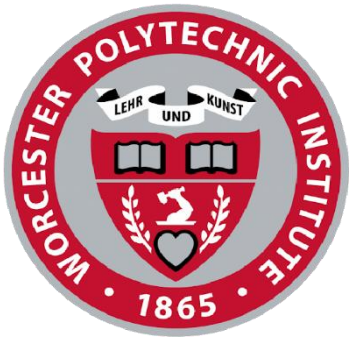
Intoxication Detection from Audio Using Deep Learning

A Major Qualifying Project Report

Submitted by:

SAINA REZVANI



WORCESTER POLYTECHNIC INSTITUTE

A report submitted in partial fulfillment
of the requirements for the degree of
BACHELOR OF SCIENCE in COMPUTER SCIENCE

Advisor:

PROFESSOR EMMANUEL O. AGU

Submitted on: April 7, 2019

Abstract

Driving under the influence is one of the largest risk factors leading to accidents. Intoxication manifests in the drinker's voice. This paper explores deep learning architectures and hand extracted features to classify voice samples as either intoxicated of sober. Our method classifies intoxicated speech with an unweighted average recall of 59.2%.

Acknowledgements

I would like to thank my adviser Professor Agu for his guidance, insight and valuable feedback for the past three terms. His effort and support has made this project possible.

Thank you to Aishwary Jagetiba for his enormous help along this project and his contribution to the research.

Thank you to graduate students Preeti Havannavar and Zhaoyu Sun, and undergraduate student, Alissa Ostapenko who have contributed to this project and spent countless hours dedicated to this research.

I would like to also thank the WPI Academic & Research Computing team who assisted me by providing computing resources and worked closely with me on the Turing Research Cluster.

Finally, I would like to thank my parents who have supported me through every step of my education and projects and have paved the way for my success.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

In the United States, 30 people die every day from motor-vehicle crashes due to alcohol intoxication [1]. This is an unacceptable loss of life that amounts to more than $44 billion of annual cost for alcohol-related crashes [3]. Intoxication tests such as breathalyzers for evaluating safe motor vehicle driving, heavy equipment operation and machine tool use require active user involvement, and must be purchased and carried whenever drinking, which reduces compliance [1].

There is a clear need for more reliable, passive ways to detect intoxication. One approach is to use passive detection of intoxication from voice which has important applications to high-risk situations, such as driving and steering and there is limited research available on this specific speaker state.

1.1 The Impact of Alcohol Intoxication

Alcohol is known to affect human behavior and can be dangerous when consumed in large amounts [2]. Blood alcohol concentrations between 0.05% and 0.08% are known to impair judgment, while higher concentrations may cause nausea, slurred speech, and loss of coordination. Blood alcohol concentrations above 0.15% can leave a person unconscious and may even result in death. Due to alcohol's effects on human judgment and coordination, driving while intoxicated poses a great safety risk to other drivers and passengers on the road [3]. In fact, 10,497 people in the United States died in alcohol-impaired driving accidents in 2016, accounting for 28% of all driving-related deaths in the United States. Thus, it is important to research and develop techniques for passive detection of intoxication to alert drivers of their state before they start driving.

In addition to affecting coordination, alcohol also affects a person's speech, making it to become slower, and increase the number of pauses, stutters, and speech errors [4]. The pitch and fundamental frequency of a person's voice may also increase, but this effect is not consistent across genders.

1.2 Voice Analytics

One can use voice analytics to investigate the changes in speech patterns of intoxicated people in comparison those who are sober to facilitate detection of intoxication from voice. Voice analytics is a branch of audio processing research that analyzes spoken conversation and audio patterns using machine learning or deep learning models to extract and analyze information, including speech speaker state by analyzing patterns in human speech [43].

Speech analytics is related to voice analytics but instead focuses on analyzing the phonetics and focus on the speech content such as syllables and keywords [53]. Voice analytics, however, focuses on vocal elements including speed, pitch, tone and emphasis on certain syllables. When features are extracted from raw audio signals, they can be compared to known features that present emotions, depression or alcohol intoxication. Important advantages of voice analytics over speech analytics are it is language agnostic and it can extract mental health and psychological state information such as depression, which cannot be extracted from speech content unless the speaker specifically states they are depressed [53]. Thus, this MQP focuses on voice analytics.

Challenges: Analyzing audio data presents several challenges including the fact that voice samples are affected by gender, age, emotions, room acoustics and proximity to the user affects the voice sample [43].

There are two main approaches for voice analytics and speech processing: using classic machine learning techniques and deep learning models. Machine learning performs classification taking hand extracted features as inputs and deep learning performs additional feature extraction as well classification. Prior work has used machine learning algorithms such as HMM, GMM and SVM and has utilized deep learning models such as CNN and RNN [80], [7], [8], [73], [5]. All these techniques are explained in more detail in ***section 1.4*** and ***section 1.5***. This report focuses on using a CNN model in addition to using hand extracted features for improving feature extraction and similar to some previous work uses an RNN-based network for classification. [5]. Our method classifies intoxicated speech with an unweighted average recall of 59.2%.

Chapter 2 of this report provides a review of existing literature in intoxication detection and chapter 3 provides an overview of our proposed methods and algorithms. In chapter 4, we discuss

our experiments and finally our results in chapters 5 and 6. Chapter 7 includes our conclusions and future work.

1.3 MQP Problem Statement

Majority of previous passive intoxication detection methods from voice use classic machine learning approaches or utilize deep learning to classify an audio signal as intoxicated or sober [5], [7], [8]. In this project we explore deep learning architectures to perform feature extraction in addition to hand extracted features and ultimately improve intoxication detection.

1.4 Machine Learning in Voice Analytics

Machine Learning algorithms are able to take complex data and find patterns in the data [64]. These algorithms can put the data into different categories and make the data meaningful. The algorithms can then make highly educated guesses about inputs that are similar to the original data and thus put them into the corresponding category.

In order to classify voice, we first extract the acoustic features and utterance characteristics of the audio signal [65]. A machine learning algorithm can then categorize the data into the two classes of sober and intoxicated and make predictions for similar inputs.

Three main algorithms used in automatic speech recognition are GMMs, HMMs and SVMs [6], [7], [8]. These models are statistical models that try to characterize the properties of an audio signal [65].

1.4.1 Hidden Markov Models (HMMs)

In the 1970s two students at Carnegie Mellon implemented HMM for speech recognition for the first time [66]. Since then HMM has been the foundation of future research in speech processing. HMMs have been used for speech recognition and specifically vocabulary detection and intoxication detection [79], [80]. HMM is suitable model for time varying spectral sequences. HMM is based on a Markov chain, which is a model that utilizes the probabilities of transitioning between random variables which are called states [45]. These states correspond to the acoustic phonemes in speech processing.

Figure 1 illustrates an example of a Markov chain. This figure has three states and the transition probability of going from the snow state to sunshine is 0.4 or 40%.

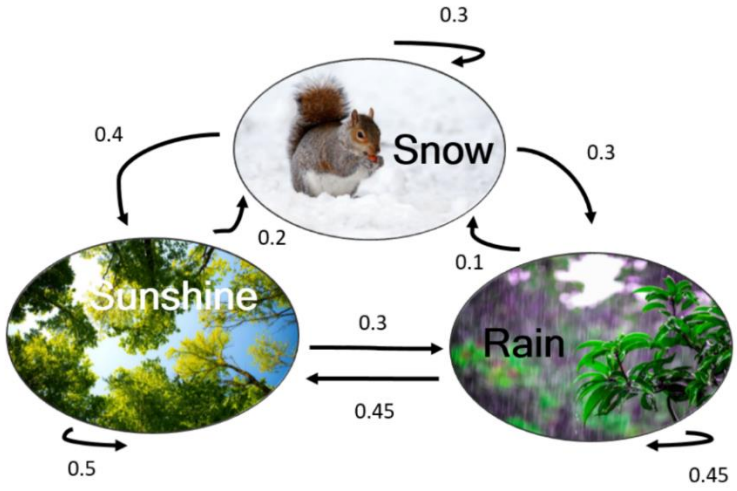


Figure 1: Markov Chain

The graph of a Markov chain and the transition probabilities between its states [45]

These transition probabilities can be represented in matrix form. In this case matrix q represents the transition probability [45]:

$$P = \begin{bmatrix} 0.3 & 0.3 & 0.4 \\ 0.1 & 0.45 & 0.45 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$$

In this example the weather can be guessed by measuring the temperature inside the house (hot or cold) but in this case the weather outside is unknown [45]. The hot and cold parameters observables while the hidden states of snow, rain or sunshine are internal states.

$$\mathbb{P}(\text{Hot}|\text{Snow}) = 0,\ \mathbb{P}(\text{Cold}|\text{Snow}) = 1$$
$$\mathbb{P}(\text{Hot}|\text{Rain}) = 0.2,\ \mathbb{P}(\text{Cold}|\text{Rain}) = 0.8$$
$$\mathbb{P}(\text{Hot}|\text{Sunshine}) = 0.7,\ \mathbb{P}(\text{Cold}|\text{Sunshine}) = 0.3$$

In this case, the HMM can be used to determine the probability that it is cold for two consecutive days [45] as:

$$\mathbb{P}((\text{Cold}, \text{Cold}), (\text{Rain}, \text{Snow})) = \mathbb{P}((\text{Cold}, \text{Cold})|(\text{Rain}, \text{Snow})) \cdot \mathbb{P}(\text{Rain}, \text{Snow}) =$$
$$\mathbb{P}(\text{Cold}|\text{Rain}) \cdot \mathbb{P}(\text{Cold}|\text{Snow}) \cdot \mathbb{P}(\text{Snow}|\text{Rain}) \cdot \mathbb{P}(\text{Rain}) = 0.8 \cdot 1 \cdot 0.1 \cdot 0.2 = 0.016$$

In speech processing, the observables are the acoustic features in each frame and the states correspond to the acoustic phonemes. The probability of going to one phoneme to another can give us information about a specific emotion or intoxication. HMM takes the acoustic features and performs classification on the data which in our case corresponds to sober and intoxicated.

1.4.2 Gaussian Mixture Models (GMMs)

GMM is another statistical model that was initially proposed for parametrizing the spectrum of speech and ever since has been commonly used for speech recognition and processing [67]. GMMs are used for speaker dependent speech recognition [69], emotion detection [68] and intoxication detection [6], [7]. GMM models the feature distribution of speech [65]. Gaussian Mixture Models are probabilistic models that find clusters of data points (subpopulation) in a dataset (overall population) that share some common characteristics [44]. These data points correspond to the smallest segments of sound called phone in speech processing. Since the subpopulation assignment is not known, this model constitutes a form of unsupervised learning [44]. Figures 2 and 3 illustrate the clustering process of the GMM model.

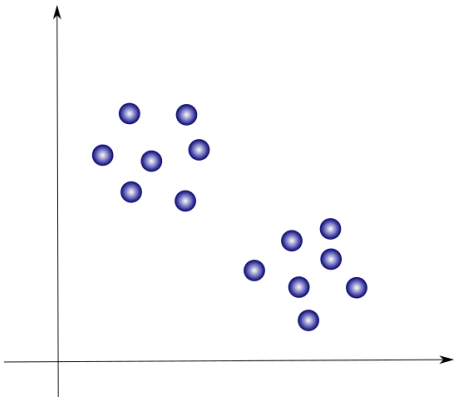


Figure 2: Unclustered Data

Overall population/dataset [44]

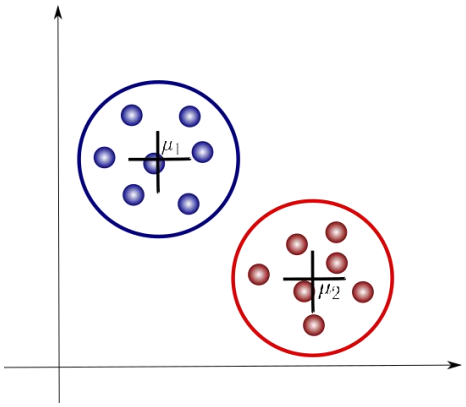


Figure 3: Clustered Data Using GMM

A GMM model finds subpopulations/clusters [44]

A Gaussian Mixture consists of multiple Gaussians. Each Gaussian represents a cluster and includes the following parameters: Mean μ (center), covariance $\sum$ (width) and mixing probability π (the size of the Gaussian function) [44]. Figure 4 illustrates these parameters.

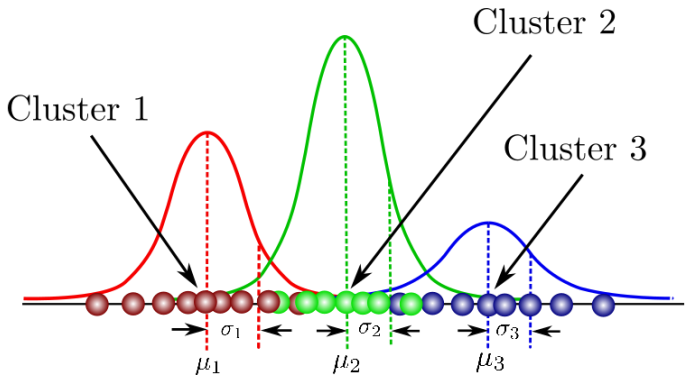


Figure 4: Gussian Parameters

Graphical representation of each cluster and its parameters [44]

1.4.3 Support Vector Machine (SVM)

SVM is a phenome classifier that has been particularly used for emotion and depression detection from voice [70], [71]. Due to their success for classification on acoustic features, SVMs have also been widely used for intoxication detection [6], [7], [8]. SVM is a machine learning algorithm that finds a hyperplane within an N dimensional space that separately classifies the datapoints that share common features (classes) [49]. In this algorithm N refers to the number of features. The best hyperplane has the largest distance from the datapoints as that represents confidence. This hyperplane classifies the dataset and can be used for two or more classes [49]. Figure 5 and 6 illustrate how SVM works and illustrates its hyperplane.

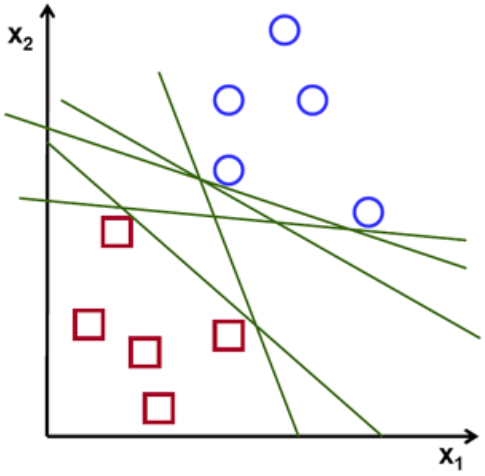


Figure 5: SVM Possible Hyperplanes

Possible hyperplanes to classify the dataset into classes [49]

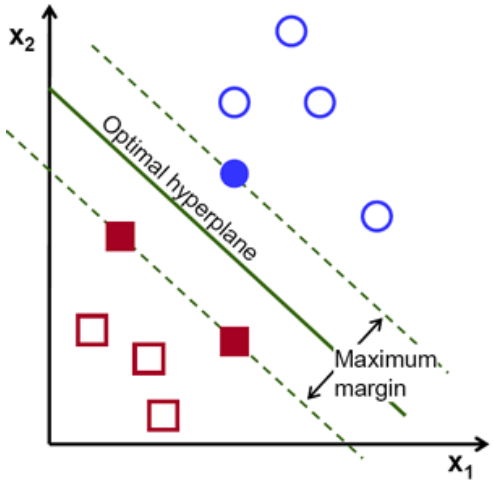


Figure 6: SVM Optimal Hyperplane

Optimal hyperplane that has the maximum margin from the classes showing confidence in the classification [49]

1.4.4 Log Mel Spectrogram

Hand extracted features are characteristics that can be extracted from an audio signal in small regions also known as frames or longer segmented regions [44], [45]. These features represent the different characteristics of a signal such as its loudness, sharpness and energy. Hand extracted features are explained in more detail in ***section 2.4***.

One of the main types of hand extracted features used in speech processing is log Mel spectrogram. The Mel scale is generated by taking the entire frequency of an audio signal and dividing it into bins of pitches that sound the same to the human ear [47]. Figure 7 illustrates the Mel Scale.

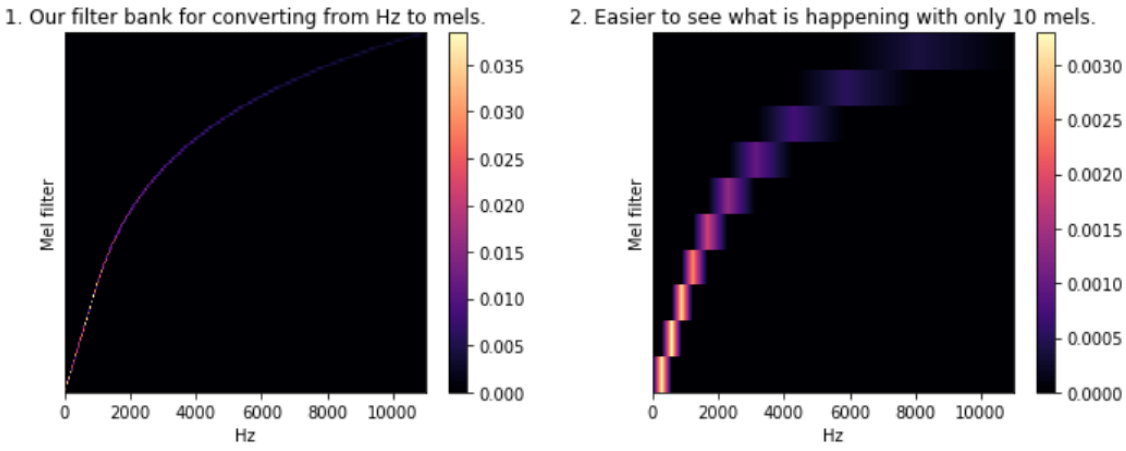


Figure 7: Mel Scale

The Mel scale of a sample audio [47]

A spectrogram is the Fourier Transform of sound, which takes the signal in the time domain as its input and returns the decomposition of the signal in frequencies [47]. Log Mel spectrogram is the spectrogram of signal in the Mel scale. Figure 8 provides an example of a log Mel spectrogram taken from a sound sample.

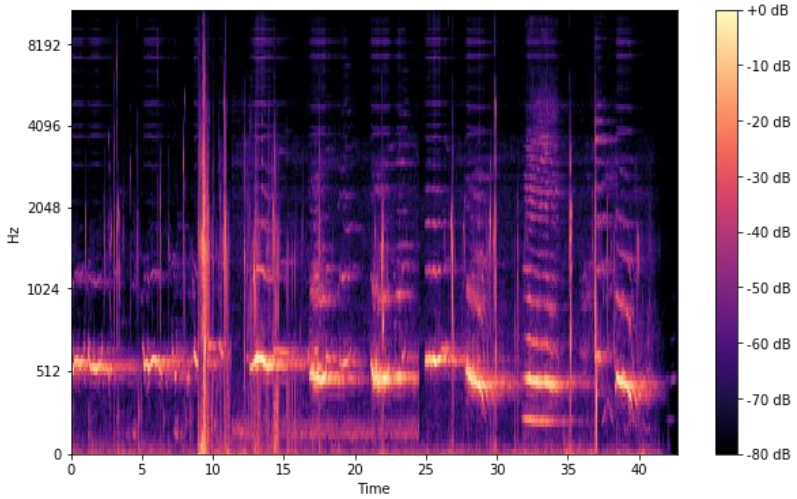


Figure 8: Log Mel Spectrogram

Spectrogram of a sample audio in the Mel Scale [47]

1.5 Deep Learning in Voice Analytics

Deep learning has had demonstrated success in a variety of other speech processing tasks as well including emotion detection, speaker recognition, and audio event detection [12], [13], [10]. Deep Learning is an enhanced version of Machine Learning that uses hierarchal neural networks that find and amplify even the smallest patterns in the data [64]. While machine learning models only perform classification, deep learning models perform additional feature extraction which then improves the classification accuracy [5]. These deep models learn high-level features on top of the Low-Level Descriptors (LLDs) and frequently outperform standard machine-learning approaches. Two of the main algorithms in voice analytics and classification include CNNs and RNNs.

1.5.1 Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a deep learning algorithm that is utilized for a wide range of imaging and computer vision tasks including object recognition, speech detection and DNA sequencing [72], [73], [74]. CNNs take images as its input and can also be utilized for non-imaging tasks if the inputs can be converted to images. Voice analytics methods often convert inputs to spectrograms (an image) that are then analyzed using CNNs.

CNNs are able to detect unique features of each image by looking for the low-level features of the image such as corners, curves and constructing a more abstract concept through the convolutional layers [46]. Each convolutional layer is similar to a feature detector. By determining these features, CNN is able to differentiate an image from another one and detect the object in the image. In addition to convolutional layers, a CNN has pooling layers which decrease the spatial size of the features and extracts the dominant features. CNN has a set of fully connected layers that take an input volume an N dimensional vector where the dimensions correspond to the classes or categories the image can belong to [46]. This is the final step that classifies an image. Figure 9 illustrates the architecture of a CNN.

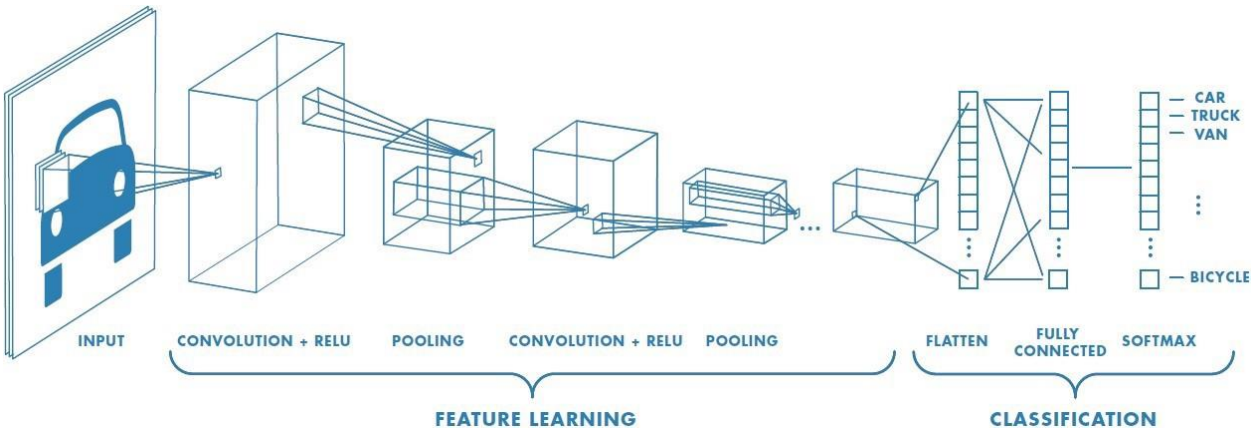


Figure 9: CNN Architecture

The architecture of a convolutional neural network (CNN) [46]

1.5.2 Recurrent Neural Network (RNN)

The Recurrent Neural Network (RNN) is a neural network architecture that remembers the past and makes the future decisions based on the knowledge it has, so the output of each step is passed to the next step [51]. In each time step, the same function computes the input, but it also uses the output of the last time step. As a result, steps are not independent of each and all the steps are related. RNNs have this functionality due to its hidden layer which takes the output of the previous time step. RNN uses its internal state to remember information and process a sequence of data. Consequently, it is useful for speech recognition where in order to predict a word of a sentence, it is necessary to remember the previous words of that sentence.

A bidirectional RNN is basically two regular RNN networks that are combined in opposite directions [51]. The input enters the two networks and moves in the forward and backward directions. In each step the output of the two layers are combined through various methods. Information the from past and future is preserved in two hidden states in each step. Figure 10 shows the architecture of an RNN.

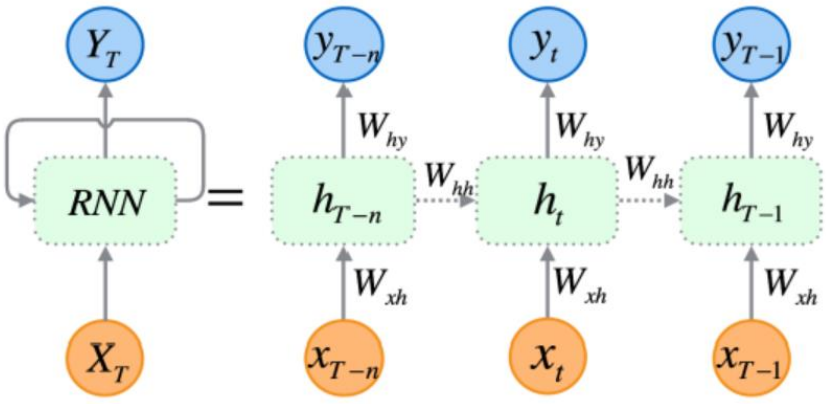


Figure 10: RNN Architecture

Representation of a Recurrent Neural Network (RNN) [51]

1.5.3 Long Short-Term Memory (LSTM)

One issue with many neural nets is that they suffer from a short memory [52]. This implies that in cases where a sequence is long, LSTMs are unable to carry all the information from the earlier time steps to the later time steps and this can miss some important information. LSTMs have gates that allow only the relevant information in the long chain of the sequence for classification. These gates are located in cell states that act as the memory of the architecture and decide which data to pass along during the training process. It can also forget information that is not relevant to the prediction [52].

Gated Recurrent Units (GRUs) are an upgraded version of LSTM where the cell state is removed and information is transferred in the hidden state [52]. It also has an additional reset gate that determines how much of the past information needs to be forgotten and thrown away.

Bi-LSTMs are bi-directional neural networks meaning that they remember the inputs from past to future and future to past and preserve the information from future data as well [52]. Figure 11 illustrates the architecture of a Bi-LSTM.

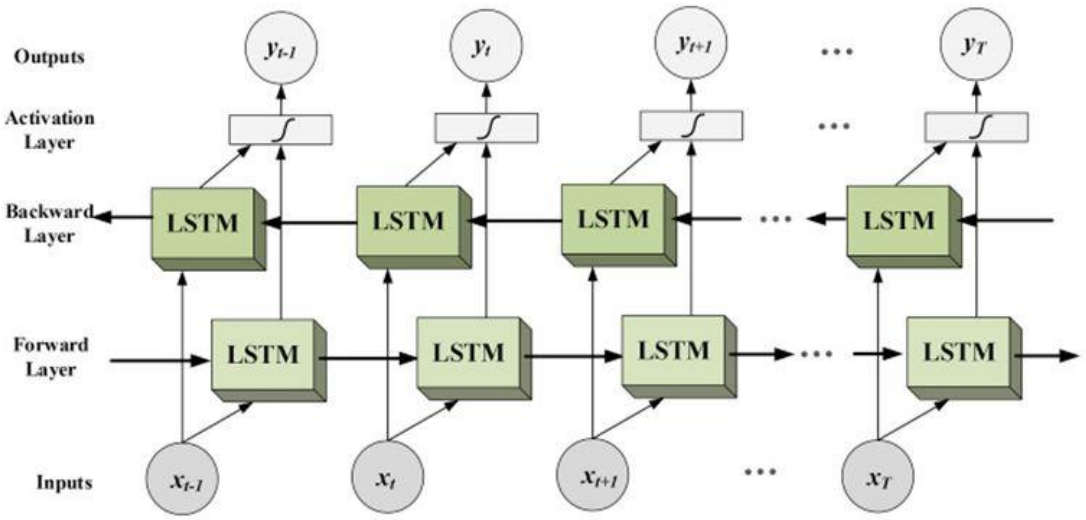


Figure 11: LSTM Architecture

The architecture of LSTM [58]

1.6 Contribution of this MQP

To improve upon previous work which uses classic machine learning techniques and only learns from hand-extracted features, we implemented a deep learning architecture for high level feature extraction and classification. The work in this MQP makes the following contributions:

1. This project explores a CNN-based architecture for intoxication detection from voice using log Mel spectrograms as input. We experiment with the VGGish and ResNext50 architectures for feature extraction and use different combinations of dense layers and pooling techniques for classification. VGGish and ResNext50 are CNN-based architectures that are explained in more detail in ***Section 3*** along with the different attention and pooling layers.
2. We also experiment with using VGGish pre-trained on Google AudioSet data as a feature extractor with global and attention-based pooling for classification. These CNN-based architectures require little feature engineering compared to previous methods and are easy to train on different domains and datasets.
3. To help counteract the sober and intoxicated class imbalance in our dataset, we also experiment with different data augmentation techniques that previous methods have not explored, including adding noise, shifting pitch, and stretching the audio.
4. Our results indicate that the ResNext50 network was able to successfully adapt to our data. We hypothesized that with further hyperparameter tuning and dataset adaption, the model could produce competitive results for intoxication detection on the ALC dataset. We mainly use unweighted average recall (UAR) as our metric of evaluation. Recall is the Number of correctly classified positive examples divided by the total number of positive examples. UAR is the mean of recall values for all the classes.
5. On our data, the model achieves an unweighted average recall (UAR) of 59.2%. Our work lays important groundwork for future research into CNN-based architectures for intoxication detection. With future experimentation and adaption to the ALC dataset, the CNN architecture can be used to classify intoxicated speech with a higher UAR.

Chapter 2. Literature Review

2.1 INTERSPEECH 2011 Challenge: Intoxication Detection Sub-challenge

Previous work into intoxication detection were primarily entries to the 2011 INTERSPEECH Challenge. The Challenge focused on intoxication detection and sleepiness detection from voice data, two speaker states that were less researched at the time. Passive sleepiness and intoxication detection from voice data have applications in the security and medical domains, especially in situations such as driving, steering, and controlling [37]. The Intoxication Detection sub-task of the Challenge was a supervised binary classification task using the Alcohol Language Corpus (ALC) Dataset, which is described in the next section.

2.2 Alcohol Language Corpus (ALC) Dataset

We use the Alcohol Language Corpus (ALC) from the Bavarian Archive for Speech Signals for binary classification of sober and intoxicated speech. ALC dataset used in the INTERSPEECH Intoxication Detection Challenge consists of sober and intoxicated German speech recordings of 162 males and females in an automotive environment. Unlike previous work in *alcoholized* speech, which has been primarily composed o

n male subjects in a lab setting with estimated speaker blood alcohol concentrations, the recorded speech in the ALC dataset features a variety of prompt styles and speakers of different genders, ages, and speaking styles. The dataset also includes metadata about each speaker as well as metadata about each recording, including blood alcohol concentration (BAC) of the recorded speaker and the environment of the recording.

Each speaker is recorded in a sober state and in an intoxicated state with BAC between 0.030% and 0.175% g/dL. The prompt sets for the sober and *alcoholized* tests contain read speech, such as tongue twisters and spellings, and spontaneous speech, including free response and command speech. All recordings are between 0.5s and 60s. There are 60 prompts in the sober test and 30 prompts in the *alcoholized* test, resulting in an unbalanced dataset in which only 1/3 of all recordings consist of intoxicated speech [41]. All audio samples were recorded in an automotive environment, at a sampling rate of 44.1 kHz. This sampling rate was kept constant for our experiments.

In our experiments, speakers with BAC of less than 0.08% are labeled as *Sober* and speakers with BAC 0.08% and above are labeled as *Intoxicated*. The following section outlines the techniques we use for data pre-processing and augmentation, as well as the model architectures we use in our experiments. Table 1 shows the different categories the data was gathered from.

2.2.1 Data Collection Procedures

Digit strings refer to credit card, phone and license plate numbers. Tongue Twisters were testing for articulation errors. Read commands were from the voice control application of a car. Address refers to home addresses that are more complicated to either pronounce or are long. The picture description dialogue was 60 seconds and speakers do not have to be talking for the entire 60 seconds hence pauses were taken into account. Spontaneous command was from the same car application used for read command.

speech type	*item type*	*intoxicated/control (A)*	*sober (N)*
read speech	digit string	5	10
	tongue twister	5	10
	read command	4	9
	address	5	10
	spelling	1	1
spontaneous speech	picture description	2	4
	question answering	1	1
	spontaneous command	5	10
	dialogue	2	5
sum		30	60

Table 1

ALC Data Collection: ALC recording types and their respective numbers in set A and N [41]

In the Intoxication Detection sub-task of the Challenge, speech recordings were labeled according to the Blood Alcohol Content (BAC) of the speaker, and they had to be classified as either *alcoholized* for BAC exceeding 0.5 per mill or *sober* for BAC equal to or below 0.5 per millilitres. The challenge contestants were given training and test sets and reported model performance using Unweighted Average Recall (UAR) [37].

2.3 Data Balancing

Many real-world datasets are unbalanced and may not have an equal number of instances of each class. This is common issue in machine learning because class imbalance negatively affects model performance as it causes models to learn the class distribution without properly classifying the data itself. Applying data balancing *data augmentation* techniques to training data improves model performance significantly, for both machine learning and deep learning models [15] [16]. Fukuda et al. [17] define *data augmentation* as artificially creating additional training samples to increase the diversity in training data. In Tang et al. [15], re-sampling techniques with and without replacement are used for data balancing and data augmentation is performed by changing the speed ratio of the raw audio. In Biadsy et al. [16], they maintain a balanced representation of all classes in train, validation, and test sets by performing stratified random sampling. Data augmentation using various signal and data processing techniques avoids model overfitting and improves the robustness of the model. Fukuda et al. [17] propose voice transformation (modification of glottal source and vocal tract parameters), noise addition, and speed modification as techniques for data augmentation that helps in recognition of foreign accented speech.

The Alcohol Language Corpus (ALC) dataset is unbalanced, with about 80% of samples labeled as sober and 20% labeled as intoxicated. To help model learning, we apply different audio augmentation techniques which are explained in ***Section 4.2***.

2.4 Low-Level Descriptors (LLD)

Interspeech's official audio feature set consisted of 4368 Low-Level Descriptors (LLDs) extracted using openSMILE and known to be useful for intoxication detection [72]. Low-Level Descriptors are features that can be extracted from an audio signal in small regions also known as frames or longer segmented regions. OpenSMILE is a library that performs automatic feature extraction from audio signals for music and speech machine learning classification [72]. It is widely used in the area of emotion detection and its features can be used for intoxication detection. Contestants could also extract additional low-level and hierarchical features for audio classification. Hierarchical features can be extracted through learning in different layers of a deep learning model.

Low-Level Descriptors fall into a number of categories. Four main categories of audio descriptors include temporal, spectral, cepstral and perceptual [54], [55].

1. **Temporal audio descriptors:** are features related to time or features that can be extracted from a signal over a time interval.
2. **Spectral audio descriptors:** are features related to the shape and structure of the audio signal such as the amplitude of a sinusoidal.
3. **Cepstral audio descriptors**: are features related to the cepstrum analysis of a signal which is a nonlinear signal processing technique used for speech and speaker recognition. Cepstrum is the inverse Fourier transforms (IFT) of a log of a signal spectrum which extracts components such as excitation and vocal tract system from a signal [54] [55]. Figure 12 is a simple representation of cepstrum.

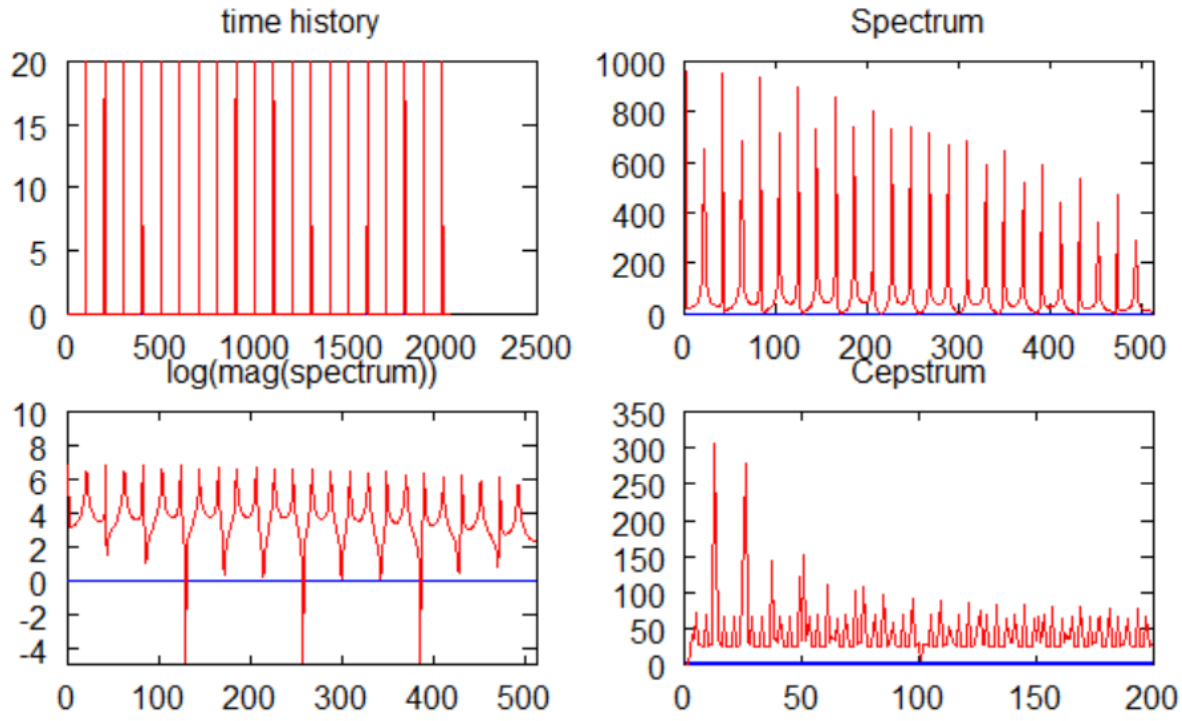


Figure 12: Cepstrum

The steps of calculating the ceptrum which is the inverse Fourier transform of log of a signal [56].

Finally, Perceptual audio descriptors are features related to the texture of the sound such as loudness and sharpness. Table 2 represents some of the common Low-Level Descriptors [48].

Low-Level Descriptor Type	Low-Level Descriptor	Description
Temporal	Energy envelope descriptor	Root mean square of the mean energy of an audio signal used for silence detection
	Zero crossing rate descriptor	The number of times the signal amplitude experiences a change of sign which is useful for differentiating music from speech
	Autocorrelation coefficient descriptor	The spectral distribution of an audio signal over time. Autocorrelation compares a signal to itself with lag time. This descriptor can be used to differentiate different musical instruments
Spectral	Spectral moments descriptor	Features of the spectral shape such as spectral flatness, symmetry, width and centroid. This descriptor can be used to determine sound brightness and mood
	Formant descriptor	The peaks of the sound spectrum of voice in a signal. This descriptor is useful for vowel and phoneme detection.
Cepstral	mel-frequency cepstral coefficient (MFCC) descriptor	The inverse discrete cosine transform of the energy of an audio signal in the Mel-scale frequency bands. This descriptor is useful for
Perceptual	loudness descriptor	The intensity of the sound
	Sharpness descriptor	Weighted centroid of specific loudness or the spectral centroid

Table 2

Low-Level Descriptors

Our work uses the ALC dataset used in the Intoxication Detection sub-challenge for supervised binary classification of speech segments as either sober or intoxicated. We use the ALC dataset to create our own splits for train, validation, and test sets and log Mel spectrogram features as input. We use UAR to evaluate our models to maintain uniformity in comparing results obtained from the original challenge.

2.5 Machine Learning Architectures for Audio Intoxication Detection

The machine learning architectures presented at the INTERSPEECH 2011 Challenge provide a solid baseline for audio classification on the ALC dataset. These models are typically composed of Hidden Markov Models (HMMs) or Gaussian Mixture Models (GMMs) that learned on low-level descriptors (LLDs) hand-extracted from speech in the Alcohol Language Corpus (ALC). After speech is analyzed using HMMs or GMMs, it is classified as either sober or intoxicated with Support Vector Machines (SVMs) or Gaussian-based classifiers [37]. The work is guided by previous studies that indicate low-level acoustic and prosodic features, such as fundamental frequency and rhythm, are affected by alcoholic intoxication [2]. Table 3 outlines the architectures, feature types, and performance results for previous work on the ALC dataset.

Model Description	Features Used	Accuracy (%)	UAR (%)
SVM classifier with linear kernel (baseline model) [8]	openSMILE LLDs	65.9	66.4
3 GMMs fused together using a linear SVM for classification and additional speaker normalization techniques [6]	• openSMILE LLDs • Praat contour features • hierarchical features	**70.47**	**70.54**
GMM-supervectors with linear SVM for classification [7]	• openSMILE LLDs • prosodic • text-based	68.6	68.5
bidirectional RNN with GRUs [5]	FBANK (sub-signals corresponding to smaller regions of a signal spectrum)	**75.9**	**69.2**

Table 3

Previous work on the ALC dataset

2.5.1 WEKA

The baseline architecture in the Challenge used the WEKA data mining toolkit [75] for classification. WEKA is an open source software that is used for predictive models and data mining. WEKA has tools and algorithms implemented for pre-processing data, machine learning tools for classification and regression, clustering and multiple visualization tools.

2.5.2 SMOTE

Synthetic Minority Oversampling Technique (SMOTE) was also used to balance the dataset in [8]. This approach is used when the classes have an unequal amount of data or specifically when the minority class needs to be balanced with the other classes. SMOTE picks a random example from the minority group and then a k number of the nearest neighbors are found. One of these neighbors are chosen randomly and a new example is synthesized at a point between the original point and the chosen neighbor [50].

2.5.3 INTERSPEECH Approaches

The baseline model chose a Support Vector Machines (SVM) with linear Kernel and the WEKA toolkit for classification [8]. The baseline model used the features IS2009EC, IS2010PC, and ISSSC2011 feature sets that were the official sets of the Emotion, Paralinguistic, and Speaker State challenges. These features were extracted by the openSMILE library. The baseline model achieved a UAR of 66.4% on the test set.

The winning submission, presented by Bone et al. [6] used a fusion architecture consisting of 3 GMMs and two feature extractors, one for extracting eight additional acoustic features using Praat and another for calculating hierarchical features, such as mean and standard deviation of the low-level features. The various prosodic and spectral features are fused and classified with a linear SVM, producing a UAR of 70.5% on the test set. Prosodic features include intonation (pitch), stress (loudness) and rhythm

Praat is a software that can be used for speech analysis, synthesis and manipulation. Praat can calculate the change of F0 (fundamental frequency) contours over time (specifically in each utterance for speech analysis). F0 is the frequency of vibration for the same phenomenon pitch [57]. This report also uses speaker normalization techniques in their model. Different speakers can have acoustic variations for phonologically identical utterances. Speaker normalization can help make the model robust to acoustic variations for the same words or phenomes.

Another approach by Bocklet et al. [7] uses score-level fusion to combine the classification outputs based on phonetic and word-level disfluency features, such as false starts, pauses, and unintelligible words, in addition to classification based on spectral and prosodic features. They achieved a UAR of 68.8% on the test set. Other approaches experimented with various feature extraction and audio normalization techniques, achieving UAR around 67%.

Although the models presented at the INTERSPEECH Challenge provide a strong baseline for audio classification, they often require complex feature engineering and speaker normalization techniques [6] to account for variability in speakers and environments. Thus, we decided to use current deep learning approaches for audio classification, which are outlined in the following sub-section. These architectures require little feature engineering and are robust to different speaker styles and acoustic environments.

2.6 Deep Learning in Audio Classification

2.6.1 RNN Networks for Speech Processing

Berninger et. al [5] lay the foundation of using a deep neural network for the speaker intoxication detection task on the ALC dataset. They use a bi-directional Recurrent Neural Network (Bi-RNN) with 2 Gated Recurrent Unit (GRU) layers and Gaussian dropout layer for the binary intoxication detection task. The Bi-RNN model has a forward GRU layer and a backward GRU layer to capture dependencies in the speech signal in both the forward and backwards directions while avoiding the vanishing gradient problem. The CMU Sphinx toolkit is a software that utilizes state-of-the-art algorithms for speech recognition and acoustic model training. [5] use the CMU Sphinx speech recognition toolkit with 40-dimensional filter bank (FBANK) features from speech segments in the ALC dataset. The spectrogram representations of the audio signals are input to the network. The model achieves an accuracy of 71.30% and UAR of 71.03%, outperforming the winning submission of the 2011 Challenge with minimal feature engineering.

Han et al. [11] show that Deep Neural Networks (DNNs) paired with an Extreme Learning Machine (ELM) outperforms standard HMM-SVM approaches for emotion classification. ELMs are feed forward neural networks that have one or more hidden layers in which the parameters do not need to be tuned. They use two DNNs to learn high-level short-term audio features at the segment-level and utterance-level before using an ELM for classifying the emotion of the utterance. Recurrent Neural Networks, especially LSTM and GRU cells, have been effective for capturing long-term context of audio segments and are robust to different speaker styles [12], [14]. To capture long-term dependencies in the speech signal, Lee et al. [12] use a bi-directional Long Short-Term Memory (bi-LSTM) model with an ELM, achieving a 12% absolute improvement in UA over the DNN-based model for emotion classification. Pooling the output of bi-LSTM layers using local attention further improves emotion recognition on the same corpus [13]. The attention mechanism allows the model of Mirsamadi et al. [13] to ignore frames which do not contain emotionally salient information.

2.6.2 CNN Networks for Speech Processing

Other deep learning techniques for speech processing use Convolutional Neural Networks (CNNs) in the classification pipeline. Deep CNN models are robust to different audio environments and speaker styles [21]. They require little feature engineering and learn high-level feature representations as they train [10], [14].

Hershey et al. [10] show that CNN architectures such as AlexNet, VGG, Inception, and ResNet, which are typically used for image classification, are also effective for large-scale acoustic signal processing on the Youtube-8M dataset. This dataset contains about 8 million videos with 500K hours of annotated content and a vocabulary of 4800 visual entities. Each video has at least 1000 views and is between 120 and 500 seconds [59]. The VGGish model was successfully used as a feature extractor in a CNN-DNN pipeline [42] for audio classification on the Google AudioSet dataset [40]. The Google AudioSet consists of 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos that are classified into 632 audio event categories/classes. The labeling of the segments is based on context such as links, metadata and content analysis [60]. CNNs have also been used for health-related speech processing tasks.

Wu et al. [9] use a Convolutional Neural Network (CNN) to detect pathological voice disorders on the Saarbrucken dataset. This dataset is a collection of 2000 German voice recordings. They compute spectrogram representations of normal and pathological speech in the dataset and input these representations into a CNN pre-trained with a Convolutional Deep Belief Network (CDBN). The CNN comprises of 10 convolutional and max-pooling layers followed by a Dense layer for classification. The CNN-based model performs with 71% accuracy on the test set and achieves an F1 score of 72%. The F1 score is explained more in ***section 4.4***. Moreover, CNNs can be used to successfully classify audio end-to-end from raw data, requiring no additional feature extraction techniques [14].

Previous machine learning techniques for intoxication detection rely on a set of hand-extracted Low-Level audio Descriptors (LLDs) to learn and require careful model adaptation and tuning to account for speaker variability. CNNs have performed well in audio classification tasks and are robust to different speaker styles and acoustic environments [36]. Moreover, they are easy to train and perform well on weakly-labeled datasets [42] and several CNN-based pre-trained neural network architectures exist for transfer learning. Given the imbalanced classes in

the Alcohol Language Corpus (ALC) and the variability in speaker gender and style in the corpus, we focused our research on CNN-based architectures for intoxication detection.

Chapter 3. Proposed Intoxication Detection Architecture

Previous techniques for intoxication detection relied on classifying hand-extracted Low-Level audio Descriptors (LLDs) and did not experiment with using CNNs in the audio classification pipeline. This work presents a CNN-based architecture with different pooling techniques for supervised binary classification of sober and alcoholized speech. We trained the VGGish and ResNeXt50 CNNs on the ALC dataset and applied global average pooling for classification. These CNNs, typically used for image classification, perform well for large-scale audio classification [10]. We also experiment with a VGGish model pre-trained for audio classification on the Google AudioSet dataset [40] and five different pooling techniques for classification [42]. Although the model is pre-trained on out-of-domain data, it performs well on audio classification in a weakly-labeled, unbalanced dataset [42] without overfitting. The architecture is promising for classifying speech audio in the ALC dataset, which is also unbalanced. To help prevent model overfitting, we employ different data augmentation techniques that previous pipelines have not explored.

We use a deep Convolutional Neural Network architecture for classifying sober vs. intoxicated speech. We explored two different CNN architectures for audio classification, namely VGGish and ResNeXt50, that use pooling and fully-connected (dense) layers for audio classification using *log Mel spectrograms* as input. *Log Mel spectrograms* have been used in a variety of deep speech classification models [5], [10], and they give an acoustic time-frequency representation of the spectral information in audio. Mel filter banks are applied to the power spectrum of audio to mimic the scaling of frequencies in the cochlea [30]. An overview of our system architecture is shown in Figure 13. Figures 14 and 15 illustrate the architectures of VGGish and ResNeXt50 models, respectively.

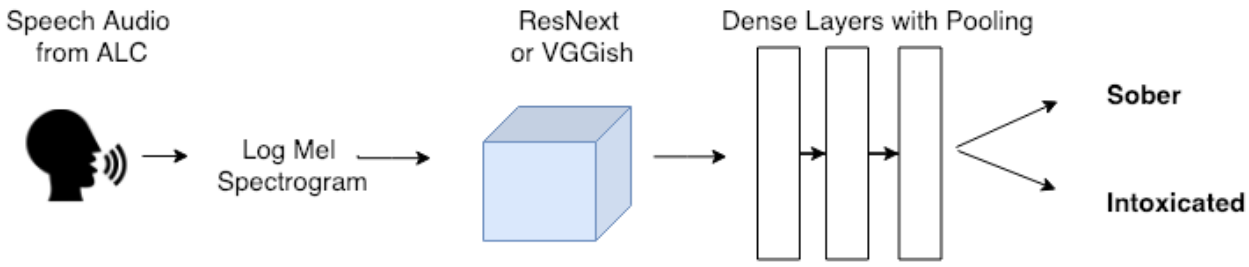


Figure 13: Proposed Architecture

Log Mel Spectrogram features are extracted for each audio segment in the ALC data. These features are input into either VGGish or ResNeXt50 and passed through dense and pooling layers for classification. Each audio segment is classified as either Sober or Intoxicated.

For each model, we experimented with different combinations of dense and pooling layers for classification. In ***section 5***, we refer to these model-specific pooling experiments as *VGG_pool* and *ResNeXt50_pool*, for the VGGish and ResNeXt50 networks, respectively. The following sections describe the VGGish and ResNeXt50 models in more detail, as well as the model-specific classification strategies we experimented with.

3.1 VGGish Model

The VGGish model [10] is a variant of the VGG [39] architecture. It has four groups of alternating convolutional and max pooling layers. Small, 3x3 filters are applied in each convolutional layer. The blocks of convolutional layers are followed by three Dense (fully-connected) layers followed by a softmax layer for classification. Figure 14 illustrates the VGGish architecture.

VGGish-specific classification involves inputting the output of VGGish through dense layers without pooling. The output of VGGish is fed through 2 fully-connected layers with ReLU activation and a final fully-connected softmax layer for classification.

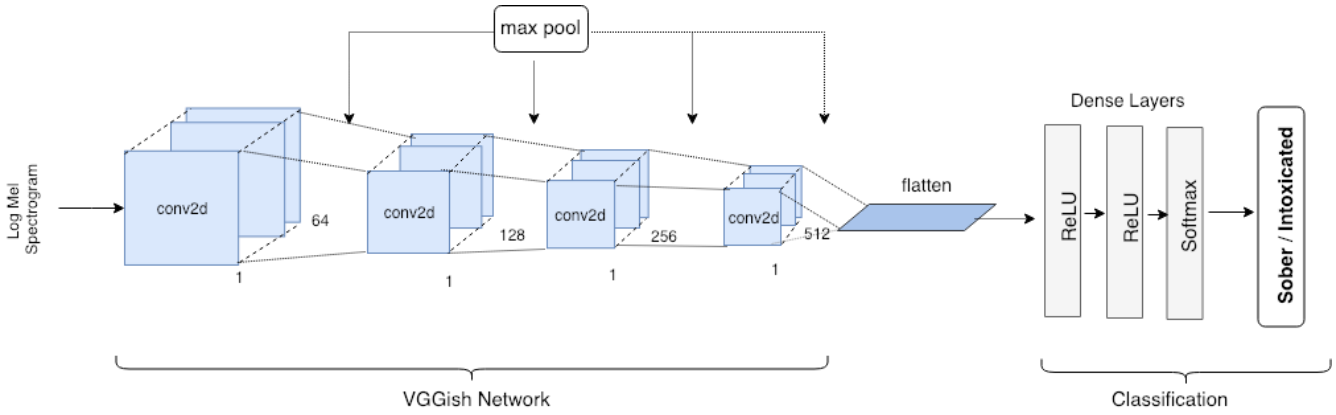


Figure 14: VGGish Architecture

The input to the VGGish model is a log Mel spectrogram representation of an audio segment. The VGGish model has four alternating convolutional and pooling layers which capture high-level temporal information of the audio signal. The VGGish-specific classification mechanism uses 3 Dense layers: the first two layers use ReLU activation, and the last layer uses softmax activation for final classification. Input audio is classified as either sober or intoxicated.

3.2 ResNeXt50 Model

The ResNeXt50 model [38] consists of stacked blocks of aggregated transformations. The architecture combines two different strategies: repeating layers of the same shape and a *split-transform-merge* technique. The technique of repeating layers is similar to the VGGish model. Split-transform-merge refers to dividing the input into low-dimensional embeddings, applying small-scale filters to each embedding, and combining all transformation outputs using summation or concatenation.

To prevent overfitting, we added 3 Dropout layers that alternate with a global average pooling layer and two dense layers: one with ReLU and another with sigmoid activation. Figure 15 illustrates a block of the architecture of the ResNeXt50 model, followed by the pooling and dense layers we employed for the ResNeXt50 network.

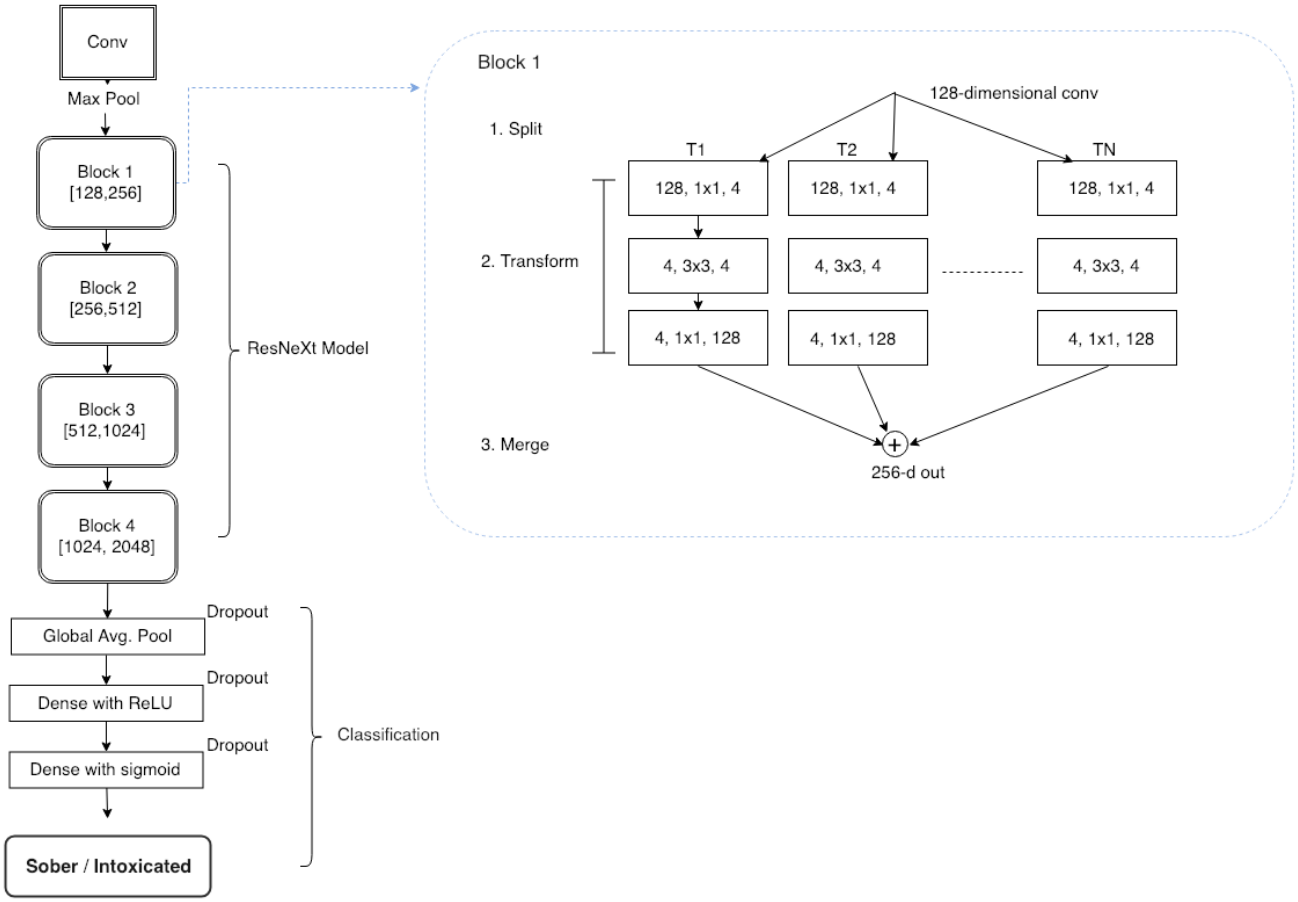


Figure 15: ResNeXt50 Architecture

The diagram on the left outlines the full stacked-block architecture of the ResNeXt50 model with the model-specific layers used for classification. The diagram on the right depicts the typical structure of a block within the ResNeXt50 model, highlighting the split-transform-merge technique each block employs. Each block has 1...N transformations that are concatenated.

3.3 Pooling Layers

The following sub-section describes the maximum pooling, average pooling, and attention-based pooling we experiment with.

3.3.1 Maximum Pooling

Maximum pooling functions such as the global max pooling layer in a Convolutional Neural Network (CNN) consists of a single Dense layer with sigmoid activation. The maximum prediction is used for classification.

3.3.2 Average Pooling

Average pooling is similar to maximum pooling, except a segment is classified based on the average of individual classification predictions.

3.3.3 Attention-based Pooling

Attention-based pooling, weights are computed over the input sequence and the final classification prediction of the instance is based on a weighted sum over the sequence [42]. Mirsamadi et al. [13] effectively used a similar local attention-based pooling strategy to compute a weighted average over RNN outputs. We experiment with single-level attention, multi-level attention, and feature-level attention pooling.

3.3.4 Single-level Attention Pooling

In the *single-level attention pooling* technique, a single attention mechanism computes weights over the input audio sequences. The single-level attention mechanism consists of 2 Dense layers. The first layer uses a sigmoid activation function, while the second applies a softmax activation function over the output of the first layer to compute attention weights.

3.3.5 Multi-level Attention

Multi-level attention technique consists of two attention-pooling layers and concatenates their outputs and applies a sigmoid activation function to compute attention weights [42].

3.3.6 Feature-level Attention

Finally, the *feature-level attention* technique consists of three dense, fully-convolutional layers employing different activation functions, namely linear, sigmoid, and ReLU. The feature-level attention mechanism also uses Dropout to prevent overfitting [31].

3.3.7 Decision Pooling

For both CNN networks, we experimented with using a *decision-pooling* strategy which consists of 3 Dense layers followed by one of five decision pooling techniques [42] for binary

classification. In ***Section 4***, we refer to the pooling technique of these experiments as **Decision Pooling**. An overview of the decision pooling technique is shown in Figure 16.

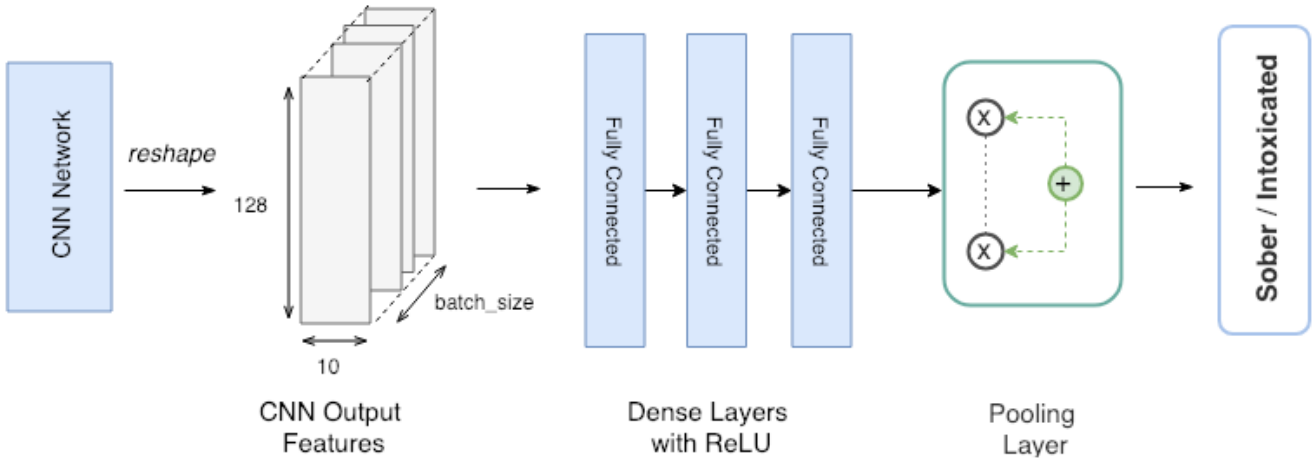


Figure 16: CNN + Pooling Layer

The diagram above illustrates the decision pooling and classification technique we experimented with [42]. The audio features extracted from a CNN feature extractor, either VGGish or ResNeXt50, are reshaped to have a shape of (10,128), or (16,128), respectively. The audio features are fed batch-wise into 3 fully-connected layers with ReLU activation. The outputs of the dense layers are pooled by either maximum pooling, average pooling, or attention-based pooling, before the audio segment is classified as either sober (1) or intoxicated (0). [42]

We use the VGGish model shown in Figure 14 or the ResNeXt50 model shown in Figure 15 to extract high-level bottleneck embeddings from the log Mel spectrogram representation of audio. The audio clips in the Alcohol Language Corpus range from 5 seconds to 60 seconds in length, resulting in variable-length output feature shapes. To account for variability in clip length and resulting feature set shape, we reshape each feature set output from VGGish to a shape of (10,128), zero-padding shorter clips and clipping longer segments to keep a constant shape of (10,128), and reshape each feature set output from ResNeXt50 to a shape of (16,128) using Reshape layer.

After post-processing the extracted features into equal-length, 128-dimensional feature vectors, the features are input into 3 Dense layers followed by a pooling mechanism to classify the input audio segment as either *Sober* or *Intoxicated*. The 3 fully connected-layers compute high-level embeddings from the features output from either VGGish or ResNeXt50. Pooling reduces the feature map of embeddings while retaining information of an activation of features [31].

Chapter 4. Experiments

4.1 Hardware and Software

Our experiments focused on how different data balancing techniques and pooling layers affect classification performance on the ALC dataset. Experimental setup is outlined as follows:

- All models are implemented in Keras 2.2.4 [32], Tensorflow-GPU 1.12.0 [32], and Python 3.6 [76].
- Feature preprocessing and data augmentation techniques are implemented using NumPy 1.16.1 libROSA 0.6.3 [29].
- Models are trained on NVIDIA Tesla V100 GPUs provided by the Worcester Polytechnic Institute (WPI) Academic Research Computing (ARC) Turing cluster [34].

4.2 Data Augmentation

Using the NumPy 1.16.1 [77] and the libROSA 0.6.3 [29] Python libraries, we implemented five data augmentation techniques to the training and validation data to increase robustness and balance the data. These techniques include adding noise, shifting the pitch, time-stretching audio samples, feature normalization, feature centering, horizontal flip and zoom.

The techniques applied to the dataset are explained in Table 4. These augmentation techniques attempt to make the model more robust to noisy data while increasing the number of intoxicated samples in the dataset. Noise, shift and stretch gave us the better results compared to the other augmentation methods.

Augmentation Technique	Description
Noise	Vector of random noise added to audio sample [78]
Shift	Audio samples are shifted forward by one time step [78]
Stretch (speed)	Time-stretch an audio sample by speeding it up or slowing it down at a fixed rate using libROSA [29]
Feature Centering	Subtracting the mean of the specific feature/variable from all the data instances to remove bias and have the mean set to zero
Feature Normalization	Scaling the features to be in the range of 0 and 1
Horizontal Flip	Reversing the columns of random images in the dataset [63]
Zoom	Zooming in random images in the dataset. Keras either adds new pixel values around the image or interpolates the image [63]

Table 4

Data augmentation Techniques

Figures 17 shows the noise, shift and stretch augmentation techniques. The first graph represents the original data before augmentation.

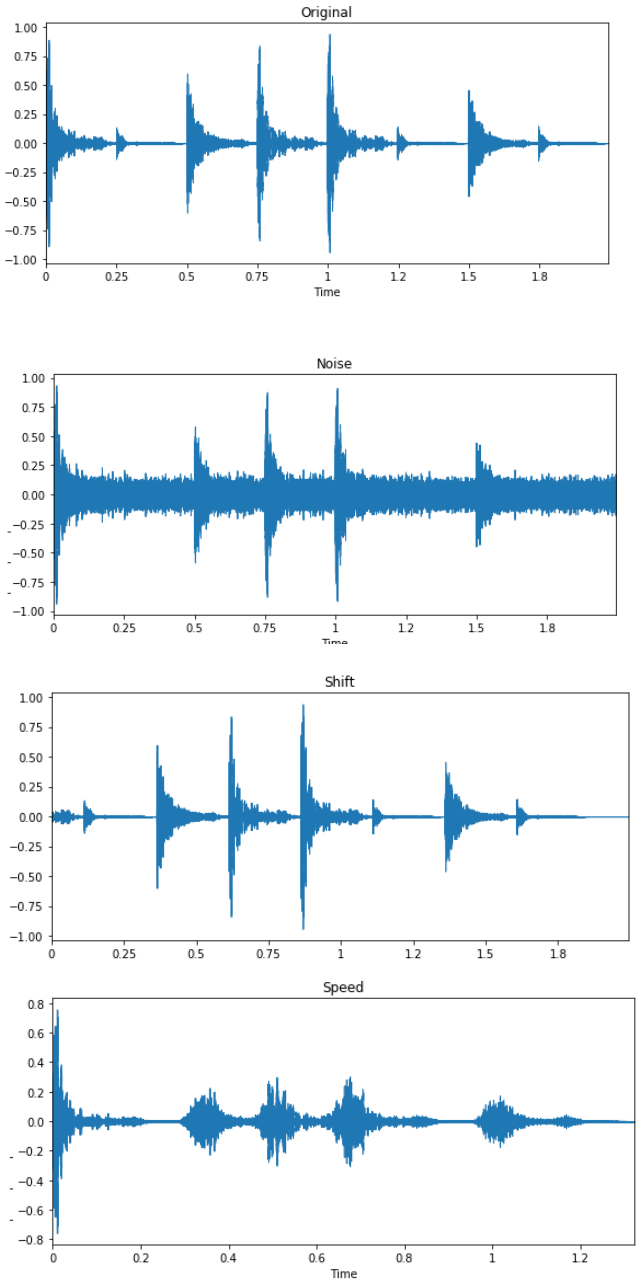


Figure 17: Augmentation Techniques

The first graph shows the original data, second graph shows added noise, third graph represents the data shifted backwards by 0.125 seconds, and third graph shows the audio at a slower speed (stretched out) [78]

4.3 Pilot Experiments: Hand Extracted Features

4.3.1 One Dimensional Features

We first extracted one dimensional features: MFCC, Chroma and Prosodic.

- **MFCC:** has been historically useful for speech analysis tasks [35]. MFCCs are short-term spectral features extracted from audio frames in overlapping Hamming windows. Cepstral features are extracted from each frame, and the Discrete Fourier Transform (DFT) is taken over each frame. After binning spectral information according to the Mel scale, a transform is applied to the information to de-correlate it and produces MFCCs.
- **Chroma:** features [26] model pitches that the human ear perceives. Pitch is separated into *tone height* and *chroma*, where tone height indicates how high or low a pitch is and chroma indicates the note of the pitch. There are 12 chroma values. Chroma features group together all spectral information of a particular chroma value in an audio segment. The Chroma representation of an audio segment can be computed from the log-frequency spectrogram of the audio.
- **Prosodic:** features are those relating to different acoustic qualities of speech, including tone, pitch, stress, fundamental frequency, and rhythm and are calculated over an entire speech segment [7]. They are useful for capturing differences between speech styles and languages.

4.3.2 Two Dimensional Features

The second approach was to extract two dimensional features: Scalogram and Log Mel Spectrogram. Log Mel Spectrogram has already been defined in ***section 1.4.4***. Scalogram is computed by applying independent wavelet filters at different time scales of the audio to extract multiscale features. Capturing the features at different time scales makes scalograms stable to time-warping and thus more robust to noise. Using scalograms with CNNs has been effective for acoustic scene classification and outperformed the traditional approach in [27]. A summary of the hand extracted features is presented in Table 5.

Feature Type	Feature Extraction Technique	Description
One Dimensional	MFCC	Mel-binned cepstral coefficients giving spectral audio information [7]
	Prosodic	Features relating to tone, stress, fundamental frequency of speech [7]
	Chroma	Model the 12 pitches perceived by the human ear [26]
Two Dimensional	Scalogram	Combination of wavelet filters applied at different time scales of audio, giving a multi-level representation of audio [27]
	Log Mel Spectrogram	Fourier Transform of signal in the Mel scale.

Table 5

Pilot Experiments: Audio Features

4.4 Evaluation Metric

We evaluated our models' performance using the same official metrics of the INTERSPEECH 2011 Challenge, namely Unweighted Accuracy (UA) and Unweighted Average Recall (UAR) [37]. TP, FP, TN, and FN represent the raw count of class-specific True Positives, False Positives, True Negatives, and False Negatives identified by the model. Table 6 provides the formula for each reported metric.

TP = True Positive	FP = False Positive
TN = True Negative	FN = False Negative

Evaluation Metric	Formula/Description
Unweighted accuracy (UA)	$\frac{TP + FP}{TP + FP + TN + FN}$
Recall	$\frac{TP}{TP + FN} = \frac{TP}{Total\ Actual\ Positive}$
Precision	$\frac{TP}{TP + FP} = \frac{TP}{Total\ Predicted\ Positive}$
F1 Score	$2 * \frac{Precision * Recall}{Precision + Recall}$
Unweighted Average Recall (UAR)	Mean of each class-specific recall scoreEg. Three classes have the recall values: R1, R2, R3 UAR = mean (R1 , R2 , R3)

Table 6

Evaluation metrics used to assess model performance

4.5 Pilot Experiments: Architecture

We experimented with three architectures: RNN-based architecture, combination of the two CNN and RNN architectures and CNN-based architecture.

4.5.1 Attention Layer

Attention mechanisms have been effective for speech classification architectures [22], [42], [13]. The attention mechanism consists of a feedforward layer which computes a vector of weights over the input audio sequence at a given time step. It uses information from the RNN hidden state of the previous time step to compute weights over the input. The weights indicate which parts of the input the model should focus on at a given time step [62]. The attention mechanism is especially useful when training on noisy sequences, because it guides the model to pay more attention to relevant parts of the input audio sequences [23]. We experiment with computing attention weights over the audio features output from VGGish before inputting them

to a bi-directional RNN, and we also experiment with computing attention over the outputs of the forward and backward LSTM layers before pooling them for classification.

4.5.2 RNN-Based Architecture

In this architecture, we combined a BiLSTM network with attention layer and different pooling layers. We also used different augmentation methods on the input data mentioned in ***section 4.2***. Table 7 represents the different experiments and their accuracy.

Augmentation Method	**Feature Extraction Technique**	**Architecture**	**UAR**
Noise	Prosodic	BiLSTM + attention layer + feature_level_attention_pooling	0.501
Shift	Chroma	BiLSTM + attention layer + decision_level_max_pooling	0.499
Shift	**MFCC**	**BiLSTM + attention layer + decision_level_max_pooling**	**0.502**
Shift	MFCC	multiBiLSTM + attention layer + decision_level_single_attention_pooling	0.500
Shift	MFCC	Attention layer + multiBiLSTM + feature_level_attention_pooling	0.493
Stretch	Mixture Features	BiLSTM + attention layer + decision_level_max_pooling	0.500

Table 7

Pilot Experiments: RNN-based architecture

As shown in table 7, shift augmentation method, MFCC features and the BiLSTM architecture combined with attention layer and decision level max pooling gives the best UAR of 0.502. Figure 18 represents RNN-based architecture.

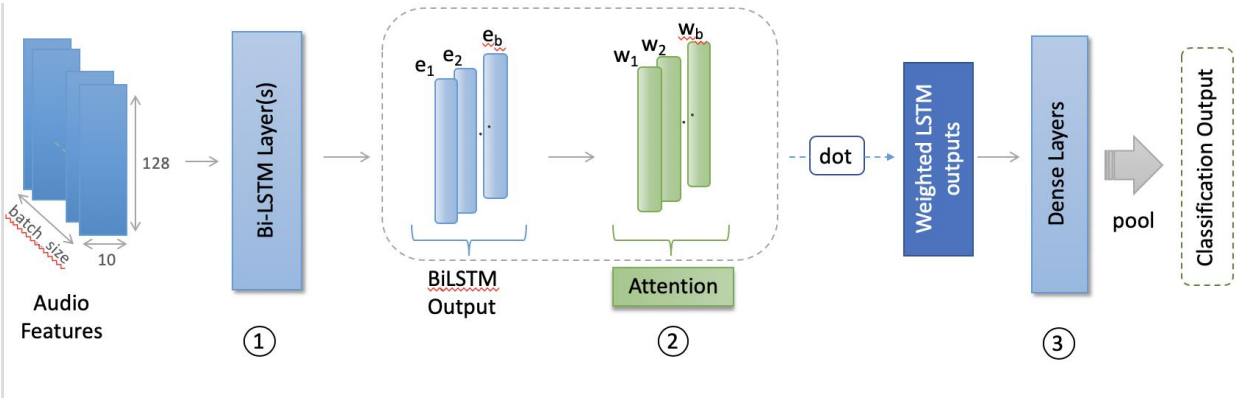


Figure 18: RNN-Based Intoxication Detection Architecture

RNN-based Architecture

4.5.3 CNN + RNN

In this architecture, instead of using hand extracted features only, we used CNN as an additional feature extractor for high-level descriptors and RNN for classification of the sober or intoxicated. For feature extraction we used fully connected layers (VGGish network) and different pooling layers. We used log Mel spectrogram features as input to the VGGish network as it performed better than scalogram.

After post-processing the output of VGGish features into equal-length, 128-dimensional feature vectors, the features are fed into a bi-LSTM model with attention and pooling layers. From the experiments in ***section 4.3.2*** we concluded that BiLSTM + attention layer + decision_level_max_pooling gives the best results, so we use this architecture for classification in the CNN+RNN experiments. We call this combination BiLSTM_ATT. Bi-directional Long Short Term Memory cells (bi-LSTMs) are used in speech classification tasks to learn high-level representations of acoustic features [5], [14]. The forward and backward layers allow the model to learn long-term context on top of the local acoustic information capture by a CNN [61]. Table 8 represents the best results of pilot experiments for CNN+RNN.

Augmentation Method + Hand Extracted Feature	**Feature Extraction Network**	**Classification Network**	**UAR**
Noise + Log Mel Spectrogram	**VGGish + feature_level_attention_pooling**	**BiLSTM_ATT**	**0.557**
Shift + Log Mel Spectrogram	VGGish + decision_level_single_attention_pooling	BiLSTM_ATT	0.522
Stretch + Log Mel Spectrogram	VGGish + decision_level_average_pooling	BiLSTM_ATT	0.534

Table 8

Pilot Experiments: CNN (feature extraction) + RNN (classification: Sober or Intoxicated)

Table 8 shows that applying the noise augmentation method on the data and passing hand extracted log Mel spectrogram to the VGGish network combined with the feature_level_attention_pooling layer and then passing the output of the feature extraction network to the RNN network gives us the best result. Figure 19 represents the discussed architecture.

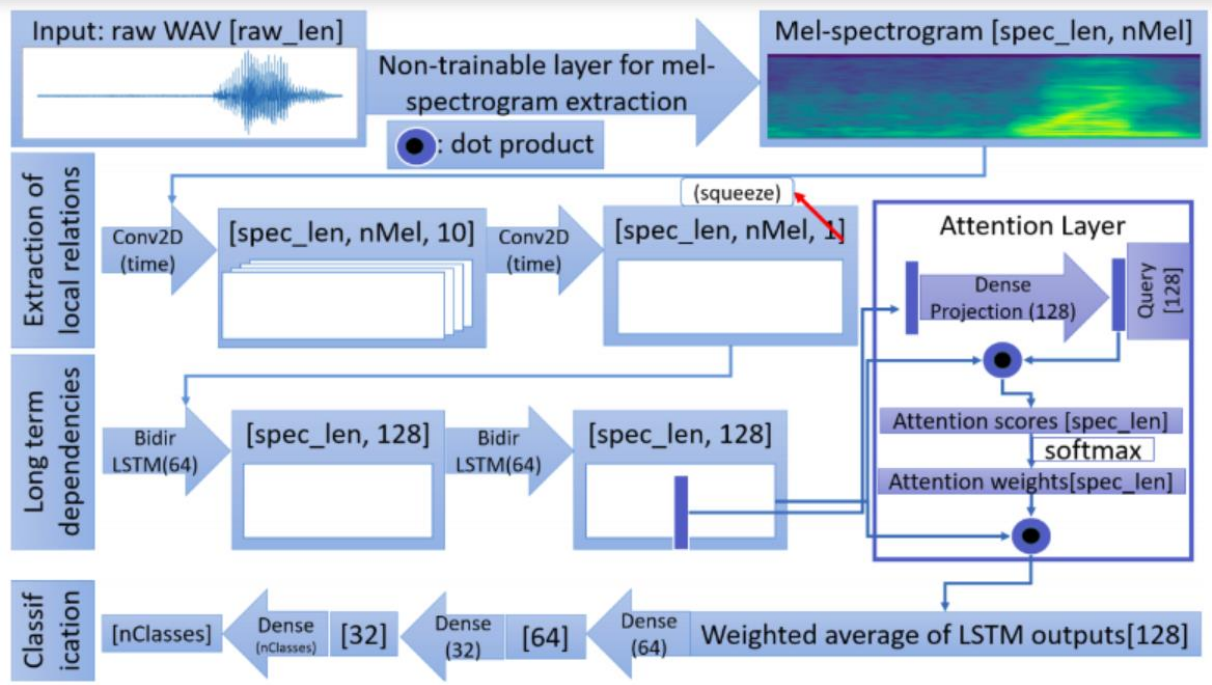


Figure 19: CNN + RNN Intoxication Detection Architecture

The CNN + RNN Architecture combined with pooling layers [81]

4.5.4 CNN-based architecture

Our third architecture used log Mel spectrogram features with a CNN-based architecture and different pooling methods. CNN-based architectures are efficient and effective for emotion detection and large-scale speech classification tasks [9], [10]. The CNN captures high-level temporal information from input audio sequences, requiring minimal feature engineering and performing well for audio classification [9]. The state-of-the-art algorithms we experimented with can all be found in table 9. We trained VGGish and ResNeXt50 CNN [38] on the ALC dataset with different data augmentation techniques such as noise, shift, and stretch. Noise and stretch data did not converge while training and the model was over-fitting. Figure 16 represents the CNN architecture with attention and pooling layers which is our chosen architecture for this paper. CNN-based networks had the best results out of the three approaches we experimented with. ***Section 5*** discusses these results in more detail.

Chapter 5. Results

We experimented with training on the VGGish and ResNeXt50 CNN [38] on the ALC dataset with different data augmentation techniques like noise, shift, and stretch. Noise and stretch data did not show convergence while training and the model was over-fitting.

Table 9 highlights the results we achieved using different CNNs, data augmentation types, and dense and pooling layers for classification on log Mel spectrogram input. *VGG_Pool* and *ResNext50_Pool* pooling refers to model-specific pooling strategies discussed in ***Section 3***. *Decision-based* pooling refers to the decision pooling discussed in ***Section 4***, with 3 fully connected layers and different pooling techniques. Overall, different augmentation techniques and pooling techniques did not impact model performance, so notable results are highlighted across all experiments.

Neural Network Architecture	**Augmentation Type**	**Pooling Type**	**UA**	**UAR**
VGGish	Shift	VGG_Pool	0.648	0.503
VGGish	Noise	Decision_Pool, Feature_Level_Attention	0.671	0.557
VGGish	Shift	Decision_Pool, Single_Level_Attention	0.656	0.522
VGGish	Stretch	Decision_Pool, Average_Pooling	0.663	0.534
ResNeXt50	Shift	ResNext50_Pool	0.653	0.500
ResNeXt50	**Shift**	**Decision_Pool, Feature_Level_Attention**	**0.682**	**0.592**

Table 9

UA and UARs for ResNeXt50 and VGGish networks trained on ALC

As shown in table 9, we achieved the best result with the ResNeXt50 model, using decision pooling with feature-level attention. This result was achieved using additional augmentation techniques on the log Mel spectrogram input, namely shifting, zooming, and horizontally flipping. The width_shift_range was 0.5 while the zoom_range = 0.2.

We also used *featurewise_center = True*; *featurewise_std_normalization = True* to achieve centering and normalization of data. These augmentation techniques are explained in more detail in ***section 4.2***. The best result was obtained at learning rate of *1e-4*. Adam optimizer was used while training with epsilon value set to 1e-8. We chose to experiment further with ResNeXt-50 because the results of the Pilot Experiments on pre-trained CNNs, as shown in table 11, indicate that the ResNeXt50 model was able to identify some alcoholized samples. It had the highest True Positive (TP) count (442) with respect to other networks, on the same sample size of 39808 samples.

The results for the Pilot Experiments described in ***Section 4*** are shown in Tables 9 and 10. Table 10 shows the results of using different features, namely Chroma, MFCC, Prosodic, and a combination of these features, with a BiLSTM network with attention and decision_level_max_pooling. As indicated by the UARs ranging from 0.499 to 0.502, this network did not train, so we did not expand upon its details.

Feature Type	Pooling Type	UAR
Chroma	Decision_level_max_pooling	0.499
MFCC	Decision_level_max_pooling	**0.502**
Prosodic	Decision_level_max_pooling	**0.502**
Mixture Features	Decision_level_max_pooling	0.500

Table 10

UA and UARs for ResNeXt50 and VGGish networks trained on ALC with different hand extracted features

Table 11 shows the results of using different pre-trained CNNs, including VGG16 and ResNeXt50, for classification. All of these received UARs around 0.50, indicating that they were overfitting to the training data and could not learn. The raw True Positive (TP) count, with respect to the Alcoholized class, is reported along with the UAR to give a clearer idea of the model's classification performance. Based off the results of these pilot experiments, we decided

to focus our work into training CNN networks with the different pooling strategies outlined in ***section 3***. BiLSTMs and Attention mechanisms were not improving our results, so we simplified the model and focused on data augmentation and normalization techniques to improve model training and help the models adapt to the ALC data.

Network	UAR	TP
Xception	**0.503**	144
InceptionV3	0.501	58
ResNeXt50	**0.503**	442
DenseNet121	0.500	4
MobileNetV2	0.500	0
VGG16	0.500	0

Table 11

Pilot Experiments: Sample results for different pre-trained CNN architectures

Chapter 6. Discussion

As the results in Table 9 indicate, most model configurations overfit the data. The best model configuration, which used the ResNeXt50 CNN with Decision Pooling and feature-level attention, could not achieve the classification accuracy and UAR of the baseline architecture in INTERSPEECH 2011. This model achieved a UAR 59.2% and an UA of 68.2%, which is lower than the baseline UAR of 65.9%. The ResNeXt50 model was chosen based on the results in Table 11. This table shows pre-trained CNN architectures, including ResNext-50, that were used for classification of ALC data. Many networks achieved UARs of 0.50, likely because the CNNs were trained on out-of-domain data and had to be tuned to classify ALC data. However, the ResNeXt-50 model yielded the highest count of True Positives, namely 442 samples, with respect to the alcoholized class, out of a sample size of 39808 samples. Thus, we chose to focus our experiments on training ResNeXt-50 on ALC data with different pooling strategies. The VGGish model was used for comparison, because it performed well on AudioSet data [42].

Aside from the ResNext-50 with decision pooling and feature level attention, the second-best result highlighted in Table 9 was achieved with the VGGish model using the same decision pooling technique. This model achieved a UA and UAR of 0.671 and 0.557, respectively. Normalization and image augmentation were also applied to this model. Aside from this result, different data augmentation techniques and pooling layers had little effect on model performance and did not assist in training. Although audio augmentation techniques helped to increase the proportion of positive (intoxicated) samples, the dataset was still imbalanced and we attribute it to the fact that we performed clipping of variable length audio samples to maintain uniform length of 10 seconds for each audio sample. Other pooling techniques, such as global and average pooling, may have added distortion and discarded original information about position of particular values in the audio feature vectors [31]. Moreover, a different set of parameters may have to be used for speech classification on the ALC dataset to help the VGGish and ResNeXt50 models extract features that are relevant for intoxication detection.

Chapter 7. Conclusion

Our work explores how CNN architectures like VGGish and ResNeXt50 can be used in the audio classification pipeline for the intoxication detection task. Our work establishes a solid baseline for further experimentation into CNN techniques for intoxication detection. Although many of the experiments we ran did not train, adding different augmentation and normalization techniques to the standalone CNN architecture helped with model training and improved UAR by 9.2%, compared to the worst-performing model in the table. The best result we achieved was using the ResNeXt50 model, using decision pooling with feature-level attention. The model achieved a UAR of 59.2%. This result is 9.2% higher than the results we were getting from standalone VGGish and ResNeXt50 pooling, which yielded UARs of 50.3% and 50.0%, respectively. The VGGish model with decision pooling and feature-level attention also received high results, yielding a UA of 65.1% and a UAR of 55.7%. These results for VGG and ResNeXt50 with decision Pooling were achieved using a variety of normalization and augmentation techniques, which indicates that further exploration into feature extraction and data balancing may help the model train.

7.1 Future Experiments

In the future, we can continue experimenting with hybrid RNN-based architectures and CNN-based architectures. RNN-based architectures capture long-term dependencies in audio and perform well in audio classification tasks [5], [13], [12]. CNN architectures have also been successful for audio classification [14], [10].

Moreover, we can continue building from the Pilot experiments we ran, using BiLSTMs, attention, and different feature extraction techniques. We can also look into using Gramian Angular Summation Fields (GASF) and Markov Transition Fields (MTF) for direct input to ResNext50 or VGGish. These feature types encode time series into images and have been useful for speech processing tasks [28].

Other experiments we can explore for future work are outlined below:

1. Current approach for data augmentation modifies the audio data. Data augmentation could be done by adding background noise to the audio without modifying the speaker audio itself.
2. Data oversampling techniques can be applied to balance the classes using SMOTE [19] and ADASYN [20].
3. To overcome the imbalance in class, auto-encoders could be tried for outlier detection of the alcoholized samples rather than employing data augmentation or data oversampling techniques.
4. Exploring a multi-class classification task with audio samples split into multiple classes based on BAC values.
5. Performing a regression task in which the model must predict speaker BAC given input audio.
6. Attention mechanisms have been useful for audio classification in an RNN-based network [22]. We can further explore using attention over audio features with a bi-directional RNN network.
7. We did not change the Sampling Rate (SR) of the audio we were using, keeping it at 44.1 kHz. Perhaps, a lower sampling rate could help the model learn by providing lower-dimensional audio input without discarding too much important information in the audio.

Bibliography

[1] ”Drunk Driving” NHTSA.gov. https://www.nhtsa.gov/risky-driving/drunk-driving. Accessed on May 3rd, 2019.

[2] ”Blood alcohol concentration (BAC) and the effects of alcohol”. SAHealth.gov. https://www.sahealth.sa.gov.au/wps/wcm/connect/public+content/sa+health+internet/health+topics/health+conditions+prevention+and+treatment/alcohol/blood+alcohol+concentration+bac+general+information . Accessed on Feb 16th, 2019.

[3] ”Impaired Driving: Get the Facts”. CDC.gov. https://www.cdc.gov/motorvehiclesafety/impaired_driving/impaired-drv_factsheet.html. Accessed on Feb 16th, 2019.

[4] Tisljar-Szab ´ o, Eszter, Ren ´ ata Rossu, Veronika Varga, and ´Csaba Pleh. ”The effect of alcohol on speech production.” ´Journal of psycholinguistic research 43, no. 6 (2014): 737-748.

[5] Berninger, Kim, Jannis Hoppe, and Benjamin Milde. ”Classification of Speaker Intoxication Using a Bidirectional Recurrent Neural Network.” In International Conference on Text, Speech, and Dialogue, pp. 435-442.Springer, Cham, 2016.

[6] Bone, Daniel, Matthew P. Black, Ming Li, Angeliki Metallinou, Sungbok Lee, and Shrikanth Narayanan. ”Intoxicated speech detection by fusion of speaker normalized hierarchical features and GMM supervectors.” In Twelfth Annual Conference of the International Speech Communication Association. 2011.

[7] Bocklet, Tobias, Korbinian Riedhammer, and Elmar Noth. ”Drink and Speak: On the automatic classification of alcohol intoxication by acoustic, prosodic and text-based features.” In Twelfth Annual Conference of the International Speech Communication Association. 2011.

[8] Schuller, Bjorn, Stefan Steidl, Anton Batliner, Florian ¨Schiel, and Jarek Krajewski. ”The INTERSPEECH 2011 speaker state challenge.” In Twelfth Annual Conference of the International Speech Communication Association. 2011.

[9] Wu, Huiyi, John Soraghan, Anja Lowit, and Gaetano DiCaterina. ”A deep learning method for pathological voice detection using convolutional deep belief networks.” In Interspeech 2018. 2018.

[10] Hershey, Shawn, Sourish Chaudhuri, Daniel PW Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal et al. ”CNN architectures for large-scale audio classification.” In 2017 ieee international conference on acoustics, speech and signal processing (icassp), pp. 131-135. IEEE, 2017.

[11] Han, Kun, Dong Yu, and Ivan Tashev. ”Speech emotion recognition using deep neural network and extreme learning machine.” In Fifteenth annual conference of the international speech communication association. 2014.

[12] Lee, Jinkyu, and Ivan Tashev. ”High-level feature representation using recurrent neural network for speech emotion recognition.” In Sixteenth Annual Conference of the International Speech Communication Association. 2015.

[13] Mirsamadi, Seyedmahdad, Emad Barsoum, and Cha Zhang. ”Automatic speech emotion recognition using recurrent neural networks with local attention.” In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2227-2231. IEEE, 2017.

[14] Trigeorgis, George, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A. Nicolaou, Bjorn Schuller, and Stefanos Zafeiriou. ”Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network.” In 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp. 5200-5204. IEEE, 2016.

[15] Tang, Dengke, Junlin Zeng, and Ming Li. ”An Endto-End Deep Learning Framework with Speech Emotion Recognition of Atypical Individuals.” Proc. Interspeech 2018 (2018): 162-166.

[16] Biadsy, Fadi, William Yang Wang, Andrew Rosenberg, and Julia Hirschberg. ”Intoxication detection using phonetic, phonotactic and prosodic cues.” In Twelfth Annual Conference of the International Speech Communication Association. 2011.

[17] Fukuda, Takashi, Raul Fernandez, Andrew Rosenberg, Samuel Thomas, Bhuvana Ramabhadran, Alexander Sorin, and Gakuto Kurata. ”Data Augmentation Improves Recognition of Foreign Accented Speech.” Proc. Interspeech 2018 (2018): 2409-2413.

[18] Toman, Markus, Geoffrey S. Meltzner, and Rupal Patel. ”Data Requirements, Selection and Augmentation for DNN-based Speech Synthesis from Crowdsourced Data. ”Proc. Interspeech 2018 (2018): 2878-2882.

[19] Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. ”SMOTE: synthetic minority over-sampling technique.” Journal of artificial intelligence research 16 (2002): 321-357.

[20] He, Haibo, Yang Bai, Edwardo A. Garcia, and Shutao Li. ”ADASYN: Adaptive synthetic sampling approach for imbalanced learning.” In 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 1322-1328. IEEE, 2008.

[21] Amodei, Dario, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper et al. ”Deep speech 2: End-to-end speech recognition in english and mandarin.” In International conference on machine learning, pp. 173-182. 2016.

[22] Pei, Wenjie, Tadas Baltrusaitis, David MJ Tax, and LouisPhilippe Morency. ”Temporal attention-gated model for robust sequence classification.” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6730-6739. 2017.

[23] Lin, Min, Qiang Chen, and Shuicheng Yan. ”Network in network.” arXiv preprint arXiv:1312.4400 (2013).

[24] Kong, Qiuqiang, Yong Xu, Wenwu Wang, and Mark D. Plumbley. ”Audio set classification with attention model: A probabilistic perspective.” In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 316-320. IEEE, 2018.

[25] Yu, Changsong, Karim Said Barsim, Qiuqiang Kong, and Bin Yang. ”Multi-level attention model for weakly supervised audio classification.” arXiv preprint arXiv:1803.02353 (2018).

[26] Muller, Meinard. ”Short-Time Fourier Transform and Chroma Features.”

[27] Ren, Z., Pandit, V., Qian, K., Yang, Z., Zhang, Z. and Schuller, B., 2017, December. Deep sequential image features on acoustic scene classification. In Proc. DCASE Workshop, Munich, Germany (pp. 113-117).

[28] Wang, Zhiguang, and Tim Oates. ”Imaging time-series to improve classification and imputation.” In Twenty-Fourth International Joint Conference on Artificial Intelligence. 2015.

[29] McFee, Brian, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. ”librosa: Audio and music signal analysis in python.” In Proceedings of the 14th python in science conference, pp. 18-25. 2015.

[30] Gibson, James, Maarten Van Segbroeck, and Shrikanth S. Narayanan. ”Comparing time-frequency representations for directional derivative features.” In Fifteenth Annual Conference of the International Speech Communication Association. 2014.

[31] Choi, Keunwoo, George Fazekas, and Mark Sandler. ”Automatic tagging using deep convolutional neural networks.” arXiv preprint arXiv:1606.00298 (2016).

[32] Girija, Sanjay Surendranath. ”Tensorflow: Large-scale machine learning on heterogeneous distributed systems.” Software available from tensorflow. org (2016).

[33] Chollet, Franc¸ois. ”keras. GitHub repository.” https://github. com/fchollet/keras. Accessed on Jan 25th, 2019.

[34] ”WPI - Academic & Research Computing”. ARC.WPI.edu. https://arc.wpi.edu/computing/hpc-clusters/. Accessed on March 4th, 2019.

[35] Logan, Beth. ”Mel Frequency Cepstral Coefficients for Music Modeling.” In ISMIR, vol. 270, pp. 1-11. 2000.

[36] Hannun, Awni, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger et al. ”Deep speech: Scaling up end-to-end speech recognition.” arXiv preprint arXiv:1412.5567 (2014).

[37] Schuller, Bjorn, Stefan Steidl, Anton Batliner, Florian Schiel, Jarek Krajewski, Felix Weninger, and Florian Eyben. ”Medium-term speaker states—A review on intoxication, sleepiness and the first challenge.” Computer Speech Language 28, no. 2 (2014): 346-374.

[38] Xie, Saining, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. ”Aggregated residual transformations for deep neural networks.” In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1492-1500. 2017.

[39] Simonyan, Karen, and Andrew Zisserman. ”Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:1409.1556 (2014).

[40] Gemmeke, Jort F., Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. "Audio set: An ontology and human-labeled dataset for audio events." In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 776-780. IEEE, 2017.

[41] Schiel, Florian, Christian Heinrich, and Sabine Barfusser. "Alcohol language corpus: the first public corpus of alcoholized German speech." Language resources and evaluation 46, no. 3 (2012): 503-521.

[42] Kong, Qiuqiang, Yong Xu, Wenwu Wang, and Mark D. Plumbley. "Audio set classification with attention model: A probabilistic perspective." In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 316-320. IEEE, 2018.

[43] Giannoulis, Dimitrios, Emmanouil Benetos, Dan Stowell, Mathias Rossignol, Mathieu Lagrange, and Mark D. Plumbley. "Detection and classification of acoustic scenes and events: An IEEE AASP challenge." In 2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, pp. 1-4. IEEE, 2013.

[44] "Gaussian Mixture Models Explained" towardsdatascience.com. https://towardsdata science.com/gaussian-mixture-models-explained-6986aaf5a95. Accessed on March 24th, 2020.

[45] "Introduction to Hidden Markov Models" towardsdatascience.com. https://towardsdata science.com/introduction-to-hidden-markov-models-cd2c93e6b781. Accessed on March 24th, 2020.

[46] "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way" towards datascience.com. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. Accessed on March 24th, 2020.

[47] "Getting to Know the Mel Spectrogram" towardsdatascience.com. https://towardsdata science.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0. Accessed on March 24th.

[48] "Low-Level Feature Descriptors" lesliesikos.com. https://www.lesliesikos.com/low-level-feature-descriptors/. Accessed on March 24th, 2020.

[49] "Support Vector Machine — Introduction to Machine Learning Algorithms" towardsdata science.com. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. Accessed on March 24th, 2020.

[50] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. “SMOTE: synthetic minority over-sampling technique.” Journal of artificial intelligence research 16 (2002): 321-357.

[51] Cui, Zhiyong, et al. "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction." arXiv preprint arXiv:1801.02143 (2018).

[52] “LSTM’s and GRU’s as a solution” towardsdatascience.com. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21. Accessed on March 25th, 2020.

[53] “Voice Analytics vs. Speech Analytics: Whats the Difference?” rankminer.com. https://www.rankminer.com/post/voice-analytics-vs-speech-analytics-difference. Accessed on April 3rd, 2020.

[54] “Cepstrum Analysis” mathworks.com. https://www.mathworks.com/help/signal/ug/cepstrum-analysis.html. Accessed on April 3rd, 2020.

[55] “Cepstral Analysis of Speech” amrita.edu. http://vlab.amrita.edu/?sub=3&brch=164&sim=615&cnt=1. Accessed on April 3rd, 2020.

[56] “The dummy’s guide to MFCC” medium.com. https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd. Accessed on April 3rd, 2020.

[57] “The use of Praat in corpus research” fon.hum.uva.nl. http://www.fon.hum.uva.nl/paul/papers/PraatForCorpora2.pdf. Accessed on April 3rd, 2020.

[58] “Deep Dive into Bidirectional LSTM” i2tutorials.com. https://www.i2tutorials.com/technology/deep-dive-into-bidirectional-lstm/. Accessed on April 3rd, 2020.

[59] Abu-El-Haija, Sami, et al. "Youtube-8m: A large-scale video classification benchmark." arXiv preprint arXiv:1609.08675 (2016).

[60] Gemmeke, Jort F., et al. "Audio set: An ontology and human-labeled dataset for audio events." 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2017.

[61] Michon, Elise, Minh Quang Pham, Josep Crego, and Jean Senellart. "Neural network architectures for Arabic dialect identification." In Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018), pp. 128-136. 2018.

[62] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

[63] "How to Configure Image Data Augmentation in Keras" machinelearningmastery.com. https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/. Accessed on April 3rd, 2020.

[64] "What is machine learning?" technologyreview.com. https://www.technologyreview.com /s/612437/what-is-machine-learning-we-drew-you-another-flowchart/. Accessed on April 6th, 2020.

[65] Rabiner, Lawrence R. "A tutorial on hidden Markov models and selected applications in speech recognition." Proceedings of the IEEE 77.2 (1989): 257-286.

[66] Gales, Mark, and Steve Young. "The application of hidden Markov models in speech recognition." Foundations and Trends® in Signal Processing 1.3 (2008): 195-304.

[67] Stuttle, Matthew Nicholas. A Gaussian mixture model spectral representation for speech recognition. Diss. University of Cambridge, 2003.

[68] Muthusamy, Hariharan, Kemal Polat, and Sazali Yaacob. "Improved emotion recognition using gaussian mixture model and extreme learning machine in speech and glottal signals." Mathematical Problems in Engineering 2015 (2015).

[69] Vyas, Manan. "A Gaussian mixture model based speech recognition system using MATLAB." Signal & Image Processing 4.4 (2013): 109.

[70] Sinith, M. S., et al. "Emotion recognition from audio signals using Support Vector Machine." 2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS). IEEE, 2015.

[71] Valstar, Michel, et al. "Avec 2016: Depression, mood, and emotion recognition workshop and challenge." Proceedings of the 6th international workshop on audio/visual emotion challenge. 2016.

[72] Eyben, Florian, Martin Wöllmer, and Björn Schuller. "Opensmile: the munich versatile and fast open-source audio feature extractor." Proceedings of the 18th ACM international conference on Multimedia. 2010.

[73] ”Audio and Image Features used for CNN” medium.com. https://medium.com/datadriveninvestor/audio-and-image-features-used-for-cnn-4f307defcc2f. Accessed on April 6th, 2020.

[74] “Simple Convolutional Neural Network for Genomic Variant Calling with TensorFlow” towardsdatascience.com. https://towardsdatascience.com/simple-convolution-neural-network-for-genomic-variant-calling-with-tensorflow-c085dbc2026f. Accessed on April 6th, 2020.

[75] Hall, Mark, et al. "The WEKA data mining software: an update." ACM SIGKDD explorations newsletter 11.1 (2009): 10-18.

[76] Van Rossum, Guido, and Fred L. Drake Jr. Python tutorial. Vol. 620. Amsterdam: Centrum voor Wiskunde en Informatica, 1995.

[77] Oliphant, Travis E. A guide to NumPy. Vol. 1. USA: Trelgol Publishing, 2006.

[78] “Data Augmentation for Audio” medium.com. https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6. Accessed on April 6th, 2020.

[79] ”Speech Recognition — GMM, HMM” medium.com. https://medium.com/@jonathan_hui/speech-recognition-gmm-hmm-8bb5eff8b196. Accessed on April 6th, 2020.

[80] Gajšek, R., F. Mihelič, and S. Dobrišek. "Speaker state recognition using an HMM-based feature extraction method." Computer Speech & Language 27.1 (2013): 135-150.

[81] “Recognizing Speech Commands Using Recurrent Neural Networks with Attention” towardsdatascience.com. https://towardsdatascience.com/recognizing-speech-commands-using-recurrent-neural-networks-with-attention-c2b2ba17c837. Accessed on April 6th, 2020

Appendix: Additional Experiments

Table 12 represents some of the other pilot experiments we performed using CNN as a feature extractor and RNN for classification. In these experiments we used a Bi-SLTM network and an attention layer followed by decision_level_max_pooling. However, these architectures did not perform very well. We called this architecture BiLSTM_ATT. However, we tested with different architectures for feature extraction.

Augmentation Method + Hand Extracted Feature	Feature Extraction Network	Classification Network	UAR
Noise + Log Mel Spectrogram	VGGish + decision_level_max_pooling	BiLSTM_ATT	0.504
	VGGish + decision_level_average_pooling	BiLSTM_ATT	0.507
	VGGish + decision_level_single_attention_pooling	BiLSTM_ATT	0.501
	VGGish + decision_level_multi_attention_pooling	BiLSTM_ATT	0.504
Shift + Log Mel Spectrogram	VGGish + decision_level_max_pooling	BiLSTM_ATT	0.500
	VGGish + decision_level_single_attention_pooling	BiLSTM_ATT	0.494
	VGGish + feature_level_attention_pooling	BiLSTM_ATT	0.519
	VGGish + decision_level_multi_attention_pooling	BiLSTM_ATT	0.500
Stretch + Log Mel Spectrogram	VGGish + decision_level_single_attention_pooling	BiLSTM_ATT	0.503
	VGGish + decision_level_multi_attention_pooling	BiSLTM_ATT	0.508
	VGGish + feature_level_attention	BiLSTM_ATT	0.500

Table 12

Additional CNN + RNN Experiments