Stony Brook University

Auditory Classification of Vehicles for Scene Understanding

A Thesis Presented

by

Scott Bejan Kaghaz-Garan

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Computer Engineering

Stony Brook University

December 2013

Stony Brook University

The Graduate School

Scott Bejan Kaghaz-Garan

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis.

Dr. Alex Doboli – Thesis Advisor
Associate Professor, Department of Electrical and Computer Engineering

Dr. Sangjin Hong – Second Reader
Associate Professor, Department of Electrical and Computer Engineering

This thesis is accepted by the Graduate School.

Charles Taber
Dean of the Graduate School

Abstract of the Thesis

Auditory Classification of Vehicles for Scene Understanding

by

Scott Bejan Kaghaz-Garan

Master of Science

in

Computer Engineering

Stony Brook University

2013

There have been numerous studies on the classification of auditory signals. In contrast, there have been very few studies in the implementation and classification of vehicles using purely auditory signals. This thesis presents an implementation of auditory vehicle identification using support vector machines. It explores how granular classification can be, from what type of vehicle to what action the vehicle is preforming. The granularity of the classification will greatly aid in auditory scene understanding. The classifications are done with computational complexity in mind, so embedded systems can utilize the findings. A simple averaging algorithm will also be explored that aids in classification significantly.

To My Parents, Family, and Cindy

Contents

List of Figures

List of Tables

Acknowledgements

Firstly, i would like to thank Dr. Alex Doboli for his continuous support and guidance throughout the duration of this thesis. Not only that, but since I was in undergrad his passion for engineering motivated me and I know that I would not be at this point in my academic career without him. It is without a doubt a privilege to have Professor Doboli as my advisor.

I would also like to thank Anurag Umbarkar for all the help he has provided. Anurag has been like a mentor to me, helping me every step of the way throughout this thesis. Utilizing the knowledge he has gained from all the research he has accomplished. He would always be available and helped with any issues I had.

My parents have always supported and believed in me. I would like to thank them for all their love and support. Even though we are over six thousand miles apart, I know they are just a call away. They are always ready to keep me in check and bring the best out of me whenever i need it.

Chapter 1

Introduction

In a world where vision is the predominantly used sense, the other four senses are usually neglected with regard to how much information can be extracted. Sound is a prime example. To demonstrate, one could close their eyes; remove the bombardment of data being processed by the visual cortex, and focus on listening. It will soon become apparent how many sounds are present and they are fairly simple to identify. Not only that, but it is also possible to give an approximation of the location of these sounds. By compiling all the sounds that have been identified with their corresponding locations, a scene can be built in the mind. The scene can now be used to track objects, predict movements, and even build a visualization of the environment: a plethora of information all from subtle fluctuations in air pressure known as sound waves. This process is called auditory scene analysis [1].

When trying to tackle the problem of auditory classification many factors must be addressed: What type of sounds are to be classified? What type of environment will the classification take place? What type of noises will be encountered while classifying? Are all important questions. In this thesis the type of sound that will be classified is vehicle sounds. The environment would most likely be an outdoor area with the possibility of reverberation

from surrounding objects. Finally, the noises that could be encountered would be ambient sounds, other vehicle sounds, and humans to name a few.

The challenge of classifying a vehicle is that, as humans, it is quite difficult to differentiate between them. The best that we can do is perhaps hear the difference between a car and a bus. It does not come naturally to us, but by preforming tests and finding the relationships between them, though signal processing techniques, this issue can be overcome.

Exploiting auditory identification can be useful in various applications such as security systems, smart noise cancellation, robots, and traffic management. It is also less computationally intensive than visual processing. This leads to cheaper implementations and the possibility of utilizing embedded systems. Imagine opening an application on a smart phone that could detect if there was an issue with a car, just by the sound of the engine. Maybe a traffic management system, where classifiers would manage traffic lights and though a network choose the optimal timing for each traffic light, thus reducing traffic. Finally, both systems could be combined creating a system that could predict if a vehicle would breakdown. The system would then promptly respond by changing traffic patterns and alerting drivers of the faulty vehicle, thereby reducing the possibility of an accident and traffic congestion. The opportunities are vast and this thesis hopes to bring them a step closer.

The report is organized as follows: Chapter 1 describes the sound components from a vehicle and the basics of machine learning. Chapter 2 introduces the theory and explanation of support vector machines. Chapter 3 describes feature extraction and the algorithms that are used. Chapter 4 explains how the application was implemented. Chapter 5 reveals the experimental results for classification in various circumstances. Chapter 6 presents the conclusions and future scope for this application.

1.1 Sound Components of a Vehicle

The acoustic signal a vehicle produces can be derived from the various components in use during operation. For example, a car's acoustic signal would consist of the engine, tires, exhaust system, aerodynamic effects, and mechanical effects[1]. The significance of each of these components depends on the assembly, design, speed, and type of parts used[2]. The main components have been generalized in [2] where the following four noise components were described: engine noise, tire noise, exhaust noise, and air turbulence noise. The following will explore the first two.

In an internal combustion engine there is a weak and strong deterministic tone. The weak tone is from the firing rate of any given cylinder, known as the cylinder fire rate which can be defined as $f_0 = x/120$, where x is the revolutions per minute (RPM) of the engine. The strong tone is called the engine fire rate $F_0 = f_0 * p$ where p is the number of cylinders.

In this paper the main component that will be of interest is the engine noise. Tire noise will be addressed and the effect of changing the driving surface will also be explored.

1.2 Basics of Machine Learning

In computing, the ability for a computer to think or make complex decisions is called artificial intelligence (AI). A subfield of AI is machine learning since the concepts and techniques are building blocks that are used in AI. Machine learning can be characterized as the generalization of data, finding unique differences between them, and thus learning properties about them. It can be grouped into two branches: supervised and unsupervised learning.

Supervised learning is when human intuition is used to help with the generalization of

[1] Axle rotation, break pads, and suspension usually have small impact on the overall acoustic signal unless the vehicle has not been maintained correctly.

[2] Some cars have aftermarket parts equipped such as race exhausts/mufflers.

some data. This problem can be either classification or regression. The classification problem consists of two or more separate types of data, or features. These features have already been labeled by a person to indicate what parts of the data are related to each other. The result is the ability to add new data to the model and retrieve or classify what type of data it is consistent with. The regression problem consists of the previous classification definition except that the output is in the form of one or more continuous variables.

Unsupervised learning takes in a set of data without any labeling or target values. This type of technique is used to try and discover groups of similar data, called clustering. Or, to estimate the distribution of data in a given space, called density estimation.

This report will utilize support vector machines, a supervised learning model.

Chapter 2

Support Vector Machines

A support vector machine (SVM) is a supervised machine learning model that is used to recognize patterns in data [3]. The classification is done using an optimization algorithm. This algorithm determines how to split the data into two separate spaces using a line. This chapter will discuss the different cases a SVM can fall under and how training as well as classification of data occurs.

2.1 Linear Case

The linear case is the simplest and most basic form of a SVM. In this case there are two data sets that are completely separable, meaning the two sets can be perfectly divided with a linear line. The line that splits both data sets is referred to as a hyperplane. In figure 2.1 a linear SVM with its hyperplane and data points can be seen. There are many places the hyperplane could be positioned to split both data sets, but a SVM should find the optimal location where the hyperplane will be equidistant from points that are closest to each other that belong to different data sets. The equal space between the hyperplane is called the

margin, and the data points closest to the hyperplane are called the support vectors. In other words a SVM is an optimization problem where we are trying to get the maximum margin between the support vectors.

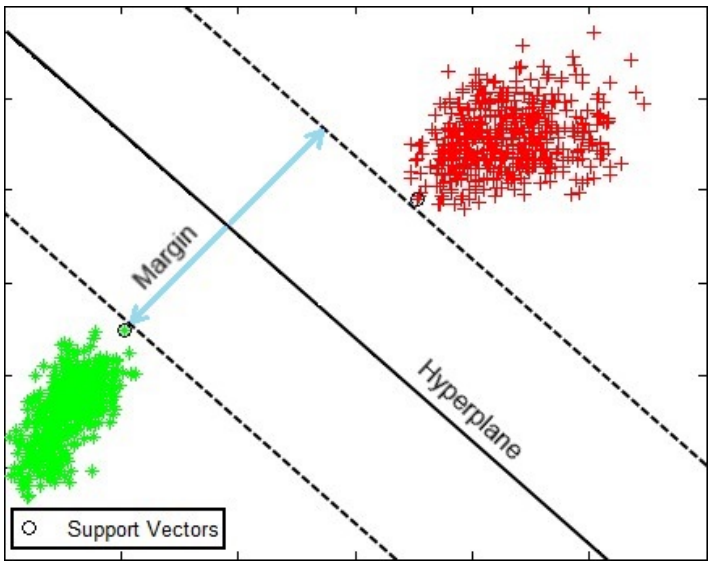


Figure 2.1: A linear, separable, support vector machine

If given a training set T

$$T = \{(x_i, y_i) | x_i \in \mathrm{Re}, y_i \in \{-1, 1\}\} \tag{2.1}$$

we would like to find the set of points x that fulfills

$$w \cdot x - b = 0 \tag{2.2}$$

where w is the vector perpendicular to the hyperplane (normal vector), and b is a constant. Then, the two dotted lines in figure 2.1 that the margin separates would be:

$$w \cdot x - b = 1 \tag{2.3}$$

$$w \cdot x - b = -1 \tag{2.4}$$

Furthermore, it can be established that the + labeled data would be in the area $w \cdot x - b \geq 1$ while the $*$ labeled data would be in $w \cdot x - b \leq -1$. The margin length can be expressed as $\frac{2}{\|w\|}$. Finally, it is possible to change from a maximization to a minimization problem:

$$\min_{w,b} \left(\frac{1}{2} \parallel w \parallel^2 \right) \qquad \text{Subject to: } y_i(w \cdot x_i - b) \geq 1 \tag{2.5}$$

2.2 Non-linear Case and the Kernel Trick

When two data sets are not separable it is possible to move them into a higher dimensional space. Moving into this space may cause the sets of data to become separable. As an example, in figure 2.1 assume the two sets were not separable. Instead of staying in the 2-D space (x,y) we can move to the 3-D space (x,y,z) by using some function to take the features as arguments and output them on the Z plane. These functions are called kernels. Once in the higher dimensional plane, a linear hyperplane may be found and the optimization problem from the previous section can be computed. After the hyperplane is found, the SVM must return back to its original dimensionality. When this occurs the linear hyperplane in the higher dimension may appear non-linear in the original dimension as shown in figure 2.2. This process is called the kernel trick and different kernels will produce varying results. Table 2.1 introduces a few well known kernels.

SVM Kernels	
Linear	$k(x_i, x_j) = x_i \cdot x_j$
Polynomial	$k(x_i, x_j) = (x_i \cdot x_j)^d$
Gaussian radial basis function (RBF)	$k(x_i, x_j) = \exp(-\gamma \parallel x_i - x_j \parallel^2), \ \gamma > 0$

Table 2.1: Common SVM kernels

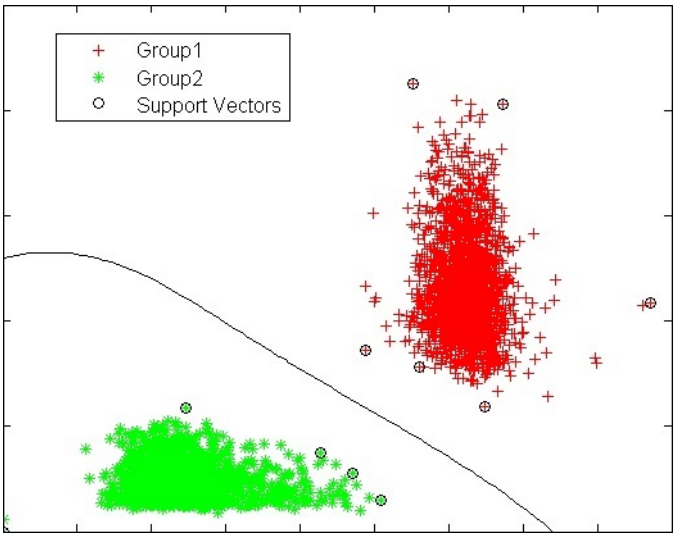


Figure 2.2: A Non-linear, separable, support vector machine

2.3 Non-separable Case

When the kernel trick fails to make two sets of data separable the soft margin method is introduced. In figure 2.3 the non-separable case is depicted. The soft margin method will try to split the data sets in two as best as possible. This is done with respect to maximizing the distance to the nearest fully split data points. To achieve this a new variable called 'slack' is introduced ξ_i. The slack quantifies the degree of misclassification of the data sets.

$$y_i(w \cdot x_i - b) \geq 1 - \xi_i \qquad 1 \leq i \leq n \tag{2.6}$$

The optimization now will become a trade-off between the maximization of the margin and minimization of the slack variables. With a linear penalty function the optimization problem

becomes:

$$\min_{w,\xi,b} \left(\frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{n} \xi_i \right) \tag{2.7}$$

This is subject to equation 2.6 for any $i = 1, \ldots, n$.

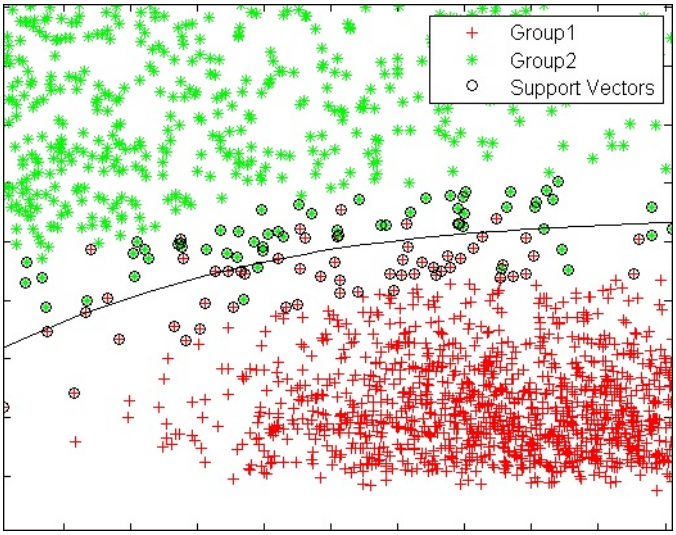


Figure 2.3: Non-separable support vector machine using RBF kernel

2.4 Multi-class

So far the discussion has been about how an SVM can classify two separate data sets. In the real world this is not very practical since there are going to be many different data sets that one may want to classify. To deal with this multi-class problem two methods will be introduced. The first being one versus all and the second being the SVM Tree.

2.4.1 One versus all

The one-versus-all approach groups data into two partitions: a specific group to be classified, and all the other groups combined [4]. In figure 2.4, for every class to be classified $N - 1$

SVMs are required. The strategy for classifying is a ranking system. Each time a SVM classifies the specific group a point is recorded. Input data must enter every SVM and be classified and the SVM with the highest amount of points would be the output classification. The computational time required to classify one data set would be $O(N-1)$.

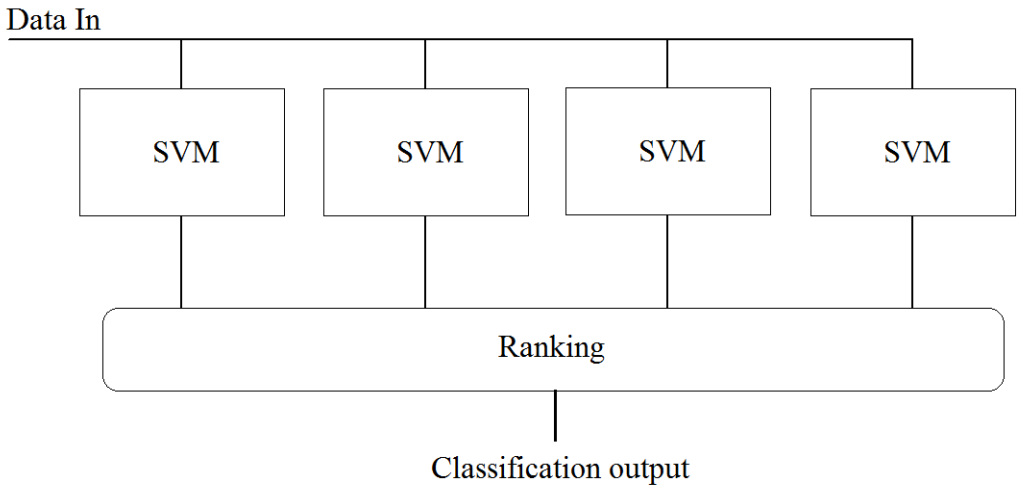


Figure 2.4: SVM Multi-class: One-versus-all structure

2.4.2 SVM Tree

The SVM tree approach groups data in a different manner. Since the basis for this method is a tree structure the data will be placed into hierarchical groups [5]. Each group should be created with data that has some similarities to ensure separability. Referring to figure 2.5, at each level of the tree, the grouping expands outward; this implies the classification is moving from more general to more specific. At the leaves of the tree the classification is complete. This method requires $N-1$ SVMs, one for each node of the tree. The computation time would be $O(\log_2(N-1))$ since at each node we descend to, the problem is being split in half.

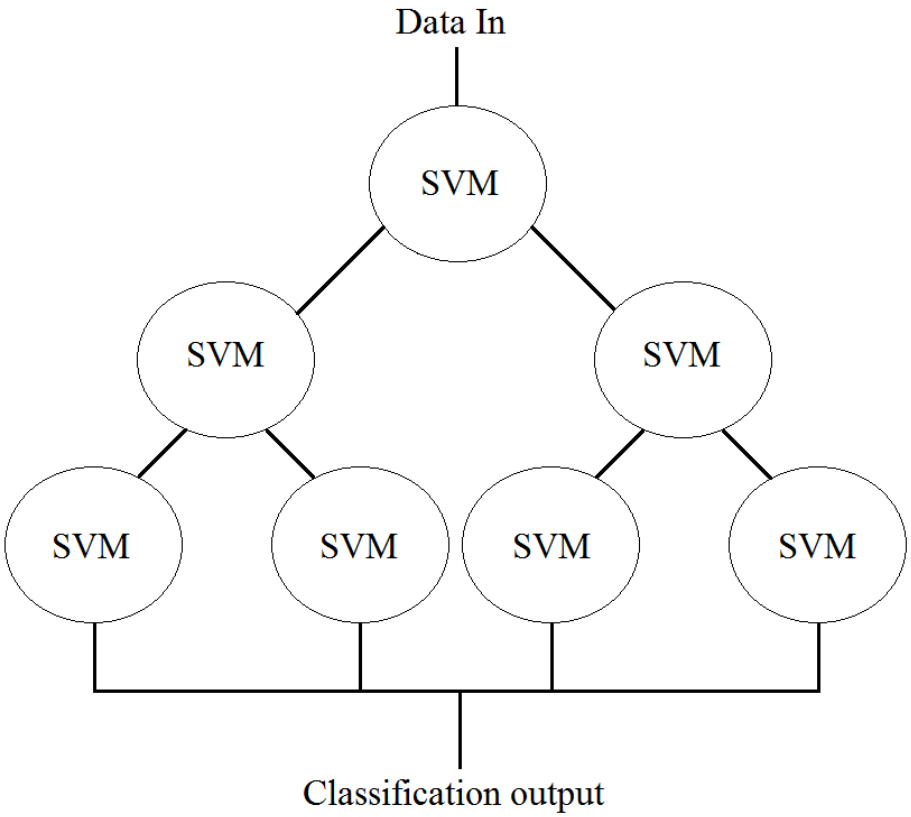


Figure 2.5: SVM Multi-class: Tree structure

Chapter 3

Feature Extraction

Feature extraction is the process of transforming raw auditory data into informative data, or features. There are a number of features to choose from [6–9] but the ones described in this chapter were used due to there relative simplicity. By using computationally simplistic features it is viable for low performance systems to utilize this system, such as embedded devices.

3.1 Time-domain Features

Volume of a signal is derived by taking the RMS of the amplitudes of samples in a small window.

$$RMS = \sqrt{\frac{\sum_{n=1}^{N} x[n]^2}{N}} \tag{3.1}$$

where, x[n] is the magnitude of time sample with index n, N is the total number of samples. Zero crossing refers to the number of sign changes of amplitude in a window.

$$Z_t = 0.5 * \sum_{n=1}^{N} |sign(x[n]) - sign(x[n-1])| \tag{3.2}$$

Short time energy shows a indication of the change in amplitude over time. This is particularly useful when determining a sound source instead of ambient noise.

$$N_j = \sum_{i=1}^{S} x_i^2 \tag{3.3}$$

Energy entropy displays abrupt changes in the amplitude of the audio signal

$$I_j = -\sum_{n=1}^{K} \sigma_i^2 \log_2(\sigma_i^2) \tag{3.4}$$

3.2 Frequency Spectrum Features

Spectral centroid is the center of gravity of magnitude spectrum of Short Time Fourier Transform (STFT). It is a measure of spectral shape.

$$C_t = \frac{\sum_{n=1}^{N} M_t[n] * n}{\sum_{n=1}^{N} M_t[n]} \tag{3.5}$$

Where, Mt[n] is magnitude of Fourier transform at frame t and frequency bin n.

Spectral rolloff is the frequency R_t below which 85% of the magnitude distribution is concentrated.

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n] \tag{3.6}$$

Spectral flux is the squared difference between normalized magnitudes of successive spectral distributions. It is a measure of the amount of local spectral change.

$$F_t = \sum_{n=1}^{N} (N_t[n] - N_{t-1}[n])^2 \tag{3.7}$$

Compactness is the measure of how dense, or compact the audio signal is.

$$C_i = \sum_{k=2}^{K} |\frac{20}{3} * \log(N_i[k]) - \frac{20}{3} * \log(N_i[k-1] * N_i[k] * N_i[k+1])| \tag{3.8}$$

Chapter 4

Implementation

The system's implementation can be split into three parts: the sound library, the feature extraction software, and the SVM classifier. The sound library used in SVM training and classifying was recorded at 44.1 kHz using an IPhone 4. Both the feature extraction and SVM software have been developed using Matlab script.

4.1 Sound Library

In Table 4.1, the entire contents of the sound library can be seen. For each vehicle there were multiple sound samples taken to create uniformity. Every sound source has had no post-recording processing done, but some sources have been cropped to begin at a classification event.

Sound Library	
Cars	Volkswagen GLI 2008 Scion Tc 2007
Buses	Stony Brook University Bus New York City Bus (hybrid and regular)
Big Rig	Car Hauler
Trains	Long Island Rail Road (LIRR) New York City Subway (Metro) CSX Cargo

Table 4.1: Contents of the sound library

4.2 Feature Extraction Implementation

Feature extraction is a preprocessing step in both training and classification of the SVM subsystem. In chapter 3, eight features were introduced that will be used for audio fingerprinting.

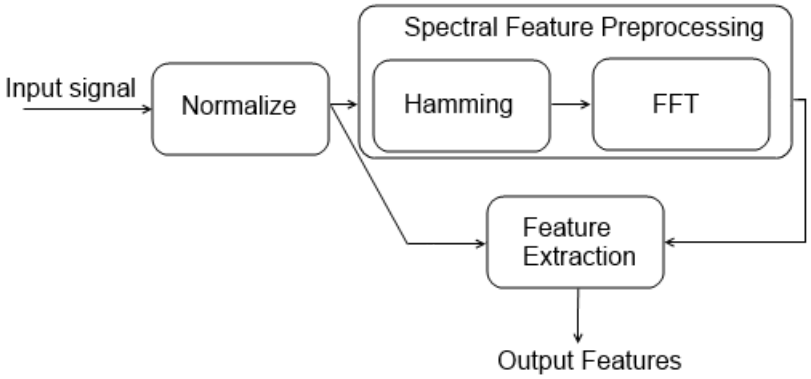


Figure 4.1: Feature extraction overview

In figure 4.1, a depiction of how feature extraction was implemented is shown. We start with a raw input data that gets normalized. Normalizing the data helps ensure that the data is not skewed due to scaling. At this point all time-domain features are ready for processing. Frequency Spectrum features must go though additional preprocessing. The first being the hamming window function. This function is used to help filter frequencies

that are not of interest such as ambient noise. Finally, the signal is ready to enter the fast Fourier transform (FFT). This transformation moves the data from the time-domain into the frequency domain.

4.3 SVM Implementation

The SVM used for implementation is the SVM tree discussed in section 2.4.2. There are a total of eight SVM classifiers used, one for each node of the tree[1]. The tree structure that is used can be seen in figure 4.2.

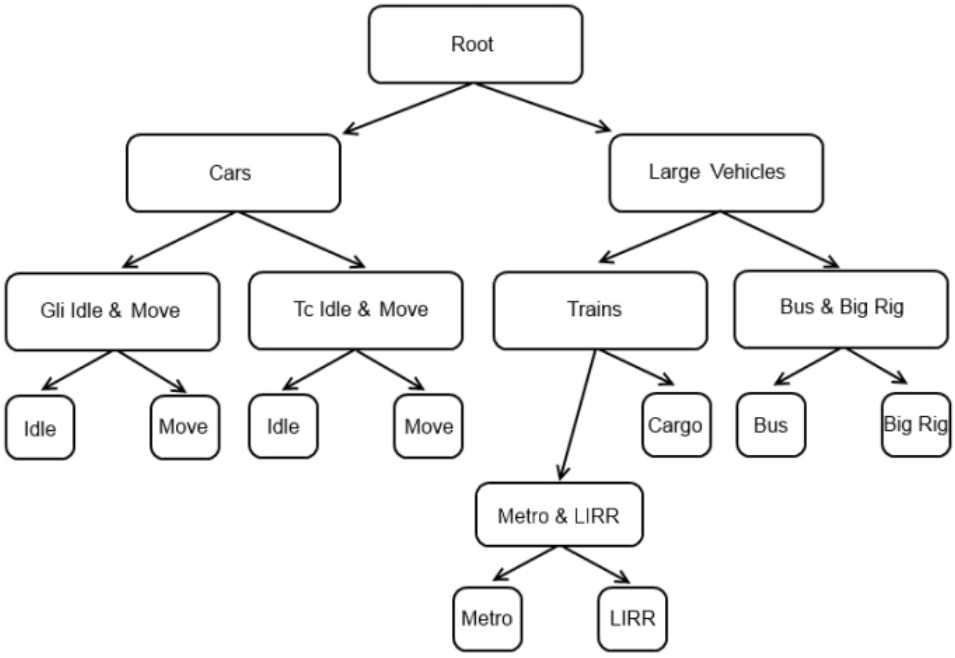


Figure 4.2: SVM tree implementation

[1]The leaves are not included because they are not classifiers, they are the result of the entire classification

4.3.1 Choosing the Optimal Features

An optimal feature set is one where both classes are completely separable and corresponding hyperplane is linear. The features chosen for each node of the SVM tree were carefully picked by comparing all eight feature sets and calculating which ones were most separable. In the feature extraction software described in the previous section an additional function was created to show these results visually and arithmetically. This function can be seen in figure 4.3.

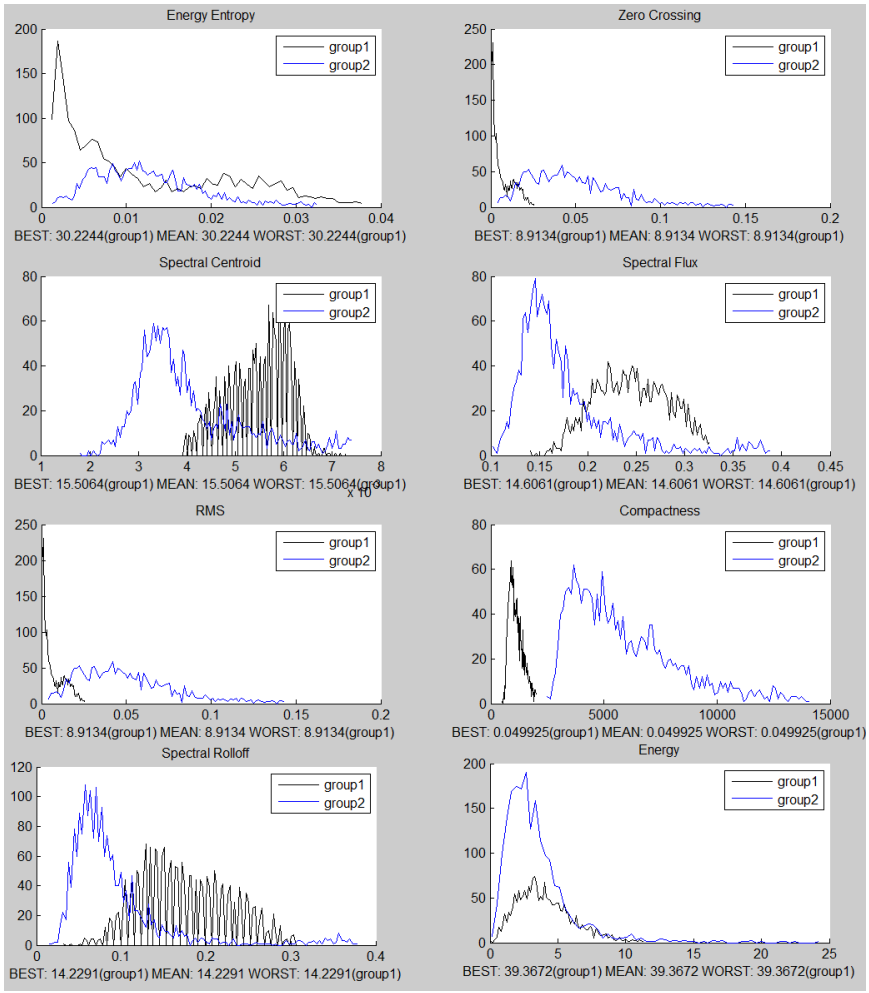


Figure 4.3: All possible features for root node of the SVM tree

The figure is comprised of all eight features discussed in chapter 3. Each plot is a density map of the data, where the x-axis is the number of data points and the y-axis is the amplitude of the data points. Inspecting the figure shows that the compactness and zero crossing feature sets have the lowest percent overlap in their data. The root node SVM can now be constructed using these two feature sets. This process continues until all the nodes in the SVM tree have been built with there optimal feature sets. In table 4.2, all the nodes and optimum features in the SVM tree are transcribed.

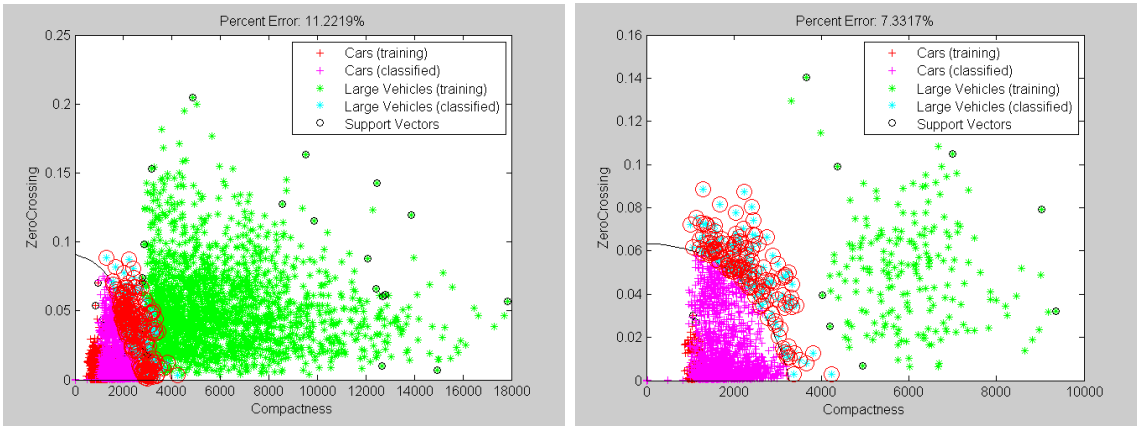
SVM Tree Features			
Name	Node #	Feature 1	Feature 2
Root	1	Zero Crossing	Compactness
Cars	2	Zero Crossing	Compactness
Large Vehicles	3	Energy Entropy	Compactness
GLI Idle & Move	4	Energy Entropy	Spectral Centroid
Tc Idle & Move	5	Energy Entropy	RMS
Trains	6	Spectral Flux	Zero Crossing
Bus & Big Rig	7	Compactness	Spectral Centroid
Metro & LIRR	8	Spectral Flux	Spectral Centroid

Table 4.2: SVM tree features sets

4.3.2 COM: The Center of Mass of Data Distribution

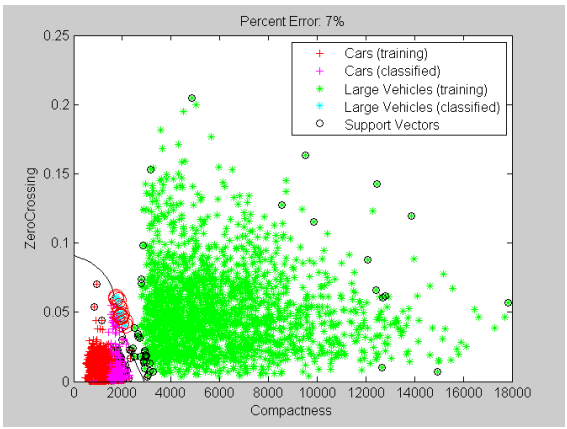
In the natural world noise is commonplace. A bird chirping, the wind through the leaves of a tree, or perhaps the roar of a jackhammer in the distance. All this noise affects the SVM's ability to classify. Fortunately, some noise occurs in small intervals and at low intensities and can be viewed as isolated points in the data. To counteract these points a technique is used to find the center of mass of the data distribution (COM).

COM is a very simple technique where every ten points in a data set are averaged together and the results are used in place of the original data. This will cause the data set to become more uniform overall and mask some of the natural noise recorded. In figure 4.4, the effects of COM can be viewed visually. The observations that can be derived from the figure is

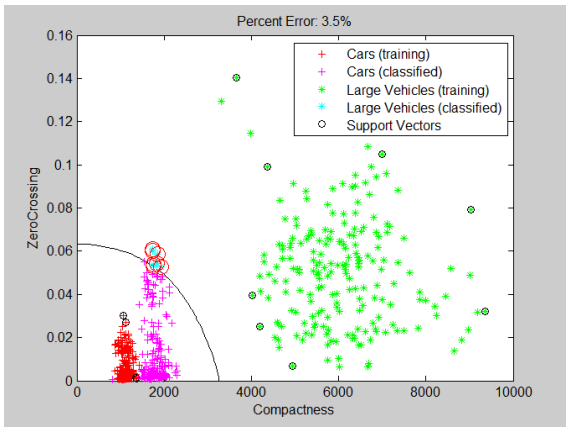


(a) Normal classification

(b) COM of training data



(c) COM of testing data



(d) COM of both

Figure 4.4: Change in data distribution using COM

that when COM is introduced the data is shifted away from the hyperplane. In (a) it can be seen that there is a lot of overlap in the classifications and training data. In (d) the data has been shifted away from each other resulting in much higher accuracy.

Chapter 5

Experiments

In this chapter we will discuss the experiments conducted on the implementation described in the previous chapter. The experiments have been divided into two groups. One being the cars and the other the large vehicles. This grouping was done because of the reliability of the training data. When gathering sound samples in the field it became clear that some vehicles could not be recorded without the presence of large background noises. These background noises are particularly true for the large vehicles, they were recorded in New York City. The sound samples for the cars were recorded in a suburban area where there was much less background activity. Since the cars have the most reliable samples we will shift most of the experimental focus towards them.

5.1 Car Experiments

Three scenarios were used to conduct experiments on the cars:

- (Pass) Car passing on paved road at about 15 miles per hour
- (Gravel) Car passing on gravel road at about 15 miles per hour

- (Idle) Car in the stopped idle state

The first scenario depicts normal driving conditions. All samples were taken on an average paved roadway in dry conditions. Some small obstructions, such as pebbles, were present. The goal for this scenario is to see if when a small noise from tires, pavement, and obstructions are present is it possible to classify between two cars.

The second scenario shows an abnormal driving condition. All samples were taken on a gravel road in dry conditions. The goal for this scenario is to see if when a large noise from tires and gravel is present is it possible to classify between two cars.

Finally, the third scenario is an ideal stopped condition. All samples were taken in an open area in dry conditions. The goal for this scenario is to see if it is possible to classify between two cars only in there idle state, with no other vehicle noise.

The use of COM in classification will also be tested on each of these scenarios. They will be split into these four tests:

- Normal implementation
- COM introduced on training data
- COM introduced on testing data
- COM utilized on both training and testing data.

5.1.1 Volkswagen GLI Results

In this section the results of the Volkswagen GLI experiments are provided. Each table is constructed with the top row informing at what level of the tree the error exists and the first column being the tests.[1] At each node data is tested. If the points of data at the current node are not classified correctly they are removed from the set as not to skew classification further down the SVM tree. The total error is calculated by the following equation:

$$TotalError = \frac{|P - R|}{P} * 100\% \tag{5.1}$$

Where P is the total number of testing points and R is amount of points removed after classification.

Error Percentage of GLI Testing					
Test	Root	Node 2	Node 4	Total Error	Average Error
Pass 1	4.83	19.2	9.99	30.82	43
Pass 2	5.32	48.21	9.64	55.6	
Pass 3	6.23	34.55	6.41	42.57	
Gravel 1	59.4	0.73	9.8	63.7	58.71
Gravel 2	24.47	19.53	1.722	40.27	
Gravel 3	68.7	1.91	9.283	72.15	
Idle 1	4.05	0.052	21.54	24.76	21.33
Idle 2	10.1	0.333	0.837	17.9	

Table 5.1: Results from GLI identification tests

[1] Refer to table 4.2 for node names and features used.

Error Percentage of GLI Testing					
Test	Root	Node 2	Node 4	Total Error	Average Error
Pass 1	2.19	24.47	1.89	27.53	40.14
Pass 2	0.049	50.15	6.09	53.21	
Pass 3	0.59	37.76	2.49	39.68	
Gravel 1	7.88	0.1	1.03	8.936	16.14
Gravel 2	1.14	19.21	0.56	20.58	
Gravel 3	16.9	0.6	1.8	18.9	
Idle 1	3.7	0.31	36.32	38.86	30.48
Idle 2	11.2	0.5	11.8	22.1	

Table 5.2: Results from COM of training data in GLI identification tests

Error Percentage of GLI Testing					
Test	Root	Node 2	Node 4	Total Error	Average Error
Pass 1	0	0.5	6	6.5	13.33
Pass 2	0.5	21.5	4.08	26	
Pass 3	0.5	4	3.01	7.5	
Gravel 1	80	0.54	3.8	84	62.50
Gravel 2	8.5	0.5	0.5	9.5	
Gravel 3	90	1.09	3.31	94	
Idle 1	0	0	8.54	8.54	4.27
Idle 2	0	0	0	0	

Table 5.3: Results from COM of testing data in GLI identification tests

Examining figure 5.1, one can see that using the COM technique on the test data produces the best results[2]. Comparing table 5.1 and 5.4, idling showed an 80 % decrease, while passing produced a 69 % decrease in erroneous classifications.

[2]There are no gravel results depicted table 5.1. They were removed based on results from the next section.

Error Percentage of GLI Testing					
Test	Root	Node 2	Node 4	Total Error	Average Error
Pass 1	0	0.5	0	0.5	14.67
Pass 2	0	33	2.5	35.5	
Pass 3	0	8.5	0	8.5	
Gravel 1	0	0	1	1	3.83
Gravel 2	0	0.5	0.5	1	
Gravel 3	8.5	0	1	9.5	
Idle 1	0	0	20.6	20.6	10.55
Idle 2	0.5	0	0	0.5	

Table 5.4: Results from full COM in GLI identification tests

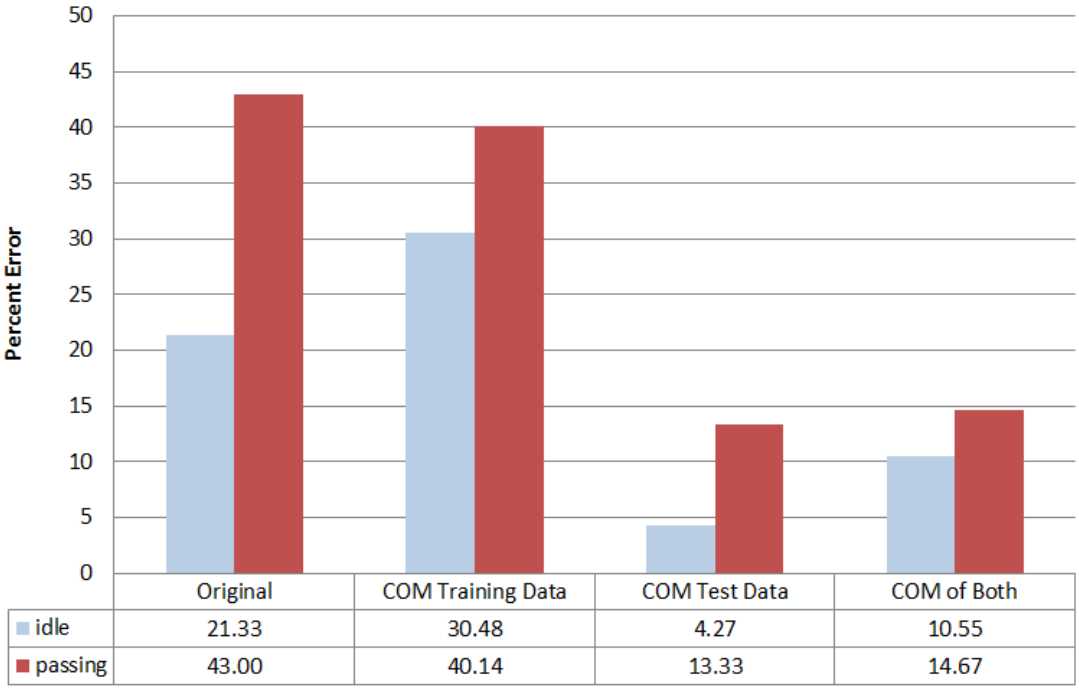


Figure 5.1: Effect of COM on GLI tests

5.1.2 Scion TC Results

This section provides the Scion TC results. As with the previous section, below are the four tables of the experimental data.[3]

Error Percentage of Scion TC Testing					
Test	Root	Node 2	Node 5	Total Error	Average Error
Pass 1	0.049	19.77	0	19.81	28.98
Pass 2	0.099	19.95	0	20.02	
Pass 3	0.099	25.13	29.31	47.13	
Gravel 1	49.45	97.63	0	98.8	98.63
Gravel 2	36.58	97.7	0	98.55	
Gravel 3	37.26	99.2	0	98.55	
Idle 1	0	15.2	0.05	15.28	18.19
Idle 2	0	21.1	0	21.1	

Table 5.5: Results from Scion TC identification tests

Error Percentage of Scion TC Testing					
Test	Root	Node 2	Node 5	Total Error	Average Error
Pass 1	0	1	0	1	12.99
Pass 2	0	1.49	0	1.49	
Pass 3	0	7	29.64	36.5	
Gravel 1	56	100	-	100	100
Gravel 2	34	100	-	100	
Gravel 3	41.7	100	-	100	
Idle 1	0	0	0	0	1
Idle 2	0	2	0	2	

Table 5.6: Results from COM of test data in Scion TC identification tests

[3]Review table 4.2 to understand what features were used and what each node represents.

Error Percentage of Scion TC Testing					
Test	Root	Node 2	Node 5	Total Error	Average Error
Pass 1	0	22.3	0	22.3	24.13
Pass 2	0.049	26	0	26.04	
Pass 3	0.847	21.7	2.81	24.06	
Gravel 1	1.29	97.3	0	97.3	97.34
Gravel 2	2.39	96.36	2.81	96.5	
Gravel 3	6.8	98.12	0	98.249	
Idle 1	5.87	15.18	2.3	22.01	20.705
Idle 2	1.15	18.3	0.18	19.4	

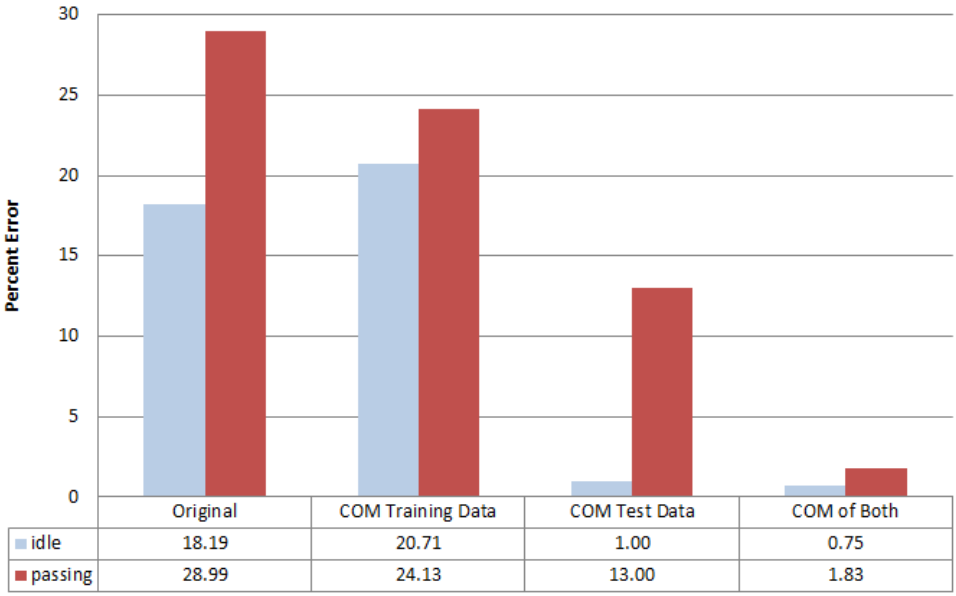
Table 5.7: Results from COM of training data in Scion TC identification tests

Error Percentage of Scion TC Testing					
Test	Root	Node 2	Node 5	Total Error	Average Error
Pass 1	0	0.5	0	0.5	1.83
Pass 2	0	0.497	0	0.497	
Pass 3	0	4.5	0	4.5	
Gravel 1	0	99.5	100	100	100
Gravel 2	0	99.5	100	100	
Gravel 3	2.5	99.4	100	100	
Idle 1	0.5	0.5	0	1	0.75
Idle 2	0	0.5	0	0.5	

Table 5.8: Results from full COM in Scion TC identification tests

The data in tables 5.5, 5.6, 5.7, and 5.8 show a huge failure in the system, the gravel tests, boasting error rates of near or at 100 %. This is completely different than the GLI gravel tests where the error rates fluctuated drastically. For this reason the gravel tests will be removed from figure 5.2.[4]

[4]A discussion on why the erroneous gravel results occurred is located in section 5.3.



	Original	COM Training Data	COM Test Data	COM of Both
idle	18.19	20.71	1.00	0.75
passing	28.99	24.13	13.00	1.83

Figure 5.2: Effect of COM on Scion TC tests

Examining figure 5.2, it can be seen that unlike the GLI results it is better to COM both the training and incoming data. Both idle and passing classifications have remarkably good error rates, 0.75 % and 1.83 % respectively. By comparing table 5.5 and table 5.6, there is a 96 % increase for idling and 93.6 % increase in classification accuracy for passing.

5.2 Large Vehicle Results

As stated in the beginning of the chapter, the experimental data for each of the large vehicles are highly skewed because of large background noises. Since in practice the system would be used in many environments, this is a good way to show how noise polluted areas would interfere with classification.

Viewing table 5.9, the data shows that the total error is high for large vehicle classifications. As the classification goes on deeper into the SVM tree, the error rate grows. That is

Error Percentage of Large Vehicle Testing						
Test	Root	Node 3	Node 6	Node 7	Node 8	Total Error
Bus	1	8.58	-	16.02	-	24
Big Rig	0	36.5	-	18.89	-	48.5
CSX	0	4.5	12.04	-	-	16
Metro	0	0	100	-	-	76
LIRR	0	19	72.5	-	12.72	100

Table 5.9: Results for full COM in large vehicle identification tests

to say as the classification becomes more specific, the error rate increases. Although that is true, the data also suggests that even with the presence of large noise it is still possible to classify each of these vehicles as large vehicles with very high precision. Averaging the error at the root node for each test yields an average of 0.2 % error rate.

The data also indicates that, under high noise, classification can be done on trains with moderate precision; the average error rate for trains is 7.83 % while the bus and big rig did poorer at 22.54 %. Continuing deeper into the classification would result in higher error leaving the results unusable, but having the ability to classify in this detail under the environmental noise conditions is quite impressive.

5.3 Effects of Noise Induced by Reverberation

In the previous sections the experimental data indicated the presence of noise leading to high error rates. The COM technique helped immensely in most cases. The worst case was the gravel tests where error rates were near 100 %. The following is a possible explanation of why this occurred.

In the passing test, the noise created by movement was restricted to the contact of the tires to the ground and the movement of the car though the air. At low speeds these noises are subtle at best and should only slightly skew classification. Now when the paved road

is replaced with gravel, the amount of noise that is created is increased immensely. The new noise becomes high enough it starts to drown out the sounds of the engine making classification extremely difficult.

To demonstrate this effect, another experiment was conducted. Each car was placed inside a garage and was recorded idling. The effect of reverberation inside the garage will distort the recorded engine sound resulting in miss-classification, even though they were classified with low error previously.

Error Percentage of Cars Idling with Reverb					
Test	Root	Node 2	Node 4	Node 5	Total Error
GLI idle	100	-	-	-	100
Tc idle	100	-	-	-	100

Table 5.10: Results from full COM of cars idling in garage

The data in table 5.10 confirms the idea that extraneous sounds are drowning out the sound signature of the each of the cars. Trying to classify either of the cars resulted in 100 % error. To ensure that reverberation was the cause, one last test was needed. While idling in the garage, the car's engines were revved quickly. The idea behind this is that making the engine louder quickly should disrupt or overpower the reverberation process and give a clearer signal to the classifier.

Error Percentage of Cars Revving with Reverb					
Test	Root	Node 2	Node 4	Node 5	Total Error
GLI idle rev	12	0	100	-	100
Tc idle rev	4	100	-	-	100

Table 5.11: Results from full COM of cars revving in garage

Table 5.11 shows that the prediction was true; at the root node both cars were able to be classified. The car classifier, node 2, was not able to classify properly. A possible reason for this is that the sound of revving an engine was never added to the sound library.

Chapter 6

Summary

6.1 Related Work

There have been many studies on sound-based classification [6, 8–10] and others using specifically SVMs [11–13]. However, there have been far fewer focused on light-weight implementation and classification of vehicles. The importance of this comes from the need to achieve low cost solutions so the utilization of this technology can become mainstream in practical applications.

Guo et al. [11] described an implementation similar to this report. By using a SVM tree and a set of feature extraction techniques, classification of 16 sounds were tested. Various approaches and feature sets were used during classification, the best producing an error rate of 11 %.

Portelo et al. [12] investigated the one-versus-all SVM and Hidden Markov Models (HMMs); classification of 15 sounds were tested. The feature set consisted of either perceptual linear predictive (PLP) or Mel-frequency cepstral coefficients (MFCC), both well known features for speech recognition. Various blockbuster movies were used as testing

input.

Theodoros et al. [13] contributed with an implementation focused on violence content in sound. The training set of the SVM consisted of sounds of shooting a gun, fighting, yelling, etc. By using eight audio features in one SVM, it was possible to achieve an accuracy rate of 85.5 %

6.2 Future Scope

In this paper, only two feature sets were used per SVM. By increasing this number it should be possible to obtain higher accuracy rates as described in [13]. Also, other features not introduced in chapter 3 could be explored. The seven classes used in this work is quite small for complete vehicle classification, adding additional vehicle sounds to the sound library would increase the amount of information that could be extracted. Implementing ways of mitigating noise from signals and the ability to classify multiple sounds from a single sound sample would greatly increase accuracy and practicality.

To create full auditory scene understanding, sound localization must be used. A compilation of this work with the sound localization system Umbarkar [14] has implemented, with a PSoC micro-controller, would create a full functioning auditory scene analyzer. Finally, if the system were also combined with Rajagopal's clustering algorithms [15] for traffic scene understanding, the result would be a sound-based traffic analyzer. The device would be able to make smart decisions on when to change traffic lights and perhaps move traffic patterns away from accidents.

6.3 Conclusions

This thesis provides an implementation of auditory scene classification using support vector machines. The report gives a brief overview of how SVMs work as described in literature. The ability of splitting two data sets by solving an optimization problem is explained. Then, a description of the signal processing techniques that are used in feature extraction is provided. Next, is the systems implementation; where the sound library, feature extractor, and SVMs are described and how each were implemented. Finally, the report explains the experimental results and the conclusions that can be derived from them.

Overall, the implementation's accuracy was quite high, even though the large vehicles and gravel tests had high error rates. The reasons behind their erroneous behavior are clear and can be mitigated in the future. The use of COM had an enormous effect on all classifications, resulting in an average of 84.65 % increase in accuracy for car classifications. Having a technique so simple yet powerful has massive advantages. Eventually, this implementation will be able to achieve scene understanding

Bibliography

[1] Albert S. Bregman. Auditory scene analysis. *MIT PRESS*, 1990.

[2] V. Cevher, R. Chellappa, and J.H. McClellan. Joint acoustic-video fingerprinting of vehicles, part i. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–745–II–748, 2007. doi: 10.1109/ICASSP.2007.366343.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning 20*, pages 273–297, 1995.

[4] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research 5*, pages 101–141, 2004.

[5] B. Fei and Jinbai Liu. Binary tree of svm: a new fast multiclass training and classification algorithm. *Neural Networks, IEEE Transactions on*, 17(3):696–704, 2006. ISSN 1045-9227. doi: 10.1109/TNN.2006.872343.

[6] Nevenka Dimitrovac. Dongge Lia, Ishwar K. Sethi and McGeec Tom. Classifcation of general audio data for content-based retrieval. *Recognition Letters 22*, 2001.

[7] Ingo Mierswa and Katharina Morik. Automatic feature extraction for classifying audio data. *Machine Learning Journal*, 58:127–149, 2005.

[8] Fabian Mrchen, Alfred Ultsch, Michael Thies, Ingo Lhken, Mario Ncker, Christian Stamm, Niko Efthymiou, and Martin Kmmerer. Musicminer: Visualizing timbre distances of music as topograpical maps. Technical report, 2005.

[9] George Tzanetakis, Georg Essl, and Perry Cook. Automatic musical genre classification of audio signals. In *IEEE Transactions on Speech and Audio Processing*, pages 293–302, 2002.

[10] C. Canton-Ferrer, T. Butko, C. Segura, X. Giro, C. Nadeu, J. Hernando, and J.R. Casas. Audiovisual event detection towards scene understanding. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 81–88, 2009. doi: 10.1109/CVPRW.2009.5204264.

[11] G. Guo and S.Z. Li. Content-based audio classification and retrieval by support vector machines. *Neural Networks, IEEE Transactions on*, 14(1):209–215, 2003. ISSN 1045-9227. doi: 10.1109/TNN.2002.806626.

[12] J. Portelo, M. Bugalho, I. Trancoso, J. Neto, A. Abad, and A. Serralheiro. Non-speech audio event detection. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1973–1976, 2009. doi: 10.1109/ICASSP.2009.4959998.

[13] G. Theodoros, K. Dimitrios, A. Andreas, and T. Sergios. Violence content classification using audio features. *4th Helenic Conference on AI, SETN 2006*, pages 502–507, 2006.

[14] Anurag Umbarkar. Improved sound-based localization through a network of reconfigurable mixed-signal nodes. M.s. thesis, Stony Brook University, 2010.

[15] Shreyas K. Rajagopal. Traffic scene understanding using sound-based localization, svm classification and clustering. M.s. thesis, Stony Brook University, 2010.