mathrm{E(Q).E(Q')}}} \tag{8.7}$$

$$MI(Q,Q') = \sum_{q_i \in Q, q'_\ell \in Q'} p(q_i, q'_\ell).\log \frac{p(q_i,q'_\ell)}{p(q_i).p(q'_\ell)} \tag{8.8}$$

where $p(q_i)$, $p(q'_\ell)$ and $p(q_i, q'_\ell)$ denote the probabilities of samples being in the clusters q_i, q'_ℓ, and the intersection of q_i and q'_ℓ, respectively, with $\mathrm{E(Q)}$ and $\mathrm{E(Q')}$ indicating the entropies of the clusters.

8.4.1 Parameters Setting Experiments

In this subsection, a comparison between the original OFC, OFC+Smoothing, and OFC+Fading and OFC+Smoothing+Fading for different Chunk sizes *L* and segments lengths $\mathbb{S}$ is provided. Tables 8.2-8.9 summarize the effect of different parameter settings. In these tables, the average *Purity* and *NMI* values for different Chunk sizes *L* and segment lengths $\mathbb{S}$ are provided for the original OFC, OFC+Smoothing, OFC+Fading and OFC+Smoothing+Fading versions. The study of the effect of changing the three parameters of the algorithm, that is Chunk size *L*, segments length $\mathbb{S}$ for subband feature extraction, and smoothing window W, is reported for different Chunk sizes $4 \leq L \leq 30$ and durations $60 \leq \mathbb{S} \leq 240$msec for no smoothing window, i.e. $W=1$, as well as for these smoothing window sizes $2 \leq W \leq 10$.

The effect of changing the Chunk size and the segment duration $\mathbb{S}$ on the number of times that the clustering algorithm generated the right number of clusters was also examined. It was found that on average, the original OFC algorithm provided lower *Purity* and NMI values and these values were sensitive to changes in the Chunk size and segment length, and 21% of the time the number of clusters matched the number of groundtruth clusters. By adding only the Smoothing step to the

Table 8.2. Average cluster *Purity* measure for chunk size *L* versus feature extraction segment length $\mathbb{S}$ for OFC + Smoothing + Fading

Chunk Size *L* / Segment length $\mathbb{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	96.9	95.5	96.2	96.2	96.3	96.5	96.5	96.5	96.5	96.5	91.6	97.6	95.9	95.9
100	**97.1**	**98.2**	**98.1**	**89.1**	**98.3**	**98**	**98**	**98.2**	**97.2**	**97.3**	**97**	**97**	**95.3**	**97**
160	93.3	97.5	96.7	97.7	98	97.6	97.6	98.7	95	93.2	98.9	99.4	97.7	98
200	80	95.2	83.7	90.6	97.7	95.2	97.5	93	98.3	98.1	96.8	90.4	98	97.3

Table 8.3. Average cluster *Purity* measure for chunk size *L* versus feature extraction segment length $\mathbb{S}$ for OFC + Smoothing

Chunk Size *L* / Segment length $\mathbb{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	96.9	31.3	30	33	65.6	32.9	46.8	41.1	33.5	40.7	39.9	90.1	56.4	56.6
100	90.5	94.1	89.9	93.2	69.6	86.7	97.7	98.1	97.5	97.7	97.7	85.9	85.4	85.2
160	90.5	88.6	98.1	94.5	95.6	96.5	97.3	97.2	97.3	93.1	96.7	98.6	98.1	97.9
200	90.8	94.7	93.6	94.2	90.4	92.8	94.9	91	96.6	90.7	91.1	92.9	94.9	97.3

Table 8.4. Average cluster *Purity* measure for chunk size *L* versus feature extraction segment length $\mathbb{S}$ for OFC + Fading

Chunk Size *L* / Segment length $\mathbb{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	87.1	48.3	83.9	58	96.1	72.6	56.3	37.7	42.8	42.5	48.7	66	91.5	96.2
100	83.3	89.7	93.8	87	98.3	86.8	90.1	97.5	97.2	97.8	97.9	97.9	97.5	97.9
160	89.2	98.4	90.3	98.4	97.6	98.4	99	97.9	95.9	97.2	97.9	98.9	98.9	98.9
200	92.1	90	91.2	88.7	97.7	92	97.3	97.7	98.3	97.3	97.8	94.9	97.1	96

Table 8.5. Average cluster *Purity* measure for chunk size *L* versus feature extraction segment length $\mathbb{S}$ for original OFC

Chunk Size *L* / Segment length $\mathbb{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	91.7	78.8	84.1	91.2	66	65.9	65.6	36.5	36.5	47	36.5	41.1	79	64.2
100	97.3	94.7	93	97	92.4	90.5	89.4	92.4	82.3	87.7	86.1	81.1	97.5	97.5
160	93.6	92.6	89.8	92.6	91.7	91.1	95.3	92.2	93.1	94.6	93.1	93.2	92	91.2
200	95.1	96	88.5	89.6	91.1	94.8	93.1	91.1	97.4	93.1	94.7	96.5	92.2	92

OFC algorithm, the matching percentage time was increased to 25%. By keeping the Chunk size and subband segment length fixed and by varying the smoothing window size, it was seen that in general these measures increased slightly and became less sensitive to the variations in the Chunk size and subband segment length. Although by setting large window sizes, these measures were slightly increased, the reaction to background noise changes was delayed. Therefore, in our

Table 8.6. Average cluster NMI measure for chunk size L versus feature extraction segment length $\mathcal{S}$ for OFC + Smoothing + Fading

Chunk Size L / Segment length $\mathcal{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	93.4	91.3	92.4	92.6	92.9	93	93	93	93	92.9	89.1	94.3	92.1	92.2
100	**93**	**91**	**93**	**89**	**94**	**93**	**93**	**93**	**93**	**93**	**93**	**95**	**92**	**93**
160	82.9	79.3	90.9	84.5	87.3	84.8	88.9	91.5	90	89.9	97.1	95.7	92.6	93.6
200	83.5	77.8	76.2	90.6	84.9	79.9	87.8	85.3	94	92.8	88.1	89.6	89.9	90.5

Table 8.7. Average cluster NMI measure for chunk size L versus feature extraction segment length $\mathcal{S}$ for OFC + Smoothing

Chunk Size L / Segment length $\mathcal{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	93.4	75.9	74.7	74.5	74.3	74.6	74.2	74.6	77.3	82.7	82.8	87	87	87.2
100	78.7	95	87	92.9	89.6	89.1	92.4	93.6	92.5	92.9	93.1	88.2	87.1	87
160	83.6	82.1	88.7	83.3	79.8	83.8	85.1	88.1	87.1	87.5	92.4	96.4	92.7	93
200	73.3	79.3	75.6	85.2	73.7	78.5	81.6	81.7	85.7	84.3	88.1	87.1	86.7	90.5

Table 8.8. Average cluster NMI measure for chunk size L versus feature extraction segment length $\mathcal{S}$ for OFC + Fading

Chunk Size L / Segment length $\mathcal{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	86.8	91.3	90.6	92.7	91.8	92.9	77.1	77.5	78.7	77.7	78.1	89.5	90.8	91.2
100	93.2	89.7	88.1	85.3	92.2	85.2	90	93.7	92.2	93.3	93.6	95.3	93	93.6
160	85.3	80.8	90.4	86.1	86.9	88.4	89.8	87.4	88.1	90.1	91.6	95.2	95.2	94.7
200	87.8	81.2	84	85.2	83.7	83.1	82.1	85.7	88.1	86.4	87.3	86.9	91.9	86.5

Table 8.9. Average cluster NMI measure for chunk size L versus feature extraction segment length $\mathcal{S}$ for original OFC

Chunk Size L / Segment length $\mathcal{S}$	4	6	8	10	12	14	16	18	20	22	24	26	28	30
60	88.3	87	90.3	92	74.7	74.6	74.2	74.3	74.5	74.5	74.4	74.7	74.6	83.4
100	93.5	83.7	82.8	86.8	83.8	83.5	85.7	90.9	76.6	75.8	77	76	92.8	94.7
160	78	76.1	86.4	77.6	78.8	79.8	85.3	82.5	81.3	83.5	83.3	91.1	88.4	88
200	80.3	83.1	77.7	82.9	77.6	79.6	78.2	78.5	90.9	82.2	83.1	83.4	84.9	89.4

experimentations, $W = 5$ was considered as this setting allowed the background noises to be updated every 2.5 seconds and smoothing window sizes larger than 5 provided more or less similar results. The cluster purity for $W = 5$ across different Chunk sizes and segments lengths was examined and it was found that Chunk sizes $L \geq 12$ and subband segment lengths $100 \leq \mathcal{S} \leq 220$msec provided the best performance in terms of correctly detected number of clusters and accuracy of

clustering with *Purity* ≅ 96% and NMI ≅ 0.92. When the Fading function was added to the OFC algorithm, the matching percentage was improved to 36%, whereas after adding the Smoothing step and the Fading function together to the OFC, the matching percentage reached 52%. Figure 8.5 shows the improvement for each of the five classes separately. Note that having larger Chunk sizes ($L \geq 12$) led to a cluster set closer to the groundtruth cluster set and therefore to higher cluster purity even when the size of the smoothing window was kept small. It was found that when L was set small for a noise environment, usually more than one cluster was created, where these clusters mostly included data from only one class. Hence, the purity and NMI values stayed more or less the same. The larger the Chunk size was set, the more information about the input noise environments and thus more accurate clusters were obtained. However, larger Chunk sizes created higher time delays in the creation of new clusters. A trade-off between the Chunk size and real-time throughput was established by setting L=20. It should be noted that this tradeoff is very much application dependent and can be set by the user depending on the application. Finally, it is worth

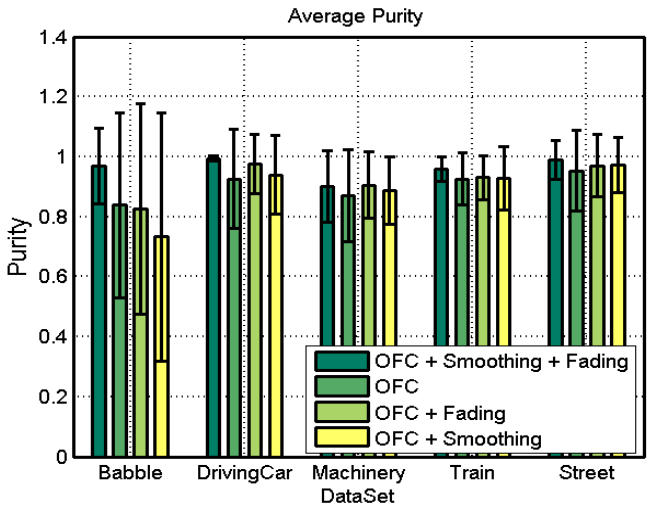

Figure 8.5. Average *Purity* value for five different classes using original OFC, OFC+Smoothing, OFC+Fading and OFC+Smoothing+Fading

mentioning that by having smoothing window sizes higher than 5, changes in the segment length $\mathcal{S}$ had a negligible impact on the performance, thus $\mathcal{S}$ =100 msec was considered in our subsequent experimentations. It is important to note that although different applications may require different parameters than the above, the guidelines discussed are applicable to any other application of interest.

Finally, the effect of changing the value Θ was also studied. It was found that as long as appropriate values for the Chunk size L and segment duration $\mathcal{S}$ were selected, Θ could be selected to be any value higher than $\Theta = 2 \times L$. For the application under consideration in this work, by considering the Chunk size to be 20 and the OFC frame duration to be 500 msec, the maximum age gap between the feature vector frames in Chunk for $\Theta = 2 \times L$ was set to $2 \times L \times \lambda = 2 \times 20 \times 500 = 20$ seconds. This value indicated that the gap age was not more than 20 seconds between the clustering decision instances.

8.4.2 Clustering Evaluation

Figure 8.6 shows the performance of the algorithm in terms of classification rate, new cluster creation with the smoothing step using the parameter setting as noted above $\mathcal{S}$ =100 msec, L=20, W =5 for a typical experiment. In this figure, the class labels 1, 2, 3, 4 and 5 refer to babble, driving car, machinery, train and street classes, respectively. Note that the label 0 means the OFC frame is not assigned to any of the existing clusters. In this set of experiments, the groundtruth clusters were changed in on-the-fly manner and the developed algorithm was used to identify the clusters without any training. The confusion matrix for this experiment after applying the entire sound file or clusters is shown in Table 8.10 in terms of all the misclassification errors. Table 8.11 shows the

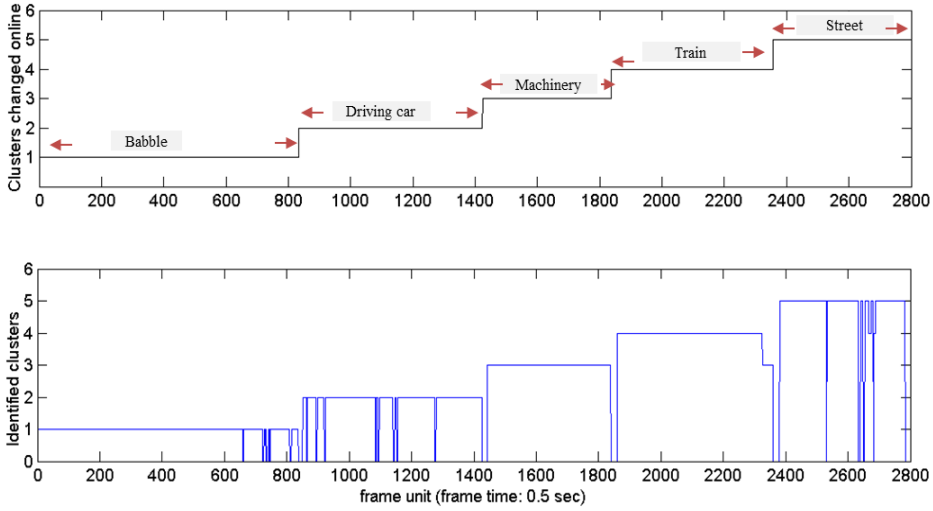

Figure 8.6. A typical classification outcome for different background noises in terms of actual clusters and created clusters

performance of the algorithm for seven different experiments consisting of different sound files in terms of clustering purity, *NMI,* processing time per feature vector frame, and the online identified number of clusters versus the actual number of clusters. In the sound files used for mall and church, there existed many variations of sound which led to more than one cluster to be created for these environments.

8.4.3 Real-time Field Testing

Field testing of the unsupervised classification algorithm was also conducted to verify its real-time throughput by checking to see whether any captured signal frames would get skipped. The

Table 8.10. Typical confusion matrix in percentages for the classification outcome reported in Figure 8.6

Detected class / Actual class	Restaurant	Driving car	Machinery	Train	Street
Restaurant	99.7	0	0	0	0.3
Driving car	0	99.6	0.4	0	0
Machinery	0	1.5	98.5	0	0
Train	0	0	8.6	91.4	0
Street	0	0	0	2	98

Table 8.11. Average cluster purity and *NMI* measures across seven different sound files

Environments	*Purity*	*NMI*	Actual number of clusters	Online identified number of clusters	Processing time (msec)
Restaurant, Driving car, Machinery	0.99	0.98	3	3	59
Train, Street Church	0.98	0.87	3	4	59
Driving car, Train, Plain	0.99	0.97	3	3	59
Restaurant, Pub, Mall	0.99	0.93	3	4	59
Fan, Plane, Office	0.99	0.98	3	3	59
Restaurant, Driving car, Machinery, Plane, Street	0.98	0.96	5	5	60.9
Restaurant, Driving car, Machinery, Plane, Vacuum, Train, Street, Church	0.90	0.88	8	9	65.3

algorithm was run in real-time on a laptop platform equipped with a 2.4GHz clock processor while taking the laptop to three environments of restaurant, driving car, and office in a random order. The background noise signals were captured using the laptop microphone. The results of a typical run of the algorithm are shown in Figure 8.7. When the laptop was taken to a new noise

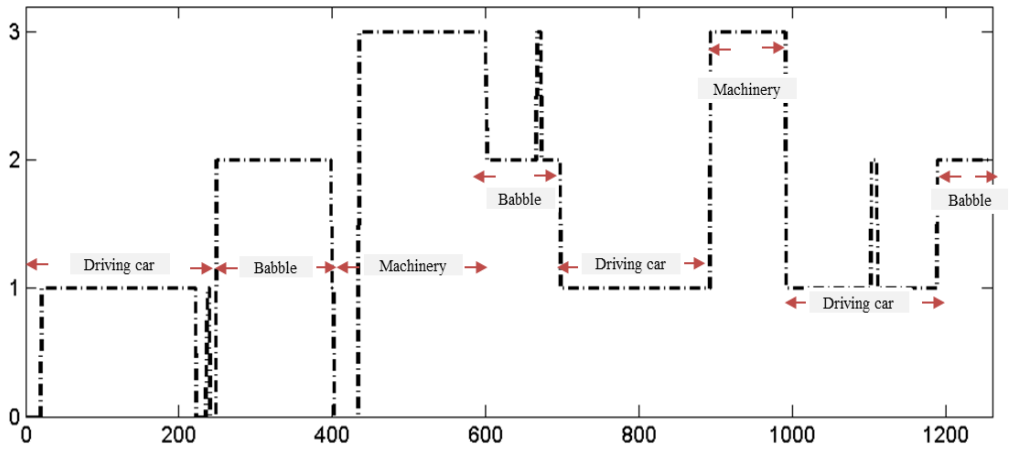

Figure 8.7. A typical field testing outcome of the developed unsupervised noise signal classification running in real-time; x-axis denotes frames and y-axis denotes cluster label

environment for the first time, a new cluster was detected and thus a new cluster was created for that environment. Then, when returning to a previously identified noise environment, the algorithm detected that the incoming signal frames were from an existing cluster or class and assigned them accordingly. The total processing time of a typical OFC or decision frame for the entire processing pipeline was less than 90 msec, which included the time for generating the OFC frame and performing the clustering. It is worth pointing out regardless of the number of clusters created, it took the same processing time of 90 msec for 500 msec OFC or decision frames.

A comparison with the recently introduced online clustering algorithm SVStream was also conducted. It was found that the computational time for a signal of a typical decision frame duration of 500 msec increased over time during the streaming process when using the SVStream algorithm and this time became as high as 10 seconds as the number of created spheres grew in this algorithm. In terms of the cluster purity measure, for the same dataset and the same parameters,

it was found that the SVStream algorithm generated close to 200 clusters which did not correspond to the true number of noise clusters or classes.

A videoclip of the algorithm running in real-time on the laptop can be seen at the website

www.utdallas.edu/~kehtar/UnsupervisedClassificationNoise.wmv

8.5 CONCLUSION

A real-time unsupervised background noise classification algorithm has been developed in this chapter which allows environmental background noise signals to be classified in an online fashion without having any knowledge of the number of clusters or noise classes. Compared to the existing noise classification algorithms, the developed algorithm has the advantages of not requiring any training and also not requiring the number of noise classes to be specified. The performance of the algorithm was assessed by examining actual noise signals in real-time and in an on-the-fly manner. The experimental results have indicated the effectiveness of this algorithm in terms of both clustering performance and computational efficiency. Our future work involves implementing this online noise classification algorithm on smartphone platforms as an app to allow its utilization in a user-specific manner.

8.6 REFERENCES

[1] E. Alexandre, L. Cuadra, M. Rosa, and F. López-Ferreras, "Feature selection for sound classification in hearing aids through restricted search driven by genetic algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2249–2256, 2007.

[2] V. Gopalakrishna, N. Kehtarnavaz, T. S. Mirzahasanloo, and P. C. Loizou, "Real-time automatic tuning of noise suppression algorithms for cochlear implant applications," *IEEE Transactions on Biomedical Engineering*, vol. 59, pp. 1691-1700, 2012.

[3] P. Nordqvist, and A. Leijon, "An efficient robust sound classification algorithm for hearing aids," *The Journal of the Acoustical Society of America*, vol. 115, pp. 3033–3041, 2004.

[4] E. Alexandre, L. Cuadra, L. Álvarez, M. Zurera, and F. Ferreras, "Automatic sound classification for improving speech intelligibility in hearing aids using a layered structure," in *Lecture Notes in Computer Science*. vol. 4224, New York: Springer-Verlag, 2006.

[5] Y. Hu and P. Loizou, "Environment-specific noise suppression for improved speech intelligibility by cochlear implant users," *The Journal of the Acoustical Society of America*, vol. 127, no. 6, pp. 3689–3695, 2010.

[6] F. Saki, T. Mirzahasanloo, and N. Kehtarnavaz, "A multi-band environment-adaptive approach to noise suppression for cochlear implants," *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, (EMBC)*, pp. 1699–1702, 2014.

[7] J-C. Wang, J-F. Wang, C-H. Lin, B-W. Chen, and M. Tsai, "Gabor-based nonuniform scale-frequency map for environmental sound classification in home automation," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 607-613, April 2014.

[8] F. Saki and N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

[9] P. Khunarsal, C. Lursinsap, and T. Raicharoen, "Very short time environmental sound classification based on spectrogram pattern matching," *Journal of Information Sciences*, vol. 243, pp. 57-74, 2013.

[10] S. Chu, S. Narayanan, C. Kuo, "Composite-dbn for recognition of environmental contexts," *Proceedings of Signal Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Hollywood, CA, pp. 1 –4, 2012.

[11] Y. Li and Y. Li, "Eco-Environmental Sound Classification Based on Matching Pursuit and Support Vector Machine," *Proceedings of the 2nd International Conference on Information Engineering and Computer Science,* no. 61075022, pp. 1–4, 2010.

[12] H. Lozano, I. Hernáez, and A. Picón, "Audio classification techniques in home environments for elderly/dependent people," *Proceedings of the International Conference on Computers for Handicapped Persons*, pp. 320–323, 2010.

[13] S. Chu, S. Member, S. Narayanan, and C. J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 17, no. 6, pp. 1142–1158, 2009.

[14] H. Byeong-jun, H. Eenjun, "Environmental sound classification based on feature collaboration," *Proceedings of IEEE International Conference on Multimedia and Expo*, pp. 542–545, 2009.

[15] S. Ntalampiras, I. Potamitis, N. Fakotakis, "Automatic recognition of urban environmental Events," *Proceedings of International Association for Pattern Recognition Workshop on Cognitive Information Processing,* pp. 110–113, 2008.

[16] C. Kraetzer, A. Oermann, J. Dittmann, and A. Lang, "Digital audio forensics: A first practical evaluation on microphone and environment classification," *Proceedings of the 9th Workshop on Multimedia and Security (ACM),* pp. 63–74, 2007.

[17] G. Kim and P. Loizou, "Improving speech intelligibility in noise using environment-optimized algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 2080–2090, 2010.

[18] F. Saki and N. Kehtarnavaz, "Online frame-based clustering with unknown number of clusters," *Pattern Recognition*, vol. 57, pp. 70-83, 2016.

[19] C. Aggarwal, J. Han, J. Wang, and P. Yu, "A framework for clustering evolving data streams," *Proceedings of the 29th International Conference on Very Large Data Bases*, vol. 29, VLDB Endowment, pp. 81–92, 2003.

[20] C. Aggarwal, J. Han, J. Wang, and P. Yu, "On high dimensional projected clustering of data streams," *Data Mining and Knowledge Discovery*, vol. 10, pp. 251-273, 2005.

[21] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," *Proceedings of the 6th SIAM International Conference on Data Mining*, pp. 328-339, 2006.

[22] P. Patil, Y. Fatangare and P. Kulkarni, "Semi-supervised learning algorithm for online electricity data streams," *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, vol. 324, pp 349-358, 2015.

[23] C. D. Wang and J. Lai, "Position regularized Support Vector Domain Description," *Pattern Recognition,* vol. 46, no. 3, pp. 875–884, 2013.

[24] C. D. Wang, J. H. Lai, D. Huang and W.-S.i Zheng, "SVStream: a support vector based algorithm for clustering data streams," *IEEE Transaction on Knowledge and Data Engineering,* vol. 25, no. 6, pp. 1410-1424, 2013.

[25] D. Tax and R. Duin, "Support vector domain description," *Pattern Recognition Letter,* vol. 20, no.11, pp. 1191-1199, 1999.

[26] D. Tax, *One-Class Classification*, PhD thesis, Delft University of Technology, http://ict.ewi.tudelft.nl/~davidt/thesis.pdf, June 2001.

[27] S. Chachada and C. Kuo, "Environmental sound recognition: A survey," *Proceedings of IEEE Signal and Information Processing Association Annual Summit and Conference (APSIPA),* Asia-Pacific, 2013, pp. 1-9, 2013.

[28] L. Lu and H. Zhang "Content analysis for audio classification and segmentation," *IEEE Transactions on Speech and Audio Processing,* vol. 10, pp. 504-516, 2002.

CHAPTER 9

CONCLUSION AND FUTURE WORK

This dissertation work has involved the development of sound signal classification approaches that are computationally efficient to enable their real-time deployment in hearing improvement devices. Both supervised and unsupervised learning schemes have been developed. The contributions made in this dissertation are summarized below:

1- The results obtained have shown that the developed environmental noise supervised classification algorithm outperforms the state-of-the-art supervised classification algorithms in terms of both classification rate and computational efficiency.

2- A real-time implementation of the above algorithm has been achieved on smartphone platforms.

3- An online clustering or unsupervised classification algorithm has also been developed which allows streaming data to be clustered without knowing the number of clusters or classes.

4- A real-time implementation of the above algorithm has been achieved on laptop platforms, which allows environmental background noise signals to be classified in an online fashion.

5- A computationally efficient hierarchical classification approach to distinguish different environmental sound signals has also been developed for deployment in hearing improvement devices.

Possible future research extensions include:

1- Developing a hybrid environmental sound classification approach by combining the developed supervised and unsupervised classification approaches.

2- Noting the developed clustering algorithm is general purpose, applying it to other applications in signal and image processing, such as human action recognition and real-time video segmentation.

BIOGRAPHICAL SKETCH

Fatemeh Saki received a BS degree in electrical engineering from the Shahid Chamran University of Ahvaz in 2008, and a MS degree in electrical engineering from Iran University of Science and Technology in 2010. She is currently a PhD candidate in the Department of Electrical Engineering at The University of Texas at Dallas and will be graduating in May 2017. Her research interests include real-time audio and image processing, pattern recognition and machine learning. Her research contributions have appeared in 4 journal papers (2 additional journal papers are under review) and 14 conference papers. She has also co-authored a textbook for Signals and Systems Laboratory courses using smartphones. She has been an active member of her research community as a reviewer of many journal and conference papers. Based on her academic record, she was awarded the Jonsson School Graduate Study Scholarship 2012 and was also awarded the Louis Beecherl Jr. Graduate Fellowship in both 2015 and in 2016 by the Erik Jonsson School of Engineering and Computer Science at The University of Texas at Dallas.

CURRICULUM VITAE

Fatemeh Saki

Address: The University of Texas at Dallas, Department of Electrical Engineering

Email: fatemeh.saki@utdallas.edu

EDUCATION

- ✓ **PhD Electrical Engineering (Signal Processing)** Sept. 2012—May 2017 (expected)
  The University of Texas at Dallas**,** Richardson, TX
  Dissertation advisor: Prof. Nasser Kehtarnavaz
  Dissertation title: Classification of Sound Signals via Computationally Efficient Supervised and Unsupervised Learning Schemes

- ✓ **MS Electrical Engineering (Biomedical Imaging)** Aug. 2008—Dec. 2010
  Iran University of Science and Technology, Tehran, Iran
  Thesis advisor: Dr. Shahriar Baradaran Shokouhi
  Thesis title: Using Opposition Based Learning Neural Network for Medical Diagnosis of Mammography Images.

- ✓ **BS Electrical Engineering** Aug. 2004—July 2008
  Shahid Chamran University of Ahvaz, Iran
  Thesis advisor: Dr. Fariba Eftekhar
  Thesis title: Detecting Traffic Load Using MATLAB Image Processing Toolbox

PROFESSIONAL EXPERIENCE

- ✓ **Journal Reviewer**:
  - IEEE Transactions on Biomedical Circuits and Systems
  - Computer Methods and Programs in Biomedicine
- ✓ **Conference Reviewer:**
  - 12th IEEE Biomedical Circuits and Systems Conference (BIOCAS 2016)
- ✓ Student Member, IEEE, 2009—present
- ✓ Student Member, IEEE Signal Processing Society, 2012—present
- ✓ Student Member, IEEE Engineering in Medicine and Biology Society, 2013-2015

TEACHING EXPERIENCE

- ✓ **Applied Digital Signal Processing** (Smartphone Implementation) Spring 2016
- ✓ **Signals and Systems Laboratory** 2015-2016
- ✓ **Digital Image Processing** Fall 2015

RESEARCH INTERESTS

- ✓ Machine Learning
- ✓ Signal Processing

- ✓ Biomedical Image Processing
- ✓ Real-Time Processing on Embedded Processors

PUBLICATIONS

Book

N. Kehtarnavaz and **F. Saki**, *Anywhere-Anytime Signals and Systems: From MATLAB to Smartphones*, Morgan and Claypool Publishers, 2016.

Journal Papers (under revision)

1. **F. Saki**, and N. Kehtarnavaz, "Real-time unsupervised classification of environmental noise signals", under review *IEEE Transactions on Audio, Speech, and Language Processing*.
2. **F. Saki**, and N. Kehtarnavaz, "Hierarchical classification of sound signals for hearing improvement devices", submitted to *IEEE Journal of Selected Topics in Signal Processing*.

Journal Papers (accepted/published)

1. **F. Saki**, and N. Kehtarnavaz, "Online frame-based clustering with unknown number of clusters," *Pattern Recognition*, vol. 57, pp.70-83, 2016.
2. **F. Saki**, A. Tahmasbi, H. Soltanian-Zadeh, and S. B. Shokouhi, "Fast opposite weight learning rules with application in breast cancer diagnosis," *Computers in biology and medicine*, vol. 43, no. 1, pp. 32-41, 2013.
3. A. Tahmasbi, **F. Saki**, and S. B. Shokouhi, "Classification of benign and malignant masses based on Zernike moments," *Computers in biology and medicine*, vol. 41, no. 8, pp. 726-735, 2011.

Conference Papers

1. A. Sehgal, **F. Saki**, and Kehtarnavaz, "Real-time implementation of voice activity detector on ARM embedded processor of smartphones," to appear in *IEEE International Symposium on Industrial Electronics (ISIE)*, Edinburgh, Scotland, 2017.
2. N. Kehtarnavaz and **F. Saki**, "Smartphone-based anywhere-anytime signals and systems laboratory," *Proceedings of 42nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, 2017.
3. **F. Saki**, N. Kehtarnavaz, "Automatic switching between noise classification and speech enhancement for hearing aid devices," *Proceedings of 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 736-739, 2016.
4. **F. Saki**, A. Sehgal, I. Panahi, and N. Kehtarnavaz, "Smartphone-based real-time classification of noise signals using subband features and random forest classifier," *Proceedings of 41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2204-2208, Shanghai, China, March 2016.
5. R. Pourreza-Shahri, S. Parris, **F. Saki**, I. Panahi, and N. Kehtarnavaz, "From Simulink to smartphone: signal processing application examples," *Proceedings of 40th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1861-1865, Australia, April 2015.
6. **F. Saki**, T. Mirzahasanloo, and N. Kehtarnavaz, "A multi-band environment-adaptive approach to noise suppression for cochlear implants," *Proceedings of 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (*EMBC'14*), pp. 1699-1702, Chicago, August, 2014.
7. **F. Saki** and N. Kehtarnavaz, "Background noise classification using random forest tree classifier for cochlear implant applications," *Proceedings of 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP*), pp. 3591-3595, Italy, May 2014.

8. R. Pourreza-Shahri, **F. Saki**, N. Kehtarnavaz, P. Leboulluec, H. Liu, "Classification of ex-vivo breast cancer positive margins measured by hyperspectral imaging," *Proceedings of 20th IEEE International Conference on Image Processing (ICIP),* pp.1408-1412, Australia, September 2013.
9. A. Tahmasbi, **F. Saki**, and S. B. Shokouhi, "CWLA: a novel cognitive classifier for breast mass diagnosis," *Proceedings of IEEE, 18th Iranian Conference on Biomedical Engineering,* pp. 255-259, Tehran, Iran, 2011.
10. A. Tahmasbi, **F. Saki**, S. B. Shokouhi, "Classification of breast masses based on cognitive resonance," *Proceedings of* 3rd *ICSAP,* pp. 97-101, Singapore, vol.1, 2011.
11. A. Tahmasbi, **F. Saki**, H. Aghapanah, and S. B. Shokouhi, "A novel breast mass diagnosis system based on Zernike moments as shape and density descriptors," *Proceedings IEEE, 18th Iranian Conference on Biomedical Engineering,* pp. 100-104, Tehran, Iran, 2011.
12. **F. Saki**, A. Tahmasbi, and S. B. Shokouhi, "A novel opposition-based classifier for mass diagnosis mammography images," *Proceedings of IEEE, 17th Iranian Conference on Biomedical Engineering,* pp. 1-4, Isfahan, Iran, 2010.
13. A. Tahmasbi, **F. Saki**, and S. B. Shokouhi, "An effective breast mass diagnosis system using Zernike moments," *Proceedings of IEEE, 17th Iranian Conference on Biomedical Engineering,* pp. 1-4, Isfahan, Iran, 2010.
14. A. Tahmasbi, **F. Saki**, and S. B. Shokouhi, "Mass diagnosis in mammography images using novel FTRD features," *Proceedings of IEEE, 17th Iranian Conference on Biomedical Engineering,* pp. 1-5, Isfahan, Iran, 2010.

SELECTED HONORS AND AWARDS

- ✓ Louis Beecherl, Jr. Graduate Fellowship 2016-2017
- ✓ Louis Beecherl, Jr. Graduate Fellowship 2015-2016
- ✓ Jonsson School Graduate Study Scholarships 2012-2013
- ✓ President's 1st Place Award for Excellence in Research, Iran University of Science and Technology 2011
- ✓ Member of Exceptional Talent's Institute of Iran 2004-2010
- ✓ President's 1st Place Award for Excellence in Education, Shahid Chamran University of Ahvaz 2008