Automatic Urban Sound Classification
Using Feature Learning Techniques

Christopher B. Jacoby

Submitted in partial fulfillment of the requirements for the
Master of Music in Music Technology
in the Department of Music and Performing Arts Professions
Steinhardt School
New York University
Advisor: Dr. Juan P. Bello

April 28, 2014
Abstract

Automatic sound classification is a burgeoning field in audio informatics, which has been growing in parallel to developments in machine learning and urban informatics. In this thesis, we define two primary bottlenecks to research in automatic urban sound classification: the lack of an established taxonomy, and the meager supply of annotated real-world data. We begin by assembling a new taxonomy of urban sounds to create a foundation for future work in urban sound classification, which is an evolution on previous perceptually-focused environmental sound taxonomies. We describe the process of curating a sizable dataset of real-world acoustic sounds collected from the Freesound archive. The completed dataset, aptly dubbed UrbanSound, contains 27 hours of audio, with 18.5 hours of annotated events across 10 classes. We further prepare a subset of the dataset which is separated into 10 folds for cross-validation.

We begin analysis of our dataset with a baseline classification system using the off-the-shelf tools Essentia and Weka. Our baseline approach, based on a typical MFCCs and a Support Vector Machine with Gaussian kernel, gets just over 70% overall classification accuracy. We then take an established algorithm for feature learning, which has been applied effectively in image recognition and music informatics, and apply it to the classification of urban environmental sounds. We compare the results of using of Spherical k-Means to traditional k-Means, and additionally compare the results of those to static (not learned) random bases in order to prove that our feature learning algorithm is learning useful features. While our feature learning system does not outperform our baseline system overall, when we carefully review the results at a per-class level, we discover that our system is indeed performing better than the baseline for many of the classes, and in some cases much better. The feature learning system improves the results of the background classes over the random bases, and therefore some useful features are learned by the system.
Acknowledgements

The author wishes to thank the following people:

Justin Salamon, Juan Bello, Tae Hong Park, Tlacael Esparza, Eric Humphrey, whose collaboration and guidance made this work possible.


Friends, etc: Ross, Ross, Bryan, Chantae, Kit, Mog, Corey, Lynda, Robert, Rowan, the Bouwerie Boys, Braintrust, Skip Wilkins, Larry Stockton, Jennifer Kelly, Ismail Jouny, Oscar Bettison, Dan Trueman, and Jeff Snyder.

And finally: Meghan, Mom & Dad, and the rest of the family.
## Contents

1 Introduction 7

2 A Taxonomy of Urban Sounds 11
   2.1 Prior Work ............................................. 11
      2.1.1 Perception of Environmental Sounds ............... 11
      2.1.2 Environmental Sound Taxonomies .................. 12
   2.2 Process .................................................. 13
      2.2.1 Freesound Tags .................................... 13
      2.2.2 Sound Walks ....................................... 14
      2.2.3 311 Noise Complaints .............................. 15
   2.3 The Compiled Taxonomy ................................ 17

3 Dataset Creation 19
   3.0.1 Data From Freesound ............................... 19
   3.0.2 Phase 0: Download & Collect Raw Data ............. 21
   3.0.3 Phase 1: Select the Files ........................... 21
   3.0.4 Phase 2: Annotate the Files ......................... 22
   3.0.5 Phase 3: Prepare a Canonical Dataset ............... 23
   3.0.6 Final Dataset Statistics ............................. 26

4 Baseline System 28
   4.1 System Overview ......................................... 28
   4.2 Baseline Results ........................................ 31

5 Feature Learning System 35
   5.1 Background ............................................. 35
   5.2 Analysis Pipeline ....................................... 38
      5.2.1 Engineered Features .............................. 38
      5.2.2 Learned Features ................................. 42
      5.2.3 Classification .................................... 44
## List of Figures

1. Urban Sound Taxonomy ................................................. 18
2. Top Freesound search results for “City” and “Urban.” ........ 20
3. Raw Data Summary .................................................... 21
4. Dataset Creation Process ............................................. 22
5. Accuracy varying maximum slice duration from 10s-1s. ....... 25
6. Occurrence Duration Per Class ........................................ 27
7. Baseline Block Diagram ................................................ 28
8. Accuracy over varying number of Mel filters. .................... 30
9. Accuracy over varying number of MFCC coefficients. ........ 31
10. Classification accuracy for each class, by foreground (FG) and background (BG), and together. ...................... 32
11. Feature Engineering Pipeline ......................................... 39
12. Feature Learning Pipeline ............................................. 39
13. Feature Learning Block Diagram ..................................... 46
14. Overall Feature Learning Results .................................... 48
15. Feature Learning System Accuracy per Class ..................... 50
16. Classification accuracy by foreground (FG) & background (BG) for various parameters. ................................. 53
1 Introduction

Buses whir and screech as they stop at every block to pick up and drop off passengers. The air rumbles as planes fly overhead. Birds chirp, dogs bark, cars honk, and sirens wail. Sound and noise permeate the daily life of the city dweller - there is no escape. High population density creates an onslaught of mechanical and human noises created by day to day activities, and the services required to keep that population functioning and happy.

The analysis and understanding of sound in the urban environment is very important to the growth of cities and populations of the future. Studies have shown a direct impact on health and child development from noise pollution [Stansfeld and Matheson, 2003, Ising et al., 2004, Waye et al., 2001]. As populations continue to grow around the world, so will the cities they live in; urban noise will continue to create a bigger and bigger health crisis globally. It is our responsibility as researchers and scientists to understand these problems so that solutions can be created. Any solution proposed will need to maximize success and minimize economic impact in order to be practically implemented. Therefore it is important to carefully analyze the existing challenges to discover the most pervasive problems.

Urban sound research is still in its infancy, but with the recent growth of big data and urban data analysis, research has accelerated. Globally, sonic analysis of urban environments is facilitated by multi-modal sensor networks in projects such as Sensor City in the Netherlands [Daniel Steele, 2013]. The Sensor City project covers the city of Assen in the Netherlands with a network of regularly placed sensors, which record data about sound, weather, time, and more. It is one of the first installations of a system to analyze urban acoustic environments on such a scale. In New York City, the Citygram project is a similar initiative, which intends to offer a low-cost sensor network for analyzing urban environments in various ways [Park et al., 2013].

The sound of a place can be called its “soundscape” [Brown et al., 2011]. There is significant work towards understanding the concept of soundscape, and the human perception of soundscapes. Comparatively little computational re-
search has been done in the analysis of urban soundscapes, and a majority of the work in this area has been focused on the analysis of scenes, instead of the identification of specific sound sources in those scenes [Giannoulis et al., 2013a]. However, a large body of work exists in the related areas of speech and music [Hamel et al., 2013]. Many of the techniques from these related fields remain applicable to the analysis of urban sounds.

The focus of this thesis is the automatic categorization of urban environmental sounds using a combination of audio informatics and machine learning techniques. An automatic categorization process takes a segment of audio, and returns the “classes” or “labels” present in the segment. Automatic categorization is a subset of “machine hearing”, as defined by Richard Lyon [Lyon, 2010], and has applications in audio informatics research, such as instrument identification, space identification, etc. Unlike those tasks, however, we are concerned with labeling sound in urban environments, which are rife with diverse human and machine sounds. These sounds offer a number of analysis challenges; they are timbrally diverse, are likely to have significant background noise, and have no macro level temporal organization, unlike music. Automatic classification of urban sounds has countless possible applications in urban informatics research, including: analyzing noise pollution, studying traffic patterns, emergency vehicle tracking, and countless others. The same techniques are also useful in other fields such as robotics and video analysis, where sound is part of a greater multi-modal sensing system.

There are three overarching problems in the analysis of urban environmental sounds: (1) the lack of a standardized taxonomy for the classification of urban sounds; (2) the lack of annotated urban sound event data; and (3) a lack of robust features for identifying sounds in noisy urban environments. In this thesis, we propose first steps to solve these challenges. Firstly, we propose a taxonomy of urban sounds to create a common vocabulary for research in urban soundscapes. Secondly, we describe our process of creating an large annotated dataset, dubbed
UrbanSound, with source audio taken from the Freesound\(^1\) web archive. To our knowledge, this dataset is the largest free dataset of annotated urban sounds which is publicly available. We offer baseline results on this system, using off-the-shelf software packages, which allow us to explore the challenges presented by this new dataset. Finally, we explore contemporary approaches using feature learning to try to learn features robust to noise.

This project is an opportunity to advance the state of the art in automatic categorization of urban sounds with developments in feature learning which have recently been applied to similar audio problems [Coates and Ng, 2012, Dieleman and Schrauwen, 2013, Hamel and Eck, 2010]. In particular, we aim to use the techniques outlined by Coates and Ng in regards to feature learning using spherical k-Means, and to replicate part of the work by Dieleman in using learned Spherical k-Means features for audio classification [Coates and Ng, 2012, Dieleman and Schrauwen, 2013]. For this we implemented a one-layer feature learning system using spherical k-Means, attempting to replicate the results produced by Dieleman for our particular task [Dieleman and Schrauwen, 2013]. We compared those results with engineered features composed of simple time-frequency representations, including the Mel-spectrogram and MFCCs (the baseline).

As with all research, this thesis is a collaborative project with roots in many different places. This project began as part of a greater collaboration at NYU between the Steinhardt Music Technology department and the NYU Center for Urban Science and Progress. Ultimately, the algorithms produced in this research are intended to be deployed in tandem with the Citygram/Sound Project sensors as they are deployed around the city, in order to provide real-time analysis of the New York City soundscapes. This work began as a class project with Tlacael Esparza for Music Information Retrieval, taught by Juan Bello. For that project, we performed unsupervised clustering of MFCCs using simple k-Means. In this thesis, the taxonomy, dataset, and baseline approaches were created in collaboration with Justin Salamon under the advisement of Juan Bello. The further exploration on

\(^1\)http://www.freesound.org
feature learning and spherical k-means is my work alone, under the advisement of Juan Bello.

The remainder of this document is structured as follows. In Section 2, we look at prior taxonomies of environmental sound, and describe our procedure for creating a new taxonomy of urban sounds. In Section 3, present a new dataset of annotated urban sounds, and enumerate the steps in creating that dataset. In Section 4, we create a baseline analysis system for our dataset, and analyze the initial features and challenges that it demonstrates in our dataset. Section 5 describes an exploration into feature learning as an iteration on the baseline. Here we present and discuss the results of our experiments. Finally, in Section 6, we review and present conclusions and future work.
2 A Taxonomy of Urban Sounds

A “taxonomy”, as defined by the Oxford English Dictionary [Oxford University Press, 2014], is “a classification of something; a particular system of classification.” The taxonomy is an important tool for facilitating automatic classification research; it establishes precisely what is being classified, the scope of each class, and the relationships between the classes. As we are interested in facilitating a community research effort towards urban sound analysis and classification, we began our work by creating a taxonomy of urban sounds, with the aim of establishing groundwork for future research in this area.

We designed our taxonomy with the following requirements in mind: (1) it should factor in previous research and previously proposed taxonomies; (2) it should be as detailed as possible, specifying low-level sound sources, such as “dog bark”, “car brakes”, or “jackhammer”; and (3) it should focus on sounds relevant to urban sound research. In this section, we take a brief look at previous work on taxonomies of environmental sound which influenced the creation of our own urban sound taxonomy. We then describe the results of several studies we performed which influenced the preparation of our taxonomy, and conclude this section with our compiled Taxonomy of Urban Sounds.

2.1 Prior Work

2.1.1 Perception of Environmental Sounds

We begin by looking at the literature on human perception—the way that we hear and distinguish individual events and sounds in everyday listening. The perception of environmental sounds have been the focus of soundscape research for several decades. William Gaver describes perceptual analysis of sounds in terms of “everyday listening”, which is defined in relation to Pierre Schaffer’s mode of “musical listening” [Gaver, 1993]. In everyday listening, the listener acquires relevant information about the environment he or she is experiencing, such as the size of the space they are in, or the presence of an approaching vehicle. In contrast, the
focus of musical listening is on the characteristics of the sound itself.

Another important work in perception is the “Categorization of Environmental Sounds”, in which Gustavino describes the results of a perceptual study on how people perceive environmental sounds [Guastavino, 2007]. Gustavino’s study focuses on the perception of specifically urban sounds, established with conclusions from free-response perceptual studies. She concludes that the semantic features of urban sounds are perceptually more relevant to listeners than abstracted stimuli.

2.1.2 Environmental Sound Taxonomies

Various taxonomies of environmental sound have been proposed since the 1970s. Each author has his or her own focuses in their taxonomies, however proposed taxonomies to date tend to have one primary element in common, which Gustavino enumerates with her perceptual study: the distinction between sounds relating to the presence or lack of human activity. One of the first taxonomies of environmental sound was proposed by Schafer in [Schafer, 1977], which divides sounds at the top level into six categories: “natural”, “human”, “society”, “mechanical”, “silence”, and “indicators”. This taxonomy forms the basis off which many later taxonomies iterate. Following Schafer, the Gaver taxonomy is one of the most referenced taxonomies related to environmental sound. However, Gaver’s work defines sound events by the materials that interact during the event that caused the sound and the method of interaction between the materials, not in terms of the semantic event itself, as Gustavino specifies is perceptually relevant.

By far the most comprehensive study of “soundscape” and presentation of a related taxonomy to date is the work by Brown [Brown et al., 2011]. Brown’s taxonomy splits from the root node, “The Acoustic Environment”, into “indoor” and “outdoor”. The taxonomy is then divided in a parallel way to Gustavino’s: both split from “sounds generated by human activity” and “sounds not generated by human activity”. Brown’s taxonomy further divides sounds in a hierarchical system, where the levels are split into “places”, “categories of sound sources”, and “sound sources”. “Sound sources” are the “leaves” of the tree, although many of
the leaves are not equally low-level, conceptually speaking; the leaves “fireworks” or “speech” are significantly more specific than the leaves “rail traffic” or “recreation”. Even so, Brown’s taxonomy is the most specific one we found, and the most directly related to describing urban sounds.

One final approach to the creation of an environmental sound taxonomy uses a computational method [Gygi et al., 2007]. This work analyzes a collection of sounds based on a hierarchical clustering with their acoustic similarity, and combines those results with several perceptual studies. While this work does not present a clear taxonomy, it offers a useful look at both the perceptually relevant and the acoustically relevant qualities of sounds.

2.2 Process

Our taxonomy is derived from the work of Brown and Gustavino. However, their taxonomies are more general and include environmental sounds of all types, whereas we are interested in studying only urban sounds, as stated in our taxonomy requirements above. We began by isolating the sections of the previous taxonomies that were related directly to urban sounds. We then looked at available sources of information about urban sounds, and used them to build upon the initial taxonomies. Towards that end, we looked at the tags available from Freesound.org, performed “sound walks”, and analyzed 311 noise complaints; these processes are described in the following sections.

2.2.1 Freesound Tags

Freesound.org is a website and database of collaboratively uploaded audio [Font et al., 2013]. Freesound files contain user-specified metadata, including tags, descriptions, and much more. The website is thoroughly searchable through a hypermedia API, allowing one to query it from the website, but also from the programming language of one’s choice.

Initially, it seemed as though it might have been possible to curate a taxonomy
out of the tag data available directly from Freesound, using tag queries for “city”, “urban”, and similar words. However, tags on Freesound are only editable by the user that uploaded the sound, meaning that there is no community confirmation that the tags are accurate. Therefore, it would be computationally intractable to construct a meaningful taxonomy directly from the FreeSound tags [Font and Serra, 2012]; an automatically generated “folksonomy” of this type would require extensive curation and processing to create something meaningful.

While others have used this folksonomy effectively for things like tag recommendation systems to reduce the tag noise [Frederic Font, Joan Serra, 2012], we used this tag data simply to inform our decisions in the creation of our taxonomy. Instead of directly using tags that appear in Freesound to create our taxonomy, we simply performed a statistical analysis of sounds with search queries related to “city” and “urban” to influence our decisions. A bar graph of the top performing queries is shown in Figure 2.

2.2.2 Sound Walks

As researchers studying urban sounds in New York City, it seemed natural to directly explore the actual environment we would be analyzing, in order to become intimately familiar with the sounds we are dealing with. To that end, three of us\(^2\) performed two “sound walks” each; one from our home neighborhood, and then one all together in Union Square, Manhattan. Each listener spent twenty to thirty minutes in each location, listening to the sounds in that space and writing them down. From this experience, we compiled a list of discrete sounds that we heard. See table 1 for a sample of the sounds we recorded from our sound walk.

This experience allowed us to look more carefully at what exactly we were going to be analyzing, from the field. It helped us to solidify taxonomy requirement (2), and thereby enabled us to narrow our goals to the detection of specific low-level events, like “car brakes” or “dog barking” instead of whole entities like “car” or “dog”.

\(^2\)Justin Salamon, Charlie Mydlarz, and myself
Table 1: Summary of Sound Walk Sources

<table>
<thead>
<tr>
<th>Vehicles &amp; Transportation</th>
<th>Motorized Vehicle sounds</th>
<th>Nature</th>
<th>People-related</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car</td>
<td>Engine passing</td>
<td>Bird</td>
<td>Drill</td>
</tr>
<tr>
<td>Taxi (=car)</td>
<td>Engine accel.</td>
<td>Squirrel</td>
<td>Saw</td>
</tr>
<tr>
<td>Bus</td>
<td>Engine idling</td>
<td>Leaves</td>
<td>Conversation/speech</td>
</tr>
<tr>
<td>Truck</td>
<td>Wheels passing</td>
<td>Wind</td>
<td>Foot steps</td>
</tr>
<tr>
<td>Bicycle</td>
<td>Horn/honk</td>
<td>Pigeon</td>
<td>Circular saw</td>
</tr>
<tr>
<td>Police car (=car)</td>
<td>Backing up</td>
<td>Dog</td>
<td>Coughing</td>
</tr>
<tr>
<td>Fire truck (=truck)</td>
<td>Hydraulics</td>
<td></td>
<td>Singing</td>
</tr>
<tr>
<td>Ambient traffic</td>
<td>Brakes (screech)</td>
<td></td>
<td>Laughing</td>
</tr>
<tr>
<td>Motorbike</td>
<td>Doors opening</td>
<td></td>
<td>garbage bag (dropping)</td>
</tr>
<tr>
<td>Airplane</td>
<td>(bus - beep)</td>
<td></td>
<td>Bottle (smashing)</td>
</tr>
<tr>
<td>Helicopter</td>
<td>unloading (mech.)</td>
<td></td>
<td>Bag (rummaging)</td>
</tr>
<tr>
<td>Metro door alarm</td>
<td></td>
<td></td>
<td>Sneeze</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Baby</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Camera (taking pictures)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Music (headphones/street musicians/car)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Leaf blower (engine)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Man pushing cart (wheels)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Man unlocking store lock</td>
</tr>
</tbody>
</table>

2.2.3 311 Noise Complaints

Our final primary source in determining what sounds to include in our taxonomy was the New York City noise complaints from 311 calls. We analyzed these sounds to see which sound complaints occurred most frequently. A full summary of the noise complaints can be found in Table 2.

---

3This data is available publicly from https://nycopendata.socrata.com/
Table 2: NYC 311 Noise Complaint Counts Since 2010

<table>
<thead>
<tr>
<th>Complaint Name</th>
<th># Complaints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loud Music/Party</td>
<td>119395</td>
</tr>
<tr>
<td>Loud Talking</td>
<td>47448</td>
</tr>
<tr>
<td>Noise: Construction Before/After Hours (NM1)</td>
<td>34342</td>
</tr>
<tr>
<td>Car/Truck Music</td>
<td>30854</td>
</tr>
<tr>
<td>Noise, Barking Dog (NR5)</td>
<td>25570</td>
</tr>
<tr>
<td>Noise: Construction Equipment (NC1)</td>
<td>19696</td>
</tr>
<tr>
<td>People Created Noise</td>
<td>16122</td>
</tr>
<tr>
<td>Engine Idling</td>
<td>14350</td>
</tr>
<tr>
<td>Noise: Jack Hammering (NC2)</td>
<td>13731</td>
</tr>
<tr>
<td>Car/Truck Horn</td>
<td>7693</td>
</tr>
<tr>
<td>Noise: air condition/ventilation equipment (NV1)</td>
<td>6573</td>
</tr>
<tr>
<td>Noise, Ice Cream Truck (NR4)</td>
<td>5701</td>
</tr>
<tr>
<td>Noise: Alarms (NR3)</td>
<td>5537</td>
</tr>
<tr>
<td>Noise: Air Condition/Ventilation Equip, Commercial (NJ2)</td>
<td>5155</td>
</tr>
<tr>
<td>Banging/Pounding</td>
<td>4466</td>
</tr>
<tr>
<td>Noise: Air Condition/Ventilation Equip, Residential (NJ)</td>
<td>3546</td>
</tr>
<tr>
<td>Noise: Private Carting Noise (NQ1)</td>
<td>3533</td>
</tr>
<tr>
<td>Other</td>
<td>2054</td>
</tr>
<tr>
<td>Noise: Other Noise Sources (Use Comments) (NZZ)</td>
<td>1903</td>
</tr>
<tr>
<td>Horn Honking Sign Requested (NR9)</td>
<td>1552</td>
</tr>
<tr>
<td>Noise, Other Animals (NR6)</td>
<td>695</td>
</tr>
<tr>
<td>Noise: Manufacturing Noise (NK1)</td>
<td>581</td>
</tr>
<tr>
<td>Noise: lawn care equipment (NCL)</td>
<td>573</td>
</tr>
<tr>
<td>21 Collection Truck Noise</td>
<td>569</td>
</tr>
<tr>
<td>Flying Too Low</td>
<td>290</td>
</tr>
<tr>
<td>Hovering</td>
<td>276</td>
</tr>
<tr>
<td>Noise: Boat(Engine,Music,Etc) (NR10)</td>
<td>258</td>
</tr>
<tr>
<td>Noise: Loud Music/Nighttime(Mark Date And Time) (NP1)</td>
<td>214</td>
</tr>
<tr>
<td>NYPD</td>
<td>212</td>
</tr>
<tr>
<td>Loud Television</td>
<td>174</td>
</tr>
<tr>
<td>News Gathering</td>
<td>171</td>
</tr>
<tr>
<td>Passing By</td>
<td>146</td>
</tr>
<tr>
<td>Noise: Vehicle (NR2)</td>
<td>136</td>
</tr>
<tr>
<td>Noise: Loud Music/Daytime (Mark Date And Time) (NN1)</td>
<td>49</td>
</tr>
</tbody>
</table>
2.3 The Compiled Taxonomy

Finally, we constructed our taxonomy, keeping in mind our requirements and the above data analysis. Our taxonomy considers only urban sounds, and therefore begins at a lower level than some previous environmental taxonomies. In the spirit of those previously proposed taxonomies, however, we defined four top-level groups: human, nature, mechanical, and music. To cover our second requirement, the leaves of our taxonomy are designed to be specific and unambiguous. Finally, to relate our taxonomy to urban sounds, we built our taxonomy around semantic data acquired from Freesound searches, “sound walks”, and common 311 noise complaints, as described above. The taxonomy consists of two types of nodes: the boxes with rounded edges represent the hierarchy of conceptual sources that produce sounds, where as the rectangular boxes represent specific describable sounds. While the conceptual sources form a hierarchy, the sound objects represented by the rectangles have no direct hierarchy, and can be properties of many sources. This allows a variety of “multiple inheritance”, in a way similar to object-oriented programming. For instance, police, ambulance, and taxi are all of the class “car”, but only the police and ambulance can produce the sound “siren”, where as all types of “road” vehicles share classes for “engine idling”, “brakes screeching”, etc. See final taxonomy in Figure 1.
Figure 1: Urban Sound Taxonomy
3 Dataset Creation

The results of any machine learning-based project are highly dependent on the quality and the quantity of training data available. For informatics projects in sound and music, this often requires the researcher to curate the dataset himself, and such was the case for this project. Few datasets of annotated environmental sounds are freely available. One notable example is the D-CASE Challenge [Giannoulis et al., 2013c] dataset, which accompanied the D-CASE challenge last year. However, the event-detection portion of this dataset is fairly small, containing only 24 short examples per class. The urban-related challenge presented focuses on scene analysis, not events, and the event detection challenge focuses on “office sounds”, not urban sounds. The dataset closest to our goals, which we discovered only recently, is the freefield1010 [Stowell and Plumbley, 2013]. freefield1010 is a dataset of 7690 audio clips taken from Freesound which contain the field-recording tag. While sharing many similar goals and approaches to our dataset, is focused simply on “field recordings”, and not on specific urban sounds.

3.0.1 Data From Freesound

Informed by our taxonomy, we set out to construct a suitable dataset for this research. We decided to use the audio freely available and downloadable from Freesound.org to create our dataset. We began by analyzing the urban-related sounds available on Freesound, querying the website using their publicly available API. See Figure 2 for a subset of the results of this analysis. Influenced by the distribution of available data from Freesound and the 311 noise complaints, we selected the following labels to begin our collection: “air conditioner”, “brakes”, “car horn”, “children playing”, “dog bark”, “drilling”, “engine idling”, “gun shot”, “jackhammer”, “police siren”, and “street music”.

The Freesound files metadata provide some information as to what might be found in the file; the original authors often include relatively concise descriptions of the sounds they have uploaded. Therefore, from a simple search on the sound titles and descriptions, we were able download sounds that very likely contained the desired tags described above. However, as discussed in Section 2.2.1, the Freesound metadata is only editable by the user that uploaded the sound, and therefore noisy. While it is coincidentally common for users to upload real-world field recordings to Freesound, it is by no means a requirement, and as such, the sounds returned by a search on Freesound are also likely to contain compositions,
synthesized sounds, and other undesirable sounds. To prepare a dataset for research, three steps were required: 1) selecting the files that indeed contained the sounds desired; 2) annotating each file with the location of the desired sound; and 3) preparation of the sound files into a form ready for research. Each of step of the dataset curation process is described in more detail below. A block diagram of the dataset curation process is shown in Figure 4.

### 3.0.2 Phase 0: Download & Collect Raw Data

The first step in creating our data was to download all of the sounds with metadata containing the desired tags to a single folder. We wrote a Python script to query the FreesoundAPI, process, and organize the results. A summary of the original raw data is shown in Figure 3.

### 3.0.3 Phase 1: Select the Files

Next, in order to be sure that we were only working with files that indeed contained the sound in question, we listened to each and every file to check for the presence of the sound in the file. Since the more detailed work is done in Phase 2,
all we needed to do for this phase was to determine that the sound occurred anywhere in the file. We then moved the file to a folder for “accepted” or “rejected”.

3.0.4 Phase 2: Annotate the Files

In Phase 2, we annotated the active region of each sound in each file for each class, hereafter the “activations”. To do this, we used the popular open source software Audacity\(^4\), which includes a utility for “label” tracks. Audacity’s label tracks allow annotations of arbitrarily named labels, simply by dragging over a region and typing on the keyboard. The user can then simply export the label tracks to text files, where each annotation represented by a single line containing start time, end time, and the label given.

During the process, we decided to label each sound with a saliency measure of ‘1’, ‘2’, or ‘3’. A ‘1’ indicates “foreground”, where the sound is not masked by any significant noise or other sounds. A ‘2’ indicates “background”, or that the sound is masked by noise or other sounds. A ‘3’ indicates “distant”, or that the sound is quite difficult for even the annotator to hear without filtering or turning the sound up.

The process is roughly as follows for each track:

- Open track in spectrogram view Audacity.

\(^4\)http://audacity.sourceforge.net/
• Create a label track.

• Listen to the recording, marking each occurrence of the sound with the saliency label. If sounds occur close together, mark them all as one.

A summary of the data created from phase 2 can be seen in Figure ??.

In the annotation phase, we decided due to the sonic differences in train brakes and other brakes sounds to split the brakes class into “train brakes” and “road brakes” classes. Following the completion of Phase 2, we decided to discard both of our “brakes” classes, because the “road brakes” had far too few segments compared to all of the other classes, and “train brakes” did not fit thematically with the rest of our classes, which are all sounds that would be found outside in the urban environment. Removing these two classes left us with an even 10 classes for the final dataset.

3.0.5 Phase 3: Prepare a Canonical Dataset

The final stage in the creation of our dataset required us to prepare a canonical version of the dataset for research. This process involved two stages: splitting the annotated segments up into short slices, and then allocating them to folders for cross-validation. To split the segments up into slices, we had to determine two parameters: the slice duration and the hop size.

To determine the slice duration, we first had to determine what length splits were going to be adequate to detect any particular class in our dataset. Our goal in determining the slice duration was to have the shortest possible slices while maintaining the maximum recognition rate. We began by performing a brief perceptual experiment, whereby we split the segments into slices of 1 second, 2 seconds, and 3 seconds, and listened to a random sampling to determine which classes we could personally identify in that amount of time. We discovered that for most of our classes, 2 seconds would be enough. However, for some classes, it was easy to mistake one for the other in a blind test; for instance, given a particular sample “street music” with an electric guitar playing a chord for only two
seconds, it was difficult to distinguish if it was a guitar, or train brakes. At three seconds, there was enough context provided that we did not have any difficulties of this sort. A similar but more rigorous perceptual study compared listener accuracy and confidence of recognition to slices of 2, 4, and 6 seconds [Chu et al., 2008]. The study determined that listener recognition rate increases significantly from 2 to 4 seconds, but with only a slight increase from 4 to 6 seconds. Their algorithm had a similar recognition rate at 4 seconds to the listener recognition rate. We also performed an experiment with slice duration where we varied the slice duration from 1 second to 10 seconds in 1 second intervals, and re-ran our baseline system (described in Section ?? for each. The results of this experiment are shown in Figure 5. These results show a general decrease in overall recognition rate below 6 seconds, however, the rates when looking at individual classes is far more complex. Ultimately, we chose 4 seconds, based on our stated requirement to have the shortest possible slices; the study in [Chu et al., 2008] shows adequate recognition at 4 seconds, and in our experiment, 4 seconds is the last point before the overall recognition rate begins to drop off more drastically.

For short sounds, such as “gun shots” and “dog barks”, the majority of the relevant sound tends to be contained within the first 4 seconds of a segment, and segments tend to be 4 seconds or shorter in length. However, segments longer than four seconds will either contain multiple occurrences of short sounds, or a more continuous sound, such as “air conditioner” or “children playing”. Therefore, we selected a 50% hop size for our slices, or 2 seconds. If the original segment is less than 4 seconds long, we keep the entire segment; slices generated which are less than 4 seconds from a file which is greater than 4 seconds long are discarded. For instance, a segment which is seven segments long will generate two slices: one from 0-4 seconds, and one from 2-6 seconds. The remaining one second will be discarded. This is intended to provide robustness to time shifts of sources which occur multiple times within a segment.

Machine learning datasets are commonly organized into “folds”, to facilitate 10-fold cross validation. In the case of our dataset, releasing it already split into
folds was essential, because there are a number of issues to deal with in allocating the data for cross-validation. Firstly, in order to ensure unbiased train and test examples, no example should ever be found in the test set that was used to train the model. For us, this means that all examples from any file had to be allocated to the same fold. Secondly, there was a vast difference between the number of slices available per class, because some classes contain mostly short sounds (e.g. ‘gun shots’), and other classes contain long sounds which may last several minutes (e.g. ‘street music’). When split into 4 second windows, several classes had many thousands of slices, where as others had only a hundred. While it is acceptable, in principle, to have unbalanced classes in our dataset, in order to keep the

Figure 5: Accuracy varying maximum slice duration from 10s-1s. (a) By classification algorithm. (b) By class using only SVM_RBF.
distribution between classes within a reasonable margin, we were restricted the number of slices from certain classes to 1000 across all folds.
The fold allocation algorithm is summarized as follows:

```plaintext
for all class labels do
  for all folds do
    Assign a file from the class to that fold.
  end for
  for all folds, until out of slices or (total slices ≥ 1000) do
    Copy randomly chosen slice from a file assigned to that fold into the fold folder
  end for
end for
```

### 3.0.6 Final Dataset Statistics

The final *UrbanSound* dataset contains 10 low-level classes from the taxonomy: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music. The complete dataset contains a total of 3075 labeled occurrences, for a total of 18.5 hours of audio. The distribution of total occurrence duration per class and per salience is displayed graphically in Figure 6(a).

The collection of 1314 full-length recordings, with corresponding annotation files and Freesound metadata files are all provided with the base *UrbanSound* dataset, and will be available for download online. For research on urban source classification, our “canonical” dataset, the *UrbanSound8K*, containing 4s slices of annotations of the original file, will also be available online. The canonical version contains a total of 8732 slices for 8.75 hours of recordings, split up into 10 folds. The distribution of slices per class in *UrbanSound8k* is shown in Figure 6(b).
Figure 6: Occurrence Duration Per Class
(a) Total occurrence duration per class in *UrbanSound*. (b) Slices per class in *UrbanSound8k*. Breakdown by foreground (FG) / background (BG).
4 Baseline System

4.1 System Overview

With dataset in hand, our first task was to create a baseline set of classification experiments to compare our results with. The goal of the baseline experiments was not to produce optimal parameters and to maximize accuracy, but to study the problems and characteristics of the dataset itself using off-the-shelf tools. The block diagram of the baseline system can be seen in Figure 7. It is composed of two simple parts: the Time Frequency Representation (MFCCs), and classification.

We used MFCCs for our baseline time-frequency representation, because they obtain competitive results in many other timbre-based classification systems, and have been used successfully in environmental sound analysis, such as [Chaudhuri and Raj, 2013]. See Section 5.1 for a further look at using MFCCs for audio classification systems. We extracted all features on a per-frame basis using a window size of 23.2ms and 50% frame overlap. Our baseline MFCC features were computed with 40 Mel bands between 0 and 22050 Hz, and we kept the first 25 MFCC coefficients. No pre-emphasis nor liftering was applied. After feature extraction, the values for each frame are summarized across the slice, using the following summary statistics: minimum, maximum, median, mean, variance, skewness, kurtosis, and the mean and variance of the first and second derivatives. This resulted in a feature vector with 255 dimensions for each slice. We used
Essentia\textsuperscript{5} to calculate the MFCCs and create the summary statistics across all the frames in an audio file.

We used the Weka\textsuperscript{6} data mining software to experiment with various classification algorithms. Every experiment is run with 10-fold cross validation, and in each fold, Weka’s built in correlation based feature selection is used to select attributes and avoid overfitting. We use the default parameters for each algorithm. Classification algorithms included the support vector machine with radial basis function and polynomial kernels, random forests, decision tree, and k-nearest neighbors.

We performed several experiments using our baseline classification architecture prior to the creation of our canonical dataset, in order to minimize the effect of parameters of dataset creation on classification accuracy. We tested the MFCC parameters to see how the number of Mel bands and the number of coefficients effected our baseline results. We tested the duration of our slices in the creation of our dataset, to evaluate how slice length effects classification accuracy. Finally, for the parameters determined in these experiments, we present the final classification results per class, examine the confusion matrices produced, and analyze the results.

Before settling on the final MFCC parameters for the baseline, we ran an experiment which evaluated the change in classification accuracy based on the number of Mel bands used in the computation of the Mel spectrum, as shown in Figure 8. Plot (a) shows the best overall accuracy over all classification algorithms to be the Radial Basis Function (RBF) kernel Support Vector Machine to be between 30 and 40 bands, with another rise at 80 bands. Looking at the classification accuracy across the individual classes in our dataset for just the SVM with RBF kernel (b), we see a relatively flat distribution for some sounds, and a drastic change in some other classes. Car horn is the notable example, where classification accuracy drastically increases with the increase in bands from 10 to 50; this makes sense for a very tonal sound, such as the car horn, as the resolution of discernible

\textsuperscript{5}http://essentia.upf.edu/
\textsuperscript{6}http://www.cs.waikato.ac.nz/ml/weka/
Figure 8: Accuracy over varying number of Mel filters.

frequencies increases with the number of available Mel bands. We also see that the “siren” sound, which is also tonal in character, starts relatively high at near 75%, and gradually decreases with increasing Mel bands. Since most classes start to decrease after 40 bands, we choose to use 40 bands for further experiments.

We performed a similar experiment looking at the number of MFCC coefficients Figure 9. This experiment produced significantly more obvious results; the overall classification results were relatively flat across all runs, although the in-class results have some minor oscillation. With 10 coefficients and under, results drop drastically. We chose 25 coefficients for our experiments, in order to maximize classification accuracy while minimizing dimensionality.
Figure 9: Accuracy over varying number of MFCC coefficients.
(a) By classification algorithm. (b) By class using SVM with RBF kernel.

4.2 Baseline Results

Once we established those parameters for our baseline system, we ran our baseline system on the completed UrbanSound8k dataset. All of the following baseline results are reported using the RBF kernel SVM. Our baseline classifier achieved an average classification accuracy of 69.30%, averaged across all classes.

However, there is a significant difference between the accuracy of sounds in which we applied the foreground saliency label and the background saliency label. This is an expected difference, and is logical given the structure of our baseline system. Since the baseline uses a bag of features approach, in which the features are summarized over the entire slice, wide-band noise distinguishing features of a sound are very likely to be lost. In addition, as discussed in Section 5.2.1, MFCCs
Figure 10: Classification accuracy for each class, by foreground (FG) and background (BG), and together.

are ineffective in the presence of noise. In Figure 10, we see a significant difference between the results for sounds labeled Foreground and sounds labeled Background. The only exception to this difference is the “siren” class, which is logical; firstly, the salient components of the siren class live in a higher frequency range than most of the other classes, and secondly, sirens tend to be loud in comparison to the other sounds. Of all the other classes, the one with otherwise the most similar balance between Foreground and Background is the “car horn”, which has similar sonic characteristics to sirens. Also as expected the standard error on the Background sounds is significant, meaning there is a great deal of variance in the results of the Background sounds, whereas the standard error on the Foreground sounds is significantly less.

See Figure ?? for the confusion matrices for all classes. Here we see three primarily confused pairs of classes: “air conditioner” with “engine”, “jackhammer” with “drilling”, and “street music” with “children playing”. The Foreground sound only shows similar characteristics to the overall confusion matrix, with the majority of the confusions belonging to “air conditioner” and “engine”, following by “jackhammer and drilling”. Notably, the confused sounds in the Foreground sounds are those made up predominantly of wide-band noise. There are significantly more confusions in the Background sounds; many of the sounds, including
all of those from the complete and Foreground confusions, had as many or more confusions as they had correct results. See Figure ?? for the foreground and background sound confusion matrix.
Figure 11: Baseline Confusion Matrices (SVM classifier with RBF kernel.)
5 Feature Learning System

In this section, we present our exploration of feature learning techniques as a method to improve the baseline results. We begin with a history of environmental sound classification and the components of the sound classification pipeline. Finally, we present the results of our feature learning module, and analyze the results in the context of the baseline.

5.1 Background

Analysis of environmental sound, and in particular the automatic source-based identification of events in urban recordings, is still a relatively new field. Only in the recently have significant efforts been made towards this goal. Research in automatic identification of environmental sounds is far more sparse than in the analysis of scenes. In this section we look at previous work in the related fields of source identification and acoustic event identification.

One significant landmark in the computational acoustic scene and event analysis was the creation of the IEEE AASP D-CASE challenge for Computational Auditory Scene Analysis, an international challenge with a public evaluation with the goal of advancing the state of the art in the areas of modeling acoustic scenes and detecting audio events [Giannoulis et al., 2013a]. There are several similar challenges, such as the MIREX competition, that have existed in related fields, but this is the first such challenge focusing on detection and classification of non-speech and non-music signals. This competition, presented by the Queen Mary University of London, consisted of three challenges: the first is a scene-analysis challenge, the second is an event-detection challenge, consisting of separate monophonic (non-overlapping sounds), and the third is a event-detection challenge consisting of polyphonic (overlapping sounds) components. This competition is an exciting development in the field of computational auditory scene analysis, creating opportunities for researchers across the world to compare their results in a rigorous way. However, the datasets used in the challenge were of
a relatively small scale, which makes using them for machine learning research somewhat challenging, as the results of machine learning algorithms are directly tied to the size of the training data. In addition, only the scene classification dataset made use of urban sounds; the event detection datasets were made up almost entirely of “office” sounds, making it of limited use for our research.

However, this challenge does provide a useful way to look at the sorts of algorithms that researchers are using for event detection problems. For the scene classification task, the baseline used MFCCs for features, and used a bag-of-frames approach in classification. MFCCs, or some variation on them, were by far the most common features used in the 2013 evaluation of the challenge, and support vector machines were the most common classifier. To make a decision across the entire recording, majority voting and max-pooling techniques were common. For the event detection challenge, non-negative matrix factorization was the baseline approach, and made a significant appearance in the challenge entries. MFCCs were also used in a significant portion of the entries in this category. The results of the D-CASE challenge are presented in [Giannoulis et al., 2013b].

Several recent approaches to machine hearing of environmental sounds take different directions than the standard ones discussed above. One approach to environmental scene recognition is presented in [Chu et al., 2008] Chu et al. proposed a method using Matching Pursuits (MP) to learn a dictionary of atoms for feature selection, combined with MFCC’s. This article stands out above many of the rest we looked at. It was the only one we came across in our research that made significant use of time-domain features. Chu et al. state plainly many of the failings of MFCCs. Their MP-based features not only perform well in noisy urban environments, but they also perform significantly better than MFCC’s in those cases. The MP-based features combine with MFCCs to yield an even better result.

Another approach is to use image processing techniques on the spectrogram. An example of this is found in [Dennis, 2011], where Dennis uses presents two image processing methods, the “Spectrogram Image Feature”, and “Subband Power Distribution Image Feature”, in combination with a variety of k-Nearest Neighbor,
to yield very good results over his baseline system. Even in the presence of noise, particular sounds in the spectrum typically have a characteristic shape which is recognizable to the human eye. This computer vision approach therefore has the notable feature of being relatively robust to noisy environments, by recognizing the characteristic shape of a sound in a spectrogram.

Several approaches to event and scene classification attempt to invent new time frequency representations. Richard Lyon proposes one perceptually motivated approach using a pole-zero filter cascade (PZFC) which simulates the impedance behavior of the basilar membrane [Lyon, 2010]. Lyon’s approach is designed from end to end to simulate the human hearing system, and is drastically different from other systems proposed to date. Lyon’s system combines the PZFC with a sparse-coding feature extraction, and his experimental results show significant improvement over features learned on MFCCs.

Detection of sounds in an acoustic environment requires knowledge of various time resolutions, as in many cases the difference between two sources is only identifiable by observing the sound over time. For instance, the difference between the sound of a screeching car breaks (a short, high pitched sound), and a police siren (a long high pitched sound that changes frequency slowly) may only be detectable if the sound is observed over a longer time widow. Dieleman uses three different approaches to solve this problem in [Dieleman and Schrauwen, 2013]; multi-resolution spectrograms, and the Gaussian and Laplacian pyramid. Dieleman found no clearly winner architecture among these, but found that different architectures tended to work better for different classes or different tasks, based on how time-dependent the signal is. Another multiresolution algorithm is presented in [Lallemand et al., 2012], which presents wavelet-based features, and a similarity measure for those features using Kullback-Leibler divergence. This approach gets improved results over MFCCs with Euclidean distance for an environmental sound recognition task, however, their system only looks at pair-wise similarity for evaluation results, and is therefore difficult to compare with the type of classification systems we are interested in.
In some cases, the Gaver taxonomy has been used directly in conjunction with audio analysis to perform sound classification. In machine learning research some work has recently been done using joint embedding spaces [Weston et al., 2010], where a mapping is learned from the audio to a semantic space which is simultaneously the tag space. This allows the machine to infer a tag which exists in the tag space from an audio example which was never labeled. Roma et. al perform a similar experiment using the Gaver taxonomy, with an expanded search space widened using the Wordnet [Miller, 1995] database in combination with audio from Freesound [Roma et al., 2010].

For a deeper look into the state of classification and automatic tagging in related audio fields, see [Bertin-Mahieux et al., 2010]

5.2 Analysis Pipeline

The results of any computational classification algorithm, such as the one discussed here, are highly dependent on the quality of the input. In machine learning terminology, the input representation into a classifier is typically referred to as the “features”. Traditionally, in machine hearing research communities, features have often been hand engineered using extensive domain knowledge [Humphrey et al., 2012]. For example, a knowledge of music theory allows one to create rules or heuristics on how to analyze a musical recording that can be encoded into an algorithm for analysis. At the other end of the feature spectrum from feature engineering is feature learning. This involves the use of machine learning techniques to “learn” the features from the low-level input to the system. In the following section, we discuss approaches to both of these techniques that have been used throughout the literature, and the benefits to both.

5.2.1 Engineered Features

The most common features used in audio classification of any type are MFCCs, or Mel-Frequency Cepstral Coefficients. MFCCs were designed initially for speech
In the feature engineering approach, raw audio is converted into an engineered, or human-designed time-frequency representation, which is fed directly into the classifier.

In the feature learning approach, a simple time-frequency representation is used, and a model (sometimes called a dictionary) is learned automatically by example from the data. Features are then created by taking the dot product of the input with the model.
detection and classification in the late 1990s, and have proven to be very successful for classifying phonemes in speech signals. The success of MFCCs in the speech processing community eventually interested those in the music and sound informatics communities, and MFCCs have since grown to be the de facto standard for timbral features. MFCCs are a relatively low-dimensional representation of timbre which encodes the spectral envelope of the signal in a small number of coefficients. They are computed by computing the Discrete Cosine Transform (DCT) on log of the Mel-filtered spectrogram.

It is precisely the low dimensionality that makes MFCCs especially efficient; machine learning classifiers have a tendency to overfit the data if the dimensionality of the input representation is too high. MFCCs convert a Fourier input of 1024 or 2048 coefficients to one of 13 to 25 coefficients, but maintain a significant amount of important timbral information required to discriminate sounds. This is an efficient coding, in that the coefficients generated are decorrelated from each other, which makes MFCCs a quality input into a classifier without needing to perform additional PCA. In addition, MFCCs represent the spectral envelope of a signal, and are therefore essentially transposition invariant. For speech signals especially, MFCCs tend to perform very well in classification tasks, but they have also had good results in instrument classification tasks.

While MFCCs have significant benefits, they are far from perfect. Firstly, they convey information about the static current state of a signal only, whereas all sounds are dynamic in time, having, for instance, a different timbre at the start of the sound from the middle or end of the sound. Secondly, the performance of MFCCs are known to degrade significantly in the presence of noise, and are similarly ineffective at representing signals noise-like signals with a relatively flat spectrum. This problem is especially notable for detecting sounds in urban soundscapes; unlike music, which is typically recorded in a controlled environment, nearly every signal involved in the classification process is likely to have some significant noise involved in the signal. Many of the classes of sounds we are interested in urban environments are exactly that sort of noise-like sounds–air...
conditioners, engines, etc.

Several approaches have been presented which try to solve MFCCs’ deficiencies by engineering more robust alternatives. One approach, entitled gammatone frequency cepstral coefficients (GFCCs), suggests a variant of the MFCC representation using the cubic root of the time frequency representation instead of the log, in combination with a gammatone weighted filter bank instead of a Mel weighted filter bank. [Zhao and Wang, 2013] Zhao and Wang demonstrate these modifications to create robustness of the representation to noise in the signal.

Now that we have established MFCCs as the most common features for timbral classification tasks, we take a step back to look at the Mel spectrum, the first step in the creation of MFCCs. It is calculated by applying a Mel-frequency scale filterbank on the Fourier spectrum. The Mel scale is a logarithmic scale weighted to coincide approximately with human hearing. The Mel spectrum is a common input representation for machine hearing systems. Machine hearing approaches which desire a representation closer to the original spectrum, but still reduced in dimensionality, typically use this representation, or a similar one such as the Constant-Q Transform.

The Mel spectrum is particularly useful in machine learning tasks, because it is stable to deformation using a Euclidean norm, unlike the spectrogram [Andén and Mallat, 2012]. However, the averaging used to create The Mel spectrum causes significant loss of high-frequency information unless the window size is kept small. Several approaches have been created to deal with this problem. One notable approach is the use of “scattering coefficients”, which extend The Mel spectrum using a cascade of wavelet decompositions and modulus operators to recover high frequency resolution while keeping stability to the deformation. This scattering representation successfully captures an improved timbral representation compared to The Mel spectrum. The authors present an extension of this similar to MFCCs in [Mallat et al., 2013], called “deep scattering spectrum”, which gets state of the art classification results on genre and phoneme classification. The deep scattering spectrum is “deep” in the hierarchical sense, because of the cascade of
operations; it’s complexity is engineered, not learned, as in other deep systems described in the next section.

5.2.2 Learned Features

Feature learning has recently become popular for finding solutions to signal classification problems. A feature learning system “learns” a function of the data by training it on the data itself. Feature learning algorithms are data-driven, and depend directly on the quality and quantity of the data provided in the learning stage.

Neural networks, in particular “deep networks”, are one of the primary tools for feature learning. A neural network is a complex hierarchy of linear and non-linear nodes which can learn complex functions automatically given enough input data. “Deep networks” are many-layered versions of neural networks, and the “deep”-ness of them allows them to learn hierarchies of features, enabling them to automatically learn very high level features about a signal. Neural networks were invented as early as the 1940s, and were researched extensively until the early 1990s, when successes in other machine learning algorithms eventually took the focus in the machine learning community. However, in the last decade or so, important advancements in training deep networks, combined with the general increased computing power available in society have brought neural networks back to prominence. Across computational research genres, but specifically in the image and speech detection communities, deep neural nets have recently been outperforming state-of-the-art systems where heuristics-based systems had previously dominated [Krizhevsky et al., 2012].

One of the biggest problems with deep neural networks is that the derivatives that are backpropagated during supervised training become extremely weak so as to be of minimal effectiveness by the time they reach the beginning of the network. In particular, it was shown that the use of greedy layer-wise training is what brought the neural network back to prominence [Hinton et al., 2006]. This initializes the data in an unsupervised fashion, one layer at a time, while freezing
the weights of the other layers. An unsupervised algorithm such as k-means or sparse coding is typically used for this. Unsupervised initialization of the network significantly improves the performance of neural networks.

One specific variant on this approach is the use of spherical k-Means to initialize feature learning layers, as described by Coates and Ng in [Coates and Ng, 2012]. Spherical k-Means is a slight variation on the traditional k-Means algorithm, where the centroids are constrained to the unit sphere at each update step. This has the function of using the cosine distance for similarity to points in the input space instead of the Euclidean distance, which is typically used in traditional k-Means. Coates and Ng also specify that the effectiveness of the learned centroids is significantly improved by the addition of ZCA\(^7\) whitening, which “sphere’s” the data, allowing the unit sphere-constrained centroids to maximally represent the data.

Coates and Ng describe the use of spherical k-means in image detection, but the algorithm is equally as effective in processing audio. One approach by Dieleman uses spherical k-means in combination with multiscale approaches in an audio tag prediction and a similarity metric learning task to [Dieleman and Schrauwen, 2013]. Unlike the Coates and Ng design, Dieleman uses PCA whitening instead of ZCA whitening, which serves to further decorrelate the inputs from each other. This step can significantly help the quality of the input representation, and serves to decorrelate the input representation. See [Nam et al., 2012, Hamel et al., 2012] for further examples of PCA whitening used with audio.

To compare the quality of dictionaries learned using the data, we can also look at dictionaries composed of noise. [Lutfi et al., 2009] uses a dictionary composed of random bases, with no learning, and has competitive results, in a process known as “compressed sensing.” The representation allows accurate compression of the signal using a small dictionary. This algorithm significantly reduces computation time over a learned representation, as no knowledge of the data is necessary beforehand, other than the dimensionality of the desired bases.

\(^7\)Zero-Phase Component Analysis
While feature learning is especially effective in deep architectures, it has also been proven to be effective in shallow architectures [Coates et al., 2011]. To begin with a feature learning architecture which is easily comparable with our baseline system, we present in this thesis a system which is loosely related to Dieleman’s system, using a single feature learning layer trained using spherical k-means on PCA whitened audio data.

### 5.2.3 Classification

After the choice of features, the next component in a classification system is the classifier itself. At a very basic level, a classifier simply takes an input and assigns one or more “labels” or “classes” to that input. The classifier is trained or “learned” from the data itself, and as a result the quality of the results depends entirely on the data the classifier was trained on. In this project, we use the Weka data mining software to perform our classification. While the classifier is an important component of the automatic classification system, the focus in this research is on the features input into the classifier, not the classifier itself. Therefore, after experimenting with different classifiers in our baseline system, we settle on using only the Support Vector Machine with a Radial Basis Function kernel as our classifier. With that in mind, we briefly take a look at how features are passed into the classifier.

Events in audio recordings are composed of a sequence of frames, but frames generated from windowed spectrograms are typically on the order of 25 to 50 milliseconds, which is much shorter than the duration of most events of interest. In order to classify individual events, a decision has to be aggregated across several frames. One technique for classifying a section of audio is called the “bag of frames” approach. Instead of analyzing each frame of a signal to determine which class or classes it might belong to, and then summarizing the frame-level results, bag of frames takes summary statistics across an entire input sound, be it a track or a segment of a track. For instance, with MFCCs, the final features that were used in the classifier would be the mean, variance, mean and variance of the first and
second derivatives, minimum, maximum, etc., each across all of the samples in the frame. Even though this approach loses a significant amount of information, it turns out to perform quite well on certain tasks, such as scene identification. Other approaches include passing the individual frames into the classifier, and to perform a majority vote on the resulting output to determine the final output. Hidden Markov Models are also a common solution to determining the final result from a sequence of outputs from a classifier.

5.3 Algorithm Summary

Table 3: Summary of Environmental Analysis Papers

<table>
<thead>
<tr>
<th>Paper Ref</th>
<th>Task</th>
<th>Classes</th>
<th>Features</th>
<th>Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Chu et al., 2008]</td>
<td>scene</td>
<td>street: 14</td>
<td>MP for feature selection, +MFCCs</td>
<td>k-NN, GMM</td>
</tr>
<tr>
<td>[Cotton et al., 2011]</td>
<td>“concept” (=scene)</td>
<td>25 concepts</td>
<td>PCA → k-Means</td>
<td>SVM</td>
</tr>
<tr>
<td>[Dennis, 2011]</td>
<td>scene</td>
<td></td>
<td>SIF, SPD-IF</td>
<td>k-NN</td>
</tr>
<tr>
<td>[Lallemand et al., 2012]</td>
<td>scene</td>
<td>Sound Ideas</td>
<td>multiresolution modeling wavelet subband coefficients</td>
<td>K-L divergence</td>
</tr>
<tr>
<td>[Cotton and Ellis, 2011]</td>
<td>events</td>
<td>16 meeting room events</td>
<td>MFCC, Convolutive NMF</td>
<td>HMM</td>
</tr>
<tr>
<td>[Krijnders et al., 2010]</td>
<td>events</td>
<td>7 subway events</td>
<td>chochleagram</td>
<td>“knowledge network”</td>
</tr>
<tr>
<td>[Roma et al., ]</td>
<td>events</td>
<td>Gaver classes</td>
<td>MFCC, misc spectral + BoF</td>
<td>1v1 SVM</td>
</tr>
<tr>
<td>[Valero et al., 2012]</td>
<td>events</td>
<td>7 urban classes</td>
<td></td>
<td>SVM &amp; Self-Organizing Map</td>
</tr>
</tbody>
</table>

We modeled our feature learning system after Dieleman’s from [Dieleman and Schrauwen, 2013]. The block diagram is shown in Figure 14. In the first stage, we extract a time frequency representation from the input audio. We explored the
Log Mel spectrum and MFCCs as inputs to our feature learning stage, with various
different parameters, as summarized in table 4. Once all of the audio is converted
to time frequency representation, we run spherical k-Means to learn a dictionary of
centroids from all of the data in the training set. The dictionary is then saved, and
used to generate features for all of the time-frequency inputs. Features are created
by taking the dot product of the input with the dictionary. Learned features are
summarized over each slice file, in the same way they are in the baseline. We
summarize the features using minimum, maximum, mean, and variance of the
input features. The dimensionality of the input into the classification system is
thus four times k. Finally, classification is run using the same system baseline, as
described above, except we only run the SVM classifiers.

In our feature learning stage, we explore whitening using both Zero-Phase
Component Analysis (ZCA) as recommended in [Coates and Ng, 2012], and Prin-
cipal Component Analysis (PCA), as recommended in [Nam et al., 2012, Hamel
et al., 2012]. We explored three different feature learning modules, including
Spherical k-Means [Coates and Ng, 2012], random bases [Lutfi et al., 2009], and
the build-in Scikit-Learn implementation of traditional k-Means. Random bases
are simply randomly initialized Gaussian noise of the same dimensionality of our
data. They are used in much the same way as our learned dictionary, except they
are not learned from the data. The comparison with random bases is used to eval-
uate if our feature learning system has learned useful features which are helpful
Table 4: Experiment Parameters

<table>
<thead>
<tr>
<th>Time-Frequency</th>
<th>Feature Learning Algorithms</th>
<th>k</th>
<th>other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mel Spectrum</td>
<td>Spherical k-Means</td>
<td>50</td>
<td>RELU</td>
</tr>
<tr>
<td>MFCC</td>
<td>Random Bases</td>
<td>100</td>
<td>PCA/ZCA</td>
</tr>
<tr>
<td></td>
<td>Trad. k-Means</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>500</td>
<td></td>
</tr>
</tbody>
</table>

in discriminating our particular sounds. We perform the same experiment against traditional k-Means to see if Spherical k-Means is improving our results as we might expect. We also experimented with including a rectified linear unit (RELU) after the dot product, which is a common non-linearity used in machine learning to threshold the features at the output.

All experiments were run using 10-fold cross validation, using nine folds for training and one for tests. The fold that was held out from the feature learning stage was used as the test fold. Results are reported as the average across all cross-validation runs.

5.4 Feature Learning Results

In looking at the results of our experiment, the primary question to keep in mind is: are our features learning anything useful? Are the features learned from the data helping to discriminate the sounds at all? We explore this question throughout this section.

Figure 15 displays the overall accuracy of the most meaningful of our experiments. From this diagram, it is clear that our feature learning algorithm is not performing as well as the baseline. The highest performing algorithm parameters uses Spherical k-Means with PCA whitening and k=100, and yields a classification accuracy of 67.02% This is compared to an overall classification accuracy of 69.3% for the baseline, a decrease of approximately 2.3%. However, for perspective, we observe that the random classification accuracy, which is to say, the
percentage chance of choosing the correct class if one were to choose randomly, is only 10.2%, and compared to this, both 67.02% and 69.3% are relatively high, and relatively close together, statistically speaking. [Talk about statistical significance of these, maybe?]

**Figure 15: Overall Feature Learning Results**
For various trial runs, with k=50,100,200,500, Random Bases (blue), Log Mel Spectrum with Spherical k-Means (red), Log Mel Spectrum with a traditional k-Means (green), and MFCCs with Spherical k-Means. Baseline accuracy is the horizontal (blue) line.

Digging a little deeper, we look at the classification accuracy *per class*, instead of averaged across all classes. This information can be found in Figure 16. These results show us that while, indeed, the baseline is performing better overall, there are some classes that perform significantly better than the baseline, and others
which perform much worse. Specifically, the classes “car horns”, “children playing”, and “dog barks” perform greater than 5 percentage points better than the baseline using various k of our established setup (Log Mel Spectrum, Spherical k-Means, PCA whitening). “Police Siren” performs only marginally better than the baseline, at around two and a half percent, and “street music” squeaks in at just .4% over the baseline. “Engines idling”, “gun shots”, and “jack hammers” received within a few percent of the baseline, but the “air conditioner” class came in last place, getting almost 10% under the baseline. Looking at this chart, we see the “air conditioner” class performs poorly across all of our feature learning tests, at more than ten percent below the baseline for that class, and around 25% lower than the baseline average. This class is clearly the cause for the low overall classification accuracy across all feature learning runs.

Beyond the baseline performance for the individual classes, it is also important to look at the results of the “random bases” experiment. As the dictionary used in the random bases case is made completely of Gaussian noise, and they are purely an agnostic coding of the data. Ideally, for best comparison, we would have performed one random bases experiment for each k that we ran in every other experiment, however, we ran out of time to complete all of these experiments. Therefore, in this discussion, we will look only at the k=100 cases for Spherical k-Means versus the random bases. We see looking at the chart in Figure 16 that for the k=100 case, the learned features perform better significantly than the random ones in almost every case. The exceptions are in the “Engine Idling”, “Gun Shot”, and “Jack Hammer” classes; in these classes, the learned features still perform better than the random features, however, the margin is so small as to be insignificant. Interestingly, we see that the classes in which the learned features are performing suboptimally in comparison to the baseline are also the classes in which the learned features perform poorly in comparison to the random dictionary. These particular classes, more than all of the other classes, are timbrally similar, but are also the classes that are most noise-like. This suggests that our features are not learning any invariance to noise, as MFCCs, which are
well known for performing poorly in the presence of noise, are outperforming our learned features.

In order to prove that our spherical k-means feature are indeed learning anything useful, we take a look at two more plots. Firstly, we take a brief look at a plot of the centroids by the model for a single fold, to see for ourselves visually that they are learning something useful. Figures 17a and 17b show the model learned by the Spherical k-Means algorithm for k=50 and k=100, respectively. While these plots individually are rather difficult to discern any useful information from, we can see from visually comparing the two different plots a common set of features which appear to have been learned by both systems. Two regions stand out between the two images: the top right corner and bottom, just to the right of center. The presence of these two regions helps discern that the k=100 appears to be a higher resolution representation of the k=50 version. This implies, as one might
A plot of the Log Mel spectrum clusters after the algorithm has terminated for one fold with (a) \( k = 50 \) & (b) \( k = 100 \). These results will be different for each fold, as they are tested and trained with 10-fold cross validation. The individual centroids are represented on the x-axis, and the Mel-spectrogram coefficients are represented on the y-axis. Centroids are sorted with a low-to-high frequency weighting.

expect, that an increase in \( k \) is more fully representing the input space, allowing for an increased ability to discern the sounds. However, wide-band noisy spectra are notably missing from these plots, which could explain the inability of these features to represent noisy sounds.

Finally, we look at the difference in rates of classification between foreground and background slices for five different experiments, as shown in Figure 17. To understand how the results change over differing \( k \), we look at the same groups of sounds we looked at in the previous sections. The “car horn”/“children playing”/“dog bark” results generally increase over increasing \( k \), although “car horn” has an interesting drop in foreground accuracy at \( k=500 \). Increasing \( k \) does increase recognition in the background classes, but not as much as we might expect. Most interesting in these plots is that the classification rates for the noisy classes: “air conditioner”, “drilling”, “engine idling”, and “drilling”. The background recognition rates for these classes in the random bases versus spherical k-means at \( k=100 \) shows that the feature learning is indeed learning useful features.
help with differentiation of noisy sounds in the background. However, across the spherical k-means experiments, the background accuracies decrease slightly with increasing k, meaning that the increased dictionary size is not necessarily helping at all with distinguishing the noisy classes.
Figure 17: Classification accuracy by foreground (FG) & background (BG) for various parameters.
6 Conclusions

In this thesis, we identify three problems in the field of automatic urban sound classification: the lack of a common taxonomy of urban sounds, the lack of an annotated dataset of real-world urban sounds, and the difficulty of discriminating sounds in the presence of noise. We presented solutions to the first two problems, and a baseline classification system to study the challenges and features of the dataset we created. Further, we implemented a first exploration upon the baseline system using a single layer feature-learning approach.

Our feature learning approach does not produce significant overall improvement over our baseline system, however it does perform significantly better on some classes of sounds. In particular, our system performs the best on tonal, pitch-based classes, and worst on sounds which are noisey in character. We establish through comparison with random bases that our feature learning system does in fact learn some useful features to assist in discriminating those noisy sounds in the background.

6.1 Future Work

There are many possible directions to take further explorations with this data and our feature learning approaches. There are a number of simple ways that we might be able to improve classification accuracy without changing our algorithms much. An important first step would be to analyze and validate our dataset further. A number of assumptions were made in the construction of our folds; we slice our segments into four second slices with a hop size of two seconds, but we have no way to know for certain that the annotated sound remains in the sliced segment after automatic slicing. We need to prove conclusively that every slice extracted from a segment has enough of the original sound to be possible to classify. Unfortunately, this process will be far from easy; this would likely require an extensive manual effort in the form of crowd-sourced annotations. While this may not improve accuracy relative to the baseline system, it may increase the accuracy
of the entire system, if negatives are being generated because the sound in ques-
tion is not actually present in the labeled slice. In addition, we could try various
other pre-processing techniques to try to improve our data, such as automatic gain
control or other loudness normalizations.

Secondly, there is the issue of noise. As shown in Figures 10 and 17, both the
baseline and the feature learning algorithms perform quite poorly on our back-
ground salience slices. In Section 5.1, we saw several other approaches that peo-
ple have used to build features robust to noise, and many of them would be easily
transferable to the system we have already established. For instance, Zhao’s ap-
plication of the cube root instead of the log on the Mel-spectrogram was shown to
improve robustness to noise [Zhao and Wang, 2013]. Scattering coefficients or the
deep scattering transform would be another way to improve the Mel-spectrogram
time-frequency input to our system to increase robustness to noise and increase
resolution at high frequencies, without increasing the complexity of our feature
learning system [Andén and Mallat, 2012].

Next, there is the issue of temporal dynamics of sound, and how to handle
classification using data from different time scales. In our work, we handled the
time dimension by summarizing our features over the entire slice file. While this
is effective enough for strongly tonal sounds, or sounds with distinctive timbral
characteristics in the mix, our results showed this approach is clearly ineffective
for noise-based sounds and sounds in the presence of significant noise. A number
of possible approaches exist which would be useful for handling the temporal dy-
namics of the signal. A first approach to dealing with temporal dynamics would
experiment with a frame-based approach rather than a bag-of-frames approach. To
compare directly with our classification system, a frame-based approach would
still need to have some decision-making algorithm to summarize the results of
the individual frames over entire slice. Typical algorithms for this in the D-CASE
challenge, which would be useful to try for comparison, are the majority-vote, and
Hidden Markov Models. The next approach we could take would be to implement
the multiple time scale work in [Dieleman and Schrauwen, 2013], including the
multi-resolution FFT, Gaussian, or Laplacian pyramids. For a system that is significantly more sensitive to temporal dynamics, we could train our k-means models convolutionally with patches from the input representation containing multiple time windows, thereby representing not only frequency content, but also time content with the patches.

To improve the results of our feature learning system itself, there are a limited number of options working within the constraints of the system we have right now. There are only two hyper-parameters available to play with in our current system: the number of k, and the number of PCA whitened features selected. It is quite likely, given the broad range of results from the different k we tried, that we have not yet selected the optimal parameter, and furthermore, we have not experimented with even larger feature dictionaries than the 500 we tried. Beyond hyper-parameter tuning, however, further developments in our feature-learning system are likely to be focused on learning multiple layers of features. While single layer feature learning systems have been shown to be competitive, the real benefit to feature learning comes from the use of deep architectures. These deep networks contain pooling steps and non-linearities between each layer, which allow the system to learn a hierarchy of complex features unattainable by shallow architectures.

Due to the specific problems with noise in this field, we feel that in implementing a deep learning system for urban sound classification, it would be necessary to carefully design each layer from the ground up. This would allow us to construct a system tailored to our specific classification task. The feature learning system based on spherical k-means presented in this thesis were the first steps towards that goal, a robust network capable of discerning urban sounds.
References


nal Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on, pages 69–72. IEEE.


[Roma et al., 2010] Roma, G., Janer, J., Kersten, S., Schirosa, M., and Herrera, P. Content-based retrieval from unstructured audio databases using an ecological acoustics taxonomy. In *The 16th International Conference on Auditory Display (ICAD-2010)*.


and Exposition on Noise Control Engineering (Inter-Noise-2012). Institute of Noise Control Engineering.

