SABANCI UNIVERSITY

Railway vehicle detection from audio recordings using one-class classification

by

Fahad Sohrab

Submitted to
the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

SABANCI UNIVERSITY

January 2016

Railway vehicle detection from audio recordings using one-class classification

APPROVED BY

Assoc. Prof. Dr. Hakan Erdogan (Thesis Supervisor) .....

Prof. Dr. Berrin Yanikoglu .....

Assoc. Prof. Dr. Erchan Aptoula .....

DATE OF APPROVAL: .....26th January 2016.....

To my family...

Acknowledgements

I would like to express my gratitude to my supervisor Professor Hakan Erdogan for his invaluable guidance, tolerance, positiveness, support and encouragement throughout the time during my M.S. studies. I feel honored to have worked under Professor Hakan Erdogan's supervision. I truly believe that I would have never made it this far without mentoring of Professor Hakan Erdogan. I was guided to a direction in which I could "comfortably" excel. I could not have imagined having a better adviser and mentor for my M.S. study.

I extend my gratitude to Dr David Tax for introducing me to Sensornet and providing me the opportunity of working under his supervision at Pattern Recognition Laboratory Delft University of Technology for one year.

I am thankful to Sensornet for the support during my time at TU delft. I was lucky to be part of Sensornet and getting access to their system and database. The precious data at Sensornet is of great significance and was used for all experiments in this thesis.

My thanks go to all my colleagues at Sensornet, VPA Lab Sabanci University and Pattern recognition Lab TU delft for their friendship, and for providing the right environment to focus on my work. I owe special thanks to Technical director Sensornet, Jasper Koolhas, for his encouraging leadership and tolerance.

I would like to thank Professor Berrin Yanikoglu for being a jury member of my thesis and for the unconditional support in final semester. I would also like to warmly thank Professor Erhan Abdullah (Okan University) for for taking the time from his busy schedule and accepted to be a member of jury for this thesis.

To my lab-mates (there is long list) at VPA-LAB Sabanci University and PR-Lab TU delft, thanks for the fun and support. My experience in both labs was great and I greatly look forward to having all of you as colleagues in the years ahead.

Railway vehicle detection from audio recordings using one-class classification

Fahad Sohrab

EE, M.Sc. Thesis, 2015

Thesis Supervisor: Hakan Erdogan

Keywords: Audio source recognition, MFCC, One-class classifier, SVDD

Abstract

In this thesis, we focus on detecting a train from the sound generated by it. An audio sensor is placed close to a railway track to record ambient sounds which may or may not originate from a train. In this problem, we define the target event as the recording of a train sound and non-target events are all other audio events that are recorded by the audio sensor.

In machine learning and pattern recognition, classifiers are trained from labeled data to categorize a new observation. Classifiers are usually trained from data which contain all possible classes, however it is possible that during training the classifier, for some classes the data is either not available or it is so diverse in nature that it cannot be used reliably. In case of binary classification, if one of the classes do not have reliable training data, we can use a "one class classification" strategy which only uses single class data for training.

For train detection from audio, we compared a one-class classifier called support vector data description (SVDD) with binary classifiers and showed that SVDD performs well in cases where data from the outlier class is scarce. We also tested the SVDD trained model in real time and the results indicate that the goal of reducing the false positive rate is satisfactorily achieved. The tests are performed using audio data recorded in Bathmen, a town in eastern Netherlands, by the company Sensornet for a project about railway vehicle detection and sound level monitoring.

Tek-sınıf sınıflandırma kullanılarak ses kayıtlarından demiryolu araç tespiti

Fahad Sohrab

EE, Yüksek Lisans Tezi, 2015

Tez Danışmanı: Hakan Erdoğan

Anahtar Kelimeler: Ses kaynağı tanıma, Mel Freakans Sepstrum Katsayıları (MFSK), Tek sınıf sınıflandırma, DVVA

Özet

Bu tezde, ses kayıtlarından demiryolundan geçen araçları tespit etme problemine odaklandık. Araçlardan kaynaklanması muhtemel çevresel sesleri kaydetmek için bir ses sensörü, bir demiryolunun yakınına yerleştirilmiştir. Bu problemde, araçların ses kayıtlarını hedef olaylar, diğer bütün ses kayıtlarını ise aykırı olaylar olarak tanımladık.

Makine öğrenimi ve örüntü tanımada, sınıflandırıcılar, yeni bir gözlemi sınıflandırmak için, etiketli veriden eğitilmiştir. Sınıflandırıcılar genellikle bütün muhtemel sınıfları içeren veriden eğitilirler ancak sınıflandırıcının eğitimi sırasında, bazı sınıflar için verinin mevcut olmaması veya verinin doğasının çok farklı olmasından ötürü güvenilir bir şekilde kullanılamaması mümkündür. İkili sınıflandırma durumunda, eğer sınıflardan biri güvenilir veriye sahip değilse, eğitim için sadece bir sınıfın verisini kullanan "tek sınıf sınıflandırma" stratejisini kullanabiliriz.

Sesten tren tespiti için, destek vektör veri açıklaması (DVVA) adlı tek sınıf sınıflandırıcıyı ikili sınıflandırıcılar ile karşılaştırdık ve aykırı sınıftan verinin az olduğu durumlarda DVVA'nın iyi performans sergilediğini gösterdik. Ayrıca, DVVA eğitim modelini gerçek zamanda da test ettik ve yanlış pozitif oranını düşürme hedefini tatmin edici bir şekilde gerçekleştirdik. Testler, Sensornet adlı firma tarafından demiryolu araç tespiti ve ses seviyesi izleme üzerine bir proje için Hollanda'nın doğusundaki Bathmen kasabasında kaydedilen ses verileri kullanılarak yapılmıştır.

Contents

List of Figures

List of Tables

Abbreviations

SVDD	**S**upport **V**ector **D**ata **D**escription
SVM	**S**upport **V**ector **M**achine
ROC	**R**eceiver **O**perating **C**haracteristic
MFCC	**M**el **F**requency **C**epstral **C**oefficients
FIR	**F**inite **I**mpulse **R**esponse
FFT	**F**ast **F**ourier **T**ransform
PCA	**P**rincipal **C**omponent **A**nalysis
LOSO	**L**eave **O**ne **S**et **O**ut-cross-validation
FP	**F**alse **P**ositive
TP	**T**rue **P**ositive
FN	**F**alse **N**egative
TN	**T**rue **N**egative
FPR	**F**alse **P**ositive **R**ate
TPR	**T**rue **P**ositive **R**ate
FNR	**F**alse **N**egative **R**ate
TNR	**T**rue **N**egative **R**rate
AUC	**A**rea **U**nder **C**urve

Chapter 1

Introduction

In our surroundings, we hear sounds from different sources. The sources producing these sounds might be important for the society but at same time the sound produced might be inconvenient for the citizens living in surroundings. For example, transportation systems are important for social and economic growth of cities but at same time they cause environmental problems because of noise produced by them. In recent years, with the increase in demand of transportation for passenger and freight traffic, railways are considered comparatively more environment friendly [3].

The expansion of railway tracks can help in restraining the congestion in traffic on roads. This in turn helps reduce the noise in residential areas but unfortunately train noise is perceived as an environmental problem as well [4].

In [5] the effects of environmental noise on public health is described in detail. In particular, it has been observed that citizens living near the railway track sometimes complain about the increase in level of noise by trains.

For optimally controlling the noise produced by trains, there are some rules proposed by different government agencies. For example in the technical specification for interoperability of railway noise, the European Union enact the maximum noise limit of freight trains and passenger trains moving at 80 km/hr as 87 and 80 dB (A) respectively [6].

To enforce the rules and identify mitigation solutions, the noise measurements can be recorded by a technician in a log book manually when train is noticed at different train stations or near the track at residential area. However it needs a lot of man

power to record noise produced by trains manually and government certainly cannot be everywhere at once 24 hours a day.

FIGURE 1.1: Sensornet's system overview. Image retrieved from [1].

Sensornet is a company specialized in accurate, long-term measurement of environmental noise in Netherlands. The company has installed a large number of strategically placed unmanned noise meters which continuously transmit their measurement data to a central database using Internet connections in real time [1].

The audio sensors are placed at different fixed locations and not in the moving trains because it is important to monitor region specific noise levels particularly near residential areas. Also the location specific microphones can be used for other audio source detection in future. For detecting train, use of microphone over video camera is preferred because the final goal is to monitor audio noise produced by train, which can be achieved only by using a microphone. Detecting train only from audio data will remove the costs of placing a video camera along with the costs of transmitting video data for train detection.

FIGURE 1.2: Microphones transmitting data in real time to Sensornet's database. Image retrieved from [2]

The audio data transmitted and recorded on the server in real time contain data from diverse sources. Besides passenger and freight trains, the noise of vehicles such as trucks, buses, cars, tractors and other non-train noises are recorded and transmitted to the servers as well. The challenge is to identify the source of data coming in real time. Manually labeling hundreds of audio files as trains, and throwing out the redundant audio files of non-trains every day after listening is very tedious and prone to error. Intelligent software which identifies the data between train and non-train would certainly increase the efficiency of noise measurement and classification system.

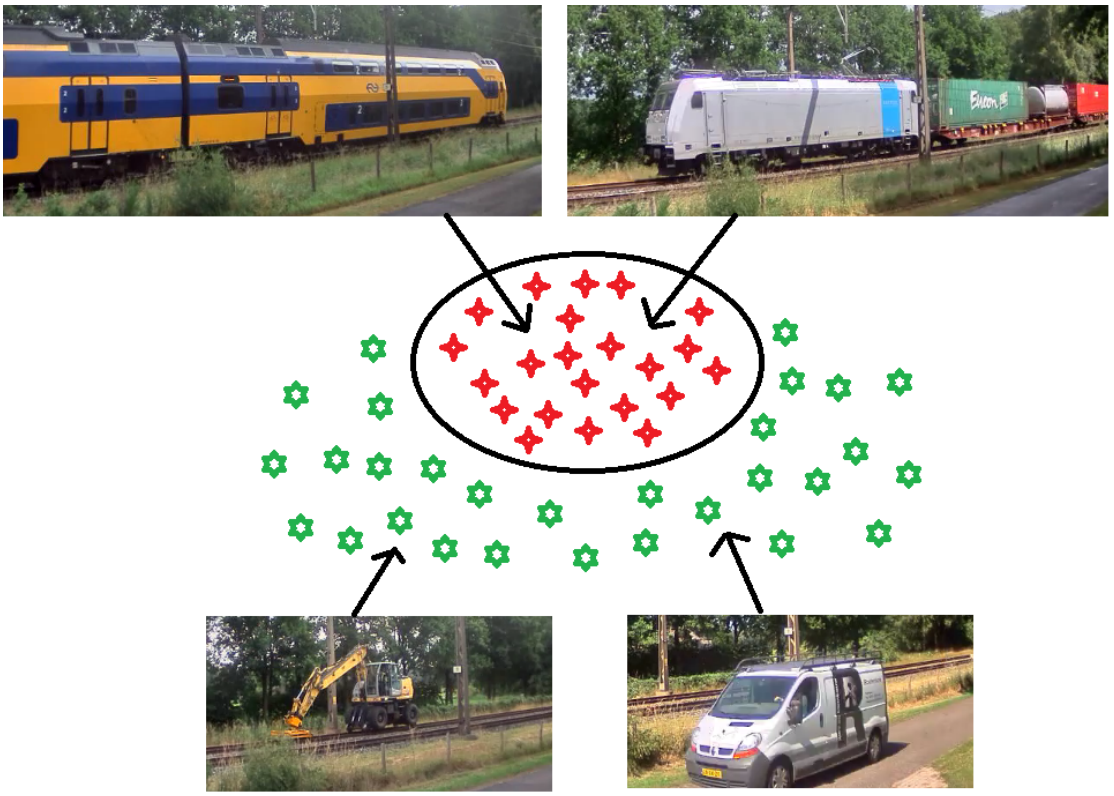

FIGURE 1.3: Abstract view train and non-train classification.

Numerous techniques and feature extraction methods are proposed in the literature for audio classification. In [7], Bark scale features and K-NN technique is proposed for audio noise classification. High speed train noise analysis in an open country environment is done in [8] while in [9] a hierarchical system for audio classification and retrieval based on audio analysis is presented. Features like linear predictive coefficients, linear predictive cepstral coefficients and mel-frequency cepstral coefficients are used to train support vector machines for classification of audio data in [10]. In [11], a study is conducted to show that cepstral-based features such as the Mel-frequency cepstral coefficients (MFCC)

and linear prediction coefficients (LPC) provide better classification accuracy compared to temporal and spectral features for classification of continuous general audio data.

For classification of data in cases where the only information available is from target class, the term "one-class classifier" was first proposed in [12]. A method aimed at recognizing environmental sounds using one-class SVMs and wavelets for audio surveillance is proposed in [13]. In [14], it is stated that one-class SVM is well suited for event-recognition tasks.

In our work, we focused on studying and developing different techniques which would help us identify sound measurements of trains in real time. To carry out this project, the thesis has been divided into five sections briefly described below.

In the first (current) section, the reader is introduced to the thesis and background, motivation and objectives are presented.

In Section 2, classification types of interest are discussed. This section is further divided into two sections. In 2.1, we describe the difference between one class and two class classification. In 2.2, we explain in detail the one-class classifier support vector data description (SVDD) and briefly discuss the difference between support vector machines (SVM) and SVDD.

In Section 3, we explain various details of the system. Section 3 is further divided into six sections. In 3.1, we discuss briefly about frame extraction. In 3.2 and 3.3 online target detection and offline target detection details are given respectively. In 3.4, extraction of MFCC features is described in detail. In 3.5, computing score for audio files in offline detection is explained briefly. In 3.6, evaluation procedure of the experiments is discussed in detail. All the numerical results from different classifiers are presented in Section 4. Finally the conclusions are presented in Section 5.

Chapter 2

Classification types

In classification of data in pattern recognition, an attempt is made to assign each input value to one of a given set of classes. The pattern recognition systems can be trained in two ways, namely supervised and unsupervised. In supervised method, the system is trained from known labeled "training" data while in unsupervised, no labeled data are available and some algorithms are used to discover unknown patterns. Since we have labeled data available for experiments, we will use supervised classification for the system.

2.1 Multi-class and one-class classification

In multi-class classification each training data point belongs to one of N different classes. A conventional multi-class classification problem can be decomposed into several two-class classification problems [15]. The goal is to construct a function which will correctly predict the class of a new point to which it belongs. In multi-class classification problems, data from all the classes are available which are used for training a classifier. In one-class classification, we are always dealing with a two-class classification problem, where each of the two classes has particular meaning and importance. In one-class classification, we have two classes, namely, a target class and a non-target class. Target class is the one which is sampled well in a sense that all information is available about this class while for non-target class either no data is available or it is so diverse and random that it cannot be modeled properly. In one-class classification, objects are identified by

learning from a training set which contains objects only from the target class and not from the non-target class. However for testing, some of the points from non-target class are taken into account. If no non-target data is available, the system can be tested on artificially generated non-target data. In problem of classifying data at Sensornet, the source of target audio data can be a freight train or passenger train. Both possibilities are combined together which form the target class. While for not-a-train class, there are theoretically infinite possibilities, the non-target source can be car, bus, tractor, scooter and so on. We consider not-a-train as the non-target class.

2.2 SVDD

The support vector data description (SVDD) is a one-class classifier which fits a closed boundary namely a hyper-sphere, around the target class. The hyper-sphere is characterized by centre a and radius R. The centre and radius are defined under some constraints [16] which minimize the volume of the hyper-sphere and include all the points x_i of training set.

The error function to minimize is as follow

$$F(R,a) = R^2 + C\sum_i \zeta_i \tag{2.1}$$

under the constraints

$$\| x_i - a\|^2 \leq R^2 + \zeta_i \ , \ \zeta_i \geq 0 \tag{2.2}$$

where ζ_i are slack variables and parameter C controls the trade-off between the hyper-sphere's volume and the errors.

By using Lagrange multipliers, Equation 2.2 can be incorporated into Equation 2.1 as

$$L(R,a,\alpha_i,\gamma_i,\zeta_i) = R^2 + C\sum_i \zeta_i - \sum_i \alpha_i\{- \| x_i - a\|^2 + R^2 + \zeta_i\} - \sum_i \gamma_i\zeta_i \tag{2.3}$$

$$L(R, a, \alpha_i, \gamma_i, \zeta_i) = R^2 + C\sum_i \zeta_i \\ - \sum_i \alpha_i\{R^2 + \zeta_i - \| x_i\|^2 - 2a \cdot x_i + \| a\|^2\} - \sum_i \gamma_i\zeta_i, \quad (2.4)$$

where Equation 2.4 should be maximized with respect to Lagrange multipliers $\alpha_i \geq 0$ and $\gamma_i \geq 0$ and minimized with respect to R, a and ζ_i. These equations are further simplified to obtain a maximization problem involving α_i's only, with some constraints [16].

Sometimes the hyper-sphere might not separate the target and non-target class adequately in original feature space. To make one-class classifier more flexible, a "kernel trick" is used [17]. In kernel trick, the data is assumed to be mapped to a higher dimension and inner products between data vectors are replaced with a function known as the kernel function.

A good kernel function would be one which maps all the target data inside the hyper-sphere and the non-target data outside the hyper-sphere in the new kernel feature space. One such kernel function is Gaussian kernel which has some favourable properties. For Gaussian kernel, a parameter σ is defined which control the width of the kernel. Small σ will result in tighter boundary of the sphere while increasing the σ to very large value will result in almost a spherical hyper-sphere.

$$k(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{\sigma^2}\right). \quad (2.5)$$

Equation 2.5 shows the Gaussian kernel function, where x_i and x_j are samples from the input feature space. During testing an object z outside the radius r of the hyper-sphere will be labelled as non-target and inside the hyper-sphere will be labelled as target

$$\sum_i \alpha_i \exp\left(\frac{-||z - x_i||^2}{\sigma^2}\right) \geq -\frac{R^2}{2} + C_R, \quad (2.6)$$

where C_R in 2.6 depends only on the support vectors x_i and not on z [16].

Figure 2.1 shows the working of SVDD over banana set. A 2-dimensional banana shaped distribution of 2-classes with 500 points each for target and non-target data is generated

artificially. The target points are indicated by the red points while the non-target points are indicated with blue points.

Four different boundaries, shown in Figure 2.1, are fitted over the data by just using the data from target class during training. For all boundaries the RBF kernel was used but with different width. For the strict boundary the width parameter used is $\sigma = 2$ while for the other boundary which appear to be flexible and almost a sphere, the width parameter is $\sigma = 8$. Another hyper-parameter C in the example is fixed as $C = 1/(N\epsilon)$. Where N is the number of target training samples and ϵ is the error on the target class supplied as default value of 0.1. It is clear that increasing the σ value results in a more flexible and more spherical boundary which can be optimized according to the requirement of data.

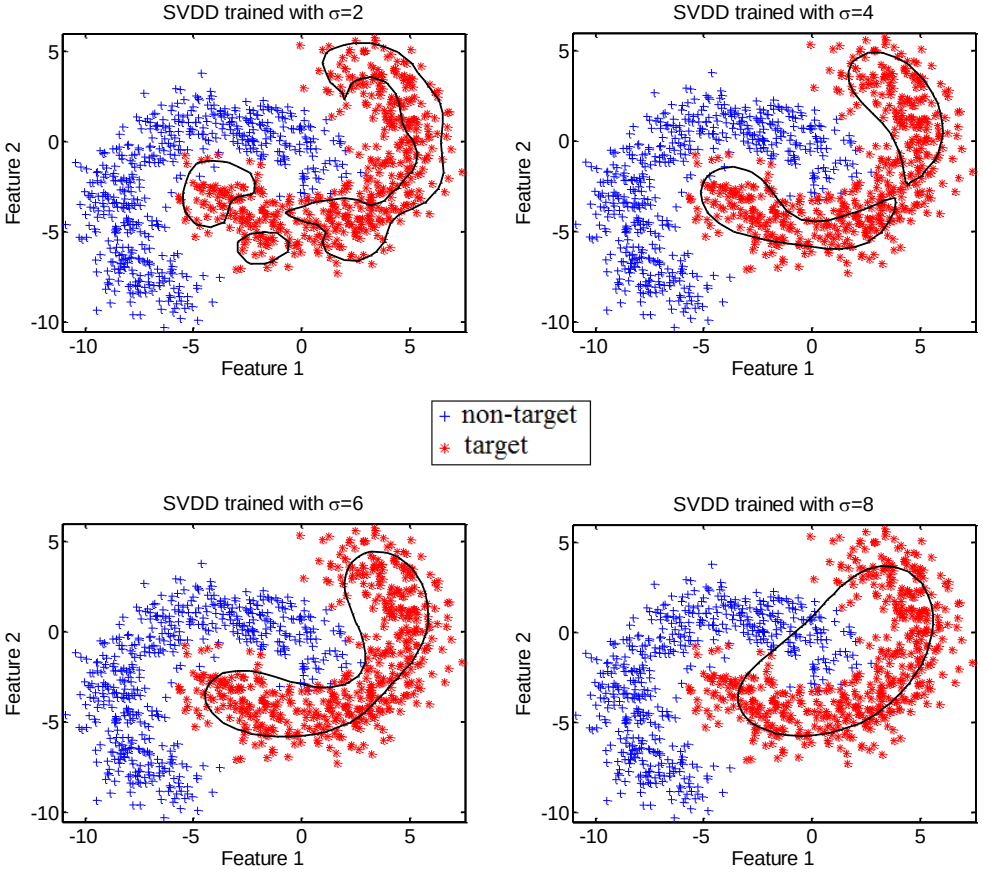

FIGURE 2.1: SVDD boundaries with different σ.

SVDD can be used for a range of applications where data collection from one of the two

classes is expensive or impossible. For example, it can be applied to machine diagnostics problem [18]. In machine diagnostics problem, SVDD is applied to find out the normal working situation of a pump in a pumping station. Similar usage of SVDD can be applied to any machine for finding out normal and abnormal working of its components. The advantage of SVDD over SVM is that for SVM there should be some samples available from every class for training while for SVDD the training is done only by considering data from one-class i.e. target class. For example in case of machine diagnostics, it is expensive to collect data of all the abnormal behaviors of machine and certainly SVM cannot be used.

Chapter 3

System description

At Sensornet, the continuous data transmitted is stored in a buffer temporarily until the system is triggered for complete audio event storage. The audio events are stored permanently in the database when noise level reaches a particular threshold (65 dBA usually) for some specific time. The system when triggered will save all the data permanently from the buffer along with the new coming data till the noise level goes well below the threshold.

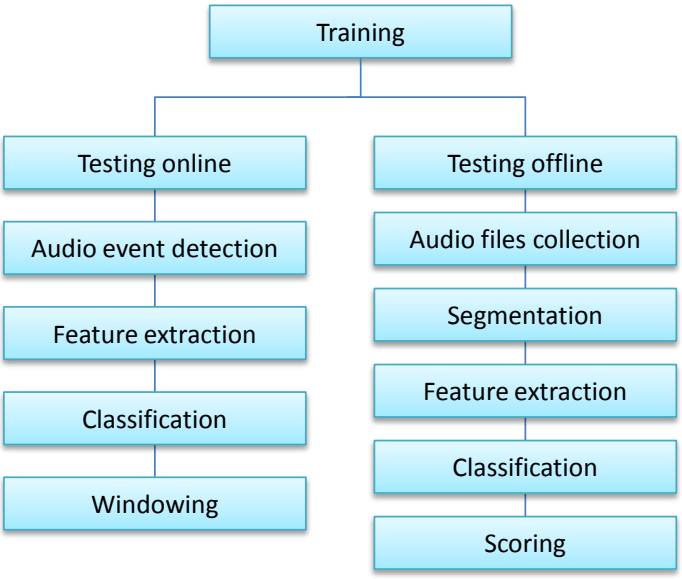

FIGURE 3.1: Flowchart for online and offline classification

Figure 3.1 shows the steps involved in the classification of audio events in online and offline systems. In online system, the features are extracted at frame level in real time without doing any segmentation. In offline system, the audio files are segmented before feeding to a trained classifier for testing. More details about online and offline target detection are discussed in Section 3.2 and 3.3 respectively.

3.1 Frame extraction

The analysis of audio events is usually done at frame level. An audio frame contains amplitude (loudness) information at a particular time period. For example an audio file with sampling rate of 32000 Hz, if analysed with a frame size of 200 milliseconds will contain 6400 samples in one frame. The audio events at Sensornet are analysed at frame level (200 milliseconds) in online system as well. The frames are analysed independently of each other and during the process there is no overlapping among the frames.

3.2 Online target detection

The audio events when detected at Sensornet are classified as target or non-target at frame level in real time. The decision is based on the percentage of target frames detected in a sliding window of 20 frames (4 seconds). The final decision can be made by deciding maximum threshold (percentage) value predicted by classifier within the sliding window.

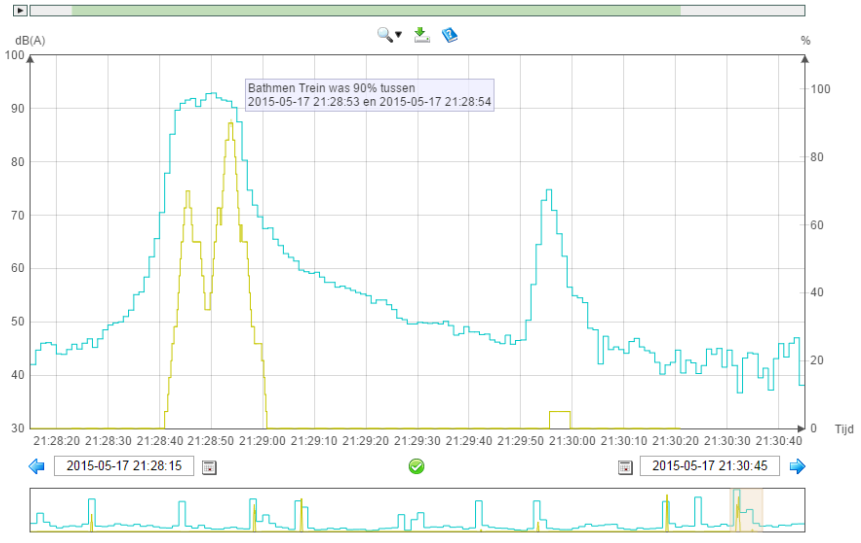

FIGURE 3.2: Online train detection example.

In Figure 3.2, an example of the audio event in the online system is depicted. The sliding window attains the maximum value of 90 percent in the event.

3.3 Offline detection

In offline setup, the detection of target event is based on frame level decisions as well. The segment level decision is made on the basis of ratio of frames within the segment detected as target by the trained classifier. In offline setup, clearly the data at the beginning and end of audio events which was stored from the buffer might cause misleading results if fed to classifier. The audio event should be segmented properly for the offline detection. In the following section, the segmentation process of audio events is explained.

3.3.1 Segmentation

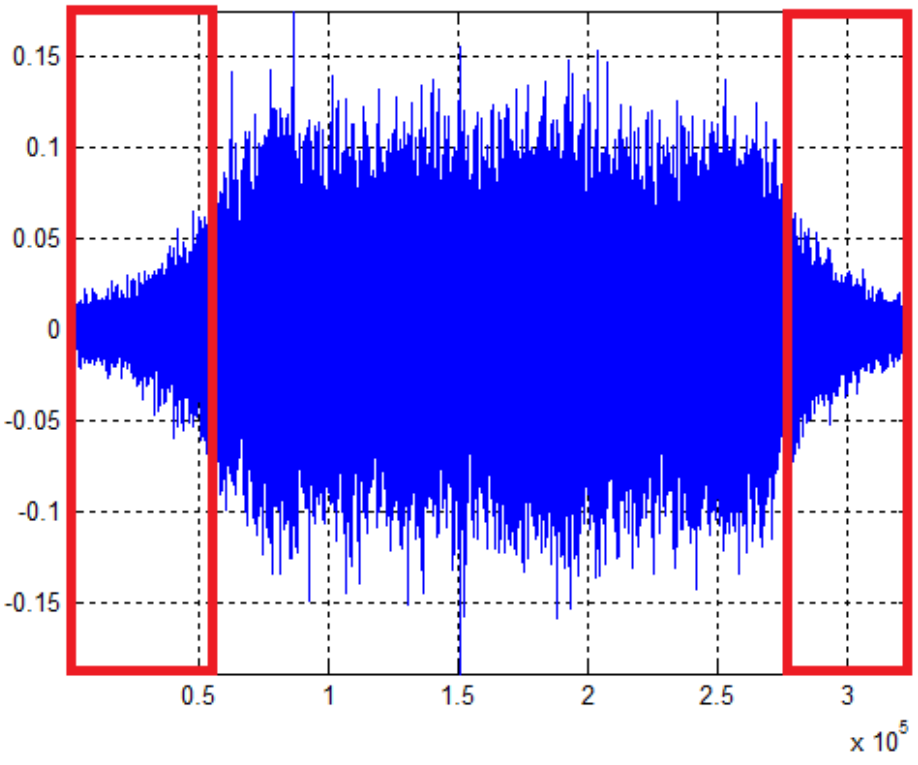

FIGURE 3.3: Audio file with low energy at starting and ending point.

In Figure 3.3, it can be clearly seen that the data marked inside the red window have low energy. To avoid any ambiguity, we follow the procedure below to cut off the silence

parts from start and end of the audio target files. In the first step, the absolute values of audio signals are passed through the simple moving average filter of 6400 points (200ms window). The output is shown in Figure 3.4

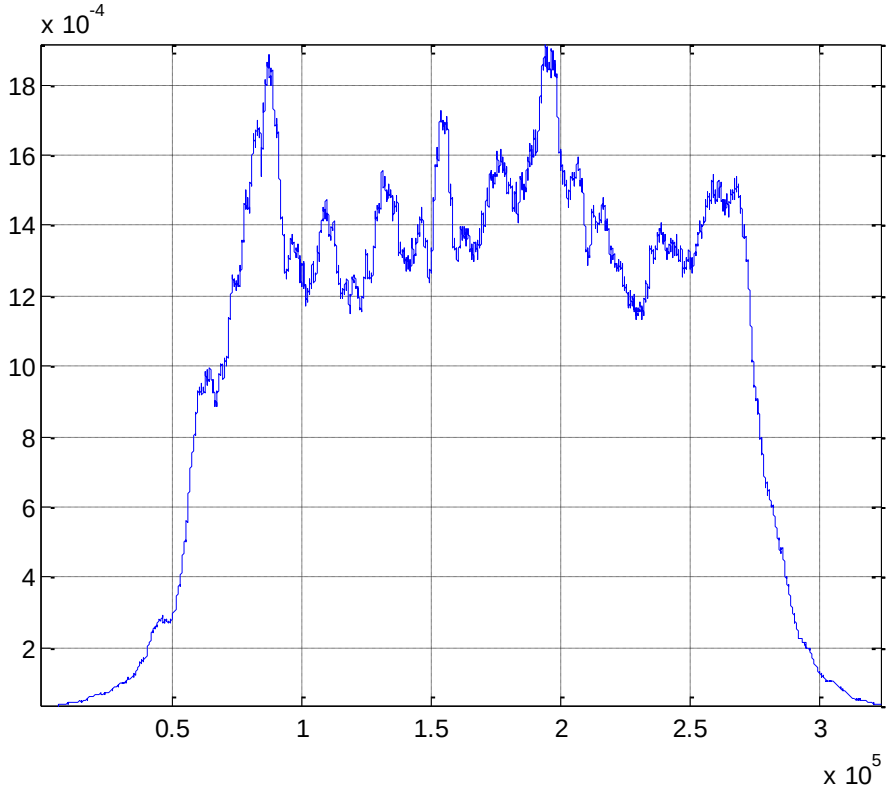

FIGURE 3.4: Output of sample train audio after passing its absolute values from moving average filter.

In next step, the threshold is decided, i.e. the starting and ending point of the signal. If the threshold is too high then the segmented signal obtained will be very small and important frames will be lost. If it's too low it will still contain low energy data. We opted for 50 percent of the maximum value in the "simple moving averaged" data. With 50 percent of maximum value, in the example the threshold obtained is $0.958 * 10^{-}3$ i.e. the indices (time) where the signal (simple moving averaged audio file) touch the threshold for first and last time are considered the cut off points for the signal. Figure 3.5 shows the audio file after segmentation.

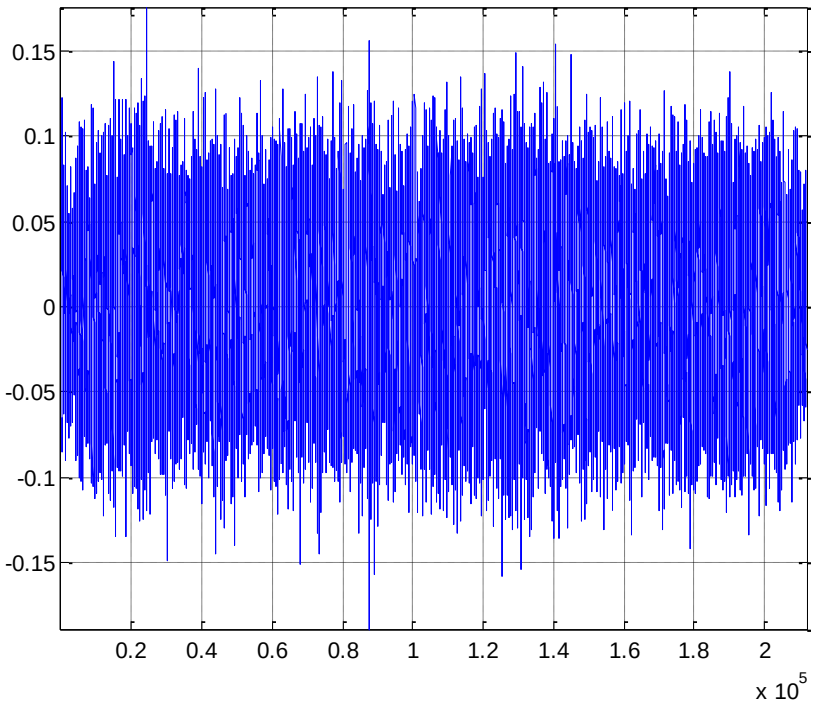

FIGURE 3.5: The segmented sample audio train.

3.4 Feature extraction

Feature extraction is most important stage for audio recognition systems. The aim in feature extraction from audio recordings is to identify the components of the audio signal that are good for identifying the content and discarding all redundant components which carries useless information.

One sample spectrogram of passenger train recording with sampling rate of 32kHz is depicted in Figure 3.6. It is clear from Figure 3.6 that the most dominant frequencies throughout the time signal are less than 4200 Hertz. The red colour indicates the strong presence while the blue indicates less energy of the frequency.

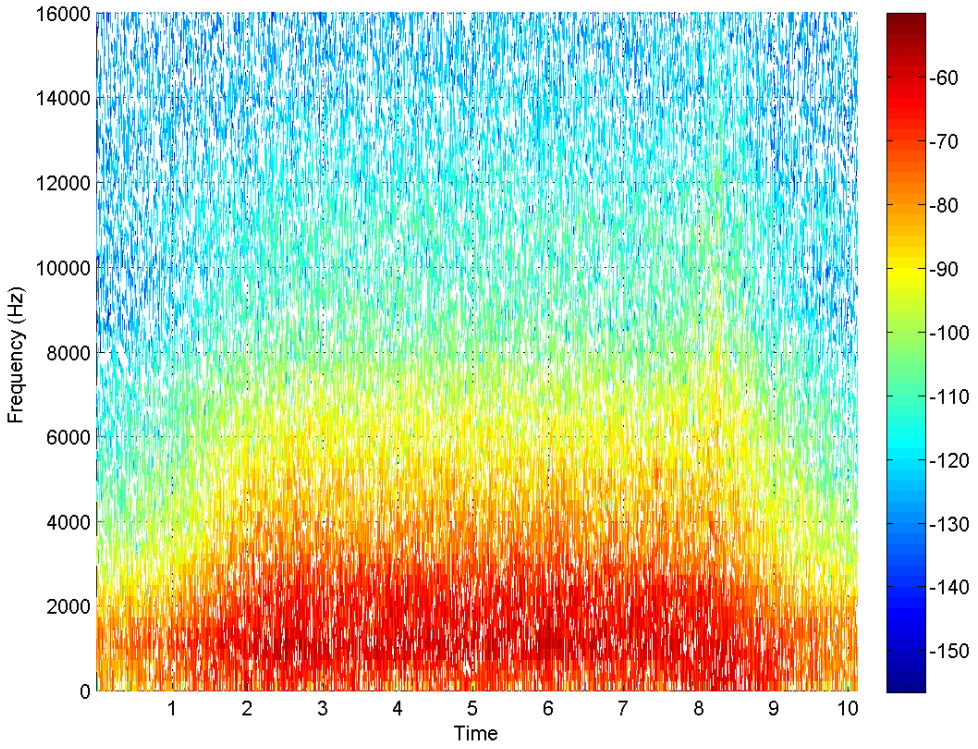

FIGURE 3.6: Spectrogram of sample passenger Train.

We opted for MFCCs as our features for train vs non train. MFCCs are used extensively for human speech recognition. Human speech is filtered by the shape of the vocal tract, tongue and teeth etc. The main idea is to determine the shape accurately which would give the idea of accurately representing the phoneme being produced. Similar idea is applied for extracting features from the audio of train signals. The shape of source producing the audio signal can be manifested in the envelope of the short time power spectrum. MFCCs are used to accurately represent this envelope. In the following sub-sections, the extraction process of MFCCs are explained

3.4.1 Pre-emphasis

In order to compensate the high frequency part of the audio signals, audio recordings are first pre-emphasised using a first order FIR filter with preemphasis coefficient α. The intention is to flatten the spectrum of the audio recordings such that dynamic range of the spectrum is reduced and low frequency components are restricted from dominating the spectral envelope.

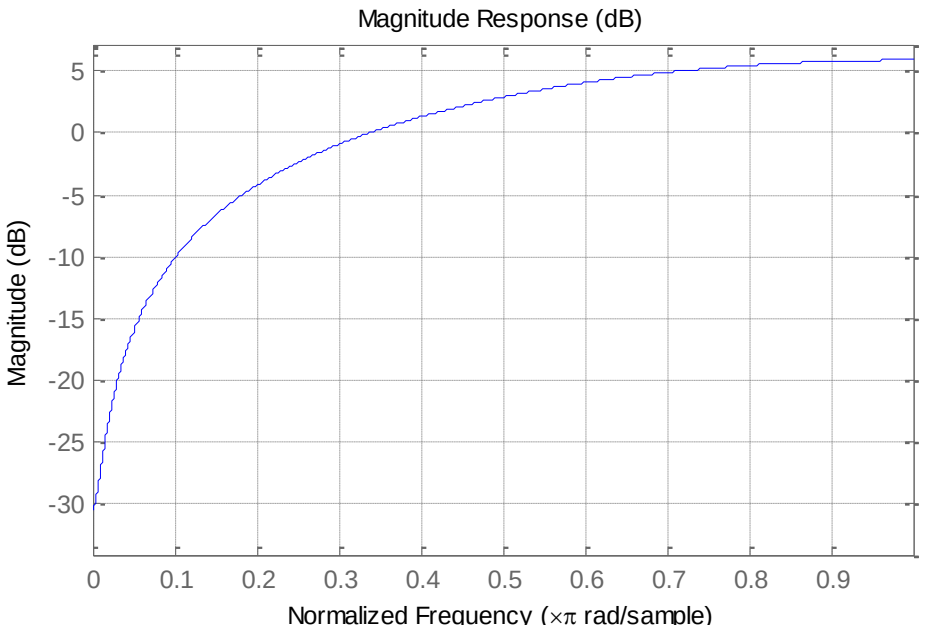

FIGURE 3.7: Pre-emphasis filter with α=0.97.

3.4.2 Windowing

The audio recordings over the whole time are non-stationary. For analyzing and extracting features, the behaviour of the signal should be close to stationary. Thus the signal is decomposed to short audio sequences, called frames, and then each frame is analysed independently from each other. If the frame size is too short then we don't get enough samples, if it is too long then signal changes too much throughout the frame and becomes non-stationary.

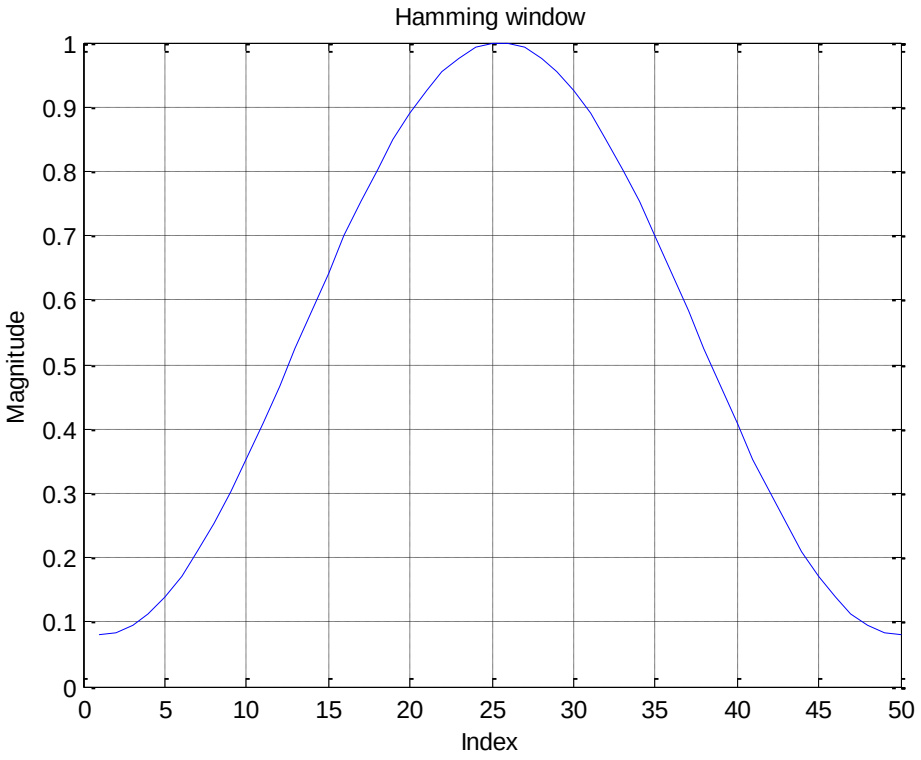

FIGURE 3.8: Hamming window which has a length of 50 samples.

To smooth the edges of frames they are passed through a Hamming window of width N. Hamming window is described in Equation 3.1 and depicted in Figure 3.8.

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi(n-1)}{N-1}\right) \ . \tag{3.1}$$

3.4.3 Mel-frequency cepstrum

In first step of computing MFCC, for every frame, fast Fourier transform (FFT) is computed and its magnitude is obtained. The magnitude of FFT identifies the frequencies present in the frames. To get an idea of how much energy exists in various frequency regions a filter bank of 20 triangular filters uniformly spaced in Mel scale are formed. The mappings from linear frequency (f) to mels (m) and vice versa are given by the following equations.

$$m(f) = 1125 \ln\left(1 + \frac{f}{700}\right) \ , \tag{3.2}$$

$$f(m) = 700 \left(\exp(\frac{m}{1125}) - 1 \right) \ . \tag{3.3}$$

Equation 3.2 is used for converting frequencies to Mel while Equation 3.3 for converting them back to hertz.

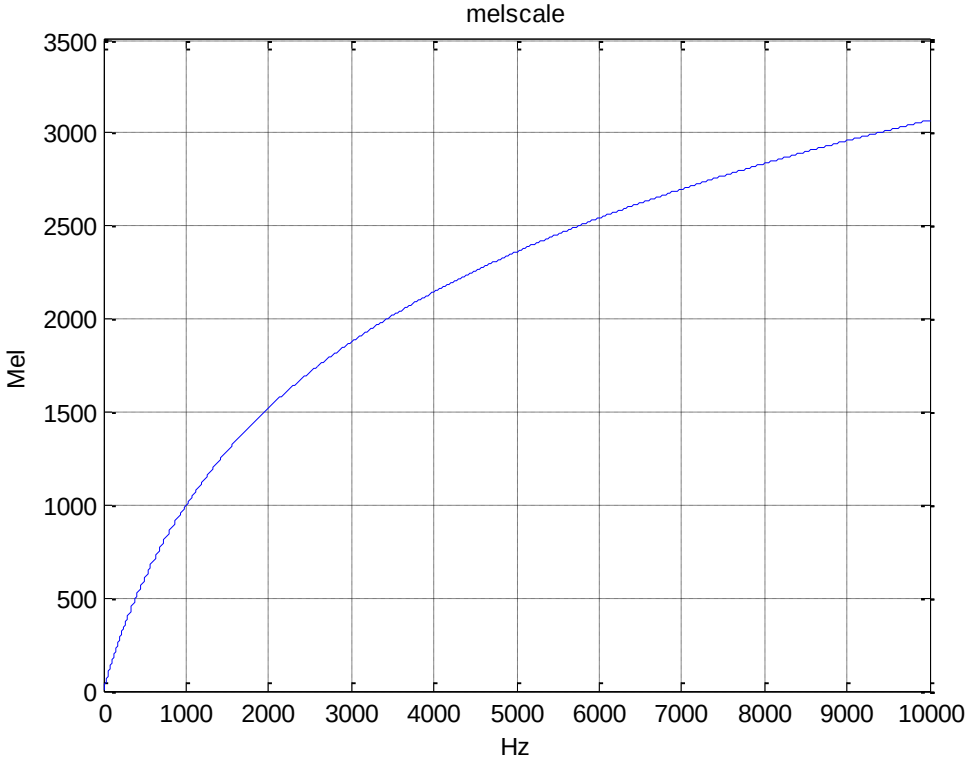

FIGURE 3.9: Hertz to mel scale transformation.

For example for range of 10Hz (15.99 Mels) to 4200Hz (2193.04 Mels), following would be the frequencies to consider. In Mel [15.99, 119.66, 223.32, 326.99, 430.66, 534.33, 638.00, 741.67, 845.34, 949.01, 1052.68, 1156.35, 1260.02, 1363.69, 1467.36, 1571.03, 1674.69, 1778.36, 1882.03, 1985.70, 2089.37, 2193.04]. The frequencies after converting to Mel scale linearly separated with equal difference of 103.67 from each other. There are two extra points which we need for starting and ending banks. After converting to Hz we get the following frequency points. In hertz [10.00, 78.41, 153.41, 235.64, 325.78, 424.62, 532.98, 651.77, 782.02, 924.81, 1081.36, 1253.00, 1441.17, 1647.47, 1873.65, 2121.63, 2393.49, 2691.55, 3018.33, 3376.59, 3769.37, 4200.00] We get the filter bank shown in Figure 3.10. The first filter start from 10Hz, which is the first point, it gets its peak at second point while goes to zero at 3rd point. Similarly the second filter starts at second

point, reach its maximum at 3rd point and goes to zero at fourth point. And similarly it continues for the rest of banks, ending at 4200Hz (2193.04 Mels).

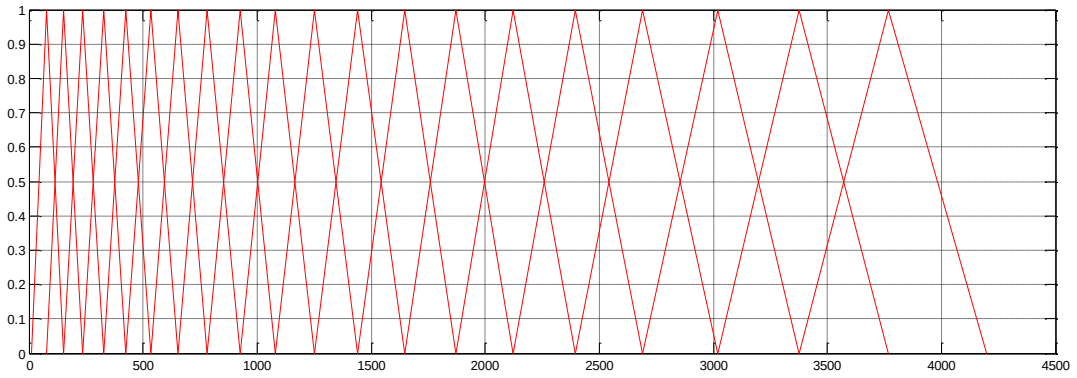

FIGURE 3.10: Filter banks.

For producing cepstral coefficients, log-compressed filterbank energies are then decorrelated using discrete cosine transform. Then, we pick some number of lower indexed outputs of the DCT as MFCC features.

Same procedure is repeated for all the audio files in the dataset. The 13 dimensional features (MFCCs) are extracted from each frame of the original segmented audio files for experiments.

Figure 3.11 summarizes the steps involve in extraction of MFCCs.

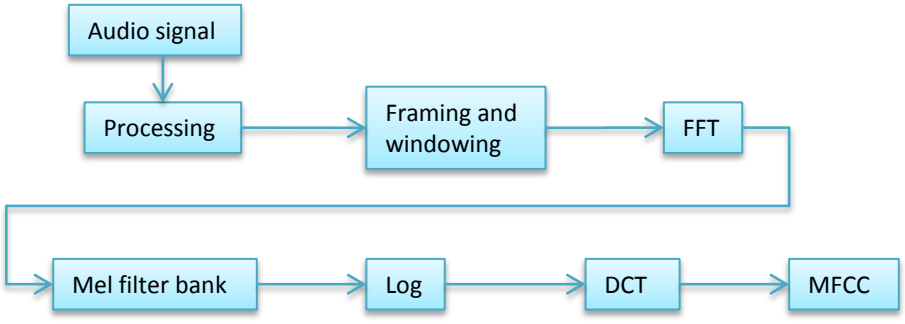

FIGURE 3.11: MFCC flowchart.

3.5 Audio type detection from a segment

An audio file is analysed as target or non-target at frame level by classifier. The classifier predict all the frames of complete audio file as target or non-target one by one. The decision in offline train detection is based on score predicted by classifier. The score for an audio segment in offline detection is the average number of target frames predicted by the classifier for that particular audio file. The final decision is taken by comparing the segment score with a threshold.

3.6 Evaluation criteria

There are several ways to evaluate the performance of a binary classifier. We will explain different evaluation methods in this section.

3.6.1 Sensitivity and specificity

For one class classifier, where the output is binary i.e. positive or negative, two kinds of errors are possible. It might wrongly label a target as non-target or non-target as target. If it correctly classifies a target as target, it's called a true positive. If the classifier correctly classifies non-target as non-target, it's called a true negative. The mistake/error if made by classifying a target as non-target is called a false negative while the mistake/error made by classifying non-target as a target is a false positive. Figure 3.12 summarises all the possible outputs.

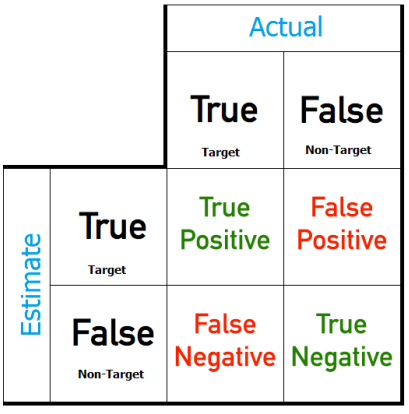

FIGURE 3.12: Summary of possible outcomes.

The true positive rate is called sensitivity while true negative rate is called specificity. The ideal classifier would be 100 percent sensitive and 100 percent specific.

True positive rate (or sensitivity):

$$TPR = TP/(TP + FN).$$

False positive rate:

$$FPR = FP/(FP + TN).$$

True negative rate (or specificity):

$$TNR = TN/(FP + TN).$$

3.6.2 Receiver operating characteristic curve

The receiver operating characteristic (ROC) curve is an effective method of evaluating the performance of binary tests. The curve is created by plotting the true positive rate against the false positive rate at various threshold settings.

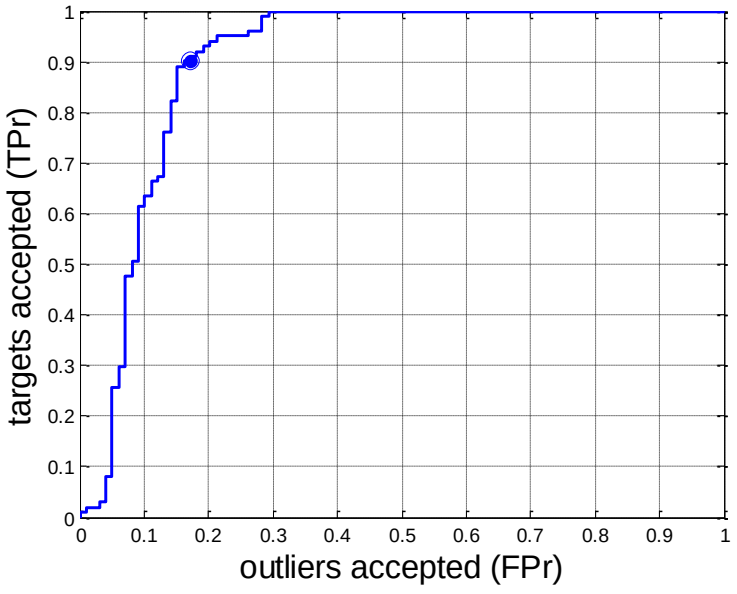

FIGURE 3.13: ROC example.

Plotting ROC curve as shown in Figure 3.13 gives good summary of the performance of a one-class classifier; however it's hard to compare two ROC curves. To summarize ROC in single number, area under ROC's are computed. Greater area under ROC means better performance of classifier. For an ideal classifier the area would be 1. This means it accurately separates the target from non-target.

3.6.3 Cross-validation

Cross-validation is primarily a way of measuring the predictive performance of a classifier. It is a technique of estimating the accuracy of a classifier on an independent dataset. In cross validation first the available data is divided to k mutually exclusive subsets of approximately equal size [19]. Then out of k subsets, a single subset is used for testing the classifier while the remaining k-1 subsets are used for training. The process is repeated k times where every subset is tested once while not used during training.

The audio recordings for our experiments are not of same length and we want to use all frames of single audio recording either for training or for testing. Usually 10-fold cross validation is used for evaluation, but in case the length of audio files are different in length, there is a possibility that from same audio recording some frames might be used for training while other frames from same recording are used for testing . It is desirable to make sets of corresponding objects (e.g. frames from the same recordings) all together in the training set or in the test set. For this purpose, we use a leave-one-out cross-validation strategy where all the frames of one recording are considered for testing while the rest of the recordings are considered for training.

Chapter 4

Experiments and results

In this chapter, we performed experiments for comparing the performance of SVDD with some other binary classifiers for the railway vehicle detection problem. Different models (classifiers) were trained and tested with MFCCs extracted from audio files provided by Sensornet. In one-class classifier, AUC is computed by testing every audio recording one by one by using leave-one-set-out cross validation. Every time the score is computed for a single file by considering the number of frames detected as a train in an audio file by the trained classifier. No data from non-target class is used during training of SVDD. For target files testing, all the target audio files except the one which is under test is used as the training set. For non-target testing, all the target data is used while no file is used for training from the non-target class.

For multi-class classifiers, leave-one-set-out cross validation is used for evaluating the performance with varying number of non-target training data samples. The purpose of this experiment is to show that when there is limited amount of non-target training data, we get suboptimal performance from a binary classifier. AUC is computed after all audio recordings (target and non-target) are tested and for each test data, we train a separate classifier. During the experiments, in the beginning, only one audio recording from non-target class is randomly selected for training the classifier. The process is repeated by increasing the number of audio recordings from non-target class (randomly selected for every test file) till all the non-target recordings except the one to be tested are used.

4.1 Data

Our data consist of 245 audio recordings in total. Out of total recordings, 200 audio files are target and the rest 45 are non-target. The target files consist of passenger and freight trains while non-target data contain recordings of tractors, cars, buses, aeroplanes, scooters and other vehicles. Further details of data are given in Appendix D

We performed experiments on the 13 MFCCs extracted from the given data. The 13 dimensional feature vectors are extracted for 200 millisecond frame from the audio files after segmentation. Matlab toolboxes Prtools [20] and DDtools [21] were used extensively for the experiments.

4.2 Multi-class results

In the following sections, different multi-class classifiers are tested with data from Bathmen. AUC is reported for range of experiments where in every experiment, different number of audio recordings from non-target class are used for training. Every time "leave-one-set-out cross validation" with random picking for non-target files is done for testing each file.

4.2.1 LDC results

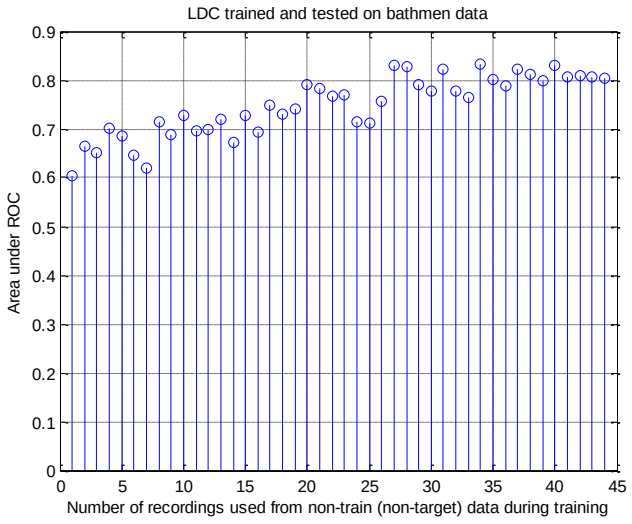

FIGURE 4.1: Full area under ROC curve, train and test on Bathmen with the LDC classifier.

Figure 4.1 shows the area under ROC curve calculated after leave-one-set-out cross validation for range of experiments using linear Bayes normal classifier.

It is clear from Figure 4.1 that training with less number of recordings from non-target class, the accuracy is very low. Starting with single audio recording from non-target class for training, the AUC is merely 0.6. The AUC gets better with increase in number of non-train recordings and it reaches the mark of 0.8 when 20 non-target recordings are used during training. Also there is fluctuation in results and the reason behind this is random nature and picking of audio files during each loop of testing a single audio recording. Certainly LDC cannot be used for classification of audio data when the non-target examples are scarce and diverse in nature.

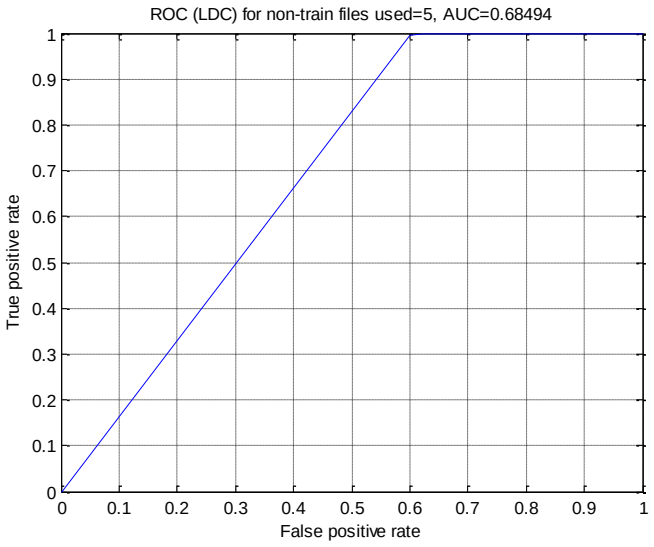

FIGURE 4.2: ROC curve for the LDC classifier when 5 outlier recordings are used

Figure 4.2 shows the ROC curve obtained after "leave-one-set-out cross validation" for linear Bayes normal classifier when five files are used from non-target class during training. In this case, 100 percent accuracy for true positive rate can be achieved but at cost of 60 percent false positive rate, which is not acceptable because of abundant non-targets in real scenario.

4.2.2 QDC results

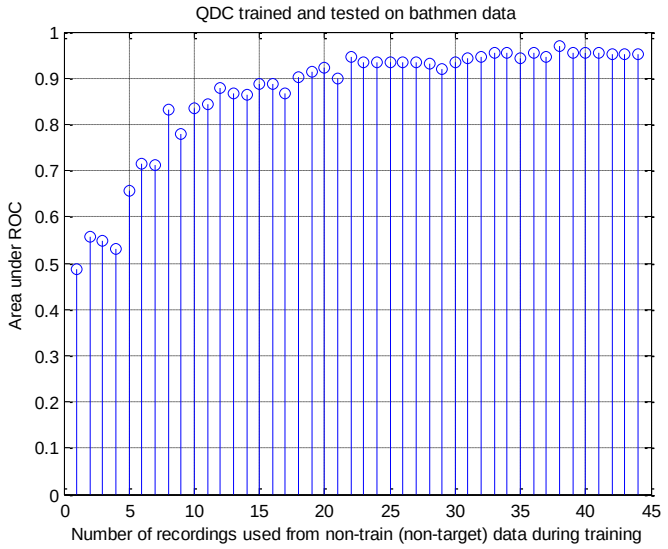

FIGURE 4.3: Full area under ROC curve, train and test on Bathmen with QDC.

Figure 4.3 shows the area under ROC curve calculated after "leave-one-set-out cross validation" for range of experiments using quadratic Bayes normal classifier.

With just one file used from non-target class, the AUC is 0.5, which means random prediction. In the beginning, the performance of classifier is worse but with increase in number of non-target audio recordings, the performance is drastically increased. The results indicate that if reliable and well sampled data is available from the non-target class, QDC can perform well. In our case, the data from non-target class is diverse in nature and we do not want to use any information from it. Since with few files (less than 5) from non-target class the AUC is less than 0.8, so its usage cannot be recommended for audio cases where less data is available for non-target classs.

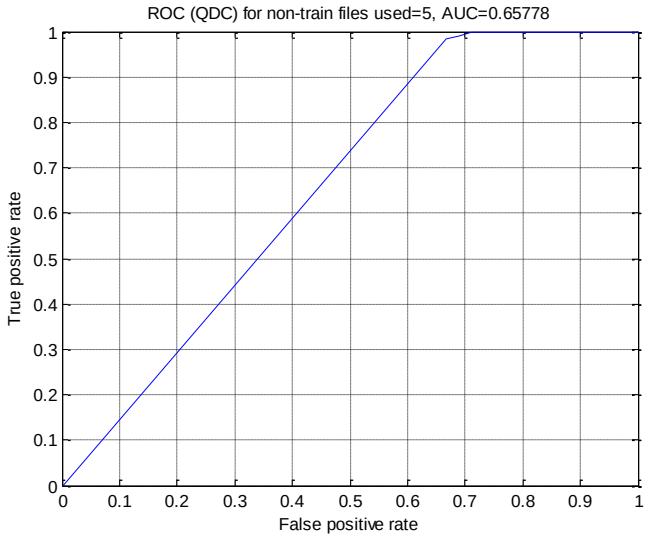

FIGURE 4.4: ROC curve for the QDC classifier when 5 non-target recordings are used.

Figure 4.4 shows the ROC curve obtained after "leave-one-set-out cross validation" for quadratic Bayes normal classifier when five files are selected randomly from non-target class during training for testing every audio file. It can be seen that true positive rate goes to 1 (100 percent) but at cost of around 0.7 (70 percent) false positive rate.

4.2.3 Naive Bayes results

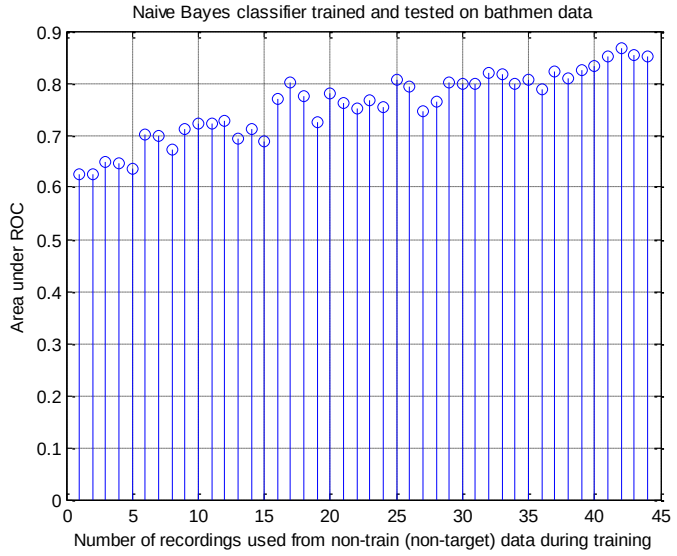

FIGURE 4.5: Full area under ROC curve, train and test on Bathmen for Naive Bayes classifier.

Figure 4.5 shows the area under ROC curve calculated after "leave-one-set-out cross validation" for range of experiments using naive Bayes classifier. Initially, with just one file used from the non-target class, the AUC is just above 0.6. There is a bit of improvement noticed when the number of non-target recordings are increased to 7. The AUC goes to 0.8 when 17 audio recordings are used for training to test every single audio file. In naive Bayes classifier, the AUC is less than 0.9 when all the non-target (44) recordings are used for training.

Figure 4.6 shows the ROC curve obtained after "leave-one-set-out cross validation" for naive Bayes classifier when five files are selected randomly from non-target class during training for testing every audio file. It can be seen that true positive rate goes to 1 (100 percent) but at cost of around more than 0.7 (70 percent) false positive rate for naive Bayes classifier when 5 non-target recordings are used for training the classifier.

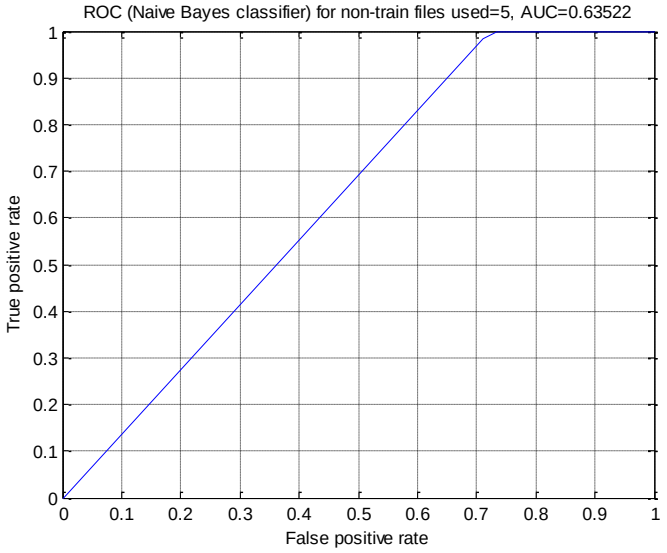

FIGURE 4.6: ROC curve for the Naive Bayes classifier when 5 non-target recordings are used.

4.2.4 SVM with RBF kernel

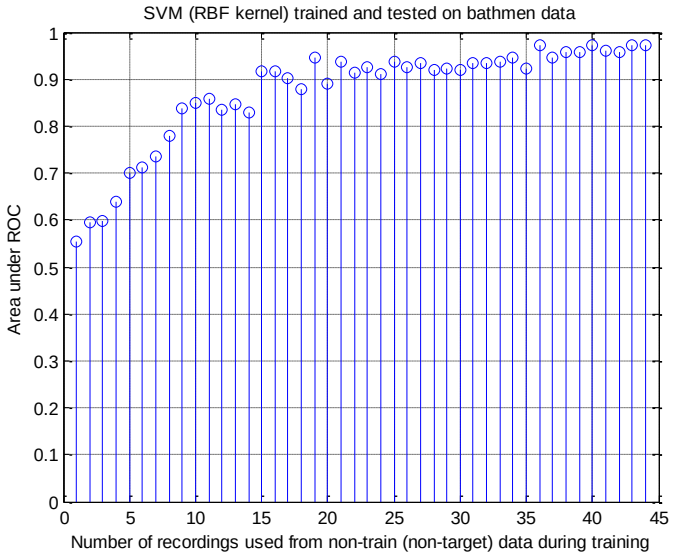

FIGURE 4.7: AUC for SVM (RBF kernel).

Figure 4.7 shows the area under ROC curve calculated after leave-one-set-out cross validation for range of experiments using support vector machines using RBF kernel with optimized parameters.

For SVM with radial basis function kernel (RBF kernel), optimized parameters are found in a separate experiment. The parameters to be optimised are C and σ. The coefficient C affects the trade-off between complexity and proportion of non-separable samples [22].

For finding optimized parameters C and σ, the data is divided into two sets (Train and test). 100 target files out of 200 total target files were selected for training while rest 100 for testing, 23 non-target files out of 45 were used for training while rest 22 for testing. The experiment was repeated for range of C and σ values combination. Results for finding the best parameters can be found in Appendix C.

Figure 4.7 shows the area under ROC curve calculated after leave-one-set-out cross validation for range of experiments using support vector machines with RBF kernel with optimized parameters. Figure 4.7 shows that initially the AUC is less than 0.6 when single non-target recording is used for training. The AUC becomes more than 0.8 when 9 or more than 9 non-target recordings are used for training the classifier.

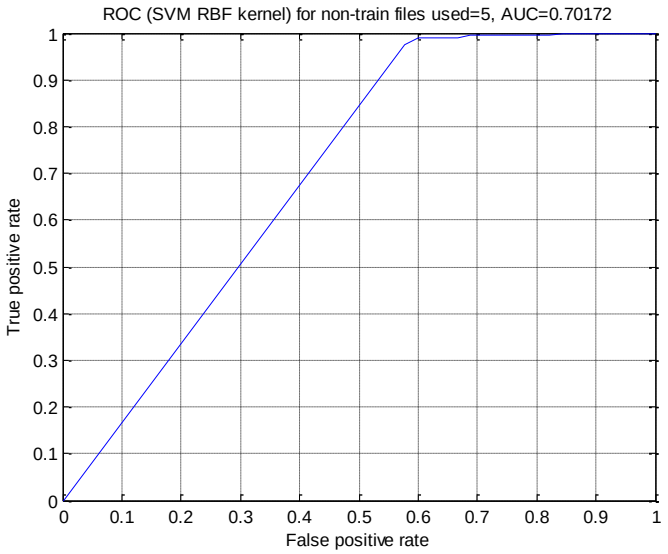

FIGURE 4.8: ROC curve for SVM (RBF kernel) when 5 non-target recordings are used.

Figure 4.8 shows the ROC curve obtained after leave-one-set-out cross validation for SVM with RBF kernel when five files are selected randomly from non-target class during training for testing every audio file.

4.3 SVDD results

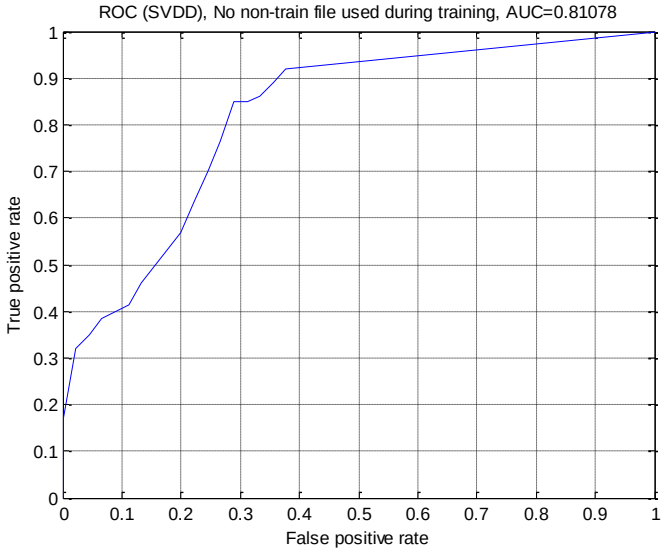

FIGURE 4.9: ROC curve for SVDD

Figure 4.9 shows the area under ROC curve calculated after leave-one-set-out cross validation for support vector data description. No Data was used from non-target class during the training. It is clear from the figure that more than 0.8 (80 percent) true positive rate is achieved with less than 0.3 (30 percent) false positive rate without using any information from non-target class during training the classifier. These results show that SVDD can perform better than binary classifiers when non-target training data is limited.

4.4 Real time results

The SVDD model was put on test in real time at a town in eastern Netherlands (Bathmen). The audio event is decided to be a target if the maximum threshold value after windowing is greater or equal to 10 percent. To evaluate the performance of the classifier, the results are cross checked with the video recorded at that particular time in real time as well.

Table 4.1 shows the detailed count of events and results for 24 hours test while Table 4.2 summaries the results in real time. The results indicate that SVDD is effective in getting low false alarms in online operation, while it may miss some percentage of genuine trains.

Time start	Time End	No. of events	No. of trains	No. of outliers	TP count	FN count	FP count	TN count
17:00	17:30	14	4	10	2	2	0	10
17:30	18:00	19	5	14	4	1	0	14
18:00	18:30	8	5	3	2	3	0	3
18:30	19:00	12	4	8	3	1	1	7
19:00	19:30	10	4	6	3	1	0	6
19:30	20:00	6	4	2	4	0	0	2
20:00	20:30	7	6	1	5	1	0	1
20:30	21:00	3	2	1	2	0	0	1
21:00	21:30	6	4	2	3	1	0	2
21:30	22:00	8	5	3	5	0	0	3
22:00	22:30	6	4	2	3	1	0	2
22:30	23:00	5	4	1	3	1	0	1
23:00	23:30	7	6	1	5	1	0	1
23:30	00:00	3	3	0	1	2	0	0
00:00	00:30	4	4	0	4	0	0	0
00:30	01:00	2	2	0	2	0	0	0
01:00	01:30	0	0	0	0	0	0	0
01:30	02:00	0	0	0	0	0	0	0
02:00	02:30	1	1	0	1	0	0	0
02:30	03:00	0	0	0	0	0	0	0
03:00	03:30	0	0	0	0	0	0	0
03:30	04:00	0	0	0	0	0	0	0
04:00	04:30	4	0	4	0	0	0	4
04:30	05:00	16	1	15	1	0	0	15
05:00	05:30	15	1	14	1	0	0	14
05:30	06:00	8	1	7	1	0	0	7
06:00	06:30	6	3	3	3	0	0	3
06:30	07:00	6	2	4	2	0	0	4
07:00	07:30	9	4	5	2	2	0	5
07:30	08:00	8	3	5	3	0	0	5
08:00	08:30	4	4	0	2	2	0	0
08:30	09:00	5	5	0	5	0	0	0
09:00	09:30	8	6	2	5	1	0	2
09:30	10:00	4	4	0	2	2	0	0
10:00	10:30	9	6	3	3	3	0	3
10:30	11:00	10	6	4	5	1	0	4
11:00	11:30	7	3	4	3	0	0	4
11:30	12:00	8	4	4	4	0	0	4
12:00	12:30	6	4	2	3	1	0	2
12:30	13:00	5	4	1	4	0	0	1
13:00	13:30	6	4	2	3	1	0	2
13:30	14:00	8	5	3	4	1	0	3
14:00	14:30	10	5	5	4	1	0	5
14:30	15:00	8	3	5	2	1	0	5
15:00	15:30	8	3	5	2	1	0	5
15:30	16:00	5	4	1	4	0	0	1
16:00	16:30	6	4	2	2	2	0	2
16:30	17:00	11	4	7	2	2	0	7
17:00	17:30	7	4	3	3	1	0	3
	Total	**328**	**164**	**164**	**127**	**37**	**1**	**163**

TABLE 4.1: Detailed count of audio events and classifier output for Bathmen

True positive rate	77.44
False positive rate	0.61
True negative rate	99.39
False negative rate	22.56

TABLE 4.2: Summary of the results in Table 4.1.

Chapter 5

Conclusion and future work

5.1 Conclusion

In this thesis, we discussed the problem of detecting target audio (train) data using one class classification. We used SVDD for training the model without using any data from non-target class and compared it with multi-class classifiers where data is used from both classes for training the classifier.

In detecting a train, it is impossible to model representative distribution of non-trains because of diverse possibilities. The results indicated that SVDD works well in cases where one of the classes is severely under-sampled due to the diverse nature of data or cost of measurement for that class.

One of the main goals at Sensornet was to decrease the false positive rate in real time and it was achieved very well as reported in real time test results.

5.2 Future work

There is possibility of extending this work further by classifying target data as passenger train and cargo (freight) train. Also the SVDD results may be further optimized by repeating the experiments for a range of σ values. In real time, the results can be further improved by adding all the missed trains into the data-set for training.

Appendix A

Matlab codes

A.1 Removing redundant starting and ending points from audio target signals

```
clc
close all
clear all

    percent=75;
    Fhandle=fopen('AllTrainsAgain.txt','r');
    F=textscan(Fhandle,'%s','delimiter','\n');
    TrainsTotal=length(F{1});

for i=1:TrainsTotal

    Filename = F{1}{i};
    fprintf('reading %s\n',Filename);
    [Ntrain,Fs]=audioread(Filename);

    NtrainT=Ntrain';

    envelope = abs(NtrainT);
    VO1 = tsmovavg(envelope, 's', 6400);
    Maxval=max(VO1)
    Threshould=(Maxval/100)*percent;

    YESNO=VO1>= Threshould;

    idx1 = find(YESNO~=0, 1, 'first');
    idx2 = find(YESNO~=0, 1, 'last');
```

```
    Cut=NtrainT(idx1:idx2);

Namefiles=['CroppedTrain' num2str(i) '.wav'] ;
wavwrite(Cut,Fs,Namefiles);
i

end
```

A.2 Extracting features

```
clc
clear all
close all

Tw = 200;           % analysis frame duration (ms)
Ts = 200;           % analysis frame shift (ms)
alpha = 0.97;       % preemphasis coefficient
R = [ 10 4200 ];  % frequency range to consider
M = 20; % number of filterbank channels
C = 13; % number of cepstral coefficients
L = 22; % cepstral sine lifter parameter

% hamming window
hamming = @(N)(0.54-0.46*cos(2*pi*[0:N-1].'/(N-1)));

%Complete trains (change list for other locations)
Fhandle=fopen('alltrainsCroppedHengelo.txt','r');
F=textscan(Fhandle,'%s','delimiter','\n');
numNoises=length(F{1});

TrainFeatures=[];
SUBlabelTrain=[];
for i=1:numNoises
    Filename = F{1}{i};
    fprintf('reading %s\n',Filename);
    [Ntrain,Fs]=audioread(Filename);
[FMFCCs, FBEs, frames ] =...
mfccforsensornetmethod(Ntrain, Fs, Tw, Ts, alpha, hamming, R, M, C, L );
    FMFCCs=transpose(FMFCCs);
    TrainFeatures=[TrainFeatures;FMFCCs];

    Size=size(FMFCCs,1);
    Indexlabel=genlab(Size,i);
    SUBlabelTrain=[SUBlabelTrain;Indexlabel];
```

```
end

sizee=size(TrainFeatures,1);
Tlabel=repmat([1],sizee,1);

%  outliers list (may change for different locations)

Fhandle=fopen('hengelooutliers.txt','r');
F=textscan(Fhandle,'%s','delimiter','\n');
numNoises=length(F{1});

OutlierFeatures=[];
SublabelOutliers=[];
for j=1:numNoises
    Filename = F{1}{j};
    fprintf('reading %s\n',Filename);
    [Ntrain,Fs]=audioread(Filename);
    [FMFCCs, FBEs, frames ] =...
    mfccforsensornetmethod( Ntrain, Fs, Tw, Ts, alpha, hamming, R, M, C, L );
    FMFCCs=transpose(FMFCCs);
    OutlierFeatures=[OutlierFeatures;FMFCCs];

    Size=size(FMFCCs,1);
    Indexlabel=genlab(Size,(j+i));
    SublabelOutliers=[SublabelOutliers;Indexlabel];

end

sizee=size(OutlierFeatures,1);
OutlierLabel=repmat([0],sizee,1);

Features=[TrainFeatures;OutlierFeatures];
labels=[Tlabel;OutlierLabel];
SUBlabel=[SUBlabelTrain;SublabelOutliers];

MissClassifyCountW1=0;
Totalfiles=max(SUBlabel);

filename = 'CroppedTrains75Hengelooutliersoutliers10to4200200wsize200.mat';
save(filename)
```

A.3 SVDD experiments codes

```
%%%%%%%%%%%%%%%%%%
```

```
% this is for sensornet
clc
close all
clear all

addpath dd_tools
addpath prtools
prmemory(inf)
load V4NEWLISTCroppedTrains50_Bathmen_50Cropoutliersoutliers10to4200200wsize200

prmemory(inf)
tic
MissClassifyCountW1 =0;
s1=25; %Sigma Value
eotg=0.1; %Error on target class

for i=1:Totalfiles
        i
    TestIndex=find(SUBlabel == i);
    Testrows=TestIndex';
    TeFeatures=Features(Testrows,:);
    Telabels=labels(Testrows);

    TrainIndex=find(SUBlabel ~= i);
    Trrows=TrainIndex';
    TrFeatures=Features(Trrows,:);
    Trlabels=labels(Trrows);

    TrData=prdataset(TrFeatures,Trlabels);
    TeData=prdataset(TeFeatures,Telabels);

    if Telabels(1)==1
        TstSet = gendatoc(TeData,[]);
    end

    if Telabels(1)~=1
        TstSet = gendatoc([],TeData);
    end

    [TrSet,I1] = oc_set(TrData,2);

    LABELSorig = getlab(TeData);
    OriginalClass=mode(LABELSorig);

    %%speedup because we dont have to train again and again for outliers class
    %%testing
    if i<202
        W1= svdd(target_class(TrSet),eotg,s1);
```

```
    end
    %%speedupends

    %W1
    TesterrorW1=testc(TstSet*W1);
    LABELSPredictW1=TstSet*W1*labeld;
    LABELSPredictW1 = double(LABELSPredictW1);
    LABELSPredictW1 = LABELSPredictW1(:,1);
    LABELSPredictW1(LABELSPredictW1==116) = 1; %116 is target
    LABELSPredictW1(LABELSPredictW1==111) = 0;
    PredictedClassW1=mode(LABELSPredictW1);
    if OriginalClass~=PredictedClassW1
        MissClassifyCountW1=MissClassifyCountW1+1;
    end

    e1(i)=TesterrorW1;
    OriginalLabel(i)=OriginalClass;
    PredictedLabelW1(i)=PredictedClassW1;

    CorrectLabelsframesW1(i)=length(find(LABELSPredictW1==OriginalClass));

end

Mean1=mean(e1);
SDeviation1=std(e1);

FileErrorW1=MissClassifyCountW1/Totalfiles;
timeElapsedd=toc

filename = 'V4_5DEC2015NEWLISTLOSOCroppedTrains50vs50bathmanDataW1_sigma25_fr0p1.mat';
save(filename)

close all
clear all
```

A.4 ROC and area under ROC for multiclass experiment

```
%%%%%%%%%%%%%%%%%% This code is for finding Area under ROCs
%%%%%%%%%%%%%%%%%% and plot the maximum area under ROC
% every time n (1:44) outlier files were selected
clc
clear
close all
```

```
for kk = 1: 44 %loop is according to number of outlier files
    kk

    %change name accordingly if different
    load(['Latest_V216OctLDC50vs50sabanciFullTest_' num2str(kk) '.mat'])

    TotalTrains= Totalfiles-numNoises;
    score(1:TotalTrains)=pctCorFrames(1:TotalTrains);
    score((TotalTrains+1):Totalfiles)=100-pctCorFrames((TotalTrains+1):Totalfiles);
    score=score/100;
    [X1,Y1,T1,AUC1] = perfcurve(OriginalLabel,score,1,'XVals',[0:0.001:1]);

    X{kk}=X1;
    Y{kk}=Y1;
    T{kk}=T1;
    AUC(kk)=AUC1;

end

stem(AUC);
xlabel('Number of recordings used from Outlier-Class during Training')
ylabel('Area uner ROC')
title({'LOSO for LDC ','Trained and tested on Bathmen'})
grid on

figure
[maxaucVALUE, maxauxINDEX]=max(AUC);
Xmax2=X{maxauxINDEX}
Ymax2=Y{maxauxINDEX}
Tmax=T{maxauxINDEX}
plot(Xmax2,Ymax2)

title(['ROC for AUC=',num2str(maxaucVALUE),'...
non-target files used=',num2str(maxauxINDEX)]);
grid on
xlabel('False positive rate')
ylabel('True positive rate')
```

A.5 Plotting filterbank

```
%This code is for finding/plotting filterbnks
clc
clear all
```

```
close all

fs=32000;
M=20;

MinF=10;
MaxF=4200;

Frameduration=200; %ms

Nw = round( 1E-3*200*fs ) ;   % frame duration (samples)
nfft = Nw;
K = nfft/2+1  ;

hz2mel = @( hz )( 1127*log(1+hz/700) );     % Hertz to mel warping function
mel2hz = @( mel )( 700*exp(mel/1127)-700 );
R = [ MinF MaxF ];

first=hz2mel(MinF);
last=hz2mel(MaxF);

difference=last-first;

onefiltersize_inMelz=difference/(M+1) %Difference in MELZ
for i=1:(M+2)
inmilz(i)=first;
first=first+onefiltersize_inMelz;
end

for i=1:(M+2)
 value=  inmilz(i);
  inhz(i)=mel2hz(value)  ;
 end

for i=1:M
yax=[0 1 0];
xax=[inhz(i) inhz(i+1) inhz(i+2)];
plot(xax,yax,'r')
hold on
grid on
end
```

Appendix B

Extra results and experiments

B.1 Extra experiments SVDD

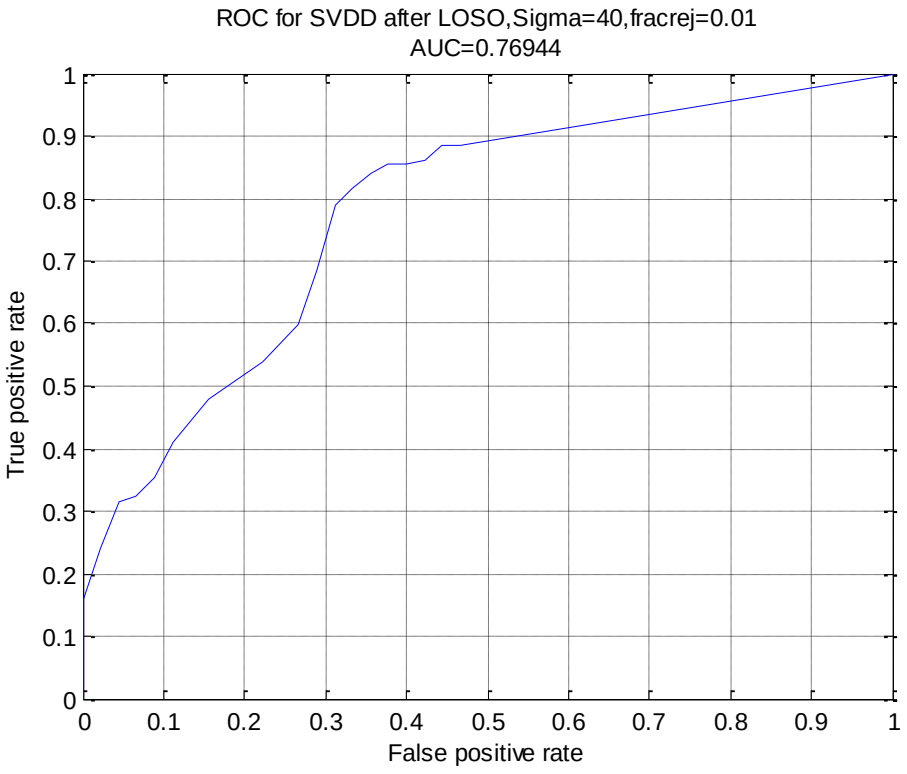

FIGURE B.1: ROC for SVDD with σ=40.

AUC = 0.7694 (Time Elapsed 151.94 hours)		
False positive rate	True positive rate	Threshold
0	0	1.0000
0	0.1600	0.8913
0.0222	0.2400	0.7692
0.0444	0.3150	0.7029
0.0667	0.3250	0.6923
0.0889	0.3550	0.6667
0.1111	0.4100	0.6071
0.1556	0.4800	0.5172
0.2222	0.5400	0.4063
0.2667	0.6000	0.3333
0.2889	0.6850	0.2100
0.3111	0.7900	0.1176
0.3333	0.8150	0.0968
0.3556	0.8400	0.0833
0.3778	0.8550	0.0714
0.4000	0.8550	0.0667
0.4222	0.8600	0.0588
0.4444	0.8850	0.0303
0.4667	0.8850	0.0227
1.0000	1.0000	0

TABLE B.1: Thresholds for Figure B.1.

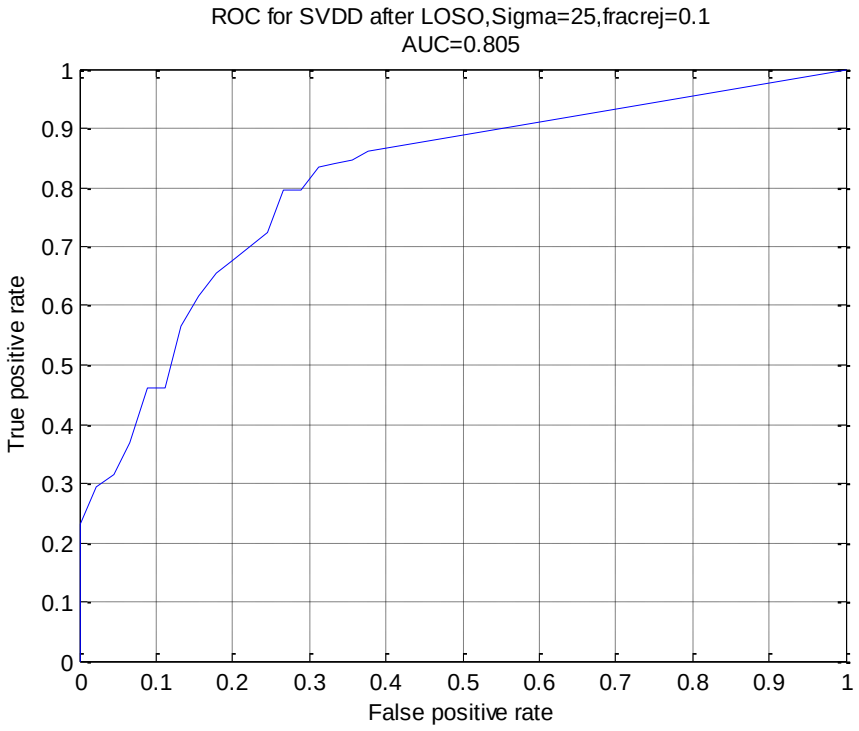

FIGURE B.2: ROC for SVDD with σ=30.

AUC1 =0.805		
False positive rate	True positive rate	Threshold
0	0	1.0000
0	0.2300	0.7813
0.0222	0.2950	0.7143
0.0444	0.3150	0.6667
0.0667	0.3700	0.6098
0.0889	0.4600	0.5217
0.1111	0.4600	0.5172
0.1333	0.5650	0.4138
0.1556	0.6150	0.3333
0.1778	0.6550	0.2857
0.2000	0.6750	0.2609
0.2444	0.7250	0.2083
0.2667	0.7950	0.1200
0.2889	0.7950	0.1148
0.3111	0.8350	0.0833
0.3333	0.8400	0.0769
0.3556	0.8450	0.0714
0.3778	0.8600	0.0156
1.0000	1.0000	0

TABLE B.2: Thresholds for Figure B.2.

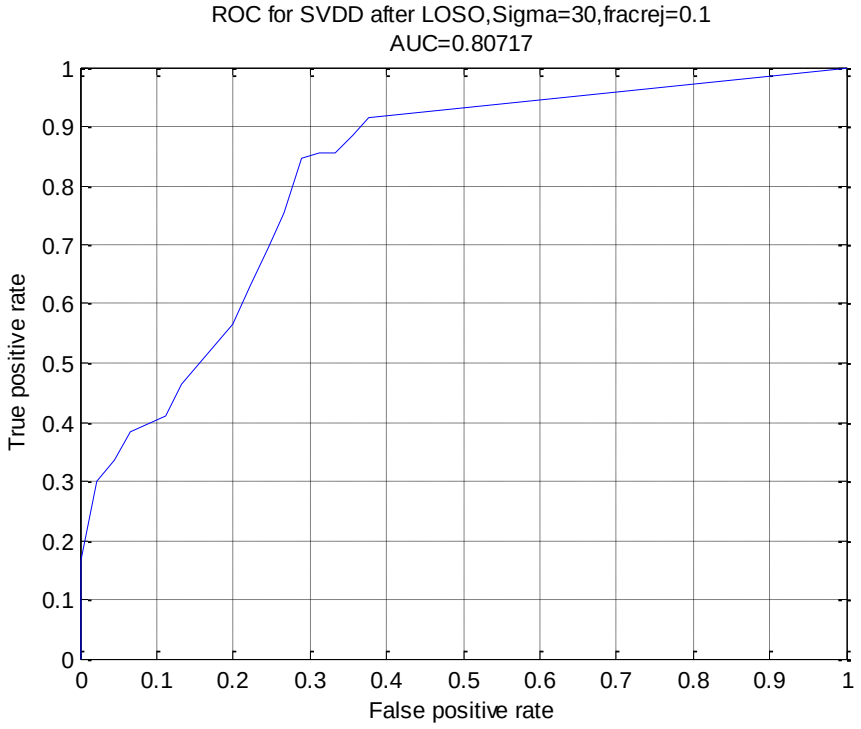

FIGURE B.3: ROC for SVDD with σ=25.

AUC =0.80717		
False positive rate	True positive rate	Threshold
0	0	1.0000
0	0.1650	0.8545
0.0222	0.3000	0.7100
0.0444	0.3350	0.6667
0.0667	0.3850	0.6121
0.1111	0.4100	0.5882
0.1333	0.4650	0.5093
0.2000	0.5650	0.4058
0.2222	0.6300	0.3333
0.2444	0.6950	0.2286
0.2667	0.7550	0.1818
0.2889	0.8450	0.1000
0.3111	0.8550	0.0870
0.3333	0.8550	0.0833
0.3556	0.8850	0.0714
0.3778	0.9150	0.0156
1.0000	1.0000	0

TABLE B.3: Thresholds for Figure B.3.

B.2 Real time results

True positive rate	75.33
False positive rate	2.08
True negative rate	97.91
False negative rate	24.66

TABLE B.4: Summary of the results in Table B.6.

True positive rate	82.20
False positive rate	0.00
True negative rate	100.00
False negative rate	17.79

TABLE B.5: Summary of the results in Table B.7.

Time Start	Time End	No. of events	No. of trains	No. of outliers	TP count	FN count	FP count	TN count
14:00	14:30	5	4	1	3	1	0	1
14:30	15:00	9	4	5	4	0	0	5
15:00	15:30	8	5	3	4	1	0	3
15:30	16:00	9	5	4	3	2	0	4
16:00	16:30	5	4	1	3	1	0	1
16:30	17:00	4	3	1	3	0	0	1
17:00	17:30	11	4	7	4	0	1	6
17:30	18:00	6	5	1	3	2	0	1
18:00	18:30	7	3	4	3	0	0	4
18:30	19:00	6	5	1	4	1	0	1
19:00	19:30	8	4	4	4	0	0	4
19:30	20:00	4	4	0	3	1	0	0
20:00	20:30	7	7	0	5	2	0	0
20:30	21:00	6	4	2	2	2	0	2
21:00	21:30	4	4	0	1	3	0	0
21:30	22:00	8	4	4	4	0	0	4
22:00	22:30	5	4	1	3	1	0	1
22:30	23:00	5	5	0	5	0	0	0
23:00	23:30	4	4	0	3	1	0	0
23:30	00:00	3	3	0	2	1	0	0
00:00	00:30	1	1	0	1	0	0	0
00:30	01:00	4	3	1	1	2	0	1
01:00	01:30	3	3	0	1	2	0	0
01:30	02:00	1	0	1	0	0	0	1
02:00	02:30	2	1	1	1	0	0	1
02:30	03:00	0	0	0	0	0	0	0
03:00	03:30	0	0	0	0	0	0	0
03:30	04:00	1	0	1	0	0	0	1
04:00	04:30	0	0	0	0	0	0	0
04:30	05:00	1	0	1	0	0	0	1
05:00	05:30	0	0	0	0	0	0	0
05:30	06:00	3	0	3	0	0	0	3
06:00	06:30	2	1	1	1	0	0	1
06:30	07:00	3	2	1	2	0	0	1
07:00	07:30	3	3	0	3	0	0	0
07:30	08:00	4	3	1	3	0	0	1
08:00	08:30	6	4	2	3	1	0	2
08:30	09:00	6	4	2	3	1	0	2
09:00	09:30	6	4	2	3	1	0	2
09:30	10:00	5	4	1	3	1	0	1
10:00	10:30	5	3	2	1	2	0	2
10:30	11:00	5	3	2	2	1	0	2
11:00	11:30	9	3	6	3	0	1	5
11:30	12:00	10	4	6	2	2	0	6
12:00	12:30	10	4	6	3	1	0	6
12:30	13:00	5	3	2	1	2	0	2
13:00	13:30	6	4	2	3	1	0	2
13:30	14:00	9	4	5	3	1	0	5
14:00	14:30	12	4	8	4	0	0	8
	Total	246	150	96	113	37	2	94

TABLE B.6: Detailed results from real time test of SVDD on March 27 2015.

Time Start	Time End	No. of events	No. of trains	No. of outliers	TP count	FN count	FP count	TN count
14:00	14:30	3	3	0	0	3	0	0
14:30	15:00	2	2	0	2	0	0	0
15:00	15:30	4	4	0	4	0	0	0
15:30	16:00	3	3	0	3	0	0	0
16:00	16:30	3	3	0	2	1	0	0
16:30	17:00	3	3	0	3	0	0	0
17:00	17:30	3	3	0	2	1	0	0
17:30	18:00	4	4	0	3	1	0	0
18:00	18:30	6	5	1	4	1	0	1
18:30	19:00	3	3	0	3	0	0	0
19:00	19:30	4	4	0	4	0	0	0
19:30	20:00	6	4	2	4	0	0	2
20:00	20:30	4	4	0	3	1	0	0
20:30	21:00	3	3	0	3	0	0	0
21:00	21:30	3	3	0	3	0	0	0
21:30	22:00	3	3	0	3	0	0	0
22:00	22:30	3	3	0	2	1	0	0
22:30	23:00	3	3	0	3	0	0	0
23:00	23:30	4	3	1	3	0	0	1
23:30	00:00	4	3	1	3	0	0	1
00:00	00:30	3	3	0	3	0	0	0
00:30	01:00	2	2	0	2	0	0	0
01:00	01:30	1	1	0	1	0	0	0
01:30	02:00	0	0	0	0	0	0	0
02:00	02:30	0	0	0	0	0	0	0
02:30	03:00	0	0	0	0	0	0	0
03:00	03:30	0	0	0	0	0	0	0
03:30	04:00	0	0	0	0	0	0	0
04:00	04:30	0	0	0	0	0	0	0
04:30	05:00	0	0	0	0	0	0	0
05:00	05:30	0	0	0	0	0	0	0
05:30	06:00	0	0	0	0	0	0	0
06:00	06:30	0	0	0	0	0	0	0
06:30	07:00	0	0	0	0	0	0	0
07:00	07:30	0	0	0	0	0	0	0
07:30	08:00	1	1	0	1	0	0	0
08:00	08:30	3	3	0	2	1	0	0
08:30	09:00	3	3	0	2	1	0	0
09:00	09:30	4	4	0	4	0	0	0
09:30	10:00	4	4	0	3	1	0	0
10:00	10:30	5	5	0	5	0	0	0
10:30	11:00	3	3	0	2	1	0	0
11:00	11:30	4	4	0	3	1	0	0
11:30	12:00	3	3	0	0	3	0	0
12:00	12:30	4	4	0	1	3	0	0
12:30	13:00	3	3	0	3	0	0	0
13:00	13:30	3	3	0	2	1	0	0
13:30	14:00	4	3	1	3	0	0	1
14:00	14:30	6	3	3	3	0	0	3
	Total	127	118	9	97	21	0	9

TABLE B.7: Detailed results from real time test of SVDD on June 28 2015.

Appendix C

Optimizing RBF kernel

	$\sigma = 2^{-10}$	$\sigma = 2^{-5}$	$\sigma = 2^{0}$	$\sigma = 2^{5}$	$\sigma = 2^{10}$
$C = 2^{-5}$	0.9278	0.9278	0.9278	0.9245	0.9225
$C = 2^{-4}$	0.9278	0.9278	0.9278	0.924	0.9225
$C = 2^{-3}$	0.9278	0.9278	0.9278	0.9235	0.9273
$C = 2^{-2}$	0.9278	0.9278	0.9278	0.9324	0.9276
$C = 2^{-1}$	0.9278	0.9278	0.9278	0.9316	0.9273
$C = 2^{0}$	0.9278	0.9278	0.9278	0.9334	0.9263
$C = 2^{1}$	0.9278	0.9278	0.9278	0.9311	0.9271
$C = 2^{2}$	0.9278	0.9278	0.9278	0.9288	0.9273
$C = 2^{3}$	0.9278	0.9278	0.9278	0.9286	0.9271
$C = 2^{4}$	0.9278	0.9278	0.9278	0.9357	0.9273
$C = 2^{5}$	0.9278	0.9278	0.9278	0.9426	0.9271

TABLE C.1: SVM (RBF kernel) optimization.

	$\sigma = 2^{3}$	$\sigma = 2^{4}$	$\sigma = 2^{5}$	$\sigma = 2^{6}$	$\sigma = 2^{7}$
$C = 2^{6}$	0.9451	0.9484	0.9469	0.9299	0.925
$C = 2^{7}$	0.9451	0.9484	0.9416	0.9291	0.9276
$C = 2^{8}$	0.9451	0.9484	0.9413	0.9375	0.9319
$C = 2^{9}$	0.9451	0.9484	0.9423	0.939	0.9281
$C = 2^{10}$	0.9451	0.9484	0.9418	0.9433	0.9319
$C = 2^{11}$	0.9451	0.9484	0.9418	0.9398	0.9281
$C = 2^{12}$	0.9451	0.9484	0.9418	0.941	0.9357
$C = 2^{13}$	0.9451	0.9484	0.9418	0.9466	0.938
$C = 2^{14}$	0.9451	0.9484	0.9418	0.9441	0.9428
$C = 2^{15}$	0.9451	0.9484	0.9418	0.9441	0.939
$C = 2^{16}$	0.9451	0.9484	0.9418	0.9441	0.9413

TABLE C.2: SVM (RBF kernel) optimization.

Appendix D

Details of data

D.1 PCA plots of features

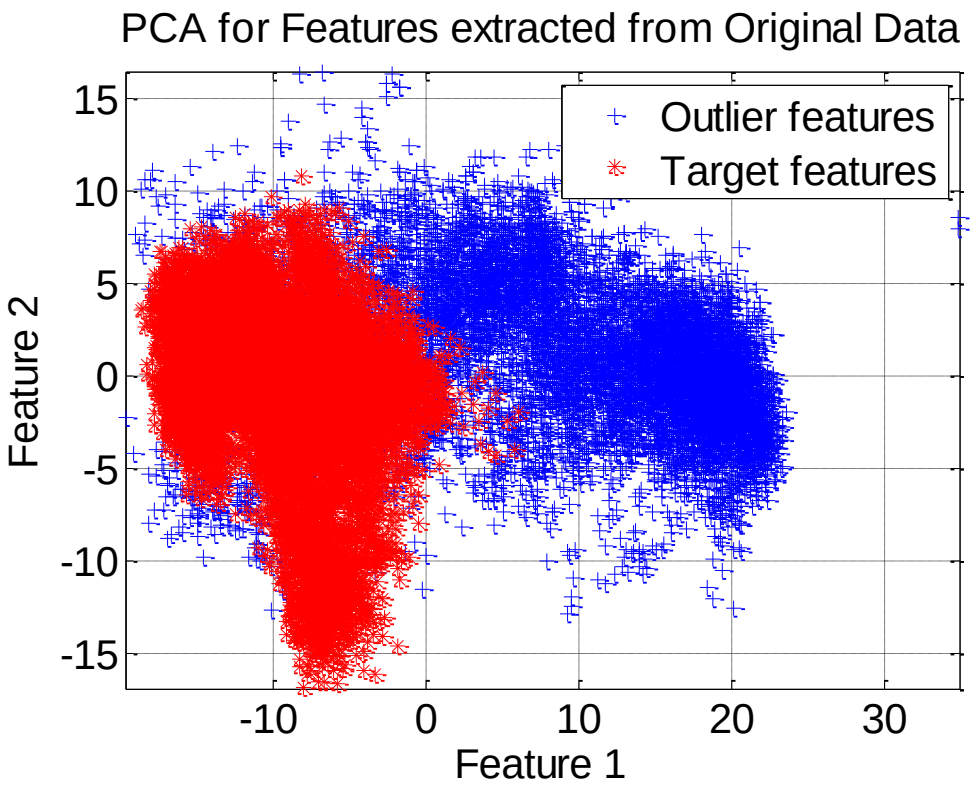

FIGURE D.1: Principal component analysis of features extracted from original audio files.

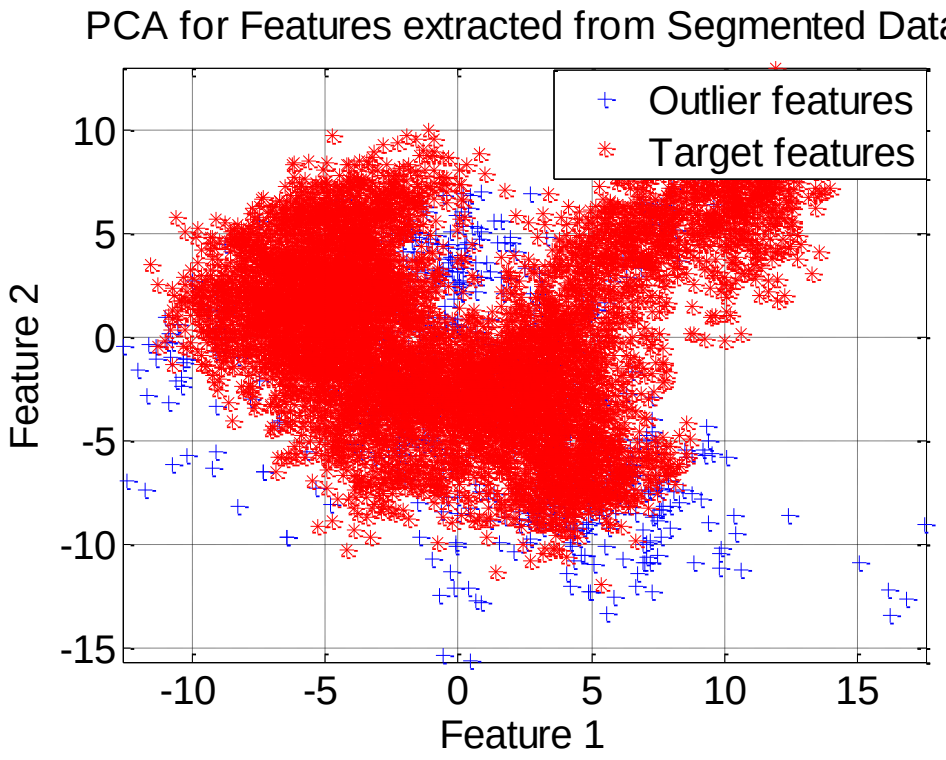

FIGURE D.2: Principal component analysis of features extracted from segmented audio files.

D.1.1 Length of audio files

Total Trains= 200 (30 Freight Trains + 170 passenger Trains)

Total Outliers=45

Sampling Frequency for all the audio files=32000

Average length of trains before segmentation= 8.8887 seconds

Average length of trains After segmentation= 6.0465 seconds

Average length of non-trains before segmentation= 34.7061 seconds

Average length of non-trains After segmentation= 2.6143 seconds

Bibliography

[1] http://www.sensornet.nl, July 2015. About Sensornet.

[2] D.M.J. Tax. https://www.facebook.com/sensornet-290718617733488, december 2015. Facebook cover photo.

[3] DJ Thompson. Noise and vibration from high-speed trains. chapter 1. theory of generation of wheel/rail rolling noise. *Thomas telford publishing, Thomas Telford LTD*, 2001.

[4] Jacqueline McGlade. Indicators tracking transport and environment in the european union. Technical report, European Environment Agency, 2009.

[5] Charlotte Hurtley. *Night noise guidelines for Europe*. WHO Regional Office Europe, 2009.

[6] Leermakers B. Biasin, D. Trans european conventional rail system subsystem. Technical report, European Railway Agency ERA, 2010.

[7] Cherdchai Eamdeelerd and Kraisin Songwatana. Audio noise classification using bark scale features and k-nn technique. In *Communications and Information Technologies, 2008. ISCIT 2008. International Symposium on*, pages 131–134. IEEE, 2008.

[8] J Quartieri, A Troisi, C Guarnaccia, T Lenza, P D'Agostino, S D'Ambrosio, and G Iannone. An italian high speed train noise analysis in an open country environment. In *Proceedings of the 10th WSEAS international conference on Acoustics & music: theory & applications*, pages 92–99. World Scientific and Engineering Academy and Society (WSEAS), 2009.

[9] Tong Zhang and CC Jay Kuo. Hierarchical classification of audio data for archiving and retrieving. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 6, pages 3001–3004. IEEE, 1999.

[10] P Dhanalakshmi, S Palanivel, and Vennila Ramalingam. Classification of audio signals using svm and rbfnn. *Expert systems with applications*, 36(3):6069–6075, 2009.

[11] Dongge Li, Ishwar K Sethi, Nevenka Dimitrova, and Tom McGee. Classification of general audio data for content-based retrieval. *Pattern recognition letters*, 22(5): 533–544, 2001.

[12] MJ David. Tax. one-class classification; concept-learning in the absence of counter-examples. *ASCI dissertation series*, 65, 2001.

[13] Asma Rabaoui, Manuel Davy, Stéphane Rossignol, and Noureddine Ellouze. Using one-class svms and wavelets for audio surveillance. *Information Forensics and Security, IEEE Transactions on*, 3(4):763–775, 2008.

[14] Asma Rabaoui, Hachem Kadri, Zied Lachiri, and Noureddine Ellouze. One-class svms challenges in audio detection and classification applications. *EURASIP Journal on Advances in Signal Processing*, 2008(834973):http–www, 2008.

[15] Keinosuke Fukunaga. Introduction to statistical pattern recognition. 1990.

[16] David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

[17] David MJ Tax and Robert PW Duin. Combining one-class classifiers. In *Multiple Classifier Systems*, pages 299–308. Springer, 2001.

[18] Alexander Ypma, David MJ Tax, and Robert PW Duin. Robust machine fault detection with independent component analysis and support vector data description. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop.*, pages 67–76. IEEE, 1999.

[19] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145, 1995.

[20] RPW Duin, P Juszczak, D de Ridder, P Paclık, E Pezkalska, and DMJ Tax. Prtools. *Pattern Recognition Tools. http://www. prtools. org*, 2004.

[21] D.M.J. Tax. Ddtools, the data description toolbox for matlab, July 2014. version 2.1.1.

[22] V Cherkassky and F Mulier. Learning from data: Concepts, theory, and methods. 1998.