Horn And Whistle Recognition Techniques For NAO Robots

Niels W. Backer
6082025
niels.backer@student.uva.nl

Bachelor thesis
Credits: 6 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
dr. A. Visser

Institute of Informatics
Faculty of Science (FNWI)
University of Amsterdam
Science Park 904
1098 XH Amsterdam

June 27th, 2014

Contents

Abstract

The efficiency and accuracy of several state-of-the-art algorithms for real-time sound classification on a NAO robot are evaluated, to determine how accurate they are at distinguishing predefined and whistle sounds in both optimal conditions, and a noisy environment. Each approach uses a distinct combination of an audio analysis method and a machine learning algorithm, to recognize audio signals captured by NAO's 4 microphones. A technical description of the audio analysis methods, Fast Fourier Transform, Discrete Wavelet Transform, and Mel Frequency Cepstral Coefficients, is provided, as well as a short summary and optimization details of the Logistic Regression, Stochastic Gradient Descent, Support Vector Machine, and AdaBoost-SAMME classifiers. Experimental results show that for each of the acquired data sets, there are multiple high-accuracy system proposals available. Since accuracy measures and F1-scores were found to be similarly high for each of the tested system proposals, no definitive determination can be made as to which one performed best on the collected data sets.

1 Introduction

Although a large portion of AI and robotics research is focused mostly on subjects such as performing high-level cognitive tasks, path planning, and robot localization, one of the most important aspects of natural intelligence is its ability to react to trigger-events in the environment. Fluent interaction with environmental signals is not only a requirement for effective Human-Robot Interaction (HRI), but also an important component for systems that have real-world applications. For these reasons, a sound recognition challenge was issued this year by the RoboCup Standard Platform League (SPL) technical committee, which organizes the yearly robot football world cup. The goal of this challenge is to make the humanoid NAO robots used in the SPL recognize audio signals, more specifically predefined start/stop signals and whistle signals, emitted from the sideline of the pitch.

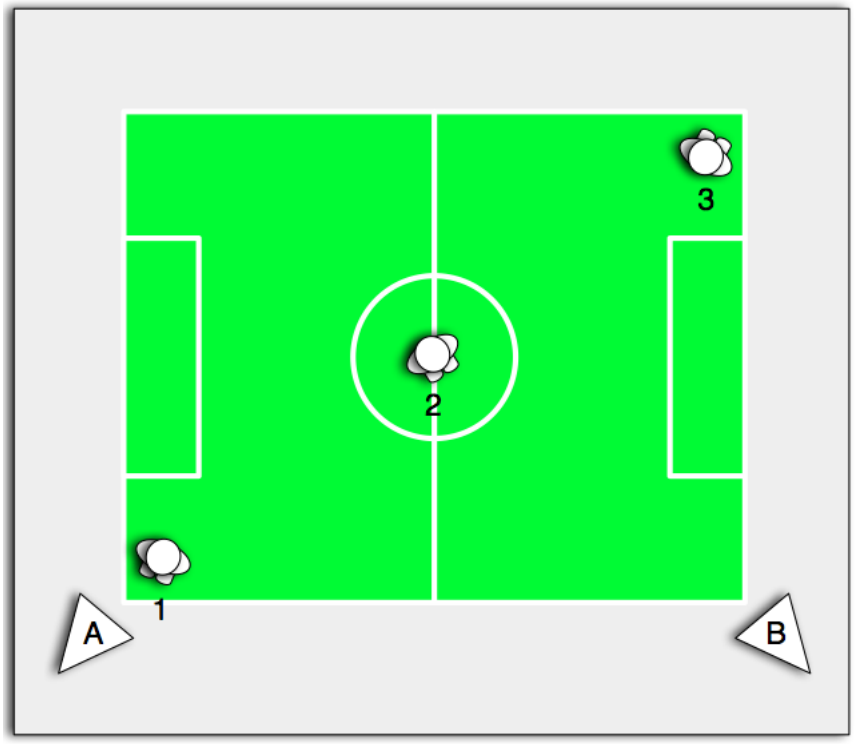

Figure 1: The NAO audio recognition challenge setup. Predefined signals will be emitted in alternating sequences from speakers A and B.

Since the RoboCup is a very noisy event, and NAO's microphones are quite small and of mediocre quality, this challenge is not very straightforward. Because audio classification has been plagued by noise in the past, this challenge calls for a qualitative assessment of a range of techniques applied to signal recognition.

Most digital audio signal processing techniques have a similar method structure. Firstly, the audio signal is converted from analog to digital, after which signal processing techniques are applied to convert the signal to a suitable representation for the task in hand. The most commonly used method for spectral analysis is the Fourier Transform, more specifically the Discrete Fourier Transform (DFT) and Fast Fourier

Transform (FFT) for discrete numerical analysis. FFT is very computationally efficient, and as such has gained popularity for a wide range of applications in the fields of mathematics, music production, and engineering.

Although the FFT certainly has its advantages pertaining to the analysis of a stationary signal, a large amount of the audio that would be useful to classify is time-bound; its spectral properties change over time, conveying information in sequences, like natural language. Since the Fourier Transform does not contain temporal information, but rather the power spectrum of frequencies over a set period of time, alternative methods have to be used for applications that aim to tackle problems such as natural language recognition. Temporal sequence information is not a necessary feature for the classification of stationary signals like a whistle, yet it is needed for many real-world problems, so these factors have to be taken into account when picking audio analysis methods.

One of these alternative methods is the Discrete Wavelet Transform (DWT), which extracts a wavelet power spectrum. Similar to the FFT, DWT can be used to extract frequencies from a signal, but has a distinct advantage for analyzing temporal sequences: It captures both frequency and location (in time) information. This feature might prove to be very useful for the classification of audio sequences. Moreover, DWT has a time complexity advantage. The FFT has a complexity of $O(Nlog(N))$, whereas DWT has a complexity of $O(N)$, which makes it a very effective method for image and audio compression, with acceptable accuracy.

The last audio analysis technique covered in this thesis is Mel Frequency Cepstral Coefficients (MFCC). Audio analysis using the Cepstral representation of the short-term power spectrum can be used to identify pitch and formant frequencies in speech, and has in fact become a very conventional way to process human speech before recognition. Its representation scale has equally spaced frequency bands, which approximates the human auditory scale more closely than the FFT representation.

The second phase is the classification (or recognition) step. While a large number of applications for classifying simple stationary signals have used a frequency mask or single-layer perceptron, the aim of this thesis is to provide a performance comparison of machine learning algorithms as well. The optimal outcome would be to find a configuration which is impervious to noise, which is a considerable obstacle for common classification methods. In order to combat the effect of noise on audio recognition, we will inspect classification algorithms, some well-established, some quite new, such as logistic regression, stochastic gradient descent, support vector machines, and AdaBoost-SAMME.

The thesis is organized as follows: The following section describes related work. In the section thereafter an overview of the FFT, DWT and MFCC is given. Section 3.3 will discuss the machine learning techniques, after which research methods will be discussed, followed by results, a discussion, and notes on future work.

2 Related Research

Sound recognition in robots is one of the major steps in creating autonomous systems that can directly interact with humankind. According to the Robocup Standard Platform League challenges set out for 2014 (RoboCup Technical Committee, 2014), there is a number of reasons to eliminate WLAN communication within RoboCup soccer matches, with WLAN not being reliable nor very human-like being the most important causes.

There have been previous attempts at tackling the recognition of whistles and predefined signals (Bonarini, Lavatelli, & Matteucci, 2005), also considering noisy environments (Lopes, Ribeiro, & Carvalho, 2010). These systems have been proven to be reliable, though they only consider one possible system architecture, using common methods. Both solutions utilize the Fast Fourier Transform (FFT), a frequency mask, and a single-layer perceptron. However, a perceptron without hidden layers can only be trained to recognize a small set of patterns, since its architecture only allows for it to discern instances in models with linearly separable patterns. Attempts at phoneme/speech recognition have successfully used neural networks as an alternative to simple perceptrons (Graves, rahman Mohamed, & Hinton, 2013).

A modified version of the FFT, the Short-term Fourier Transform (STFT), can be used to achieve single-microphone audio source localization with a Hidden Markov Model (Saxena & Ng, 2009). This approach models the Fourier Transform as a function of the incident angle of the audio source to the test setup, consisting of a microphone with an artificial outer ear (pinna).

Some alternatives to FFT as audio analysis method are the Discrete Wavelet Transform (DWT), and Mel Frequency Cepstral Coefficients (MFCC). DWT has been proven to be an effective way to analyze the temporal and spectral properties of non-stationary signals (Tzanetakis, Essl, & Cook, 2001), (Tan, Fu, & Spray, 1996), as well as MFCC for spectral properties like pitch and formant frequency analysis (Muda, Begam, & Elamvazuthi, 2010).

Various versions of the DWT, in conjunction with statistical methods such as the k-Nearest Neighbor Distance Classifier, have also been used successfully to classify musical signals (Lambrou, Kudumakis, Speller, Sandleg, & Linney, 1998). This article gives a fairly detailed overview of each method's classification capabilities. DWT analysis has also been applied to fields other than audio, such as time series data mining (Chaovalit, Gangopadhyay, Karabatis, & Chen, 2011), and the JPEG2000 image compression algorithm.

Audio analysis using the Cepstral representation of the short-term power spectrum (such as MFCC) can be used to identify pitch and formant frequencies in speech (Sathe-Pathak & Panat, 2012), and has in fact become a very conventional way to process human speech before recognition.

Some speech and dialogue based systems for the NAO also exist. For instance an event-based conversational system for Human-Robot Interaction (HRI) (Kruijff-Korbayová et al., 2011). Though not yet fully autonomously usable, it provides a solid basis for future research. One of the biggest issues in robot audio recognition is background noise. HRI approaches using the NAO built-in conversational system have had some difficulties getting around this barrier (Jayagopi et al., 2013). However, more recent research with a fairly simple approach, using the FFT of a signal and logistic regression, has obtained very promising results in noisy environments (Poore, Abeyruwan, Seekircher, & Visser, 2014).

3 Research Method

In this section, a description will be given of all methods, their implementation details, and optimizations for the task in hand. Firstly, we will discuss data collection on the NAO robot, in particular the several data sets that were recorded for this research project. In the next section all the audio analysis methods will be set out, as well as the instance vectors they produce for the machine learning phase. Hereafter, a quick recap

of the classifiers, and the design choices made for both accuracy and F1 score gain, will be given. The exact classifier settings can be found in appendix A.

3.1 Data Collection

All experiments were conducted using audio recorded indoors, on the NAO robot. The types of audio consist of two predefined signals, namely a 'start' and a 'stop' signal, both a sequence of $200Hz$ and $320Hz$ sinusoids, and a complex whistle sound, as used by a referee in any regular football match. The distance between the robot and the audio source, as well as the level of background noise, were both decided upon after preliminary audio data assessment: The recordings should not contain any clipping, empty bits of audio in between actual signals were scrubbed, and the signal should be audible to a human being.

Five data sets were recorded using NAO's microphone array. The SPL challenge requires the recognition of both predefined signals and a whistle, brought by the team itself. For both types of signals, two distinct datasets were recorded. Each dataset consists of two recordings, one with a low amount of noise and high signal strength (sets 2 and 4), the other with background noise and a low signal strength, recorded further away from the signal source (sets 1 and 3). Some background noise, containing small amounts of human speech, was also recorded, to serve as the negative samples for training (set 10).

All audio samples were manually marked as positive, $y = 1$, or negative, $y = 0$. Combined data sets were then created, namely sets 5, 6, 7, and 8, one for each type of signal, and one per amount of background noise, by concatenating data from the first four sets into a single file. Finally, a data set containing all types of audio was created, namely set 9. Not all data was used in these combined sets: In order to keep the data varied, and speed up classifier convergence, some audio was removed because it was too similar to other audio bits in the set. Data set descriptions can be found in table 1.

Each of the data sets containing the whistle and the predefined signals was scrubbed to remove audio without a signal presence.

Table 1: Data set description. All data sets were recorded at a 48000 Hz sample rate, using pcm16 encoding, on all 4 NAO microphones simultaneously.

Set	Description	Amount of audio bits	Duration (s)
1	Whistle set, with background noise	4450	56
2	Whistle set, minimal noise	4205	53
3	Predefined signal set, with background noise	2210	28
4	Predefined set, minimal noise	2225	28
5	Whistle data, sets 1 and 2 combined	3289	41
6	Predefined signal data, sets 3 and 4 combined	4436	55
7	Noisy data, sets 1 and 3 combined	6661	83
8	Clean data, sets 2 and 4 combined	6430	80
9	All other sets combined	7726	97
10	Noise set	4805	60

3.2 Audio Analysis Methods

The microphones on the NAO provide a 48 kHz audio stream from 4 microphones, situated at the front, rear, and sides of the head. Audio frames were taken at a frequency of 20 Hz, or every 50 ms. This results in signal bits of 2400 samples, which is a sufficient size not to skip any samples because of the analysis processing time, and more than enough samples to produce a precise representation of the signal. These audio bits were normalized to reduce the effect of the audio volume on the final representation.

3.2.1 Fast Fourier Transform

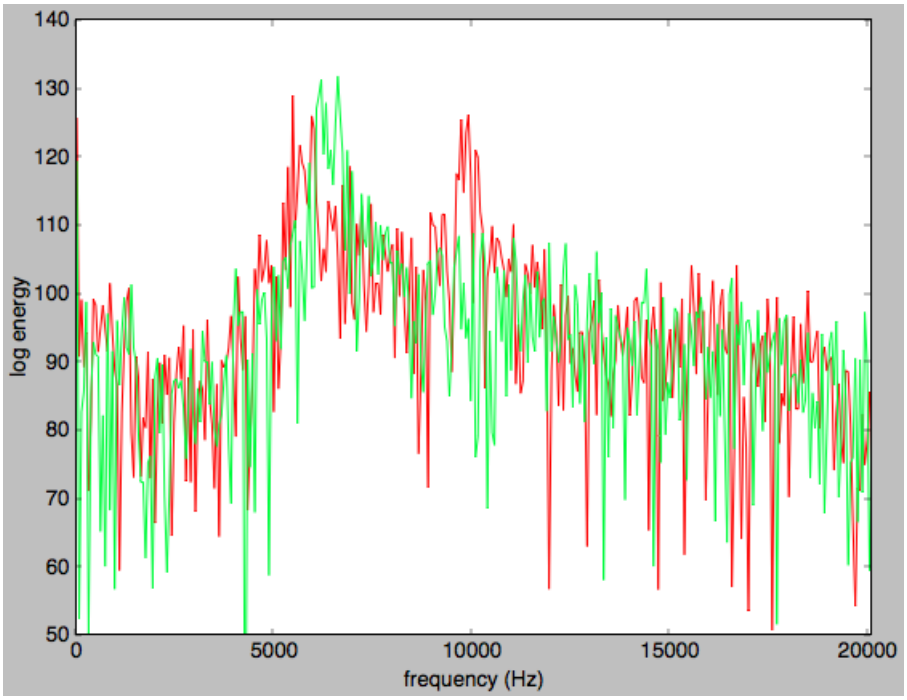

Figure 2: Two overlaid whistle FFTs. Red is a weaker whistle blow in a noisy environment (set 1), green contains a strong signal with only minimal background noise (set 2). The whistle base frequency resides between 6 and 7 kHz, while the second peak in the noisy FFT, around 10 kHz, is fan noise in the background.

There is a plethora of FFT algorithms, each of which calculates the Discrete Fourier Transform, or DFT, of a signal. The DFT Is a discretized form of a method for expressing a function as a sum of periodic components, and for recovering the function from these components. Using an input signal of N samples, the DFT is defined as

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i \frac{kn}{N}} \qquad k = 0, \ldots, N-1$$

For this thesis, the Scientific Python (SciPy) implementation of FFT was used (Jones, Oliphant, Peterson, & Alii, 2001–), specifically `scipy.fftpack.rfft`. This Real Fast Fourier Transform is an implementation of an adapted form of the

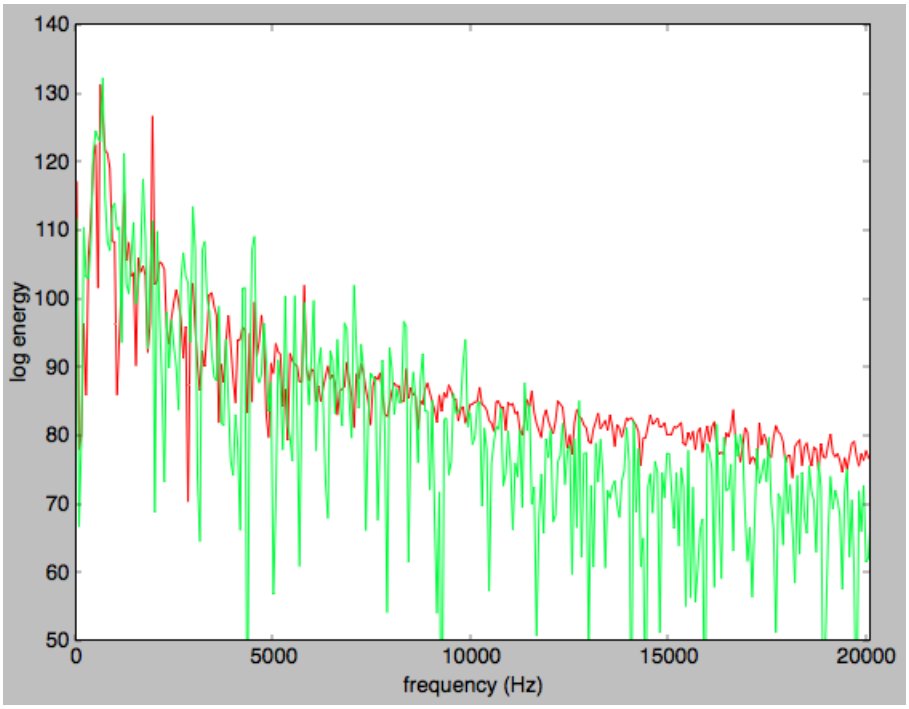

Figure 3: Two overlaid sinusoid signal FFTs of the start signal. Red was recorded in a noisy environment (set 3), green contains a strong signal with only minimal background noise (set 4). Without truncating, one can only just discern the two peaks at 200 and 320 Hz.

Cooley– Tukey algorithm (Cooley & Tukey, 1965), which can only be applied to a series of real values, such as a sound signal. By ignoring the complex conjugates, which would yield negative frequencies, it gains a considerable speed boost. This method yields $\frac{N}{2} + 1$ coefficients, of which the log values are taken, which are more suitable for most machine learning algorithms. The FFT is truncated at around 12 kHz, since early analysis shows that higher frequencies are redundant for the task of whistle recognition (see figures 2 and 3). This process generates the 680-feature length vectors used for training and testing our classifiers.

3.2.2 Discrete Wavelet Transform

For this research project, the implementation of DWT from the open source PyWavelets Python module, `pywt.WaveletPacket`, was used (Walewski & Alii, 2006—). Like most implementations of DWT, it calculates the transform of a discrete signal by passing it through a series of filters, using convolution. Instead of analyzing sinusoid levels in a signal, DWT attempts to find wavelets, which are wave-like oscillations with an amplitude that begins at zero, increases, and then decreases back to zero. There are many wavelet families, the particular wavelet used here is commonly referred to as the Daubechies 4-tap wavelet, or db4 (Daubechies & Alii, 1992). Firstly, the signal is decomposed by simultaneously applying a low pass filter and a high pass filter, which output the approximation and detail coefficients, respectively:

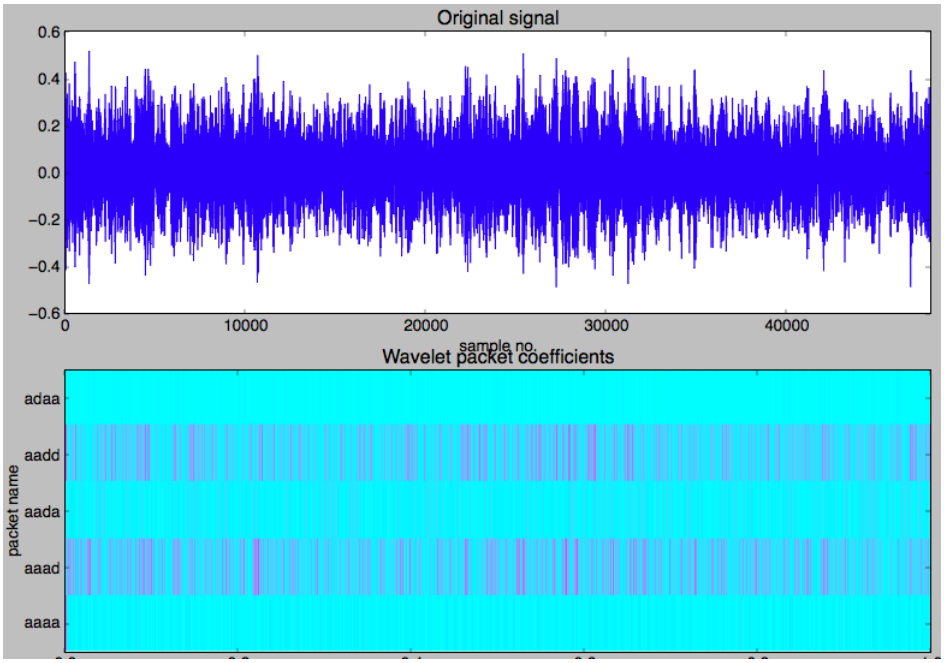

Figure 4: A second of audio from the clean whistle set (data set 2), and its DWT packet coefficients, truncated at coefficient level 5, as they were during processing. Without truncating, DWT produces 16 coefficients at level 4, labeled here with their packet node names.

$$y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k]$$

$$y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k]$$

Where h and g are the low and high filter's impulse responses, respectively. The output of the filters is then subsampled, because according to Nyquist's rule half of the samples can be discarded since half of the frequency band is removed. Hereafter the process of decomposition and subsampling is repeated in a filter bank. The repetition depth, or level, determines the frequency resolution; higher-level coefficients have a high frequency resolution, and a low time resolution. Preliminary analysis shows that a level 4 DWT provides a suitable representation for classification, as well as the fact that the higher-level coefficients contain no useful information. Therefore, the DWT is truncated at the fifth coefficient level, out of 16 levels in total. For each 2400-sample audio bit, the mean of the resulting 44 coefficient vectors is taken, to produce a 5-feature length vector for classification. Though this may seem to be a small amount of features, results show that it is sufficient for a classifier to attain a very high accuracy.

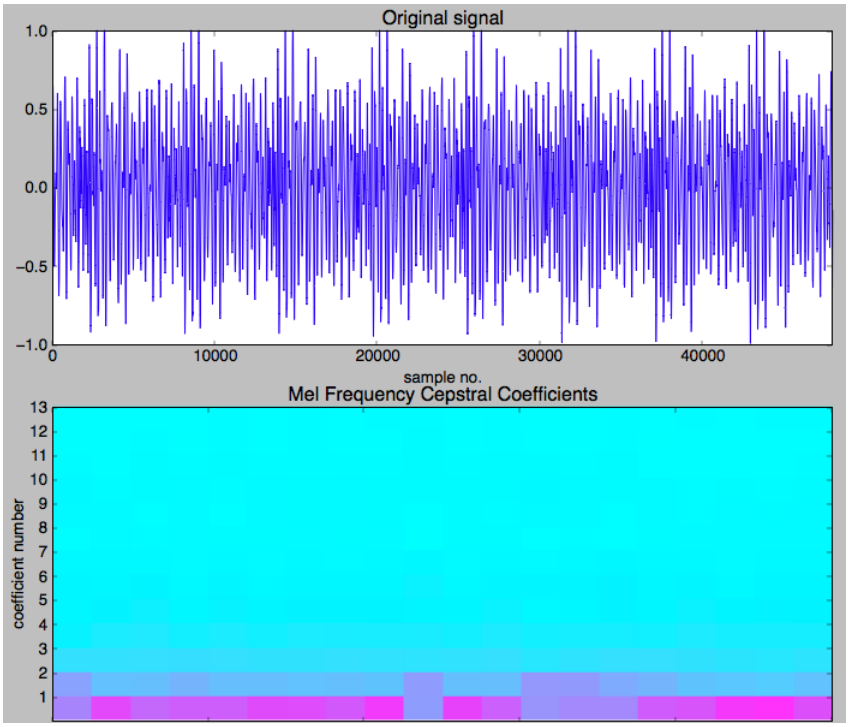

Figure 5: A second of audio from the noisy predefined recordings (data set 3), and its Mel Frequency Cepstral Coefficients. The MFCCs were calculated on 2400-sample audio bits, in the same way as was used for classification.

3.2.3 Mel Frequency Cepstral Coefficients

The mel-frequency cepstrum is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. MFCCs are commonly used as features in speech and speaker recognition systems, since its warped frequency scale approximates the human auditory system's response more closely than the linearly-spaced frequency bands of the FFT or common cepstral representation. For this thesis, the following approach was used to produce MFCCs:

1. Compute the FFT for an audio bit.
2. Map the obtained FFT onto the mel scale, using triangular overlapping mathematical windows, also known as producing a triangular bank filter.
3. Compute the log spectrum. This is done by taking the logs of the powers at each of the mel frequencies.
4. Finally, compute the discrete cosine transform of the list of mel log powers. This produces the MFCCs.

This method of calculating coefficients allows you to pick the size of the MFCC array as you choose. Result analysis shows that all characteristic components of both the

predefined and whistle signals can be captured in a 13-feature length instance vectors, which are shown in figure 5.

3.3 Machine Learning Algorithms

The machine learning algorithms used in this thesis were implemented in the Scikit-Learn Python module (Pedregosa et al., 2011). The SciKit library uses the LIBLINEAR library (Fan, Chang, Hsieh, Wang, & Lin, 2008) for logistic regression, LIBSVM (Chang & Lin, 2011) for support vector machines, and its own implementations for AdaBoost and stochastic gradient descent. Before classification, all analyzed data was scaled in order to produce data with zero mean and unit variance, which is needed for algorithms like support vector machines. This scaling was first applied to the training set, and its parameters were then used to scale the test set, in the same manner.

The data sets were then shuffled, and split with 70% of the data for the training set, and 30% reserved for cross-validation. These examples can be represented as a set of instance vectors $\vec{x}$, and mark y: $\{(\vec{x}_i, y_i)\}_{i=1}^{M}$, where $\vec{x}_i \in \mathbb{R}^M$ and $y_i \in \{0, 1\}$. The amount of samples, M, greatly exceeds the amount of features N: $M \gg N$.

All of the training and test examples can be defined as a design matrix, $\mathbf{X}$, with feature vectors as its columns. This gives us an $M \times N$ matrix, or $M \times (N + 1)$ in the case of an algorithm that uses a bias term, and a prediction term $h_\theta(\mathbf{X}_i)$.

The results in this report were obtained after each classification algorithm was run 20 times independently. For each classifier run, the data was re-shuffled and split, after which the classifiers were fitted. Their accuracy, F1, precision, recall, false positives, false negatives, true positives, and true negatives were stored and averaged over all of the runs.

3.3.1 Logistic Regression

The most efficient algorithm in terms of computing power used for this research project is the logistic regression binary classifier with l^2 normalization (Boser, Guyon, & Vapnik, 1992). This particular form of regularization was selected since previous approaches to this task yielded promising results (Poore et al., 2014). Its hypothesis space consists of sigmoid functions, $h_\theta(\vec{x}) = \frac{1}{1+e^{-\vec{\theta}^T \vec{x}}}$, where $\vec{\theta}$ are the adjustable weights, and $\vec{x}$ the sample vectors. In its implementation, $y_i \in \{-1, 1\}$, yet this is solved by automatically setting all instances intitally marked 0 to -1 prior to fitting, and resetting them for the evaluation phase. It solves the following unconstrained optimization problem:

$$min_{\vec{\theta}} \left(\frac{1}{2} \vec{\theta}^T \vec{\theta} + C \sum_{i=1}^{N} \xi(\vec{\theta}; \vec{x}_i, y_i) \right)$$

Where ξ is the loss function:

$$\xi(\vec{\theta}; \vec{x}_i, y_i) = log(1 + e^{-y_i \vec{\theta}^T \vec{x}_i})$$

In order to combat overfitting, but retain a high accuracy rate, a C-sweep (inverse regularization, $C = \frac{N}{\alpha}$) was performed, in the range of $\{0.1, 0.2, 0.4, \ldots, 81.92\}$. The upper value in this sweep was chosen based on preliminary F1-score assessment. Here a low C-value specifies stronger regularization. The C-value with the highest average F1-score over all tests on set 9 was chosen, but special attention was also given

to the amount of false positives, since a false positive in the SPL challenge would mean a score penalty. Preliminary analysis showed $C = 0.64$ to provide suitable regularization.

A bias term was also used in the discriminant function. This bias b term was handled by augmenting vector θ and each instance vector $\vec{x}_i$ with an additional dimension.

3.3.2 Stochastic Gradient Descent

The SGD classifier used here is a first-order learning routine, which, in contrast to batch gradient descent, considers a single instance vector at a time. The change of its weights is given by the gradient of the classification error, which is evaluated only for the latest in the training sample sequence. This variant is sometimes known as the On-line Gradient Descent algorithm (OGD). Its weights vector $\vec{\theta}$ is updated like so:

$$\vec{\theta} \leftarrow \vec{\theta} - \eta \nabla Q_i(\vec{\theta})$$

Where $Q_i(\vec{\theta})$ is the value of the loss function at example i, and the adaptive learning rate η is given by

$$\eta^{(t)} = \frac{1}{\alpha(t_0 + t)}$$

Where t is the weight number, $t = 1, 2, \ldots, T$. For the task of audio recognition, the decision was made to compute the gradient over a simple hinge loss function. This linear approach has previously successfully been applied to audio recognition, for instance in music annotation (Nam, Herrera, Slaney, & Smith, 2012).

3.3.3 AdaBoost-Samme

AdaBoost-SAMME (Zhu, Zou, Rosset, & Hastie, 2009) is a state-of-the-art generalisation of the AdaBoost algorithm (Freund & Schapire, 1995), which uses Stagewise Additive Modeling and a Multi-class Exponential loss function (hence SAMME). This meta-estimator begins by fitting a classifier, a decision tree classifier in this case, on the original dataset, and then fits additional copies of the classifier on the same dataset, but where the weights of incorrectly classified instances are modified such that subsequent classifiers focus more on difficult cases.

The 'weak classifier' used for this algorithm, a decision tree classifier, is first built using a greedy, top-down recursive partitioning strategy. It uses the Gini impurity function to calculate the quality of a split, which is a measure of how often a randomly picked instance would be incorrectly labeled if it were randomly labeled according to the distribution of labels it has already encountered. On its own, this decision tree classifier achieves a very low accuracy, but using AdaBoost with a standard learning rate of 1.0, the accuracy and F1-scores soar.

The limit set for the maximum amount of boost-estimators was set to 50, with the SAMME.R boosting algorithm in order to speed up computation, since AdaBoost-Samme generally takes a lot of time to converge. Because SAMME.R uses probability estimates to update its additive model, while SAMME only uses classifications, SAMME.R achieves a lower test error in fewer iterations than regular SAMME, which causes the converge time to decrease.

3.3.4 Support Vector Machine/C-Support Vector Classifier

The particular support vector machine used here is the C-support binary vector classifier, `sklearn.svm.SVC`, an l^2-regularized SVM. In its implementation, $y_i \in \{-1, 1\}$, yet this is solved by automatically setting all instances intitally marked 0 to -1 prior to fitting, and resetting them for the evaluation phase. Similar to logistic regression, this support vector machine solves the following problem:

$$min_{\vec{\theta}} \left(\frac{1}{2} \vec{\theta}^T \vec{\theta} + C \sum_{i=1}^{N} \xi(\vec{\theta}; \vec{x}_i, y_i) \right)$$

Yet with a loss function different from logistic regression, ξ, defined as

$$\xi(\vec{\theta}; \vec{x}_i, y_i) = max(1 - y_i \vec{\theta}^T \vec{x}_i, 0)^2$$

For FFT and MFCC, the error term penalty parameter $C = \frac{N}{\alpha}$ was kept at 1.0 after preliminary assessment - lowering C made model convergence unfeasible, while higher values gravely affected F1-scores. For DWT analysis, C was adjusted to 1.4. In the decision function used for FFT and MFCC, SVC uses a sign function with a third degree polynomial kernel K, without an intercept term:

$$h_\theta(\vec{x}) = sgn \left(\sum_{i=1}^{N} y_i \alpha_i K(\vec{x}_i, \vec{x}) \right)$$

Note that for most types of audio analysis, support vector machines will yield very poor results if the data has not been scaled. SVC is an inefficient algorithm for large data sets, since its fit time complexity is more than quadratic. In order to speed up convergence, a shrinking heuristic was used. This shrinking heuristic removes certain elements that may have already been bounded during the decomposition iterations (Chang & Lin, 2011). Because of the optimizations to this algorithm, convergence time was brought down to a feasible level, while retaining a low criterion for stopping tolerance ($tol = 0.001$).

4 Results

All analysis/classifier combinations were run 20 separate times on each combined data set, with a cross validation stage on the remaining 30% of the data that was not used during the training stage. The accuracy score reflects that of a single classifier, run on all audio bits in from all 4 microphone inputs, unless specified otherwise. The optimized settings for each algorithm can be found in appendix A.

4.1 Combined Data sets

Firstly, we will examine classification accuracy scores using all 4 microphones available, on the data set containing the noisy and clear recordings of both predefined and whistle signals. Table 2 shows the total accuracy percentage scores, rounded to two decimals, and their F1-scores.

This shows us that even on a mixed data set, including noisy data, a very high accuracy can be attained. As can be seen in table 2, each algorithm is shown to have reached a robust classification rate and F1-score after parameter optimalization. MFCC-based

	FFT	DWT	MFCC
SVM	98.17	97.23	**99.92**
SGD	99.19	97.81	99.91
ADA	99.89	99.84	99.90
LogReg	99.85	98.21	99.89

	FFT	DWT	MFCC
SVM	0.977	0.964	**0.999**
SGD	0.989	0.971	0.999
ADA	0.999	0.998	0.999
LogReg	0.998	0.976	0.999

Table 2: Accuracy percentage scores on the combined data set (set 9) on the left, and F1-scores on the right.

systems appear to have a slight advantage over FFT and DWT, but MFCC only performs better by such a narrow margin, that these experiments do not firmly establish either one of the audio analysis techniques as being the best.

The next set of experiments was performed on both signal types separately, in order to ascertain any discrepancies in performance. The results can be found in tables 3 and 4.

	FFT	DWT	MFCC
SVM	99.97	**100.00**	**100.00**
SGD	99.99	99.99	**100.00**
ADA	99.97	**100.00**	99.97
LogReg	**100.00**	99.98	99.99

Table 3: Accuracy percentage scores on the whistle data set (set 5).

	FFT	DWT	MFCC
SVM	99.73	99.80	99.87
SGD	99.70	99.82	**99.90**
ADA	99.87	99.84	99.87
LogReg	99.88	99.83	99.87

Table 4: Accuracy percentage scores on the predefined signals data set (set 6).

These results show that although classifier performance varies slightly between different types of audio, it is not clear whether this is due to the signal type itself, or to small imperfections in data set 6, since there is only a 0.2% accuracy rate variation.

4.2 Noisy Data vs. Clean Data

In order to obtain a clear view of the difference in classifier performance in varying conditions, each classifier/analysis combination was run on both the noisy and clear data sets (sets 1-4). The results of these tests, shown in table 5, provide the total classifier accuracy percentage scores, using all 4 microphone inputs. The same tests were performed on the predefined signal data sets, as shown in table 6. Experiments were also performed on the sets with combined signals (sets 7 and 8), as is shown in table 7. F1 scores for data sets 7 and 8 are shown in table 8.

These results show something unexpected: None of the analysis/classifier combinations seems to be heavily affected by signal noise. The accuracy rates on separate

	FFT	DWT	MFCC
SVM	98.95	98.57	99.73
SGD	99.62	98.50	99.02
ADA	**99.83**	99.28	99.59
LogReg	99.79	98.61	99.38

	FFT	DWT	MFCC
SVM	99.99	**100.00**	**100.00**
SGD	**100.00**	**100.00**	**100.00**
ADA	99.98	**100.00**	**100.00**
LogReg	**100.00**	**100.00**	**100.00**

Table 5: Accuracy percentage scores on the noisy whistle data (set 1 - left), and clean whistle data (set 2 - right).

	FFT	DWT	MFCC
SVM	95.48	99.79	99.88
SGD	99.61	99.74	99.77
ADA	99.82	99.77	**99.91**
LogReg	99.77	99.82	99.78

	FFT	DWT	MFCC
SVM	96.79	99.84	99.83
SGD	99.68	**99.88**	99.83
ADA	99.87	99.82	99.84
LogReg	99.78	99.82	99.82

Table 6: Accuracy percentage scores on the noisy predefined data (set 3 - left), and clean predefined data (set 4 - right).

sets 1-4, as well as the accuracy rates and F1-scores on combined sets 7 and 8 (see tables 7 and 8), show only minor variations. Also, table 6 shows that some classifiers show no variation at all, or even an increased accuracy rate when applied to the noisy data (MFCC + AdaBoost-SAMME.R).

	FFT	DWT	MFCC
SVM	97.44	94.99	99.60
SGD	98.74	96.25	99.22
ADA	99.69	98.99	99.65
LogReg	**99.78**	96.35	99.21

	FFT	DWT	MFCC
SVM	98.66	96.28	99.90
SGD	99.49	96.92	97.88
ADA	**99.88**	**99.88**	99.87
LogReg	99.87	97.88	99.87

Table 7: Accuracy percentage scores on the combined noisy data (set 7 - left), and combined clean data (set 8 - right).

	FFT	DWT	MFCC
SVM	0.971	0.943	0.995
SGD	0.985	0.955	0.991
ADA	0.996	0.988	0.996
LogReg	**0.996**	0.957	0.991

	FFT	DWT	MFCC
SVM	0.985	0.958	0.999
SGD	0.994	0.965	0.999
ADA	**0.999**	**0.999**	0.998
LogReg	0.998	0.975	0.998

Table 8: F1 scores on the combined noisy data (set 7 - left), and combined clean data (set 8 - right).

Table 7 and 8 shows a performance comparison between the total combined noisy and clean sets. Note that these sets contain multiple types of signals - start, stop, and whistle signals. Their overall performance decreases slightly when applied to noisy data, but only by a small margin. This implies all methods are mostly impervious to background noise.

4.3 Evaluation

The purpose of this experimental study is to assess several potential system proposals for the recognition of trigger-events in the form of audio signals, and to compare their performance in discerning both low-frequency sinusoids and a high-frequency whistle sound. Figures 6, 7, and 8 are scatter plots of $\sim 5\%$ of the processed versions of set 9, of which the dimensionality was reduced using Principal Component Analysis (PCA). These PCA representations clearly show that all audio analysis techniques provide a robust method for discerning each data point, since most of the sets are linearly separable.

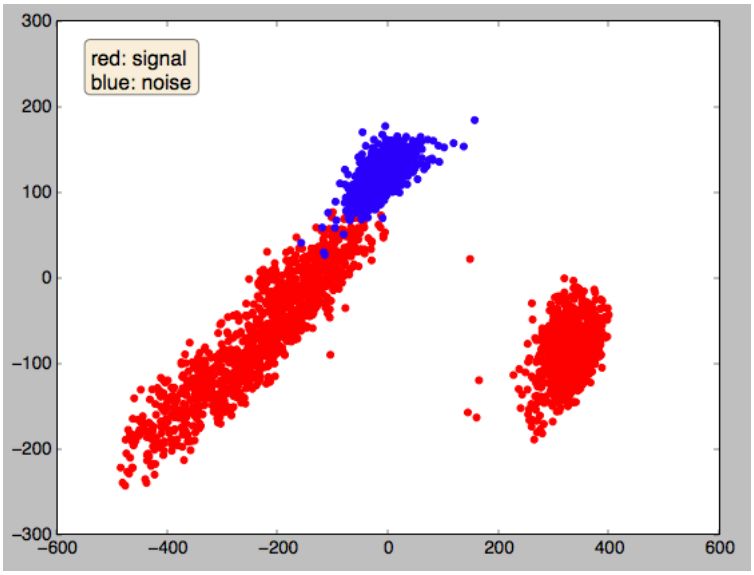

Figure 6: PCA visualization of the Fast Fourier Transforms of data set 9.

5 Discussion

All of the audio analysis methods used for this thesis have their advantages and disadvantages. FFT is a very widely used method for audio applications, mostly because of its complexity of $O(Nlog(N))$, but also because of the shape of its distribution in high-dimensional space (see figure 6), which indicates a signal presence very clearly. Classification rates were also high, even when applied to noisy data, as can be seen in tables 5, 6 and 7. Therefore, FFT clearly possesses the ability to reliably identify environmental auditory triggers. Nevertheless, as mentioned earlier, it is hard to apply FFTs to time-based sequences of sounds.

An even more efficient approach is the DWT, with its complexity of $O(N)$. The level of the DWT (its iteration depth) can also be adjusted, for even greater speed boosts, yet this may increase the classification error. The fact that DWT requires only linear extra computation time for a growing sequence, means that it has an advantage when it comes to autonomous robots, or other devices with a low computational capacity. Also, its representation, which is based of both frequency and location, allows not only for sequence analysis, but also for audio reconstruction from the DWT itself. For

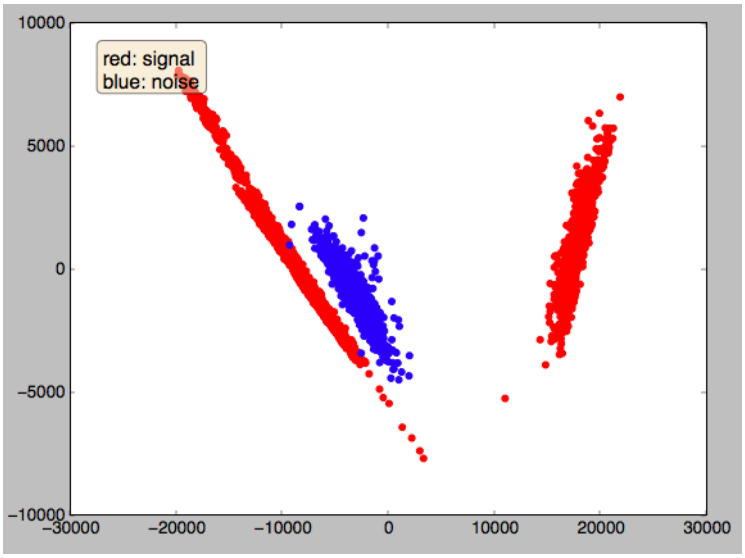

Figure 7: PCA visualization of the Discrete Wavelet Transforms of data set 9.

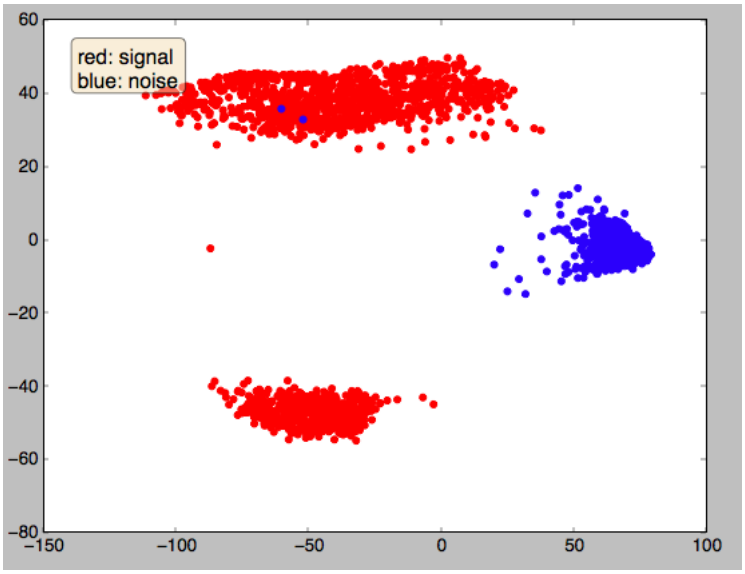

Figure 8: PCA visualization of the Mel Frequency Cepstral Coefficients of data set 9.

simple audio trigger recognition in systems where computational load is of the essence, this would be the analysis method of choice.

MFCC have the highest overall classification rates, yet its complexity is much higher than FFT, since it involves calculating the FFT, a bank filter, log computation, and a discrete cosine transform in order to generate its representation. The low error rates however suggest that MFCC are a prime candidate for multi-class classification, a high capability when it comes to discerning signals that seem closely related, and are especially suitable for time-based audio sequences, for instance in conjunction with

Dynamic Time Warping (Muda et al., 2010).

The parameters for C-support vector classification are relatively hard to optimize, since a small change in its stopping tolerance or C-value greatly affected model accuracy and convergence time - so even though this type of linear SVM is claimed to always converge (Chang & Lin, 2011), its use for audio classification is discouraged because of its slightly worse performance in noisy signal recognition (see table 7), and its impracticality during the training stage.

Considering a system with a low computational capacity, like the NAO robot, the obvious machine learning algorithm of choice would be Logistic Regression. It uses very few resources, since its decision function is a simple sigmoid function, has a fast convergence time, and lower accuracy rates than both C-SVM and SGD.

A system in which computational capacity is not a bottleneck, or one that requires a very high accuracy on different types of signals, would have a definitive advantage using AdaBoost-SAMME.R. Since this classifier also support multi-class classification without reducing the problem to multiple binary classification problems (Zhu et al., 2009), and its accuracy rate never dropped below 98.99% on any of the tests, this algorithm applied to MFCC has the capacity to discern multiple types of signals accurately, using just one classifier.

6 Conclusion

12 different system proposals for recognizing auditory triggers on a NAO robot using 4 microphones were tested and evaluated. The audio analysis was performed using Fast Fourier Transform, Discrete Wavelet Transform, and Mel Frequence Cepstral Coefficients, whereas the recognition task was handled by a C-Support Vector Classifier, Stochastich Gradient Descent, AdaBoost-SAMME, and Logistic Regression.

The results reported herein clearly prove that there is a large number of system architectures able to tackle the problem of recognizing both the predefined sinusoid signals and an arbitrary whistle signal. Experiments were carried out comparing the accuracy rates, precision rates, and F1-scores, in order to discover the optimal settings for each algorithm in both noisy and quiet environments. Performance was not severely affected by noise, and because these results are based on $50ms$ time frames, while a real signal would be at least several frames long, all of these methods possess the ability to recognize natural audio signals reliably.

7 Future Work

The details of classifying audio using the FFT and MFCC have been examined in many research efforts. The DWT however is a fairly recent approach to audio analysis, and shows a lot of promise in enabling audio classification on devices with a limited computational capacity, since its time complexity is $O(n)$, as opposed to the more computationally expensive FFT and MFCC.

Another interesting future research subject is multi-class classification. During this thesis, efforts were only made to achieve binary classification; in more natural circumstances, different signals usually have different meanings. Therefore, extending the scope of these approaches to recognizing different classes of audio would be an obvious step.

Sound localization is also an intriguing theme, in terms of its usability in real-life problems such as rescue efforts. The NAO robot has 4 microphones, and a built-in sound source localization module. However, this module is fairly inefficient, so building a different system would be an interesting project.

References

Bonarini, A., Lavatelli, D., & Matteucci, M. (2005). A composite system for real-time robust whistle recognition. *online published*.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on computational learning theory* (pp. 144–152).

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

Chaovalit, P., Gangopadhyay, A., Karabatis, G., & Chen, Z. (2011). Discrete wavelet transform-based time series analysis and mining. *ACM Computing Surveys (CSUR)*, *43*(2), 6.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, *19*(90).

Daubechies, I., & Alii. (1992). *Ten lectures on wavelets* (Vol. 61). SIAM.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, *9*, 1871–1874.

Freund, Y., & Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory* (pp. 23–37).

Graves, A., rahman Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*.

Jayagopi, D. B., Sheiki, S., Klotz, D., Wienke, J., Odobez, J.-M., Wrede, S., ... Gatica-Perez, D. (2013). The vernissage corpus: A conversational human-robot-interaction dataset. *8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*.

Jones, E., Oliphant, T., Peterson, P., & Alii. (2001–). *SciPy: Open source scientific tools for Python.* Retrieved from `http://www.scipy.org/`

Kruijff-Korbayová, I., Athanasopoulos, G., Beck, A., Cosi, P., Cuayáhuitl, H., Dekens, T., ... Verhelst, W. (2011). An event-based conversational system for the nao robot. *Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop*.

Lambrou, T., Kudumakis, P., Speller, R., Sandleg, M., & Linney, A. (1998). Classification of audio signals using statistical features on time and wavelet transform domains. *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, *6*.

Lopes, G., Ribeiro, F., & Carvalho, P. (2010). Whistle sound recognition in a noisy environment. *(not published)*.

Muda, L., Begam, M., & Elamvazuthi, I. (2010). Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *Journal of computing*, *2*(3).

Nam, J., Herrera, J., Slaney, M., & Smith, J. O. (2012). Learning sparse feature representations for music annotation and retrieval. In *Ismir* (pp. 565–570).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Poore, K., Abeyruwan, S., Seekircher, A., & Visser, U. (2014). Single- and multi-channel whistle recognition with nao robots. In *Robocup symposium (accepted for publication)*.

RoboCup Technical Committee. (2014). *Robocup standard platform league (nao) technical challenges*.

Sathe-Pathak, B. V., & Panat, A. R. (2012). Extraction of pitch and formants and its analysis to identify 3 different emotional states of a person. *IJCSI International Journal of Computer Science Issues*, *9*.

Saxena, A., & Ng, A. Y. (2009). Learning sound location from a single microphone. *IEEE International Conference on Robotics and Automation*.

Tan, B. N., Fu, M., & Spray, A. (1996). The use of wavelet transforms in phoneme recognition. *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*.

Tzanetakis, G., Essl, G., & Cook, P. (2001). Audio analysis using the discrete wavelet transform. *Proc. Conf. in Acoustics and Music Theory Applications*.

Walewski, F., & Alii. (2006—). *PyWavelets: Open source discrete wavelet transform in Python.* Retrieved from `http://www.pybytes.com/pywavelets/`

Zhu, J., Zou, H., Rosset, S., & Hastie, T. (2009). Multi-class adaboost. *Statistics and Its*.

A Classifier Settings

Classifier settings conform to Scikit-Learn's standards.

- SVM:

```
C=1.0, kernel='poly', degree=3, gamma=0.0,
coef0=0.0, shrinking=True, probability=False,
tol=0.001, cache_size=200, class_weight=None,
verbose=False, max_iter=-1, random_state=None
```

- SGD:

```
alpha=0.0001, class_weight=None, epsilon=0.1,
eta0=0.0, fit_intercept=True, l1_ratio=0.15,
learning_rate='optimal', loss='hinge', n_iter=5,
n_jobs=1, penalty='l2', power_t=0.5,
random_state=None, rho=None, shuffle=False,
verbose=0, warm_start=False
```

- AdaBoost:

```
base_estimator=DecisionTreeClassifier(compute_importances=None,
criterion='gini', max_depth=1, max_features=None,
min_density=None, min_samples_leaf=1,
```

```
min_samples_split=2, random_state=None,
splitter='best'), n_estimators=50, learning_rate=1.0,
 algorithm='SAMME.R', random_state=None
```

- Logistic Regression:

```
penalty='l2', dual=False, tol=0.0001, C=0.64,
fit_intercept=True, intercept_scaling=1,
class_weight=None, random_state=None
```