Structured Models for Semantic Analysis of Audio Content

Sourish Chaudhuri

CMU–LTI–13–005

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Bhiksha Raj (Chair)
Rita Singh
Jaime Carbonell
Dan Ellis, Columbia University
Malcolm Slaney, Microsoft Research

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Keywords: audio content analysis, audio semantic analysis, acoustic unit descriptors, structured models for audio, multi-instance learning for audio

To baseball
(and the Pittsburgh Pirates)

For their enormous contribution in helping me understand statistical modeling in the real world, and life.

Abstract

In the universe of audio signals, the notions of syntax, semantics, pragmatics, *etc.* have been associated with a very limited set of domains, such as speech and language, and musical analysis, to some extent. However, research efforts focussing on formalizing general notions of syntactic or semantic structure for universal audio analysis have been relatively limited. Prior work in analysis of audio content has largely involved identifying certain sounds in recordings, and the analysis paradigm has typically relied on a shallow analysis framework that assumes that observed acoustics map directly to the semantics.

We posit that sound possesses a hierarchical semantic structure, in reality, and a full understanding of the semantic content of recordings requires inferring this hierarchical structure. However, modeling this kind of structure in supervised settings would require richly annotated datasets, that do not currently exist and would require a significant annotation effort to develop.

The main hypothesis that drives this dissertation is that sound has its own *language* and *structure* and that the deeper, underlying semantics can be modeled using a hierarchical framework. In this dissertation, we present such a hierarchical framework and develop formal models, designed for *unsupervised* or *weakly supervised* settings, for the same.

We model the observed sound using sequences of lower level units. While these units may not carry semantic information individually, the sequences or distribution of these units should capture semantic information. In this *language* for sounds, the lower level units would be analogous to the alphabet. Such a representation of sound using a discrete sequence lends itself naturally to the hierarchical structure, where sequences of these lower level units can be mapped at higher levels to real events with clear semantic interpretations. Further, these event sequences should carry information about the overall semantic category of the audio. Depending on the restrictions we enforce at various levels of this structure, we can use such structured models to classify audio, detect sound events, segment files, or predict associated sound classes.

In this dissertation, we present structured models for the various layers in the hierarchy. We then explore 2 different paradigms for inducing a hierarchy over the low-level acoustic units. Our proposed methods work unsupervised and in a task-agnostic manner, and we demonstrate empirically, using standard audio tasks, that semantic analysis of audio using this framework is feasible and that it outperforms other plausible semantically motivated schemes. Finally, we discuss some directions for future work, and present some preliminary formulations and experiments toward addressing them. The research pursued in this dissertation demonstrates that hidden semantic structure can be automatically discovered from weakly-labeled audio data. Further, we believe that the use of such semantically informed features will enable significant improvements over the state-of-the-art, for a number of different tasks.

Acknowledgments

I must begin this section by thanking my advisor, Bhiksha Raj. I could not possibly acknowledge sufficiently his influence; I will simply say that if a genie had offered me the opportunity to wish for an ideal thesis adviser, I could not have wished for anyone more perfect than Bhiksha. The enormous support, knowledge and (perhaps most importantly) time and patience that he had to offer on every aspect of my work as well as life astounds me still. I've learnt many things from him, but the lessons I cherish most are the importance of a long-term vision and the ability to focus past the trees on the forest. I've often stopped by to chat with him with no specific discussion topics in mind, and I've never left without learning something new.

I also had the opportunity to work closely with Rita Singh on various aspects of my dissertation research. She has been responsible for the rapid development of my technical knowledge on a number of topics, and the discussions we had regarding my dissertation work had significant influence on the direction of my work.

In addition to Bhiksha and Rita, I was fortunate to have Jaime Carbonell, Dan Ellis and Malcolm Slaney on my thesis committee. I thank all of them for the technical discussions and advice I received from them in helping improve my work, and their generosity in finding time for me, whenever I needed it! Aside from the fact that their respective individual expertise was closely related to the context of my research, I was thrilled that Jaime, Malcolm and Dan agreed to be on my committee because (early in graduate school) I'd had occasion to read, and admire, and be inspired by, work done by each of them. If there is such a thing as academic fandom, then I had 5 people on my committee that I was a fan of. In addition to their technical contributions to my thesis, they were excellent people to work with, which made work fun.

My research in this dissertation was funded by Charles Stark Draper Laboratories, Cambridge, USA, and I had the opportunity to have a number of insightful discussions with their researchers, for which I am very grateful to them.

I would also like to thank Carolyn Penstein-Rose, who I had the opportunity to work with for 2 years during my MS at Carnegie Mellon University, and who guided me through many of the early bumps of graduate school, research philosophies, and technical speaking and writing. My deep gratitude to Rich Stern, John McDonough and the extended ROBUST–MLSP research groups for all their suggestions, feedback and support, and certainly a group of people that will be hard to replace; thanks to Amir Moghimi, Anjali Menon, Ben Lambert, Bosco Chiu, Chanwoo Kim, Griffin Romigh, Kenichi Kumatani, Kshitiz Kumar, Mahaveer Jain, Manas Pathak, Mark Harvilla, Michael Garbus, Nia Bradley, Pallavi Baljekar, and Sohail Bahmani.

I've had the opportunity to learn from various faculty both in courses, as well as in technical discussions in various contexts, while at Carnegie Mellon. I'd especially like to thank Alex Hauptmann, Alan Black, Carlos Guestrin, Eric Nyberg, Geoff Gordon, Jamie Callan, Lori Levin, Noah Smith, Stephan Vogel

for their time and ideas, that greatly influenced my development as a researcher and a person.

The LTI staff have been incredible in their ability to make our lives easier– special thanks to Stacey Young, Radha Rao, Dana Houston, Linda Hager, Krista McGuigan, Kelly Widmaier, and Ben Cook for all their help!

At Carnegie Mellon, I was fortunate to have an outstanding peer group who were such an integral part of the last few years of my life. They are friends, as well as collaborators, teachers, teammates, and possess the most wonderful quality of being a perpetually comforting omnipresence. Among the many people that I'm certain I'm forgetting from CMU, I'd like to thank Anindita Dutta, Anagha Kulkarni, Bhavana Dalvi, Brian Langner, Debaditya Dutta, Dipanjan Das, Elie Krevat, George Nychis, Gopala Krishna Anumanchipalli, Jeff Flanigan, Jineta Banerjee, Jose Pablo Gonzalez-Brenes, Kriti Puniyani, Le Zhao, Mahesh Joshi, Matt Marge, Pradipta Ranjan Ray, Prasanna Kumar Muthukumar, Ramnath Balasubramanyan, Rohit Kumar, Saurabh Taneja, Siddharth Gopal, Sivaraman Balakrishnan, Sourav Bhattacharya, Sudarshana Bhattacharya, Udhyakumar Nallasamy, Vasco Pedro.

I cannot stress enough how perfect a place Pittsburgh was for me for the last 6 years. The Pirates, Steelers and Penguins gave the city (and me) plenty of enjoyment, and I spent many cherished moments in the various excellent places in this city. Few places can fit as well (in time and space) as Pittsburgh did for me!

Finally, I'd like to thank my parents, Kakali and Somnath Chaudhuri, and to them I owe my quest for improvement. They've always stressed the importance of knowledge, of understanding, of improving oneself, and they've always invested more of themselves in me than one could ever ask. This dissertation is theirs as much as it is mine.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Sound has a profound effect on our perception of the world around us. It affects our sense of beauty, level of alertness, ability to focus, and often our mental state. While various acoustic phenomena, such as speech recognition, language identification, music classification, have been well studied, a significant portion of the knowledge that would help advance automatic understanding from audio still elude us. Even though some of the specific tasks that audio understanding enables such as computer aided music, content-based retrieval of audio are the subject of significant research efforts, approaches toward developing a more general understanding of sound– relationships between various sound types, semantic links between sound events, discovery of sub-event structures– remain largely unexplored.

The world around us is structured in space and time, and the evolution of naturally occurring phenomena with time is related to the previous states. Thus, the changes that occur in any given scene in the real world are sequential by nature and the human brain can perceive and understand the sequential nature of these changes, as well as the semantic relationships between the various events; *e.g.* the movement of traffic and people at an intersection are governed by the traffic laws and changes in the traffic signals; a driver will sound his horn to warn another driver or a pedestrian. While the semantic information linking event sequences exists in both the visual and audio modalities, we believe that the audio alone carries significant information, and the automatic discovery of semantic structure from audio is the focus of this dissertation.

1.1 Motivations for this thesis

Automatic analysis of audio content is a key aspect of information retrieval systems [Foote, 1997, Wold et al., 1996] that deal with multimodal files. Specifically, for the purposes

of indexing and classification, audio content analysis research has focussed on specific tasks – detection of specific sounds, classification of the audio into categories, retrieval of documents in response to queries. While state-of-the-art techniques in some of these tasks perform extremely well (e.g. gunshot detection), these techniques have been largely developed in a task-driven manner, and do not provide a more general understanding of how the acoustics evolve with real-world semantic events.

Consider, as an example, processing the acoustics of an action scene from a movie. A listener might hear a loud *crack* followed by vocalized, high pitched sounds, some human speech, a repetitive electronic wailing sound, interspersed with more rapid *crack* sounds, and loud, mechanical humming. Based on the context, he would probably infer that the first *crack* sound was produced by a character firing a gunshot, the high pitched sounds those of people shrieking and screaming as a result of the gunshot. This probably led to police cars arriving, whose sirens created the repetitive electronic wailing sound and the cars produced the mechanical humming. This was followed by the authorities exchanging fire with the character(s) who were responsible for the initial gunshot.

Note that, in the absence of the sequence of events, it would be hard to precisely identify the individual sound and the semantics around it. For instance, a *crack* sound could be produced by a gunshot, or a car backfiring, or other chance events, such as a heavy metallic object falling on another metallic surface. Given the context of siren-like sounds and people screaming, one is more likely to infer that the source of the sound was a gunshot. Similarly, the sound of a siren alone does not indicate to a listener that the scene was an action scene. Sirens may be heard in a regular scene shot in traffic, or even from ambulances responding to non-action emergencies (e.g. medical emergencies). However, in conjunction with the gunfire around it, it seems likely that the scene was an action scene.

The example was intended to illustrate that the process of extracting semantic information from audio entails a number of inferences made at various levels. The lowest level involves identifying sources that produce the sound (*crack* of a gunshot or a car; siren of an ambulance or police cars; high pitched sounds from people screaming or children playing). A higher level might involve identifying what the co-occurrence of these sounds implies (in our example, an action scene), whereas a higher level still might involve identifying the genre of the movie. We note also that increasingly higher level inferences require being able to analyze a larger window of data– a gunshot may be inferred from the *crack* sound followed by people screaming, whereas the inference of an action scene requires looking at a wider span including the subsequent arrival of the police and exchange of fire, whereas the inference of the genre of the movie requires considering a number of such scenes for

relevant evidence. This leads us naturally to a hierarchical analysis framework, that we will describe briefly in this chapter, and in more detail in Section 2.1.1 of Chapter 2.

We posit that a truly intelligent listening system should be able to analyze audio content in a similar manner. Such analysis would then enable it to tackle not only the various specific audio tasks (such as detection of sounds or objects, scene classification, genre classification, etc.) but also create a learning framework that would enable it to truly *understand* objects, their various acoustic manifestations and how they affect the world around them. For instance, consecutive gunshots on a firing range are not cause for alarm, whereas if that were to occur in a public place, a listening system should raise an alarm. The ability to learn the relationships between various sounds, their sources and the surroundings would enable the development of intelligent systems that can understand and potentially respond.

The main hypothesis that drives this dissertation is that sound has its own *language* and *structure*, especially with respect to its semantic content. We would like to be able to discover such structure automatically from audio streams and exploit it to analyze the semantics in the audio better.

One of the main challenges of attempting to automatically analyze audio content in the present setting of rapid technological advancement is that a large proportion of content that needs to be analyzed is user-generated content, created in less than ideal recording conditions with limited resources. Thus, the recordings themselves will reflect a significant amount of real-world entropy, in terms of variations of sound effects, background noise, similar objects with very different acoustic signatures, etc. This kind of audio is often referred to as *audio in the wild*, and the ideas outlined in this thesis have been developed with primarily the *audio in the wild* setting in mind.

As mentioned earlier, most of the prior work in automatic semantic content analysis used task-specific approaches, such as building detectors for specific sounds or sources, and using the detection results to characterize content. These approaches primarily involved working individually on small segments of audio using supervised methods to detect patterns of interest. As a result, especially when working with audio in the wild, they face 2 important limitations.

First, the low-level sound or object detectors require supervised training data, and typically rely on audio concept libraries. Such libraries of concepts have typically been defined manually, and trained with data captured in clean, low-noise environments. Detectors trained from these libraries, therefore, are limited in their ability to represent data from new domains or deal with the kind of randomness that audio in the wild almost

certainly contains. Besides this, they depend heavily on the audio concepts in the library being largely relevant to the kind of data in the corpora being processed. As a result, the performance of detectors trained from these libraries may suffers when applied to noisy data, such as user-generated content on Youtube.

Second, such systems rely on the low-level audio concept coverage being sufficient to bridge the gap between the specific object detectors and a potentially semantic task. As in the earlier example, if one were to be looking to identify audio files corresponding to *emergency situations*, it would require the concepts *gunshot* and *scream* to be present in the audio library, as well as concepts relating to other manifestations of *emergency situations*.

Instead, a more feasible alternative to using supervised training for specific patterns would be to learn the various sound concepts from the corpora of interest, and use not only the low-level sounds but also local patterns of occurrence of the various sounds and their relationship to other sounds to address semantic tasks. While unsupervised approaches have been employed in past work, those approaches primarily relied upon frame-level or fixed frame windows for developing an unsupervised lexicon of low-level sound concepts. Further, thus far, there has been very limited interest in identifying deeper structure and using such structure for characterizing audio content for semantic tasks.

This thesis makes significant advances in addressing each of these limitations. We propose a novel task-agnostic, hierarchical framework that attempts to analyze audio content for semantic tasks, as described earlier. In Figure 1.1, we present a conceptual representation of a hierarchical framework that envisions a system to perform increasingly complex analysis of audio. The grey circles closest to the observed audio represent short-duration lower-level acoustic units which produce sounds that human ears can perceive, such as the *crack* of a gunshot, *clink* of glass, *thump* produced by footsteps, etc. These units have acoustic characteristics, but no clear associated semantics since the semantics may be context dependent. Sequences of these units, however, will have interpretable semantics– we refer to these as *events* marked by grey rectangles in Figure 1.1. The annotations in blue correspond to (usually unavailable) human labels for these events. Further, these events themselves likely influence future events, shown by the arrows, *e.g.* the loud cheering in the audio clip because a hitter hit a home run. Figure 1.2 shows an example of the structure we envision extracting from audio streams.

This hierarchical analysis structure can be exploited for various common tasks in audio analysis since the information contained in the different layers correspond to some of the common audio processing tasks. The root node in Figure 1.2 would typically correspond to the data genre, while the *events* layer corresponds to a segmentation of the audio file.

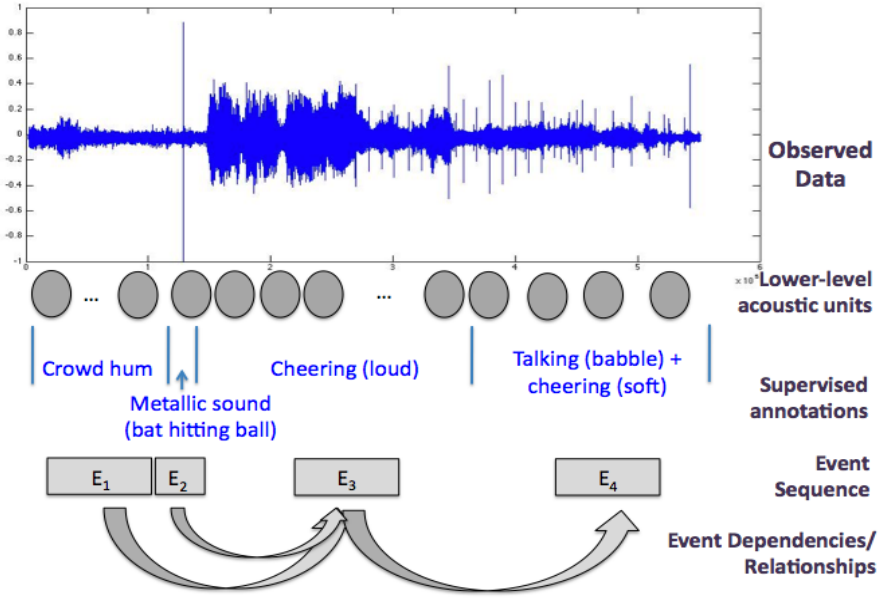


Figure 1.1: An example of the levels of analysis involved in the proposed hierarchical framework with increasingly complex semantic analysis. The color *blue* is used for data or human annotations, and *black* for the labels that should be generated by our system.

Each individual *event* or lower-level sound (or sound *source* might correspond to a sub-file level topic of interest, and temporally neighboring *event* or acoustic unit information can be used to further refine detection.

Beyond their direct applications in tasks like retrieval and recounting, structured, hierarchical relationships between various sound types, audio events would be interesting to analyze and understand, and can lead to the development of truly intelligent systems that can not just detect sounds of interest, but also understand them in a manner that humans are capable of doing. Such analysis can improve our understanding of the sequential (or co-occurrence) relationships of various sound types, e.g. music and whether sequences of audio events (notes, note sequences) can help identify composers, genres, etc.

The primary issue that arises in our setting for semantic analysis of audio is a scarcity of richly annotated data with information at various hierarchical levels that could be used in supervised settings, and efforts to obtain annotation for a reasonably sized audio corpus would likely be prohibitively expensive. To address this issue, we propose to use easily available data that only contain weak or no supervision, and perform learning in unsupervised or weakly supervised settings. The experiments reported in this dissertation will demonstrate that sufficient information can be extracted using such methods to allow semantic analysis of audio data.

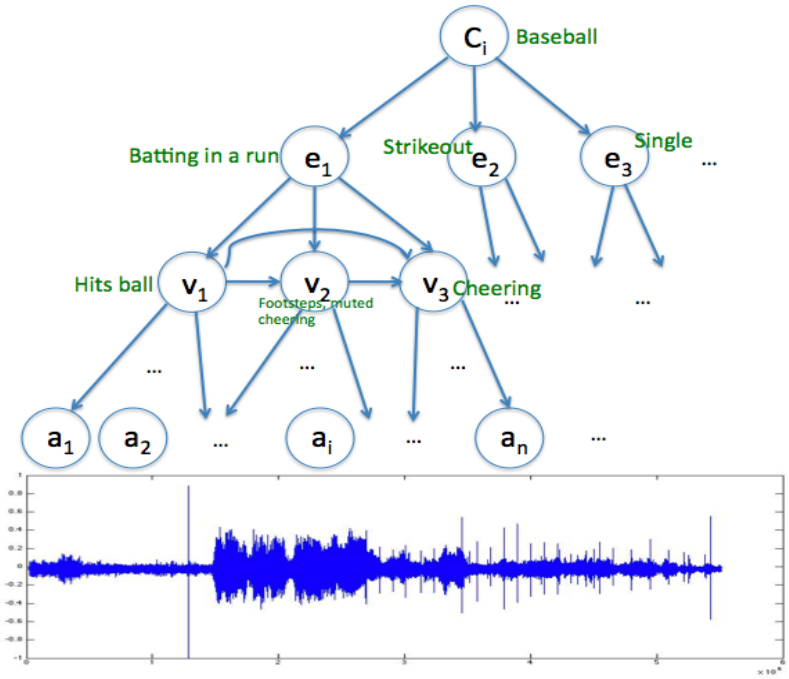


Figure 1.2: An example semantic parse for baseball.

We now proceed to present our thesis statement and contributions.

1.2 Thesis Statement

The main hypothesis that drives this dissertation is that sound has its own *language* and *structure*, especially with respect to its semantic content. In this thesis, we aim to show that:

"Structured models can be used to obtain increasingly higher-level semantic information from audio in unsupervised settings, and that such information is useful for semantically motivated tasks, even though the learning process was task-agnostic."

To support this statement, we present learning algorithms for the different layers of our framework shown in Figure 1.1, as well as test performance using the models for the various layers on the task of audio retrieval from semantic queries. In the audio retrieval task, we are given a database of recordings, and given a query, we need to retrieve all documents that are relevant to the query. Our experiments, which are discussed in detail in subsequent chapters, show that our framework for analyzing and characterizing audio data performs significantly better than current state-of-the-art techniques.

1.3 Summary of Thesis Contributions

To the best of our knowledge, this thesis is the first to propose a hierarchical, analysis paradigm (such as the one shown in Figure 1.1) for semantic analysis of audio content. The technical contributions of this thesis can be summarized as follows:

1. Proposed a novel hierarchical framework to map the observed acoustics to increasingly higher-level semantics, in a task-agnostic framework.
2. A framework for unsupervised learning of a vocabulary of low-level acoustic units that capture acoustically consistent phenomenon with no fixed window length-based assumptions.
3. An unsupervised approach to learning higher levels of the framework as local patterns over lower-level units, one layer at a time, with a prior that induces a power law distribution on the higher-level units.
4. An unsupervised approach to learning a full hierarchy jointly, starting with only the lower level acoustic units.
5. Empirically demonstrate performance improvements on current standard datasets using our approach.
6. A preliminary approach to learning a vocabulary of atomic, lower level acoustic units that assumes that the observed sound is a sparse mixture of multiple units.

The work done in this thesis also has an auxiliary contribution that could be very important for the development of research in this field, but which has not been explored directly in this thesis. The approach to induction of higher level structure leads naturally to segmentation of the audio stream that can be leveraged in the future to obtain rich annotations from human annotators who now only need to label the proposed segment as opposed to perusing the stream to both identify and label segments. This would provide a means of obtaining a set of inexpensive annotations, which can then be iteratively used in a supervised (or semi-supervised learning framework) to refine the structure induction process.

We would also like to note here that although the work in this thesis has been largely done in the context of audio, many of the modeling assumptions are general enough that they could be extended to arbitrary time-series data, where the assumptions hold. For instance, one could easily envision extending the entire framework to a cognitive framework that analyzes concurrent information from multiple streams to produce a single vector as a joint representation.

1.4 Thesis Layout

The rest of this thesis is organized as follows. In Chapter 2, we will review past work in audio content analysis as well as work in various other areas of research that relate closely to the approach that we've adopted in this thesis. We then introduce the data sets that we use for our experiments and present an empirical analysis using standard methods commonly used in content analysis tasks.

In Chapter 3, we present our approach to learning the low-level acoustic units unsupervised from data. We present experimental results using only these low-level units on a few different tasks, and investigate the effects of various shallow structure that influence the design of the learning schemes for the higher layers.

Chapter 4, we present a power-law prior driven approach to learning higher layers, one layer at a time, incrementally building over the previous layer induced.

Chapter 5 presents our approach to inducing a full hierarchy on top of the low-level acoustic units, and investigates the tradeoffs with the *one-layer-at-a-time* approach of Chapter 4.

In Chapter 6, we present results of our preliminary investigations of approaches to 2 important future extensions of the ideas laid out in this thesis– first, learning atomic low-level acoustic units, where the observed audio is generated as an additive mixture of multiple sources; and second, an approach to making the vocabulary learning process for higher layers non-parametric.

Finally, Chapter 7 concludes by summarizing the work presented in this thesis, and a discussion of future directions of research.

Chapter 2

Audio Content Analysis– Background and Related Work

In Chapter 1, we presented an overview of the content analysis problem as it relates to audio, and a brief summary of the limitations of current systems that attempt to perform automatic content analysis *in the wild*. In Section 2.1 of this chapter, we present a broader look at some of the relevant work in audio content analysis, and use this context to describe our framework in greater detail as well as highlight the main similarities and differences with past work. As mentioned earlier in Chapter 1, hierarchical annotations for the audio of the kind that our framework attempts to discover are not currently available. Therefore, any research that attempts to discover any level of semantic structure from audio would need to be able to work in unsupervised or weakly-supervised settings, since rich annotations sufficient to learn the kind of semantics we propose would be hard to obtain. We discuss some of the challenges with respect to good datasets in Section 2.1.3 and present a brief overview of some weakly supervised learning techniques in Section 2.1.4.

The hierarchical paradigm presented in this thesis is rather novel in the context of audio content analysis but bears some similarities to a large body of work in text processing. We review some of the relevant work in text processing in Section 2.2, while pointing out some of the principal points of difference in the two paradigms.

The current, openly available datasets, however, contain no supervision at all in terms of semantic structure. Thus, the models we propose in this thesis for hierarchical structure induction work unsupervised, employing latent variable models. We provide a brief introduction to latent variable modeling and the Expectation-Maximization technique for parameter estimation that we use in this work in Section 2.3.

Finally, we present 2 datasets in Section 2.4 of this chapter that we use for experimental

evaluation of our framework, and present the results of applying a pair of commonly used approaches to audio content analysis to these datasets, which we shall later compare with our techniques.

2.1 Automatic Content-based Audio Processing

The rapid increase in popularity of web-based systems that allow users to share data on the web has resulted in an unprecedented increase of user-generated multimodal content on the internet. Automatic analysis of the content of these files is essential in order to be able to index and retrieve relevant files in response to user queries. The analysis of the audio and video modalities of the content have typically been done separately, with their respective results then combined to produce a joint characterization of the file. These results could be feature level analysis results that are fused and then used to train a classifier, or they may be independently used to train classifiers and the classifier predictions for the two modalities might be combined to produce a final prediction.

Since the audio in the multimedia files can be expected to provide significant information with respect to its semantic content, and potentially complementary information to what can be obtained from analysis of video (*e.g.* due to the presence of speech), various approaches have been developed to analyzing the audio content. One of the earliest paradigms of indexing audio content was to annotate it with textual information and then use traditional IR techniques for searching. This approach works well and has the advantage of using well-understood techniques for retrieval. On the other hand, obtaining human annotation of audio is extremely time-consuming and expensive. As a result, there has been significant research effort attempting to automatically utilize the audio content for indexing of multimedia files.

Content based audio classification is essentially a 2 step pattern recognition problem– first, the audio is represented using a set of features. These features are then used for classification. The earliest efforts [Liu et al., 1997, Wold et al., 1996] sought to match perceptual features of audio files to an audio query.

As speech recognition systems improved, the task of spoken document retrieval using text queries on large speech corpora [Garofolo et al., 1997] was studied more closely. The standard approach used by systems in this setting involve transcribing the speech signal, and using text-based retrieval techniques on the transcribed text. Systems that could perform keyword spotting [Szoke et al., 2005] and query by example [Velivelli et al., 2004] were also developed for retrieval from these large speech databases.

The task of understanding non-speech sounds, however, is a much harder one. In settings where the domain of sounds is unconstrained (audio *in the wild*, as it is often called), one can imagine an infinite number of potential sounds. To avoid having to explicitly model a very large number of sounds, approaches to audio processing in this area were task-driven– *e.g.* detecting specific sounds in audio using detectors that sought to classify chunks of audio as containing the target sound or not. The most common approach is to use a *vocabulary* of sounds, comprising clearly characterizable sounds such as gunshots, laughter, speech, animal sounds, music, crowd sounds etc. Audio is analyzed by detecting the presence of sounds from this vocabulary in it and additional analysis builds on top of such detection. For instance, Chang et al. [2007] identify the presence of sounds from a vocabulary and combine this information with evidence from video. Slaney [2002] describes a system that could be used to map between regular vocabulary and sounds by association. Friedland et al. [2009] navigates Seinfeld episodes taking advantage of traditional sitcom artifacts, such as music indicating scene changes and laughter following punchlines. Other analyses detect repeated sequences in a television broadcast stream [Berrani et al., 2008], with the intent of identifying jingles, advertisements and so forth. Many of these methods work well in restricted domains, and based on these techniques, an unconstrained, completely automatic system for audio understanding can be envisioned.

Semantic analysis of audio has also been explored recently in multimodal settings. Jiang et al. [2009] utilize the concept of Short-Term Audio Visual Atoms (S-AVA) where features are extracted from both the video and the audio and are used together to develop codebooks for various semantic concepts. The codewords in the codebooks are then used as features to train classifiers to detect the concept. Lee and Ellis [2010] used a set of 25 semantic classes (*e.g.* dancing, singing, birthday) for classifying consumer video clips using only audio information. Rui et al. [2000] attempted to generate highlights for baseball audio using information from the audio track only using energy-based features, as well as phoneme-level features and prosodic features from the announcers' speech. Divakaran et al. [2003] used audio features in conjunction with video features based on motion activity for news video summarization– they used Hidden Markov Models for modeling various kinds of audio events such as speech, barking, etc. which were then used to segment the sound track, as well as detect speakers and speaker changes.

2.1.1 Structured Analysis and Our Proposed Framework

There are 2 principal distinctions between the various threads of past work discussed earlier in this area, and the approach that we propose.

First, in all of the above, the basic mechanism involves *spotting* a set of known sound types in audio, with a specific task in mind. Higher-level descriptions of audio must be obtained by further inference or by human supervision, after the sounds are detected. Especially for semantically motivated tasks, an approach based on a spotting or detection-based paradigms makes the important latent assumption that the observed acoustics map directly to the semantics. Naturally, this is not the case for approaches such as Lee and Ellis [2010] which use a set of pre-defined *semantic* classes. However, all of them do run the additional risk of trying to map acoustics across 2 datasets with dissimilar characteristics. Such dissimilarity may arise from differences in recording devices, conditions, degree of naturalness, as well as simply from the presence of significantly different concepts, as might occur when recording a similar concept (*e.g.* wedding) from different cultures.

Second, the various content analysis techniques that are popular in the literature utilize a *shallow analysis* of the surface form or representation obtained via the different paradigms. While some have looked at some degree of structure between these surface forms, as captured by language models, we will argue in Chapter 3 and 4 that the kind of structure that exists in semantic audio would be extremely hard to capture with regular language models due to their high variance.

To understand the intuition behind this, consider 2 plausible manifestations of a crime scene event as shown in Figure 2.1. Both are very similar in terms of the higher level semantic content, which involves a crime committed with a gun. However, they manifest differently enough that a simple language model would not be able to capture the similarity between the two manifestations. As indicated in the figure, noise may also be added due to imperfections at the time of decoding of the lower layer introduced by modeling issues, presence of background noise that distorts the true characteristics of the data, etc. In either case, however, the system can be robust to the shallow surface form by doing another higher level of analysis, where it starts to learn that the presence of *gunshot* and *door slam* is sufficient to indicate a crime scene, independent of the presence of *speech*, *water* or any other sounds.

In our work, we take a different approach to address these issues. All our approaches involve learning from the dataset of interest, in order to avoid domain mismatches, or insufficient concept coverage. Our methods can work unsupervised, and therefore do not the high cost of obtaining human annotations, As our baseline experiments using prior work from the literature will show, unsupervised lexicon learning techniques will typically outperform supervised, audio concept dictionary-based techniques on user-generated content.

In order to address the issue of how to map the acoustics to the semantics, we employ

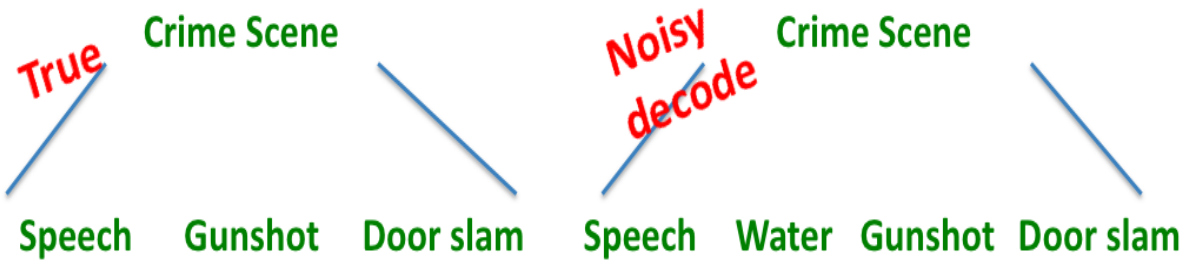


Figure 2.1: An illustrative example to show how the same semantic event might manifest differently. The difference in manifestation might be the true difference between 2 occurrences of the same event, or it may happen because the decoder that identifies the lower level events is imperfect (denoted by *noisy decode*).

the hierarchical framework that was introduced in Figure 1.1 and 1.2 in Chapter 1. We will briefly explain the use of the multiple layers in the hierarchical model of Figure 1.2 here.

A higher layer in our framework corresponds to a higher level of semantic inference. The lowest level, indexed by a_i in Figure 1.2, is the closest to the audio and it captures more of the acoustic phenomena than semantic meaning. At this level, we model all audio as being composed of a sequence of a relatively small set of atomic sound units, which we call *acoustic unit descriptors* (AUDs, henceforth). Our hypothesis in attempting to formulate a language for sound is that sound can be modeled with a small number of lower level units (AUDs, in our case). Since the number of different sound concepts in the real world can be infinite, we would need some sort of limiting criterion for computational benefits. Our assumption of a finite set of low-level sound units is motivated by the benefit of computational tractability as well as 2 other observations. First, even though the full set of sounds might be infinite, only a limited (although, possibly large) number will occur in any given dataset. Second, the assumption of a smaller than true number of units will force multiple different concepts to cluster with the most similar acoustic concepts. As a result, these unit clusters will not be true, diverse units, but capture acoustically similar phenomenon.

As a result, individual AUDs do not necessarily capture semantic information, but sequences or patterns of AUDs should capture semantic information. Every instant of an audio file is part of one such unit, and the entire audio stream can be *transcribed* in terms of these units. However, we do still expect that individual AUDs do capture some underlying semantics, even though we may not be able to quantify this in terms of generally understood semantic concepts. As an intuitive example, the crack of a bat might fall into

the same AUD cluster as the sound of a hammer. While we cannot distinguish between a *baseball* tag and a *workshop* tag, we can use it to say that such sounds are not very likely in the *musical performances* class.

Once the acoustics have been mapped to the AUDs layer, we start to look at the higher *events* layer. Our expectation is that as we go up the hierarchy, the segment captured by the span of the node becomes increasingly more semantically informative in nature. To extend the example from above, as we look at AUDs that appear in temporal proximity to the one that capture the sound of baseball bats and hammers, if we find AUDs corresponding to applause or cheering or general merriment, we would predict that the sound was from the *baseball* class, since a workshop project using a hammer rarely elicits applause.

As we go higher still, we are better placed to make even higher level inferences. For instance, even though a short segment in the recording might have been from a baseball game, the recording may actually be of a birthday party where the camera focussed on a baseball game on television for a while. Thus, to make an inference about the overall genre of the audio, we need to combine evidence from multiple segments.

In Chapter 3, we will explain our model for learning the AUDs and how we use them for audio processing tasks. Chapter 4 explains how we utilize information from local patterns over AUDs (or similar units) to develop higher layers in the hierarchy, and both Chapter 4 and 5 explore two different approaches to building up a hierarchical structure on top of the low-level acoustic units.

While techniques for analysis of general audio does not typically pay attention to structure, there has been a great deal of work on using grammars for generating or analyzing music. Early approaches to grammar based generation of music employed simple, deterministic rule-based techniques [Steedman, 1989]. Researchers have attempted to use various kinds of grammars for different tasks– tree grammars were used to compute melodic similarity computation and melody classification [Bernabeu et al., 2011], analysis of musical structure to recover the sectional form of a musical piece using MFCC, chroma and rhythmogram features [Paulus and Klapuri, 2009], and using regular grammars to model musical style for classification [Cruz-Alcazar and Vidal, 2008].

2.1.2 Similarities to Image Processing Approaches

Image processing researchers have attempted to employ grammars to improve the performance of computer vision systems. Early efforts to building such systems attempted to exploit the fact that the evolution of a visual scene is guided by different conventions, such as the laws of traffic or social conventions (such as expecting people to sit on chairs, or

expecting to see food on tables). Subsequently, research efforts have sought to generate conceptual descriptions from sequences of images [Nagel, 1988] using intermediate levels of description using verbs to describe temporal changes, events and histories in order to go from frame level understanding to a schematic story.

To the best of our knowledge, Christensen et al. [1996] were the first to suggest using grammars for the task of describing a scene, using a rule-based grammar for this task. Subsequently, the problem of object detection has motivated the formulation of grammar based approaches, where the objects of interest are modeled as being composed of parts which are objects, as well. Such models [Felzenszwalb and McAllester, 2010] also make efforts to distinguish between different compositions leading to the same *object*, which would potentially provide semantic information, *e.g.* distinguishing a smiling face from a frowning face, using the same *parts* to detect the face. Felzenszwalb et al. [2010] outline a cascade detection algorithm for a general class of models defined by a grammar formalism. This class includes tree-structured structures as well as richer models that can represent each part recursively as a mixture of other parts.

Recently, image processing researchers have proposed the generation of hierarchies from images using both visual and semantic information. Generative models have been proposed for the same [Li et al., 2010c], and the success of the models in generating hierarchies using image information augmented with text was measured on image classification tasks as well as by human judgment. While low-level image features, such as pixels, have proved to be strong features for many image tasks such as classification, work by Li et al. [2010b] found that the use of features based on detected objects in an image provide complementary information to the low-level features, and can be used to enhance performance.

2.1.3 Enriching current datasets

The most important restriction that would limit the development of techniques exploring better and more interesting ways to model semantics in the audio (and possibly, video) modality is the limited amount of annotated data available for such tasks. Ideally, one would be able to evaluate structure induction techniques directly on the structures, but this is presently impossible, since there exist no datasets with the kind of rich annotations that such structure evaluation would require. The closest parallel to the kind of data one would be looking for would be in the domain of syntactic parsing for text, which we shall discuss shortly, where treebanks have been developed for that purpose. However, the development of treebanks required consensus on a number of decisions related to syntactic structures, something which might be hard to obtain for a semantic task for audio.

As a result of the above factors, decisions on how to use annotations at various semantic levels for evaluation would need to be closely monitored. Nonetheless, the importance of generating such annotations is paramount, so that the community can begin to explore techniques to leverage them. There are 2 possible ways of obtaining such data.

The first would be to employ annotators, agree upon a minimal set of annotations and allow them to annotate the tree structure with those labels. This might, however, prove prohibitively expensive since the task of identifying and labeling boundaries has been found to be very slow, since it would require identifying boundaries of segments such as the low-level acoustic units, all the way to the highest genre node. Indeed, for speech, accurate labeling of utterances is extremely time consuming and requires trained linguists.Zhu [2004] reports that annotation at the word level can take ten times longer than the actual audio (e.g., one minute of speech takes ten minutes to label), and annotating phonemes can take 400 times as long (e.g., nearly seven hours).

The second approach would involve the setting up of weakly supervised learning techniques that can use minimal supervision to improve, or ask questions to users to maximally improve its understanding of certain structural elements, use this information to refine its structure learning, as well as ask questions to verify the correctness of its knowledge. SImilar efforts have been undertaken in different communities, with LabelMe, Captcha, etc being examples of available technology that use human input for such tasks.

Further, such systems need to be robust to various issues. Besides the obvious one of annotation errors arising out of misclassifications, Tzanetakis and Cook [1999] raise the issue of subjectivity of the listener in assigning annotations to audio, and suggest an interactive framework that combines manual and automatic annotations into a flexible, unified framework. Our framework is equipped to handle such situations, as well. We will describe in Section 3.2.3 a method that can be adapted to take into account individual preferences in generating labels for segments of audio that can easily be personalized for different users.

2.1.4 Weakly-Supervised Learning

We include here a brief summary of some general principles of weakly supervised learning approaches that we leant on, given that the degree of supervision in our setting was minimal. More importantly, we believe that as this area of research develops, being able to leverage weak supervision meaningfully will prove extremely important.

Traditional classifiers use supervised data with feature and label pairs for each data point in the training set. However, for a variety of reasons such as human effort and

time required for labeling, such labels may not be available in all settings. Unsupervised approaches, on the other hand, make use of large quantities of unlabeled data that is easily available, but do not require annotated labels for the training data. Semi-supervised learning is a special learning setting which attempts to leverage the best of both worlds. It makes use of large amounts of unlabeled data in conjunction with a small amount of supervised labeled data to build better classifiers [Zhu, 2004]. Semi-supervised approaches require less human effort and often provides significantly improved performance making the approach interesting in both theory and practice.

It should be noted here that semi-supervised learning approaches are not guaranteed to increase performance over unsupervised approaches. For instance, Elworthy [1994] observed that training a Hidden Markov Model [Rabiner and Juang, 1986] can reduce accuracy under certain initial conditions.

Various algorithms have been developed for semi-supervised learning– self training (which has been used successfully in and co-training are two of the most popular approaches. Self-training uses the labeled data to learn a classifier and then uses this classifier to predict classes on the unlabeled data. The unlabeled data points that this classifier can make predictions on with the most confidence are added to the labeled training set, and this process is repeated, with the classifier using its own predictions to teach itself. Self-training approaches have been used for word sense disambiguation [Yarowsky, 1995] and for detecting objects in images [Rosenberg et al., 1995].

Co-training [Blum and Mitchell, 1998] assumes that features can be split into two sets and that each sub-feature set can be used to train a good classifier, provided the two sets are conditionally independent given the class. Initially, two separate classifiers are trained with the labeled data, on the two sub-feature sets respectively. Each classifier then classifies the unlabeled data, and teaches the other with the few unlabeled examples that they can classify with the most confidence added to the supervised data set. Each classifier is now retrained with the additional training examples given by the other classifier, and the process repeats. Thus, the two classifiers must agree on the much larger unlabeled data as well as the labeled data. Nigam and Ghani [2000] showed empirically that co-training performs well if the conditional independence assumption indeed holds.

In the case of sound data, semi-supervised learning approaches are important for 2 reasons. First, for the kind of semantic structure we are attempting to discover from audio, annotated data with rich annotations would be expensive to obtain, and being able to leverage a small amount of annotations would be important. Second, audio datasets which provide class labels on the entire files are available. However, especially in the case

of user-generated content, labels simply refer to the subject that the audio was attempting to capture; the audio may and does contain significant amounts of content not related to the topic, due to backgrounds, voice overs and overlaid commentary or music. In principle, the second issue can be tackled using a learning setting known as multiple-instance learning [Zhou, 2004], which has been used for various tasks, such as drug activity prediction [Dietterich et al., 1997], music retrieval [Mandel and Ellis, 2008] and visual tracking [Babenko et al., 2009]. However, as we shall see in Chapter 6, successfully tackling this issue with multiple instance learning is likely to be a significant undertaking by itself.

Finally, yet another learning paradigm that may be of interest due to the scarcity of labeled data for our tasks is that of active learning [Settles, 1995]. The main idea behind active learning is that a learning algorithm can achieve greater accuracy with fewer training labels if it is allowed to choose the data from which it learns. An active learner may pose queries, usually in the form of unlabeled data instances to be labeled by an oracle (e.g., a human annotator). Active learning is used in a variety of applications, where unlabeled data may be abundant or easily obtained, but labels are difcult, time-consuming, or expensive to obtain.

2.2 Relevant Text Processing Paradigms

The various layers in the structure in Figure 1.1 correspond to understanding the semantic content of the audio at various granularities. Such a structure is quite similar to those produced when parsing natural language text to understand the syntactic structure of the text. In addition to this, the task of identifying meaningful segments from streams of the low-level acoustic units (or even at higher levels) are tasks with analogs in the text analysis domain. The task bears certain similarities at a high level to approaches in morphology induction and topic modeling. Section 2.2.1 discusses some of the relevant literature in the syntactic parsing domain, while Section 2.2.2 discusses relevant topic modeling work.

2.2.1 Syntactic Parsing

The process of parsing enables the identification of the structure of meaningful subsequences of any text sentence, including phrase boundaries spanning sets of words and identification of the parts-of-speech for the individual words, producing a syntactic structure as shown in Figure 2.2 [Charniak, 1997]. It is generally accepted that the parse tree obtained in this manner is useful in understanding the sentence automatically, and

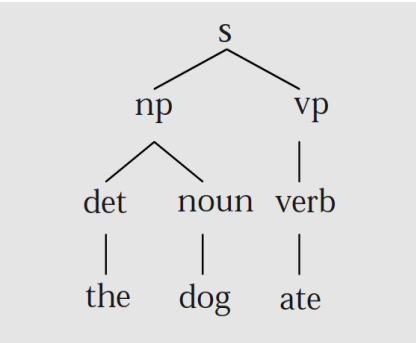


Figure 2.2: Example of a syntactic parse of a sentence (from Charniak [1997])

features derived from such parse trees are used for a variety of applications, including question-answering, machine translation, textual entailment.

The current state-of-the-art systems for natural language parsing all employ statistical methods to determine the most likely syntactic structure for any given sentence. From the parse tree shown in Figure 2.2, one can see that words are first categorized into parts of speech (POS), and phrase boundaries are obtained over sequences of words (in the Figure, NP refers to a noun phrase, and VP to a verb phrase). Finally, the sentence is a sequence of such phrases– note that a phrase can span sub-phrases as well. Consider a sentence such as "The dog ate the biscuit". In the parse for this sentence, "the biscuit" would have parts of speech "det noun", which would be spanned by an NP. Then the VP would span "verb NP", thus spanning the words "ate the biscuit". In some ways, this structure is analogous to our framework, where the AUDs learning process is independent of the higher structure induction, just as the POS tagging is done for the different words in a sentence independent of the syntactic structure prediction.

Openly available part-of-speech taggers include the Stanford POS Tagger [Toutanova et al., 2003], and the CRF Tagger [Phan, 2006], and implement various methods ranging from Hidden Markov Model formulations [Rabiner and Juang, 1986] to Maximum Entropy techniques [Toutanova and Manning, 2000] and Conditional Random Fields [Lafferty et al., 2001].

As a precursor to full parsing, many researchers worked on the task of chunking [Abney, 1991, Ramshaw and Marcus, 1995], which involved dividing sentences into non-overlapping segments based on analysis of the words over windows. However, statistical parsing techniques are currently sufficiently developed to generate full parses for sentences of the kind shown in Figure 2.2. Again, statistical parsers for full parsing of natural language sentences employ a variety of models including probabilistic context free grammars and maximum

entropy models and are now openly available [Bikel, 2004, Klein and Manning, 2003].

In Chapter 4, we develop an algorithm for developing higher layers on top of the AUD sequences. Unlike in text, where the POS tagging is done at the word level, in our case, the notion of words is ill-defined. As we've discussed earlier, we expect the semantics to be contained in the local patterns over AUDs (just as the true semantic of a word is only obtained from the local context), and thus AUDs may be considered analogous to words in text. The algorithm we propose for developing higher layers in Chapter 4 can then be considered to be performing an analogous function to *chunking* for text, with the exception that due to the nature of the audio content, the chunks represent semantically coherent segments instead of being syntactically bound as in text. Due to the fundamental differences in what they are trying to accomplish and the different challenges in the different domains, our learning algorithm is not at all similar to the standard chunking algorithm, instead bearing more similarity to the semantically driven topic modeling literature, which we discuss in Section 2.2.2.

Overall, however, the parse tree representation of natural language text is very similar to the structure shown in Figure 1.1 for analysis of audio content. One of the approaches to inducing a hierarchy that we describe in detail in Chapter 5 is inspired by work in unsupervised structure induction for text Klein and Manning [2001].

In conclusion, we can summarize the main difference between the parsing paradigm for natural languages and the discovery of structure among audio events as follows: the parses for natural language text are syntactic, and enable analyses of the syntactic relationship between words, and of the sentence as a whole. Relationships between events in audio are semantic, and the accuracies of the inferred dependencies are likely to be more sensitive to the detection of individual events. Further, text used in these settings are usually ground-truth and noise-free, while the analogous discrete sequence for sound (the Acoustic Unit Descriptors) are likely to contain significant noise.

2.2.2 Topic Modeling

In Section 2.1.1, we discussed how our modeling assumptions and choices led us to expect that the individual low-level acoustic units (AUDs) would not contain individual semantic interpretations; instead, local patterns over these AUDs would contain semantic information.

Such an analysis paradigm for semantic inference is analogous to principles used in word sense disambiguation tasks in text processing. Consider, as an example, the word *bank*. It could refer either to the financial institution or the shore of a river. Indeed, it

could even function as a verb as in the usage *banking on*, to mean *relying upon*. While various different modeling techniques have been used for word sense disambiguation, the driving principle remains one of exploiting context to determine the true sense of a word.

Our principle in trying to extract semantics from AUDs (or indeed, any layer being mapped to a higher layer) is to understand not only the semantics, but also the context and the role it plays in the development of future events and, therefore, acoustic phenomenon. Thus, in trying to develop a layer that captures scene level information, one would expect to find that the unit corresponding to a *crime scene* (to continue the example from Chapter 1) would often generate AUDs corresponding to gunshots, screams, sirens, etc. Note however that not all instances of crime scenes will have all of these but (let's assume here that a gun is the only way crimes may be committed) they will all contain the gunshot. An intelligent learning system would be able to learn that a sound like a gunshot becomes more likely to be a gunshot if screams, sirens, more gunshots, etc are present in temporal proximity (AUD sense disambiguation), and that the sequence of sound events (gunshots, etc) from the beginning of the excitement correspond to a *crime scene* unit.

Thus, at a local event level, we introduce the concept of an event as a distribution over AUDs, since not all of those elements need to be present. This is analogous to document understanding approaches using topic modeling, where the documents are distributions over words, and documents belonging to a particular class are likely to display some similarities in distribution while being different from documents in different classes, although they may use the same vocabulary. Similarly, different events will likely have different distributions over AUDs.

We will describe our specific model for inducing higher-level events in Chapter 4. In this section, we provide a brief overview of relevant topic modeling literature that will be relevant to understanding our model.

The goal of topic modeling is to find appropriate characterizations of the members of a collection (such as, documents belonging to a particular class in a corpus) that enable processing of large collections for tasks such as classication, retrieval, summarization, etc. The early methodology for such tasks involved reducing each document in the corpus to a vector of real numbers. In the popular *tf-idf* scheme [CITE Salton and McGill, 1983], a vocabulary of *words* is chosen, and, for each document in the corpus, a feature vector is created of the size of the vocabulary with the normalized frequency of occurrence of each word being the feature value. This is then combined with an inverse document frequency and used as an indicator of saliency.

While this approach was reasonably successful, the feature descriptor was typically

of high-dimensionality, and the resulting representation was not particularly insightful in understanding structural aspects of documents. To address these problems, probabilistic Latent Semantic Indexing (pLSI) using the aspect model was introduced, where documents were modeled as *distributions over topics* and topics modeled as *distributions over words*, such that the high probability words for any topic would provide insight into how the *topic* might be interpreted. This is currently the accepted paradigm for document analysis tasks in text processing. A number of variants of this approach have been developed since to add more structure information including the popular Latent Dirichlet Allocation (LDA) Blei et al. [2003]. pLSI, LDA and its variants have been used successfully for a variety of tasks, such as document classification and retrieval, conversation modeling, summarization, etc.

Briefly, the aspect model (and LDA) provide a generative schema to generate text documents probabilistically, where the for each document (document d with N_d words), a latent topic (z) is selected from a distribution $P(z|d)$ and then a word w is drawn from the distribution of words for z, $P(w|z)$, and this process is repeated N_d times to generate a document. The LDA varies in that it doesn't have a new topic distribution for every document, instead using a distribution over topic distributions. We use a similar generative process to model higher layers of the hierarchy with the AUDs being the observed layer, but with modifications to handle significant differences between the basic domains of text and audio, as well as modified modeling assumptions to deal with the fact that the observed low-level AUDs are the result of noisy decodes, whereas the observed words in text are usually ground truth.

The topic models are a specific instance of latent variable modeling, which we will briefly discuss in Section 2.3. Finally, unlike in text, where word boundaries are known, our approach to inducing events requires us to estimate the events as distribution over AUDs, as well as identify the segment corresponding to the event, jointly. One can think of an analogous task to this in text processing, where given a stream of space-removed characters in a strange language, the task is to discover the vocabulary of the language. Such approaches are, in fact, employed for morphology induction as well as vocabulary learning in the literature. However, these are not of general interest to this research area, so we will discuss them in Chapter 4 in the context of our model for performing joint segmentation and *event* learning.

2.3 Latent Variables and Expectation–Maximization

Audio data sets, especially audio in the wild, do not currently contain the kinds of rich annotation that would be required in order to train fully supervised models to predict the kind of hierarchical structure we envision in Figure 1.2. Thus, as we shall see later in Chapters 3, 4 and 5, the various units in the hierarchy are modeled using latent variables, where the only observed data is the actual sounds and the various levels of the hierarchy are estimated from that set of observations.

In general, latent variable models are a powerful tool for probabilistic modeling where the set of observations are augmented with additional hidden variables that attempt to capture a set of beliefs regarding how the observed data was generated. In our instance shown in Figure 1.2, one can think of the generative process as one where an underlying latent process first picked a genre from a distribution over genres, and then picked a sequence of events from the distribution over events conditioned on the chosen genre (baseball). For each of those events, sequences of sub-events are drawn that then manifest themselves as a sequence of low-level acoustic units (*e.g.* crack, thud, clap) which in turn produce the characteristics of the raw audio. An example of these characteristics would be Mel-Frequency Cepstral Coefficients (MFCC) that are commonly used to represent audio.

Figure 2.3 shows an example of a simple generative process, using the parameters of the grammar at each level, with 3 levels instead of the 4 used in the baseball example, and the lowest level correspond to the Acoustic Unit Descriptors. The actual observed data is not shown here, but each of the AUDs produce the equivalent of raw audio. The example also assumes that units in the same level are independent of each other, and the grey boxes with G denote the distributions from which the units in the different layers are drawn, conditioned on the unit in the layer above it.

In such latent variable models, a joint distribution is then defined over the latent and observed variables, and the corresponding distribution of the observed data can be obtained by marginalizing the latent variables. Typically, a form for the distribution of the latent variables is chosen and the training process attempts to estimate the parameters of the graphical model thus formed that can explain the generation of the data, typically either in a maximum-likelihood or maximum-a-posteriori setting. While the specific learning setup depends on the choice of the underlying model and the distributions, there are a number of commonly used examples of latent variable models such as Hidden Markov Models, Probabilistic Latent Semantic Analysis, Conditional Random Fields and Topic Models that have been successfully used for a wide range of tasks.

While various techniques have been presented in the literature for training the param-

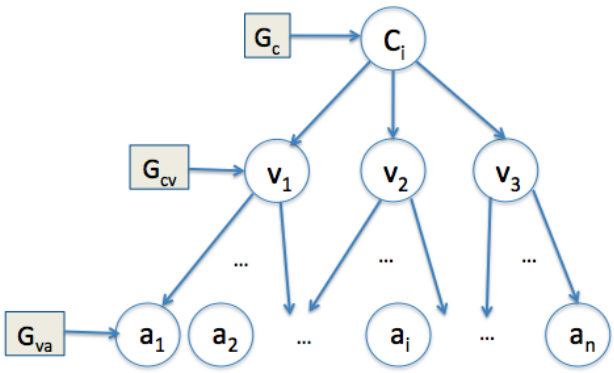


Figure 2.3: An example of a simple parse tree over AUDs..

eters of latent variable models, one of the most commonly used ones is the Expectation Maximization (EM, henceforth) approach, which we will briefly introduce here. The EM algorithm is an efficient iterative procedure to train the parameters of a latent variable model. It starts with an initial guess of the parameters that it seeks to estimate better, and arrives at a more optimal parameter estimate via iterative updates of its current parameter estimate. The EM algorithm is guaranteed to converge. Each iteration of the EM algorithm consists of two processes: The E-step, and the M-step. In the expectation, or E-step, the missing data are estimated given the observed data and current estimate of the model parameters. This is achieved using the conditional expectation, explaining the choice of terminology. In the M-step, the likelihood function is maximized under the assumption that the missing data are known. The estimate of the missing data from the E-step are used in lieu of the actual missing data.

While for detailed information about the various properties of the EM algorithm, we refer the reader to Dempster et al. [1977], it can be shown that we can improve the probability of observing the data when compared to the parameter estimate in the r-th iteration as follows.

Let us define a model where $\mathcal{X}$ are the observed data, $\mathbf{Z}$ are the latent random variables, and $\mathbf{\Theta}$ are the set of parameters we wish to estimate. One can define an objective function that is guaranteed to increase when we estimate an updated $\mathbf{\Theta}$ given our previous estimate of $\mathbf{\Theta_r}$.

$$\Delta(\Theta, \Theta^r) = \sum_Z P(Z|\mathcal{X}; \Theta^r) log \frac{P(\mathcal{X}, Z, \Theta)}{P(Z, \mathcal{X}, \Theta^r)} \quad (2.1)$$

$$= E_{Z|\mathcal{X};\Theta^r}\left[log \frac{P(\mathcal{X}, Z, \Theta)}{P(Z, \mathcal{X}, \Theta^r)}\right] \quad (2.2)$$

The LHS in this equation is guaranteed to be non-negative since setting Θ to Θ^r will result in the RHS evaluating to 0.

Our new estimate of Θ corresponds to the maximization of the function $\Delta(\Theta, \Theta^r)$, and can be obtained by differentiating the RHS in Equation 2.2. Note, however, that we can use a simpler function since the denominator within the logarithm is not a function of Θ. We can rewrite it as follows, and maximizing the RHS of Equation 2.3 will give us our optimal estimate of Θ in the following iteration.

$$\Delta(\Theta, \Theta^r) = E_{Z|\mathcal{X};\Theta^r}\Big[logP(\mathcal{X}, Z, \Theta)\Big] \quad (2.3)$$

We will use this for our derivation of the update rules in the models we subsequently present.

2.4 Data and Baselines

In this section, we will first describe the data that will be used for evaluation of the framework and algorithms developed in this thesis in Section 2.4.1. We will follow with a brief discussion of a pair of analysis techniques in Section 2.4.2– one supervised and one unsupervised– that are commonly used for audio content analysis, and establish baseline performances on the 2 datasets of interest, while explaining how these analysis techniques are used in a system to perform audio retrieval.

2.4.1 Datasets Used in this Thesis

There are 2 datasets that we will use for our experiments in this dissertation. The first is the BBC Sound Effects library (BBC, henceforth) which consist of 10 different categories and 1120 different audio clips [bbc]. This library consists of various conceptual categories of sound, and audio tracks for the various categories contain complex audio due to the presence of many different sounds; *e.g.* a supermarket audio contains voices, sound from the checkout bell, trolleys and baskets being stacked. Thus, these categories are defined

Category	No. of positive instances	Total duration (hh:mm)
Exterior atmospheres	53	03:02
Household	51	00:59
Interior Backgrounds	24	01:05
Transport	135	02:37
Animals	52	01:06
Audiences	47	00:32
Electronic equipment	69	00:47
Water	24	00:34
Birds	43	01:07
Warfare	138	00:47

Table 2.1: The various categories in the BBC dataset and occurrence statistics.

at a higher semantic level than datasets that contain instances of simpler sounds, such as gunshots, laughter, etc. The BBC Sound Library recordings are of a high and consistent quality, and allow us to compare compare different systems in a setting where additional confounding factors are not present, as is often the case in Youtube-style, user-generated content where different recording conditions and equipment introduce channel variance.

The different categories and the number of files belonging to each of those categories are listed in Table 2.1. Besides these files, which correspond to positive instances of the various categories, the other files are considered negatives for each of the 10 categories.

The second is the TRECVID 2011 Multimedia Event Detection (MED11, henceforth) dataset [MED, 2011]. This dataset consists of a total of 15 semantically defined categories. This is currently the largest semantic audio dataset available with a total of about 10,000 files and about a 1000 hours of audio (please refer to Table 2.2 for a list of the various categories). Unlike the BBC sound effects library, the recordings have been made with a wide array of different recording devices with vastly differing background noise signals, as well as overlaid music that often has very little to do with the actual content of the file.

In terms of evaluating a novel framework on data, this pair of datasets provides an interesting contrast on 3 important factors. The MED11 dataset is certainly closer to the kind of data that one can expect to encounter on large, user-generated platforms such as Youtube, and therefore, a dataset that can be used to obtain a more realistic evaluation for the development of future technologies dealing with content analysis. However, the downside is that due to the lack of information about the recording conditions, etc, under which the MED11 dataset was created, it is hard to gain a better understanding of techniques being evaluated, since it is harder to get a clear handle on cases where the technique works well, and where it does not, as a function of the various external factors

Category	No. of positive instances	Total duration (hh:mm)
E001– Attempting a board trick	172	03:02
E002– Feeding an animal	166	14:59
E003– Landing a fish	152	12:38
E004– Wedding ceremony	163	18:43
E005– Working on a woodworking project	158	19:06
E006– Birthday party	221	15:32
E007– Changing a vehicle tire	118	11:17
E008– Flash mob gathering	191	11:24
E009– Getting a vehicle unstuck	150	14:32
E010– Grooming an animal	141	11:19
E011– Making a sandwich	183	16:22
E012– Parade	167	17:14
E013– Parkour	130	11:51
E014– Repairing an appliance	136	12:32
E015– Working on a sewing project	120	09:06

Table 2.2: The various categories in the MED11 dataset and occurrence statistics.

such as noise, channel, etc. Alternately, the BBC category allows a better evaluation of techniques in terms of understanding its strengths and weaknesses, especially in comparison to other techniques. Further, the induced structures and corresponding segments in the BBC dataset can potentially be exposed to external annotators in order to attempt to understand how well the system performs at discovering structure. On the other hand, due to legal restrictions, data from the MED11 set cannot be exposed to external annotators at all.

Both of these datasets contain some proportion of human speech, but in most of their occurrences, the speech is hard to recognize since the recordings were not made with the intent of capturing the speech. They tend to appear because a number of the events have human involvement, and speech appears either in the natural course of the event or in the background.

While the 2 datasets described above are the primary datasets used for evaluation using an audio retrieval setting, in Chapter 3, we will demonstrate effectiveness of our AUDs learning framework on a pair of different tasks using the TRECVID 2010 Multimedia Event Detection dataset (MED10, henceforth) – a smaller dataset that is actually a subset of the MED11 data. MED10 comprises 1746 total clips of training data, totaling about 56 hours in length, and the 1724 clips of test data about 59 hours long. The recordings are publicly available, user-generated multimedia content uploaded to internet hosts. Each

video is annotated with one of 4 labels – *making a cake*, *batting in run*, *assembling shelter* and *other*, identifying the kind of activity being performed in it. The class *other* appears to be a catch-all class consisting of all videos that do not belong to the first 3 classes. Participants in the NIST MED evaluation were required to retrieve recordings from the testset that included a queried activity or event, and largely focused on the video features available. The use of the audio features was usually limited to speech transcriptions [Li et al., 2010a], and detection of pre-specified sound types in the audio [Hill et al., 2010].

2.4.2 Baseline Performances on Audio Retrieval

As mentioned earlier, we evaluate our framework for content analysis of audio on a semantically defined audio retrieval task. In this section, we first describe the audio retrieval task and performance measures, and we follow with a description of how content analysis techniques are used to characterize audio files for the retrieval task.

Audio retrieval task

In the audio retrieval task, the input query to the system is a semantic event (from the list of events in Section 2.4.1), and the goal of the system is to detect all recordings corresponding to the query event in a large collection of user-generated Youtube-style recordings. Thus, for each event, we have a binary one-against-all setting. Since the events in the task are semantically defined, one might expect that such a task might be addressed by looking for semantically meaningful acoustic phenomena that correlate well with the events in the audio.

The training and test data consist of positive instances from the various classes in the dataset, and a number of files that do not belong to any of those classes. For each class i, the positive instances for that class are taken along with all the other files (which are negative instances for that class), and a detector is trained for that class. (We will describe the detector in more detail in 2.4.2.)

At test time, for each file in the test set, the detector for class i predicts whether or not the test file belongs to the class i. We evaluate performance using *Missed Detection* (MD) and *False Alarm* (FA) rates, which are defined as follows: suppose there are N_t test files, of which N_i are labeled as belonging to class i by the detector for the class. However, the test set contained C_i files belonging to class i, and D_i of these were correctly detected. Then:

$$MD = \frac{C_i - D_i}{C_i} \tag{2.4}$$

$$FA = \frac{N_i - D_i}{N_t - C_i} \tag{2.5}$$

We evaluate performance using the area under DET curves which measure MD rate against FA rate. Since both missed detections and false alarms are measures of error, the lower the Area Under the Curve (AUC), the better the performance of the system.

Baseline System Description

In this section, we will describe 2 baseline systems that are used for audio retrieval following techniques currently used for audio content analysis. The audio retrieval pipeline, however, is required to perform a number of additional tasks along with the content analysis in order to create an audio retrieval system. Figure 2.4 shows a schematic that briefly outlines the various steps involved in the full audio retrieval pipeline. We briefly discuss the various steps here, including the different paradigms used for content analysis.

Pre-processing: This stage typically involves the conversion of the raw audio to a standard representation. The Mel-Frequency Cepstral Coefficients (MFCC) are one such representation of audio data, which we use in this thesis. The Mel-Frequency Cepstrum is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a non-linear mel scale of frequency. Its coefficients are derived from a cepstral representation of the audio, and may be thought of as a nonlinear *spectrum of a spectrum*. The frequency bands used to obtain the coefficients are equally spaced on the mel scale, and this spacing is expected to approximate the human auditory system's response. Typically, speech recognition techniques use the first 13 MFCC, whereas audio processing tasks have used between 13 and 30 MFCC coefficients. For all the experiments reported in this paper, we use 13-dimensional MFCC, extracted from audio sampled at 16 KHz. All the audio in our dataset were downsampled to 16KHz, if they were originally sampled at a higher frequency.

Content Analysis Techniques: Once the audio corpus has been converted into a standard representation, we can proceed with an analysis of the audio content. In the first section of this chapter, we discussed various approaches to content analysis, but they included 2 main paradigms– supervised, audio dictionary-based approaches, and unsuper-

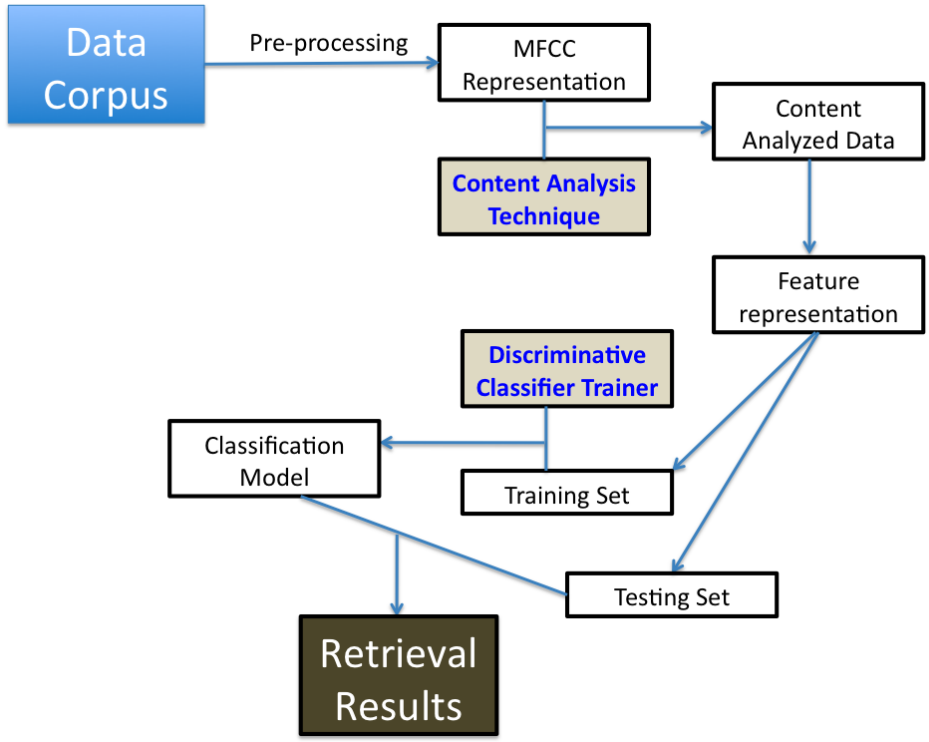


Figure 2.4: A schematic diagram of the various stages in an audio retrieval system, for a specific query event.

vised, dictionary learning-based approaches. We use an implementation of both of these as our baselines.

For the *supervised, audio dictionary-based approach*, we experimented with 2 baseline models, one using a speech corpora to train speech phonemes, and the other using a corpus of various natural sounds to train models for each of those. The speech-based dictionary was expected to account for the presence of human speech in the event recordings. The audio dictionary with examples of natural sounds was used since it seemed intuitive that using semantically meaningful units would be a reasonable way of approaching the task of detecting semantically defined audio event classes.

The speech units were trained using the HUB4 corpus [Pallett et al., 1996], consisting of business broadcast news audio, was used to train 40 phoneme models (as well as 5 fillers), using 40 filters between 50Hz to 6800Hz. Each phoneme was modeled by a 3-state Hidden Markov Models (HMMs), with the emissions being governed by mixtures of 16 gaussians. These units were then used in a manner analogous to their usage in a speech recognizer where the entire audio file was decoded as a sequence of these phonemes.

For the dictionary of natural sounds, we use the Art of Foley Sound Effects Library [Fol,

2005], which consists of 480 individual audio events. We trained Hidden Markov Models (HMMs) for each of these events with 5 states Bakis topology and emissions governed by a mixture of 8 gaussians. MFCC features, used to represent the audio, were generated with 32 filters between 100Hz and 5200Hz. As before, these models can be thought of as analogous to phonemes for speech, and we use them to decode the audio recordings as a sequence of the individual audio events.

Similarly, we set up the content analysis paradigm in the unsupervised case. Current state-of-the-art results on the MED tasks had been obtained using unsupervised learning of a lexicon using the Vector Quantization (VQ, henceforth) technique. In this technique, each frame (or a fixed window of frames) is treated as an individual instance of sound produced by one of K sources. A K-means clustering is then carried out on the corpus of frames to identify a set of K clusters and each frame is assigned to the cluster it is most likely to have been generated by. For a new recording, the set of clusters can again be used to predict (for each frame in the recording) which unit was most likely to have produced the frame. Thus, any recording can be represented by the sequence of cluster units, each corresponding to one frame in the recording.

In each of these cases (both supervised and unsupervised), therefore, for each audio recording, we obtain a discrete sequence of dictionary elements. This sequence is the output of the content analysis stage for the supervised setups.

Feature Representation: The process of designing a feature set is often critical to the task in question. Different feature sets using the same learning framework often yield different results because one set of features may be more suited to the task, or might capture more discriminative information than another. Temporal behavior of features has been shown to be important for audio and music classification, and the use of models of auditory perception in feature sets have proven to be better than MFCC based feature sets for the same tasks [McKinney and Breebaart, 2003]. Different feature sets have been shown to affect performance in other applications as well, such as web document classification [Qi and Davison, 2009].

Our focus in this thesis is on the content analysis techniques, hence we only perform a limited amount of experiments on the feature set design to choose a standard feature set representation that will then be consistently used to compare the various content analysis techniques. In the audio retrieval domain, the most commonly used technique is the *bag-of-words* representation where given a discrete sequence of symbols for a recording (in this case, the output of the content analysis stage) with K unique symbols, a K-dimensional feature vector is created where there is one feature for each symbol, and the feature value

is the relative frequency of the that symbol in the recording.

As an example, if the discrete representation from the content analysis stage of a recording is: a_1 a_2 a_1 a_1 a_2 a_3 a_5, where the vocabulary has K=5 (with a_1 a_2 a_3 a_4 a_5), then a feature representation for that recording would be: $< 3/7, 2/7, 1/7, 0, 1/7 >$

For the supervised, dictionary-based techniques, $K = 45$ for the speech phoneme dictionary, and $K = 480$ for the Foley sound effects library. For the unsupervised, VQ-based techniques, the size of the dictionary K is a hyperparameter that needs to be optimized.

We will consistently use this feature representation for comparison to the baseline with the results of the novel, hierarchical content analysis algorithms proposed, as well. Note that in Chapter 3, we will show results of some experiments with alternate feature representation that show the bag-of-words approach as superior, as well.

All the various stages in the pipeline thus far– pre-processing, content analysis and conversion to feature vector for each recording– could be done without considering the distinctions between train and test data. For the final stage of the pipeline, however, we need to separate the train and test sets, since the classifier for each query event must be built on the training set alone.

The Classification Framework: As described earlier, the audio retrieval task has been cast as a discriminative one-against-all classification task, for each query event type. The *bag of words* representation serves as the feature representation for each audio file. Given this training data with event labels for the audio files in the training corpus, we use a discriminative classifier that can decide whether a test file belongs to a specific event type or not.

Again, various discriminative classifiers have been developed in the literature, and could be used for this task. As mentioned earlier, this thesis focuses on techniques for analyzing the audio content, and different techniques need to be compared with the rest of the pipeline held fixed. For the experiments reported in this thesis, we use the Random Forest classifier [Breiman, 2001].

Random forests are an extension of decision tree classification techniques, where the training process grows many trees instead of a single one. Given a new test file, each of the trees in the forest returns a class label, which is used in a weighted vote to determine the final predicted label.

We chose random forest classifiers as they are resistant to overfitting. Training random forests is done by sampling with replacement from the training data, with about one-third of the training data typically held out. This held out data is used to get an estimate of the error as trees are added to the forest. The trees in the forest are grown as far as

System	Average AUC in BBC	Average AUC in MED11
PHONE	0.3011	0.2614
FOLEY	0.2872	0.2921
VQ	0.2143	0.2339

Table 2.3: Performances of the various baseline systems for audio retrieval

possible, and pruning is not used. For details of the training process, the reader is referred to Breiman [2001].

Once the classifier has been trained, it can then be applied to each file in the test corpus to decide whether or not that file corresponds to a positive instance of the query class. In our experiments, since we plot an area under the curve, the classifier returns a probability of its estimate that the test file belongs to the class. We then use a threshold to decide whether to label the file as a positive. While this threshold can be tuned to find an optimal operating point depending on the goals of a user of the system, we iteratively modify the threshold, thus trading off missed detections and false alarms, to obtain a curve, which we use to compare the various systems.

Experimental Results

In this section, we will establish baseline performances for the various standard techniques used in the literature for content analysis of audio on the datasets of interest. Recall that the metric used for evaluation of performance will be the Area Under the Curve (AUC) of Missed Detection rate (MD) v/s False Alarm rate, which is a measure of error. Thus, a lower AUC value signifies better performance.

We will discuss the results on the various classes in the datasets in greater detail in subsequent chapters. Here, we simply present a table with results on each dataset macro-averaged over the classes in the dataset. This table will be augmented in each of the next 3 chapters with results of using the proposed content analysis techniques on the data. Table 2.3 shows the average performance on the 2 datasets for the 3 baseline techniques described earlier.

We refer to the system using the speech data for phoneme modeling as PHONE and the one using the Foley Sound Library as FOLEY, henceforth. The system using the Vector Quantization technique is referred to as VQ.

We note that the unsupervised VQ technique significantly outperforms the supervised library-based techniques in both cases. This is not surprising and corroborates the notion that supervised, library-based techniques are unlikely to perform well on semantic tasks,

unless the set of sounds in the library were carefully chosen to be helpful for the specific task, as well as the audio instances in the library suitable in their characteristics to the type of audio likely to be found in the corpus of interest.

We note that, of the supervised systems, the PHONE system performs better on the MED11 dataset whereas the FOLEY system performs better on the BBC dataset. One possible reason for this may be that the MED11 categories contain significantly more human involvement in the data, since these are user-generated content, and therefore, the proportion of human speech, or speech-like sounds are higher in MED11. As a result, the PHONE models are able to do a better job of capturing the acoustic phenomena underlying in that dataset than in the BBC dataset where the proportion of human presence is significantly lower.

Another reason might be that both the FOLEY data and the BBC data are recorded in studio conditions with an emphasis on non-speech sounds that enable them to be a better match in terms of acoustic characteristics as well as coverage of concepts. Since the MED11 data contains a larger representation of the potentially infinite sound concepts owing to its user-generated characteristics, the coverage of these concepts provided by the FOLEY library may be far more limited.

In the next few chapters, we will compare the performance of these methods with the methods proposed in this thesis, as well as attempt to understand their relative strengths and weaknesses.

Chapter 3

Acoustic Unit Descriptors

In this chapter, we will discuss our approach to learning the low-level acoustic units of the hierarchical framework presented in Figure 1.1 in Chapter 1. These low-level acoustic units (indexed by a in the figure) are the hidden layer in the hierarchical structure that is the closest to the audio, and higher layers are built on top of this layer. When looking at the lowest layer only, we see that it consists of a sequence of atomic sound units, which we call *acoustic unit descriptors* (AUDs), which in turn, produce the observed audio. As discussed earlier, we expect that the layers of the hierarchy allow us to make increasingly higher level semantic inferences about the audio, as we go higher in it. Recall our example of a movie content analysis task, where in a particular scene, one might hear a loud *crack* followed by vocalized, high pitched sounds, some human speech, a repetitive electronic wailing sound, interspersed with more rapid *crack* sounds, and loud, mechanical humming.

The *crack* sound could be produced by different sources (*e.g.* a gunshot, or a car backfiring), but given the context of siren-like sounds and people screaming, one is more likely to infer that the sound was a gunshot. Similarly, the sound of a siren alone does not indicate to a listener that the scene was an action scene. Sirens may be heard in a regular scene shot in traffic, or even from ambulances responding to non-action emergencies (e.g. medical emergencies). However, in conjunction with the gunfire around it, it seems likely that the scene was an action scene.

In our framework, the lowest-level acoustic units are expected to capture more of the acoustic characteristics than the semantic characteristics, which are captured in the higher layers. Thus, in an ideal scenario, we would like to learn low-level sound units that can capture the *crack* sound, a unit that when repeated captures screaming, or the repetitions of the siren, etc. As illustrated by the example, while the semantic import of the individual units may not be clear, the semantics are expected to be found in the distribution or

sequences of these units, which are captured by the higher layers.

We note, however, that while clearly interpretable semantics are not present at this level, they do carry some degree of semantic information. For instance, even though the semantic space from which sounds may be produced is enormous, hearing a particular sound (even in the absence of clear associated semantics) constrains use to a subspace of the full semantic space. Thus, hearing a sharp *crack* may not tell us the exact semantic source, but we know that some sources (*e.g.* musical performances, birthday parties) are less likely to produce such a sound, while others (*e.g.* gunfire scene, baseball game) are more likely, and in this manner, the low-level units are not entirely devoid of semantics.

The primary hypothesis behind this dissertation is that sound has its own *language* and *structure*. In this *language* for sounds, the AUDs can be considered analogous to the alphabet. The middle layer in the hierarchical structure is referred to as the *event* layer. Here, the term *event* refers to semantic sub-file level events (distinct from the event categories in the various datasets) that are expected to span segments of the audio, and therefore, sequences of AUDs. We note that such a definition may be specific to certain applications, and that we may require an additional *sub-event* layer, where sub-events span AUDs, and events span sub-events, or perhaps one where events can recursively span other events. The root or highest layer in this hierarchical structure contains information about the semantic class or topic that generated the sequence of events.

Such a hierarchical structure may have layers whose analysis is treated individually when inferring the structure, and we will present an approach for the same in Chapters 4. Alternately, the full set of higher layers may be learnt jointly as well, and we will present a framework for doing so in Chapter 5. Each of those analysis techniques assumes that the raw, observed audio has been first analyzed to obtain the AUDs layer, resulting in a sequence of AUDs, and builds on top of the AUDs layer. In this chapter, we will present our approach to modeling the AUDs layer, show how any new audio may be analyzed after models for the AUDs have been learnt, and validate our hypothesis that AUDs capture underlying semantic information by using AUDs for various semantic audio processing tasks.

This chapter is organized as follows. In Section 3.1, we discuss AUDs in greater detail, including the intuition behind the formulation and an unsupervised learning paradigm. In Section 3.2, we present approaches to using these AUDs (that were learnt in a task-agnostic manner) for various tasks, and our experimental results demonstrate the efficacy of this approach on real-world data. We conclude the chapter with a discussion in Section 3.3.

3.1 Acoustic Unit Descriptors

In the hierarchical structure introduced in Figure 1.1, the latent layer containing the acoustic unit descriptors is closest to the observed sounds. Our hypothesis in attempting to formulate a language for sound is that sound can be modeled with a small number of lower level units (AUDs, in our case). Individual AUDs do not necessarily capture semantic information, but sequences or patterns of AUDs should capture semantic information. Every instant of an audio file is part of one such unit, and the entire audio stream can be *transcribed* in terms of these units. If every AUD were to have distinct semantic identity, the number of AUDs required to represent all audio would be very large. Instead, we hypothesize that if we used just a small number of AUDs, the patterns in the transcriptions of audio recordings in terms of these AUDs will still be characteristic of the larger events in the audio.

Our assumption of a finite set of low-level sound units is motivated by the benefit of computational tractability as well as 2 other observations. First, even though the full set of sounds might be infinite, only a limited (although, possibly large) number will occur in any given dataset. Second, the assumption of a smaller than true number of units will force multiple different concepts to cluster with the most similar acoustic concepts. As a result, we expect multiple semantically different but acoustically similar concepts to map to individual units. Thus, these unit clusters will not be true, diverse atomic units, instead modeling acoustic saliency shared by multiple concepts.

The transcription of audio in terms of AUDs also results in a mapping from the acoustic (or acoustically derived) feature space to a discrete symbol space. This has the auxiliary benefit of allowing us to use the knowledge gleaned from a vast amount of research in dealing with discrete symbol sequences in text processing and information retrieval, in terms of developing techniques to use such symbol sequences for modeling higher level units, using these to characterize files for retrieval, etc.

We can think of AUDs as as an analog in the general sound domain to phones for speech. However, because audio (in general) is so diverse and variable, we cannot expect to be able to interpret individual AUDs. Further, due to the assumption of finiteness of the AUDs vocabulary, the individual units are not necessarily clean representations of individual audio concepts. In a sense, therefore, the AUDs are a synthetic concept, and hence, it is not possible to have supervised transcripts of recordings in terms of AUDs. As a result, the training process for AUDs is fundamentally unsupervised.

Let us illustrate the intuition behind our formulation of the AUDs learning with an example– consider sounds from a baseball recording. Fig. 3.1 shows three video frames from

such a recording. The bat makes contact with the ball, producing a sharp metallic sound. This is followed by footsteps running, and finally, cheering as teammates congratulate the player. A listener familiar with baseball would be able to infer from the sequence of sounds that they may be from a baseball game, and that a hit or a run may have occurred. Although the precise sounds produced and their sequences may vary in nature, the overall pattern of sounds is still characteristic of the event.

Figure 3.1: Example of a sequence from a baseball video.

The sound of the ball being hit, footsteps, cheering, etc. are all *atomic* sound events that characterize the larger event of the run being batted in. Moreover, besides these key events, there are other individual nondescript atomic events such as rustling, silence, etc. which occur in the recording. In fact, every instant of the audio may be considered to be a part of one such atomic event. In our process of training the AUDs layer from a corpus of such recordings, we would like the learnt AUDs to consistently capture these atomic sound concepts from the audio.

The overall pattern of occurrence of these atomic events characterizes the larger event, and our work tries to mine AUD sequences to be able to both identify the larger event as well as attempt to find structured sub-events that lead to the conclusion, if any, as shown in Fig 1.1.

[We briefly digress here to point out an instance where one layer for events (instead of an event and a sub-event layer) could be expected to be enough. In audio of a baseball game, the sub-events that constitute events (*e.g.* plays) are short-duration events– bat hitting a ball, throwing or catching a ball– and most of these could be expected to be captured within individual AUDs. In case of events that are more complex and of significantly longer

duration, such as a car chase in a movie, it may be necessary to have a sub-event layer to model the events better.]

The process of learning of the parameters for the AUDs is described in Section 3.1.1. We propose a maximum-likelihood solution that jointly estimates the parameters of the HMMs for the *AUD*s and the (potentially, category-specific) language models of audio from a training corpus.The solution is analogous to various approaches in the literature for automatic learning of sub-word units in speech [Bacchiani, 1999, Singh et al., 2002].

3.1.1 Learning AUD Parameters

Our task in modeling the AUDs layer is to represent the data with a sequence of AUDs that are likely to have generated the data. The concept of AUDs is similar in principle to that of acoustic segment models (ASM, henceforth) [Lee et al., 1988] and Self-Organizing Units [Siu et al., 2011]. While these were derived for speech, AUDs make no assumptions about the nature of sound in general. The ASM also used an acoustically derived lexicon for word models based on subword segment models, while there is no equivalent available for AUDs which are learnt completely unsupervised. Further, *AUD*s do not spot specific events in the audio stream – the entire audio stream can be *transcribed* in terms of these units, *i.e.* every segment of audio is part of some *AUD*.

The problems we must address now at training time are twofold: A) **Learning of AUDs:** Given a set of training audio recordings, we must learn the set of *AUD*s, B) **Learning of AUD distributions:** We need to learn statistical characterizations of the patterns of *AUD* sequences for audio from different categories. The learning process is inherently unsupervised, since the AUDs are a synthetic concept, and one cannot obtain ground truth transcriptions of audio in terms of AUDs.

Each individual acoustic unit can be modeled using any structured model– one instance would be using a Hidden Markov Model (HMM) formalism, as described in Chaudhuri et al. [2011]. We represent the audio signal as a sequence of mel-frequency cepstral vectors, as is the practice in speech and audio processing and model each AUD with a left-to-right Bakis-topology HMM with Gaussian-mixture state-output densities. Since we are primarily interested in characterizing the AUDs, rather than interpreting their semantics, learning the AUDs is equivalent to learning the parameters of the HMMs for the AUDs. We model the distribution of the AUD sequences as language models over the vocabulary of AUDs. In our initial treatment of the learning process, in order to avoid loss of generality, the formulation we present can use any N-gram language model. Further, one can choose to adopt class-specific language models for certain tasks, where class labels for the recordings

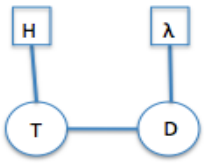


Figure 3.2: Graphical model for the generation of a recording data point D. Circles represent random variables and rectangles represent parameters. Note that D is observed, while the transcript T is not.

in the training data are provided as supervision. As we shall see later, using only one class-independent unigram language model works as well as class-specific, higher order language models. Besides, for large datasets, it provides significant improvement in the time required for decoding, due to avoiding having to decode with each of the class-specific language models. Nonetheless, we will first discuss the case where class-specific language models are trained, since the case where there is only one class-independent model is a special case of this. Since the AUDs have been conceptualized as a task-agnostic vocabulary of audio concepts, the set of AUDs is shared by all classes in the dataset.

We cast the learning problem as one of *maximum likelihood* estimation. We are given a collection of audio recordings $\mathcal{D}$. Assigned to each recording D_i in $\mathcal{D}$ is a *class label* $C_i \in \mathcal{C}$, where $\mathcal{C}$ represents the set of all classes. Although not necessary, we will assume that each recording is entirely assigned to only one class. Each audio recording D_i has an unknown *transcription* T_i as a sequence of AUDs. The AUDs are modelled by HMMs, whose parameters we collectively represent as Λ. The transcriptions of all recordings belonging to a class C are assumed drawn from an N-gram language model $H(C)$. The HMM parameters Λ and the set of language models for all classes $\mathcal{H} = \{H(C)\ \forall C\}$ are unknown and must be estimated from the data. We assume that the total number of AUDs K is known. In reality, K is a hyperparameter that may be optimized.

We assume the dependencies shown by the graphical model in Fig 3.2: the acoustic realization of any recording depends on its transcription and not directly on the language model for the class. So also, the transcriptions only govern the acoustic realization and do not directly relate to HMM parameters.

The maximum likelihood estimate for Λ and $\mathcal{H}$ is given by:

$$\Lambda^*, \mathcal{H}^* = \text{argmax}_{\Lambda,\mathcal{H}} P(\mathcal{D}|C(\mathcal{D}); \Lambda, \mathcal{H}) \tag{3.1}$$

Here $C(\mathcal{D})$ represents the classes assigned to each D in $\mathcal{D}$. In the notation above terms to

Algorithm 1 Iterative algorithm for learning AUDs and LMs

$$T_i^{r+1} = \text{argmax}_T P(T|D_i; H(C_i)^r; \Lambda^r) \quad (3.3)$$

$$\lambda^{r+1} = \text{argmax}_\lambda \prod_{D_i} P(D_i|T_i^{r+1}; \Lambda) \quad (3.4)$$

$$H(C)^{r+1} = \text{argmax}_H \prod_{D_i:C_i=C} P(T_i^{r+1}; H) \quad (3.5)$$

the right of the semicolon are parameters, while remaining terms are random variables.

In principle, the above estimator must consider all possible transcriptions for any D. Instead we will approximate it by only considering the most likely transcription for any D. Also, assuming that individual recordings D are independent, and that the class is represented primarily through the language model for the class, the estimator changes to:

$$\text{argmax}_{\Lambda,\mathcal{H}} \prod_C \prod_{D_i:C_i=C} \max_T P(D_i, T; \Lambda, H(C)) \quad (3.2)$$

We obtain the above estimate using the iterative algorithm in Algorithm 1. In the algorithm, superscripts appearing against the parameters indicate the iteration in which the estimate for the parameter was obtained.

It is simple to show that Algorithm 1 is a hill-climbing procedure that results in ever increasing likelihood for the data: Equation 3.3 ensures that

$$P(D_i, T^{r+1}; \Lambda^r, H(C)^r) \geq P(D_i, T^r; \Lambda^r, H(C)^r) \quad (3.6)$$

and Equations 3.4 and 3.5 ensure that

$$P(D_i, T^{r+1}; \Lambda^{r+1}, H(C)^{r+1}) \geq P(D_i, T^{r+1}; \Lambda^r, H(C)^r) \quad (3.7)$$

Equation 3.3 simply represents the automatic recognition of D_i using HMMs with parameters Λ^r and can be performed with the Viterbi decoder of a speech recognizer. Equation 3.4 is the learning procedure for HMM parameters Λ, given the recordings D_i and their transcriptions T_i^{r+1}, and can be performed using the Baum-Welch training module of any recognizer. Equation 3.5 represents the procedure for learning an N-gram language model $H(C)$ from the set of all transcriptions T_i^{r+1} of all recordings belonging to class C.

Algorithm 1, however, requires an initial transcription for all recordings. We obtain this by segmenting all recordings by merging adjacent analysis frames, and finally clustering the

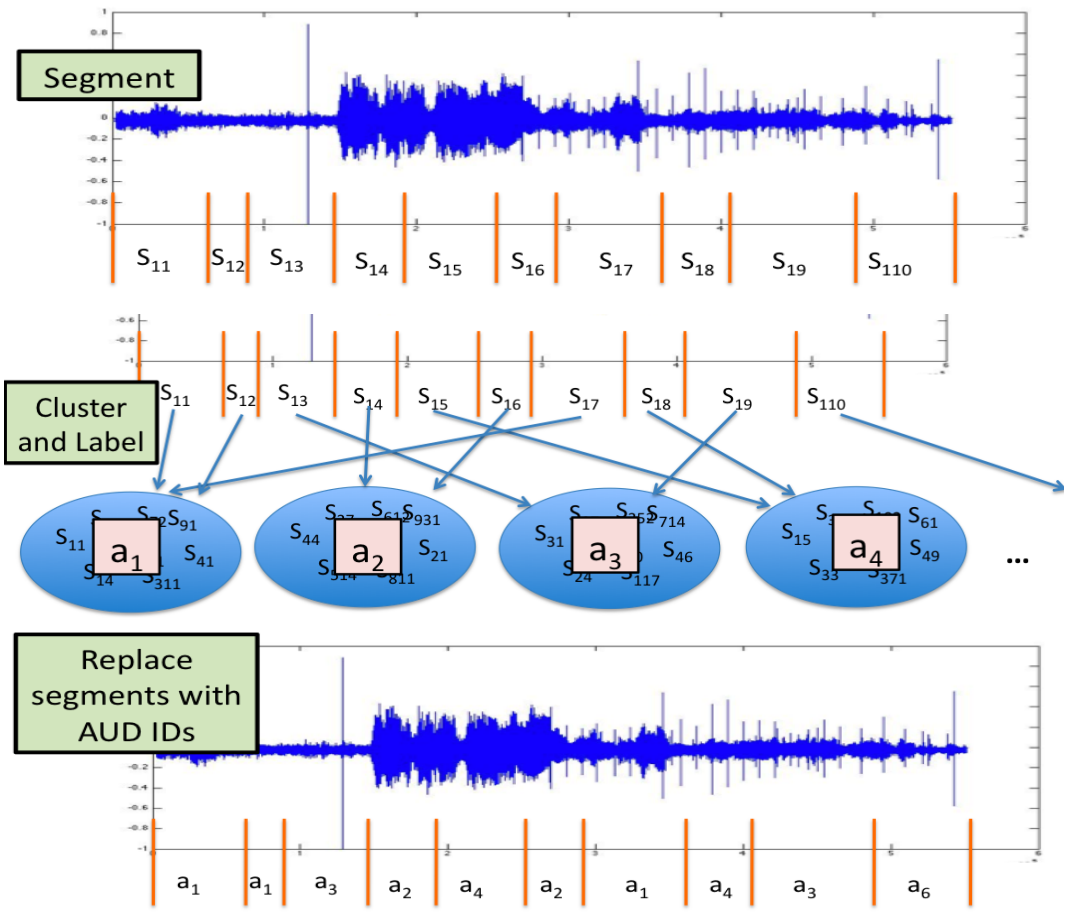


Figure 3.3: The process of initialization of the AUDs setup for training displayed in 3 steps. (Top) Segmentation of the audio based on agglomerative clustering of consecutive frames. (Middle) Clustering of the segments. (Bottom) Setting AUD ids to the segments to create AUD transcript

obtained segments into K clusters. The sequence of cluster identities corresponding to the segments composing any recording form the initial transcription for that recording. The steps used to obtain this initial segmentation in terms of AUDs beginning with features from the raw audio are shown in Figure 3.3.

In the first step, when the raw audio is segmented, this is done by converting the raw audio to MFCC frames, and the comparing the distance between consecutive frames using Euclidean distance. If the distance is above a threshold, then the two frames are merged to belong to the same segment, else a new segment is started at that boundary. Segments are constrained to be no fewer than 3 frames long at the time of initialization.

The mapping between the AUDs and the observed acoustic data is stochastic. Given models for these AUDs, we can decode new audio in terms of the AUDs in a manner similar

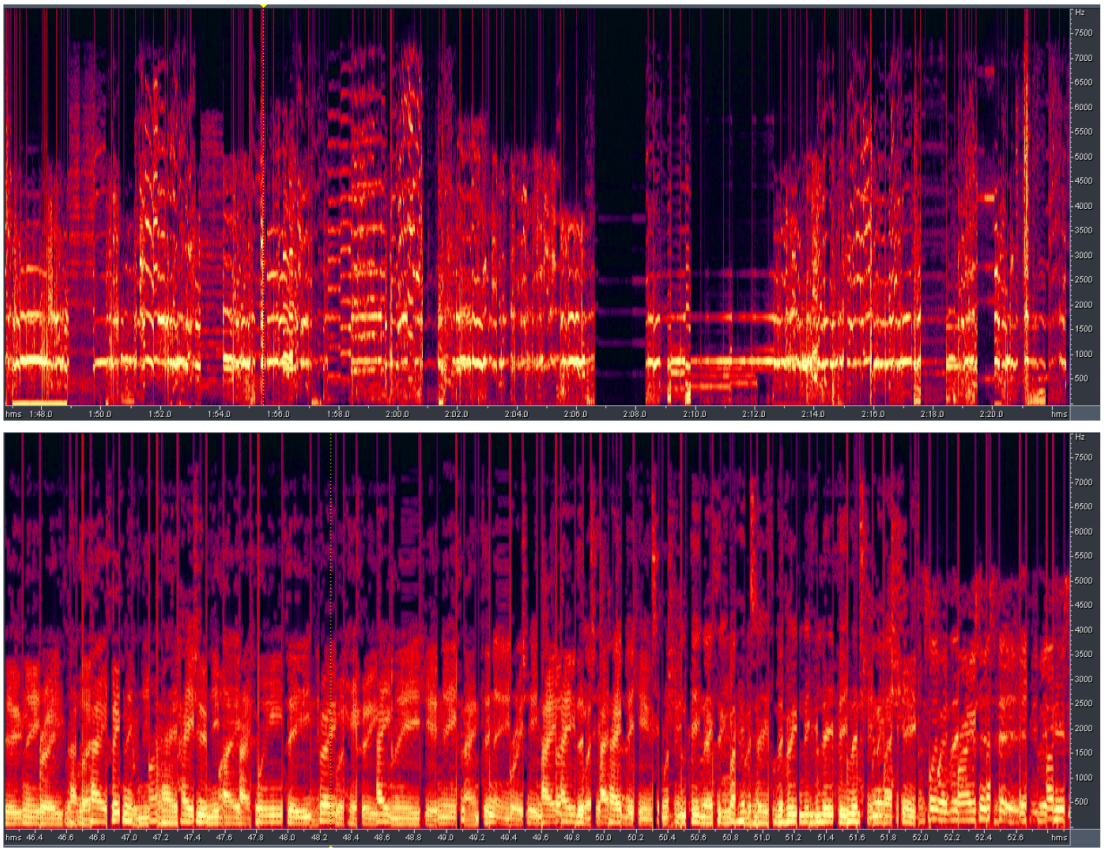


Figure 3.4: Instances of log-spectra for 2 AUDs, with all occurrences across files concatenated. (Top) Predominantly music; (Bottom) Predominantly speech. The y-axis correspond to frequency bins

to the one used in a speech recognition system to decode continuous speech in terms of phonemes, followed by selecting the highest scoring AUDs path through the lattice of AUDs. We recognize that such decodes are likely to be noisy, since the presence of channel artifacts and background noise can affect the local estimation resulting in a different concept taking precedence over the concept that should have been estimated by a perfect system. As we shall see in Chapter 4, our modeling of higher layers attempts to recover from some of the errors introduced at this stage.

The iterations of Algorithm 1 lead to progressively improved joint AUD and language models, as shown by the increasing likelihood of the data through the iterations. We present two examples of log-spectra of all frames spanned by an AUD in the dataset extracted and concatenated together in Figure 3.4. In both cases, one can see structural consistency, showing that the AUDs find acoustically similar segments as desired. While the AUDs are not required to have clear semantic interpretations, listening to the concatenated instances

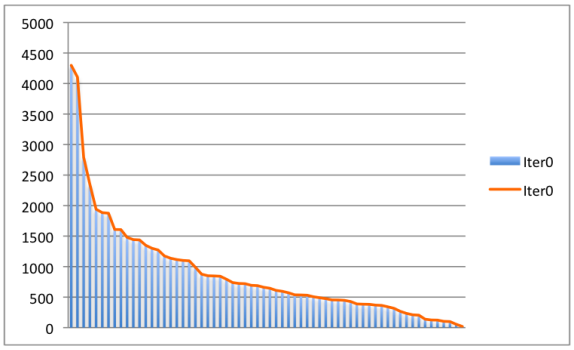


Figure 3.5: A plot of occurrence counts sorted AUDs from our initialization, which appears to follow a power law. (No. of occurrences of AUDs v/s AUD-id)

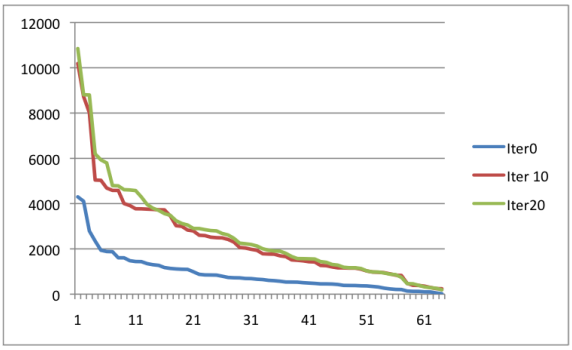


Figure 3.6: A plot of the occurrence count sorted AUDs at different points in the learning process for AUDs. (No. of occurrences of AUDs v/s AUD-id)

(converted to audio from the log-spectra) shows that the AUD on the left primarily spans music segments while the right consists primarily of speech– speech formant structures are visible in the image.

In addition to this, however, following the training process of the AUDs on the MED11 data resulted in some interesting observations.

First, we look at the distribution of AUD counts as we proceed with the learning. Our initialization of the AUDs used similarity between consecutive frames in the audio to decide whether or not they belonged to the same segment, and then clustered these segments together. Each of these clusters is a distinct AUD, and all segments belonging to a cluster were transcribed with this cluster identity. We first investigated the distribution of these segments and how they change with learning iterations.

Figure 3.5 shows the histogram where the Y-axis is the occurrence counts of the AUDs. The AUDs are sorted using occurrence counts along the X-axis. As the plot shows, the occurrence counts of the AUDs appear to follow a power law [Zipf, 1932]. Various naturally occurring phenomenon have been shown to follow such power laws [Gabaix, 1999, Li, 1992], but it does indicate that our initialization does produce something that should be expected.

Figure 3.6 shows the sorted occurrence counts of the AUDs at intervals in the learning process, where the blue line is the same as the one in Figure 3.5, immediately after initialization. Notice that the distribution of AUDs continues to follow a power law throughout, even though no explicit measures were taken to ensure this.

We also noticed that besides the fact that the sorted AUD counts appears to follow a power law at various stages of training, we find that the counts of the AUDs have approximately tripled at the end of the training process. We plot the counts after 10 and

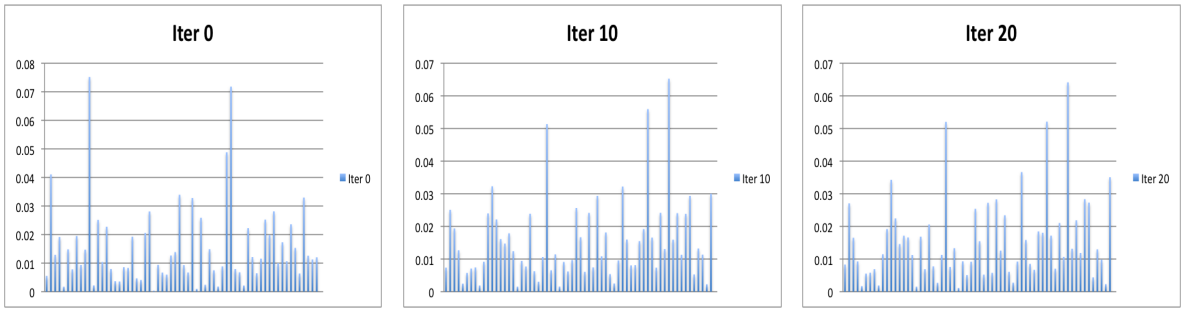


Figure 3.7: Changes in the unigram distribution of AUDs over time

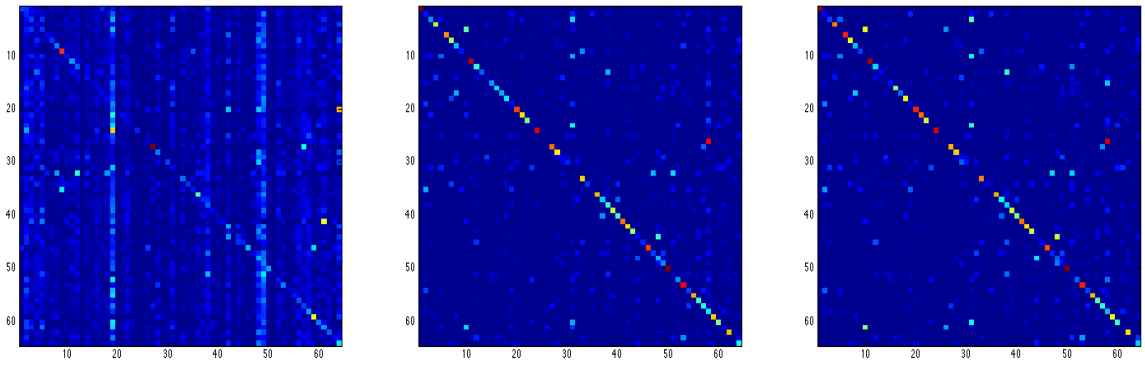


Figure 3.8: Changes in the bigram distribution of AUDs

20 iterations in the figure. We note that this change is linked to the durations spanned by the AUDs. When they were initialized on the MED11 dataset, each AUD on average captured 0.84 seconds of audio. At the end of the training process, the average length of audio spanned by the AUDs was 0.29 seconds.

Anecdotally, we note two other points about the distribution of the AUDs in the learning process, although their implications are not clear to us at this point. First, the unigram distribution of the AUDs changes significantly over time, as shown in Figure 3.7. Second, the bigram distribution of the AUDs shown in Figure 3.8 also changes significantly over time, although most of the probability mass appears to concentrate along the main diagonal, which implies that states are most likely to transition to themselves. The transition matrix also appears to get more sparse over time, implying that AUDs are most likely to occur in a small number of contexts. This should imply that bigram or higher order n-gram characterizations should provide stronger information than unigram about the semantic content. However, as we shall see from experiments in subsequent sections, this does not appear to be the case.

In Section 3.2, we will describe our approach to applying the AUDs layer to various tasks. The experimental results we describe also show that our models of the AUDs do, in fact, capture some underlying semantics and can be utilized in semantic audio tasks, by themselves to produce state-of-the-art results even in the absence of the hierarchical structure. Recall, as discussed earlier, that current techniques in content analysis research do not use hierarchical analysis techniques using the kind of deep analysis that we proposed, and thus the AUDs layer is the one that is most directly comparable to currently used techniques, in terms of depth of analysis.

Further during the course of our experiments, we used both a class-specific language model setting as well as a class-independent language model setting, as well as different degree language models, and we shall discuss insights and lessons learned from those, as well.

3.2 Applications of AUDs to Audio Processing Tasks

As noted in earlier sections, the ideal evaluation of the hierarchy, or even any single layer learnt, would be by comparing it to a ground truth baseline, such that the process of discovery of the hierarchy or the units themselves could be appropriately evaluated. Unfortunately, since audio datasets do not have nearly the kinds of annotation that would be required to evaluate the units of individual layers, we are compelled to evaluate the degree of semantic information captured by the units and layers on semantic audio tasks.

In this section, we describe our experiments with using the vocabulary of AUDs to perform 3 audio processing tasks. We describe our experiments and results on the task of audio retrieval in Section 3.2.1, on the task of audio classification in Section 3.2.2, and in an event detection-like setting in Section 3.2.3.

3.2.1 Audio Retrieval

We described the audio retrieval task and the datasets that we report results on in detail in Chapter 2, Sections 2.4.1 and 2.4.2, respectively. To briefly summarize the task, we attempt to detect all recordings of a specific semantic event type in a large collection of user-generated Youtube-style recordings– thus, for each event type, we have a binary one-against-all setting. We use 2 datasets for this task– the 15 category Multimedia Event Detection task, 2011 (MED11) dataset and the 10 category BBC Sound Effects Library (BBC).

Since the events in the task are semantically defined, one might expect that such a task might be addressed by looking for semantically meaningful acoustic phenomena that correlate well with the events in the audio. While we take a different approach, we compare performance with semantically motivated baseline techniques that were described in Chapter 2 in Section 2.4.2 and their results were summarized in 2.4.2.

We attempt to perform retrieval using the AUDs which have no clear semantic interpretation. The process of using AUD sequences to describe the audio is done according to Equation 3.3, given the models for the AUDs and their distributions. In order to compare the AUDs against the baseline systems, we use the pipeline described in 2.4.2 where the content analysis module is replaced by the AUDs framework, and it outputs a sequence of AUDs for each recording in the corpus. Similar to the approach mentioned there, we use a unigram *bag of words* to characterize the recordings in terms of AUDs. Later, in this section, we will show results using bag of words characterization but with bigrams and trigrams. We note here that the unigram characterization typically performs at least as well as the higher orders, if not better, while providing benefits in terms of model size and avoiding issues due to dataset size.

Given training data with event labels for audio files, we use a discriminative classifier that uses features extracted over the AUD transcriptions for the audio files to decide whether a test file belongs to a specific event type or not. As discussed in 2.4.2, the classifier we use is a random forest classifier. Our experiments show that the AUDs perform very well at the task of detecting the events. In fact, retrieval based on AUDs is superior to that obtained using semantically meaningful units– PHONE and FOLEY. In this section, we first describe the AUD-based feature sets we use, in addition to the bag of words. We then describe the classifier used for this task, and finally, we discuss our experimental results.

Features for Audio Retrieval

For this retrieval task, we experiment with various feature sets derived from AUD sequences in the data. AUDs can be used to transcribe audio as a sequence of discrete symbols. We expect that different classes of audio data will contain different distributions of the various acoustic phenomena, resulting in different distributions of AUDs in their recordings. We extract the frequency of occurrence of each AUD in the transcription for an audio file. The {AUD, AUDfrequency} pairs thus obtained are used to construct a feature vector for each data point.

In order to analyze the AUD-based features, we also define two other feature classes– a binary feature vector that indicates occurrence of each AUD based on the transcription,

System	Average AUC in BBC	Average AUC in MED11
PHONE	0.3011	0.2614
FOLEY	0.2872	0.2921
VQ	0.2143	0.2339
AUDs–binary	0.2065	0.2590
AUDs–count	**0.1744**	0.2273
AUDs–framecount	0.1806	**0.2189**

Table 3.1: Performances of the AUDs compared to the baseline systems for audio retrieval

and a feature vector where the feature value for an AUD is the total number of frames spanned by all instances of the AUD in the transcript for the audio.

We model each AUD modeled as a 5-state HMM, with a mixture of 8 gaussians used to control the emissions and a unigram model over the AUDs to control inter-AUD transitions.

Experimental Results

The task we use to evaluate performance is one of detecting all recordings of a specific event type from the MED11 dataset. The training and test data consist of positive instances from the set of 15 classes, and a number of files that do not belong to any of those classes. For each class i, where $i \in 1, 2..., 15$, the positive instances for that class are taken along with all the other files (which are negative instances for that class), and a detector is trained for that class. The evaluation of retrieval performance is done using the Area Under the Missed Detection vs False Alarm Rate curves, as explained in Section 2.4.2. Note that since this is an error curve, lower is better.

We described the various feature sets earlier– binary indicators for AUD presence (AUDs_binary, henceforth), AUD count vectors (AUDs_count, henceforth), AUD frame vector counts (AUDs_framecount, henceforth), phoneme count vectors (Phone, henceforth), audio library sound-type count vectors (Foley, henceforth) and Vector Quantization (VQ, henceforth).

Table 3.1 updates the table with baseline performances averaged across all categories of the 2 datasets, with the performance of the various AUD feature sets.

[Note that the number of AUDs used for the results reported in this table are 64, since we wanted to use the same setting for both datasets in this table at this time, to avoid confusion. In subsequent chapters, when the table is augmented with results from higher levels of the hierarchy, we will replace the AUDs–count result on MED11 with the best result obtained with each dataset, with 64 AUDs for BBC, and 1024 AUDs for MED11.]

While the average numbers hide some of the details of performance in the specific cat-

egories, a closer look at the category-wise numbers shows that the AUDs–count setting performs better across the board when compared to any of the 3 baselines or the AUDs–binary setting. An evaluation on MED11 shows that the biggest difference in performances of the AUDs compared to the strongest baseline VQ comes in the *feeding an animal* category. Both classifiers perform worse on this class than their average, however. The least improvement in performance, on the other hand comes from the *woodworking project* class where both systems perform above their average, with the rest being in between and *fishing* being the class where the AUDs–count achieves median improvement over the VQ baseline. The respective categories in the BBC dataset were *warfare* (least improvement), *exterior atmospheres* (most improvement) and *animals* (median improvement).

From a general comparison of category-wise results for the various AUD settings, we observe the following:

1. The AUDs–binary setting performs reasonably well on this task, and is comparable overall to the Foley, which was trained using a library. Let us discuss the intuition behind the binary setting here. The question being asked by this setting is as follows– if one were to assume that the AUDs learnt the kind of information they were expected to learn, what information does their mere presence convey? Consider an example where a *crack* sound, which is hypothetically confusable between a gunshot sound and a hammer sound only. In spite of not knowing which specific source produce it, with the knowledge that humans have, they can look at the list of categories, and make a good estimate of which categories might contain sounds from either of those sources. The more likely ones include *warfare*, *woodworking project*, *repairing an appliance*, while less likely categories include *fishing*, *making a sandwich*, *animals*, etc. Similarly, applause units are likely to occur in *audiences*, *bike trick*, etc and not in *working on a sewing project*. Thus, the mere presence of certain sounds can help us narrow the space of possible candidates, thereby providing significant information that can be used in making a decision on whether a recording belongs to a given event type. The good performance of the binary setting in comparison with the baselines shows that they might be capturing the kind of information we expect.
2. Thus, if the presence information alone gives us insight into the category, then knowledge of how often the various sounds occur should give us even more information. For instance, a file in which automobile sounds occur often is more likely to belong to the category *transport* than one where it occurs once, which may have been due to the random presence of a car in the background even though it was not the subject of the recording. This intuition appears correct, since the best performance overall is

obtained in the AUDs–count setting significantly improving over the AUDs–binary setting, as well as both Phone and Foley-based classifiers across all categories. The improvement in performance in the AUDs–count setting over the baselines shows strong statistical significance ($p < 0.01$).

3. We note that the AUDs–framecount setting performs very similarly to the AUDs–count setting, not just as an overall average, but also in the individual categories. It outperforms the AUDs–count in the MED11 dataset, but the difference is not statistically significant (p-value of 0.13). Fig 3.9 shows the performance in terms of Area Under Curve (AUC) when using the different feature sets for the first 5 classes of data in MED11 (note that lower AUC is better), and we can see that the 2 settings are almost the same. The AUDs–framecount setting counts the number of frames in a file that each AUD spans. Intuitively, then, the information contained in it is more useful if the duration of time spanned by each AUD in a recording is meaningfully important. This does not, however, appear to be the case, which tells us that the information contained by the number of frames spanned by the AUDs is no more than the information contained by its frequency of occurrence. This might be construed as an implication that the AUDs are successful at isolating atomic units. Consider the case where there is extended applause– the entire segment could be marked as the corresponding unit, or individual *clap* or *cheer* elements could be identified as an AUD. In the first case, we would expect the frequency counts to perform worse than the framecount, since it wouldn't be able to distinguish between a short, randomly introduced applause sound from a longer applause segment that was part of the topic in focus, such as applause from an audience.

Figures 3.10, 3.11, 3.12 show the DET plots for the first 5 classes in the MED11 dataset for the various settings of the AUDs. The red circle in the figures represent the NIST benchmark operating point of 75% missed detection at 6% false alarm rate.

We note from the results above that a coarse representation using data-driven AUDs with no guaranteed associated semantics outperform semantic units significantly. A limited set of semantically-defined acoustic units such as those in the Foley set or the phoneme list cannot adequately describe all events that occur in even a small collection of audio recordings. In order to cover even a moderate fraction of all possible phenomena that can occur, one would require an impossibly large vocabulary of semantic units. Even if such a set of units were available, confusions in detecting them automatically in a recording could render them ineffective. Our hypothesis is that this is where data-driven units that can learn a vocabulary to fit the data without making significant prior assumptions can

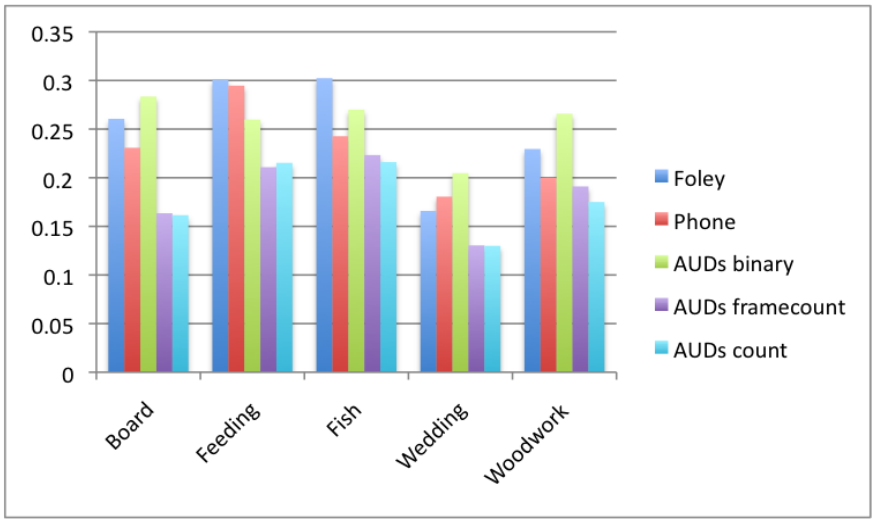


Figure 3.9: AUC for the various feature sets (lower is better)

provide a distinct advantage, and this is borne out by our experiments.

Nevertheless, the AUDs obtained through data-driven discovery are not entirely lacking in semantic association. As we see from the AUDs–binary setting in the results in Fig 3.9, merely detecting the presence of AUDs in a recording is sufficient to identify the event type with a probability that is significantly better than random. This may be interpreted as an indication that the AUDs do carry characteristic information that can distinguish one event type from another, which may in turn imply that they do capture some underlying semantic. Moreover, the AUDs themselves are generatively learnt without any explicit requirement to be discriminative; yet they perform well on a discriminative task, further strengthening the notion that they capture some underlying semantic in the data.

In the results shown so far, we only used the unigram *bag of words*. While n-gram patterns can be expected to carry more information in general, this does not turn out be the case when using AUDs. The main reason for this is likely that the decoding of audio in terms of AUDs is inherently noisy, and we do not get sequences nearly as good as we would obtain if someone were to provide the ground truth transcription in terms of acoustic units. Higher order n-gram patterns over the larger universe of sounds would be harder still because of the even larger set of possible n-grams.

Further, recordings in the same semantic category often vary greatly in how the semantics are conveyed, rendering simple bigram characterizations ineffective. To counter this, we used *flexible* n-grams, where for a given AUD, we look at all the AUDs within a context window, and increment the count of a flexible n-gram, when the context AUDs all appear in the context window. We expect this to reduce the effect of noisy decodings

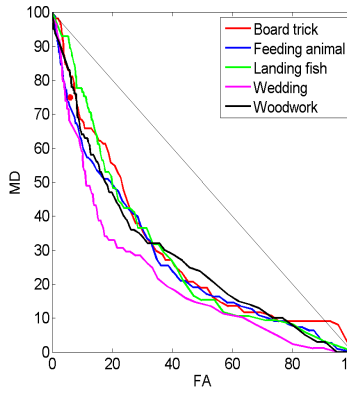


Figure 3.10: DET curve with AUDs_binary

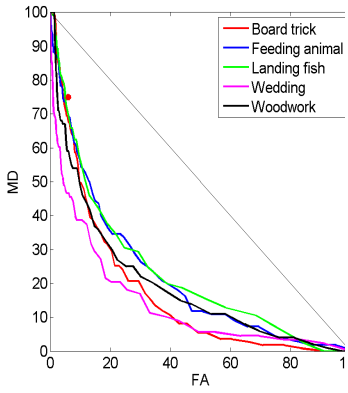


Figure 3.11: DET curve with AUDs_framecount

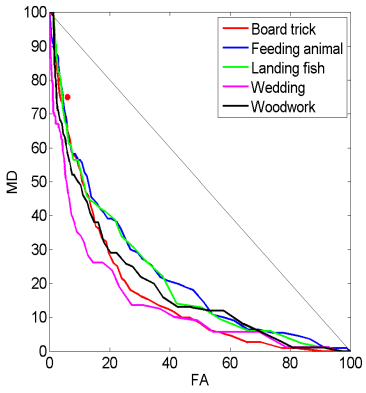


Figure 3.12: DET curve with AUDs_count

System	Average AUC in BBC	Average AUC in MED11
AUDs–count	**0.1744**	0.2273
AUDs–bigram	0.2003	0.2412
AUDs–trigram	0.2023	0.2539
AUDs–bigram-window3	0.1851	0.2264
AUDs–bigram–window6	0.1922	**0.2195**

Table 3.2: Performance of various characterizations using the AUDs

and sequence variations. Table 3.2 compares performance of the various characterizations on the standard datasets, while Figure 3.13 compares performance of the unigram, bigram and flexible bigram characterizations for using a 64-AUD lexicon, across five categories of MED11. We find that using flexible bigrams with a window length of 6 performed best.

We note here, however, that not only do the various higher order modeling of local AUD patterns not result in a significant improvement, but it also suffers from an additional data issue. As we discussed when we introduced the data, the set of data files in the BBC was around 1100 files. Thus, using a bigram characterization with 64 AUDs already results in a feature set of size 4096, which is more than the number of files in the training set. This should be expected to lead to overfitting, and might be one of the reasons higher order characterizations perform worse. The same is true for the MED11 dataset, although performance does seem to improve a little when using higher order models. We cannot, however, convincingly claim that these improvements would generalize, and they might simply be an artifact of the specific data involved.

The one-against-all setting used here allowed us to perform audio retrieval. In the next

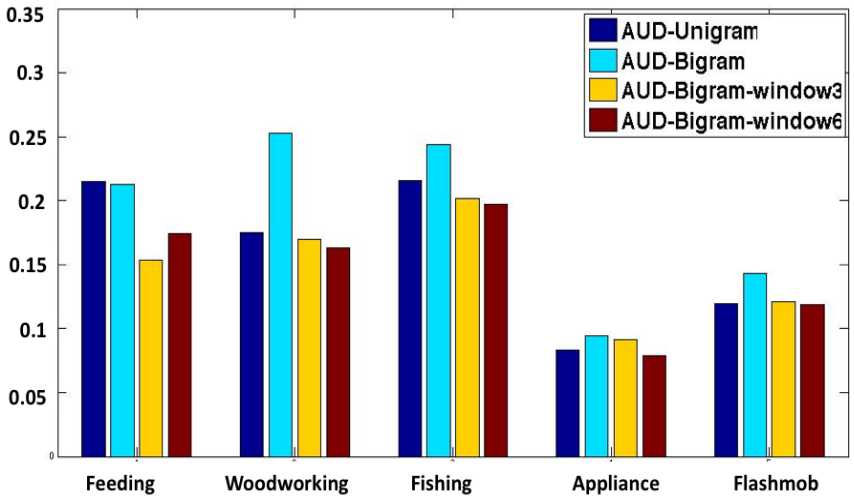


Figure 3.13: AUC for varying characterizations for various MED11 categories (**lower is better**)

section, we will also briefly investigate formulations and approaches for performing audio classification, in a multi-class setting. Finally, while we used AUDs to perform retrieval of entire audio recordings, one could also envision using them to detect smaller sub-events within audio recordings, and we will show an example using AUDs based characterization in Section 3.2.3. Similar approaches could be employed for segmentation as well as co-occurrence analysis of audio events.

3.2.2 Multi-Class Audio Classification

Earlier in this chapter in Section 3.1.1, we discussed a learning process for the AUDs and their distributions. In the experiments we describe in this section, we present results with using an AUDs-based approach to multi-class audio classification using data from the MED, 2010 task with class-specific language models. As before, the models for the AUDs are shared by all these classes. We first describe the dataset, followed by a description of how the AUDs are used to predict a class label, and finally experimental results in Section 3.2.2.

MED10 Dataset

For this task, we worked with the 2010 Multimedia Event Detection (MED) dataset (a subset of MED11) of TRECVid, provided by NIST, similar in characteristics to the MED11 dataset described in Chapter 2. The TRECVid 2010 Multimedia Event Detection dataset

comprises 1746 total clips of training data, totaling about 56 hours in length, and the 1724 clips of test data about 59 hours long. The recordings are publicly available, user-generated multimedia content uploaded to internet hosts. Each video is annotated with one of 4 labels – *making a cake*, *batting in run*, *assembling shelter* and *other*, identifying the kind of activity being performed in it. The class *other* appears to be a catch-all class consisting of all videos that do not belong to the first 3 classes. The use of the audio features in the actual MED10 evaluation was usually limited to speech transcriptions [Li et al., 2010a], and detection of pre-specified sound types in the audio [Hill et al., 2010].

Using AUDs for Class Label Prediction in a multi-class setting

Given the HMM parameters Λ for all AUDs, and the set of language models $H(C)$ for all classes $C \in \mathcal{C}$, we can now classify a new audio recording D into one of the classes by Bayesian classification:

$$C^* = \text{argmax}_C P(D|C; H(C), \Lambda) P(C)$$

To compute $P(D|C)$ we must sum over all possible transcriptions of D, which is generally computationally intractable. Instead, we can employ the common approximation of only considering the most likely transcription:

$$\text{argmax}_C \log P(C) \max_T \log P(D, |T; \Lambda) + \log P(T; H(C)) \tag{3.8}$$

Here $\max_T \log P(D, |T; \Lambda) + \log P(T; H(C))$ is the log likelihood of the most likely transcription of D for class C for and can be computed by the decoder of a speech recognizer. $\log P(D, |T; \Lambda)$ is the *acoustic score* $A(D, C)$ for class C and $\log P(T; H(C))$ is the *language score* $L(D, C)$ for the class.

However, we do not use the above procedure directly for classifying the recordings. Instead we employ a second-level classifier that uses the acoustic and language scores for the class as features. The primary reason for doing this is that even though the decoding uses a scale factor for the language model with respect to the acoustic model, there is no notion of discriminating between acoustic and language model scores with different classes. The weights learnt in the second stage of the process enable comparison of the total scores between classes.

Let $F(D, C) = [A(D, C)\, L(D, C)]^\top$ be a feature vector representing the acoustic and language scores for the data D computed for class C. For each class C we define a two-dimensional weights vector W_C. Classification is performed using the following classifica-

Algorithm 2 Learning weights for each class

M = maxiter; i = 0; $\mathbf{v} = 0$
$w_c = (0, 0), \forall c \in C$
$\mathbf{w}^{(0)} = \{w_1, w_2, ..., w_{|C|}\}$
for $m = 1$ to M **do**
  for $j = 1$ to τ **do**
    $\mathbf{w}^{(i+1)} = \min_w ||w - w^{(i)}||$
    s.t. $S(x_j, y_j) \geq S(x_j, y_c), \forall y_c$
    $\mathbf{v} = \mathbf{v} + \mathbf{w}^{(i+1)}$
    i = i + 1
$\mathbf{w} = \mathbf{v}/(N \times \tau)$

tion rule:

$$C^* = C \ : \ W_C.F(D, C) > W_{C'}.F(D, C') \ \forall \ C \neq C' \tag{3.9}$$

To train the classifier we learn weights W_C for each class as follows: For each training instance D belonging to class C_D we decode D using $H(C)$ to obtain $F(D, C)$ for every class C. A training instance is correctly classified if:

$$W_{C_D}.F(D, C_D) > W_C.F(D, C) \ \forall \ C \neq C_D \tag{3.10}$$

The weights W_C can be learned to maximize classification accuracy on the training data.

We optimize an objective function with an iterative algorithm described in Algorithm 2. The algorithm is an online margin learning algorithm adapted from the Margin Infused Relaxation Algorithm [Crammer and Singer, 2003] (MIRA, henceforth)– it seeks to update the vector of weights after it encounters each new instance so that the weighted score using the feature vector generated by the model for the true class label is greater than the weighted score using models for the other class labels. Let us assume that for each training instance (D), the features scores (features) have been computed, and the label (y_j) is known, and there are τ such training instances. Instead of simply requiring that the score using the true model be greater than the score using the other labels, one can modify this formulation to include a margin by which the score of the true label should be greater than the score of other labels. This can be done by adding a positive term to the right hand side of the constraint. This term may be a constant or may be the result of some loss function.

System	3-class	4-class
64-symbols 1-gram	24.29%	36.44%
64 symbols 2-gram	**23.49%**	35.21%
64 symbols 3-gram	25.00%	47.90%
200 symbols 2-gram	58.12%	**32.58%**
Class-specific HMM	63.12%	56.44%
Random	66.67%	75%

Table 3.3: Classification errors based on Viterbi decoding scores

Experimental Results

We learned models using the data for the 4 classes in the MED10 training data, as described earlier. The 4-class classification task contains an extremely skewed majority class: the *other* class has far more recordings than the other classes. In order to experiment with a more balanced dataset, we also experimented with 3 class classification, leaving out the data from the *other* class.

Table 3.3 reports classification error for the 3 best settings obtained by classifying data directly using their Viterbi decode scores. (Note that, again, a lower number is better since the metric used is a measure of error.) We also compare with a simple baseline model, where we simply model each of the four classes with an ergodic HMM and perform Bayesian classification. The HMMs and the *a priori* class probabilities employed were tuned for best performance in the last case. Table 3.4 reports results using weighted MIRA classifier.

Overall, our experiments indicate that bigram language models outperform both unigram and trigram models on this task. Further, using 64 sound units appears to outperform systems that use more sound units on the 3 class classification task ($p < 0.05$), but it doesn't do as well on the 4-class task, where the 200 symbol system performs better with weak statistical significance (p-value of 0.08). This supports the intuition that more units better capture a larger set of sounds. The MIRA classifier is generally significantly superior to classification based on Viterbi scores alone.

Employing our approach on the MED dataset involves significant challenges. For instance, the *other* class is not consistent in content, and contains a wide array of different audio and video. Besides the *other* class, the remaining 3 classes are not all well-structured. Events in the *batting_in_run* class have audio structure to them, as discussed earlier, but the audio in the *assembling_shelter* and *making_cake* classes are widely varied. Table 3.5 compares the accuracy for each class for the 200 symbol bigram models with the simple baseline class-specific HMM models on 4 class data.

It is not clear to us why the *making cake* class is better predicted with class-specific

System	3-class	4-class
64 symbols 1-gram	18.59%	28.42%
64 symbols 2-gram	**18.39%**	26.39%
64 symbols 3-gram	19.70%	40.28%
200 symbols 2-gram	44.37%	**22.92%**

Table 3.4: Classification errors based on the MIRA classifier

Class	Class-specific HMM	200 symbol 2-gram
assembling_shelter	68.89%	56.00%
batting_in_run	65.38%	40.38%
making_cake	56.14 %	75.86%
other	35.33%	5.3%

Table 3.5: Category specific error for the various classes

HMMs, but we the answer to that may lie in the fact that audio corresponding to this class appears to contain a considerable amount of speech in most cases. Given the small number of AUDs used in describing the data, the speech sounds appear to be clustered with other similar short-term sounds and the distribution of the AUDs are not particularly informative here.

Qualitatively, on analyzing the transcriptions generated on the training data by the iterative learning procedure (using MFCC features augmented with Δ and $\Delta\Delta$), we find that it does a consistent job in identifying some sounds, such as the sound of a baseball bat hitting the ball or clapping, while the AUDs for speech segments, for instance, appear to be more inconsistent, instead being distributed among various units. To improve performance in such scenarios, one could identify speech segments as a processing step before sound unit learning to help the system focus on non-speech acoustic events. An alternative approach to consider might be starting with a small amount of supervision in the form of data for specific class-specific characteristic sounds to help the system converge to a better solution instead of building a sound dictionary completely unsupervised.

3.2.3 Audio Event Detection at a Sub-File Level

In this section, we present our approach to sub-file level event detection tasks in a supervised setting. Here, the term *event detection* refers to detecting an event or segment of interest within a larger audio file by identifying the start and end boundaries of the event in the audio stream. As an example, consider a recording from a sports game– a significant portion of the file contains non-sporting sequences, such as crowd shots, or commercial

breaks. Automatically discovering the segments of interest (*e.g.* highlights from the game) would provide enhanced navigability to the user looking at the recording.

We describe a large-margin, discriminative training method that uses supervised training data to learn the importance of various features in capturing the context. We use a novel feature set for audio, based on acoustic unit descriptors, which are used to describe the sequence of events in the audio. The AUDs are learnt from the data in an unsupervised manner, as described earlier, although the approach used here is general and one could use any other method (*e.g.* sound dictionaries) to obtain the sequence of acoustic units [Kim et al., 2010]. We use baseball data for our experiments from the MED10 *batting in run* class, seeking to create a system that would take a file as input, and remove all non-baseball-action segments from the data. While our experiments in this paper focus on baseball data, the approach is generic and can be applied to any dataset with relevance labels.

This section is organized as follows: we first describe the data used in the experiments. We then describe the learning framework, including the problem formulation, the feature set, and the training paradigm. Finally, we discuss our experimental results.

Data

We work on a dataset that is focussed on a specific topic– baseball videos. We use the *batting in run* data from the TRECVid 2010 Multimedia Event Detection dataset (MED, henceforth) [MED, 2010]. The main reason for using the data in this class is that it is the most structured and deciding on segments of the audio relevant to the topic *batting in run* was easiest for this class. For each file, some segments in the audio were marked as *relevant*. A section is marked as *relevant* if it contains baseball action (baseball action refers to sporting action on the field, as distinct from any similar action in the crowd), while other portions of the video not related to baseball action are marked as *not relevant*.

The training data consist of 54 videos and the test data consist of 52 videos. For both sets of data, the audio is extracted from the mp4 video, and down-sampled to 16KHz, single-channel. The annotator was allowed to use the audio as well as the video to decide which segments were relevant, since the audio doesn't always make it clear whether the relevant section of the video is over. In the context of the audio for the *batting in run* data, examples of sections that are not relevant include cutting to the crowd, or having conversations with a friend, or voiced-over segments of audio. For each audio file in the training set, we have a set of segments that are marked as relevant, and the remaining segments marked as not-relevant. We use this data to extract patterns to help us identify

segments as relevant or non-relevant.

Learning Framework

We showed earlier, in Section 3.1.1, how one can transcribe audio as a sequence of AUDs using Equation 3.3. Since we have annotated data that mark segments as relevant, we can use this information to obtain the sequence of AUDs that appeared in the relevant sections of the audio. Thus, for every file in the training data, we have a full transcription in terms of the AUDs, as well as a truncated version, using only the AUDs that appear in the relevant segments. Let us refer to the original, uncompressed transcription in terms of AUDs as X (where $X = x_1x_2...x_m$), and the truncated version using the relevant segments as Y (where $Y = y_1y_2...y_k$, and $k \leq m$).

Thus, if we have n training audio signals, we use the n pairs $\{X_i, Y_i\}$, $\forall i = 1, 2, 3...n$ as the training data. Thus, we need to train a system using $\{X_i, Y_i\}_1^n$, such that at test time, given a transcription of the test audio file in terms of the AUDs (X_{test}), the system can generate a compressed version of the AUDs (Y_{test}). In order to generate the audio (or video, depending on the application) corresponding to the relevant sections, we can simply synthesize the frames that correspond to the AUDs that were retained.

We would like to note here that the problem setup that we have now is analogous to the problem of text compression by deleting words from text. The problem of text compression has usually been reduced to one of sentence compression and has been approached in a number of ways, including noisy channel models [Knight and Marcu, 2000], integer programming [Clarke and Lapata, 2006] and large margin approaches [McDonald, 2006].

We model our approach to this task as similar to the large margin supervised approaches used in sentence compression research. Unlike the text-based compression approaches which use deep syntactic features based on parse trees generated over the text, we have access only to surface features– AUD identity markers in this case, analogous to the words in the sentences in sentence compression research. We describe the feature set used in Section 3.2.3, and the learning algorithm for training parameters in Section 3.2.3.

Feature set

We jointly extract features over the original and compressed transcription pairs in the training data. First, for every pair of consecutive AUDs in the compressed transcription, we consider the bigram feature $y_{j-1}y_j$. We then extract the context information for each of the consecutive AUDs individually in the transcription– both bigram context and trigram context. We also add an indicator feature that says whether or not any 2 consecutive

AUDs in the compression were also consecutive in the original sentence. These features are intended to understand which AUDs are more likely to be relevant to the topic of the audio, and to understand the contexts in which they are retained.

We also add features for AUDs that were dropped from the original, uncompressed transcription of AUDs. This is done in the following manner: for every pair of AUDs that appears in the compressed transcription, we add features corresponding to the identity of the AUDs dropped from the uncompressed transcription. Further, for every AUD dropped from the original transcription, we add a feature that identifies the AUDs nearest to it on either side that were retained; *e.g.* if the sequence $...a_4a_7a_8a_2a_4...$ becomes $...a_4a_2a_4$ in the compressed transcription, we add features to indicate that a_7 and a_8 were dropped and that they were dropped so that a_4 and a_2 appeared consecutively in the compressed transcription. We then add the bigram and trigram contexts of the dropped AUDs as features. The motivation behind these features is to understand the different contexts that indicate whether or not an AUD should be deleted.

Training

The training process is a large margin online learning algorithm that involves a decoding algorithm that can search the space of all possible compressions in an efficient manner. Given an uncompressed transcription of AUDs $X = x_1x_2...x_m$, we can see that we can generate an exponential number of compressions, depending on the location and number of AUDs dropped.

Thus, for any compression y of an uncompressed transcription X, we need to score the compression such that the selected compression $y^* = \arg\max_y Score(X, y)$. In order to be able to efficiently compute scores over the entire space of compressions, we need to factor the function that computes the score. We do this as follows– suppose the compression y contains k AUDs. Then:

$$Score(X, y) = \sum_{j=2}^{k} g(X, y_{j-1}, y_j) \tag{3.11}$$

The function $g(X, y_{j-1}, y_j)$ represents a weighted scoring function that extracts features on pairs of AUDs that appear consecutively in the compression and the score is obtained as a weighted sum of the feature values:

$$g(X, y_{j-1}, y_j) = \mathbf{w}.\mathbf{f}(X, y_{j-1}, y_j) \tag{3.12}$$

Thus, we can compute the score between every pair of AUDs that could possibly be present consecutively in the compressed transcriptions. Note that this set of AUDs is the same as every pair of AUDs in the uncompressed transcription x_p, x_q, such that $p < q$. Thus, we compute the function g, for every pair x_p, x_q, such that $p < q$, in the original uncompressed transcription of AUDs.

Now, we can compute the best compression for the uncompressed AUD sequence X using dynamic programming over the factored scores. Before we do so, however, we need to modify X slightly, so that we have a dummy start and end position at the beginning and end respectively. We can continue to assume without loss of generality that the new X, with a START symbol at the beginning and an END position at the end is of length m. (At training time, the START and END symbols are inserted to the compressed AUD sequences as well, so that they are never dropped.) We define our dynamic programming table as follows:

$$T[1] = 0.0 \tag{3.13}$$

$$T[i] = \max_{j<i} T[j] + g(X, x_j, x_i), \forall i > 1 \tag{3.14}$$

The table T contains entries from $i = 1, 2, 3..., m$ when the uncompressed sequence of AUDs X is of length m. The entry at position i in the table, $T[i]$, is the score of the compression that ends at position i. Thus, the score of the best scoring path through the AUD sequence is given by $T[m] = T[END\_symbol]$ given the way we factored the scoring, and the path can be found by keeping backpointers to remember where the best scoring path at any index came from.

The process described above is the one used to generate compressions at test time, when we have an input uncompressed sequence of AUDs. However, it is also used at training time, as we shall now describe. We would like to obtain the set of parameters $\mathbf{w}$, such that we can generate the compression y for the training instances X with as little error as possible.

We use an adaptation of the Margin Infused Relaxation Algorithm (MIRA, henceforth) [Crammer and Singer, 2003], a discriminative large margin online learning algorithm, to train weights. The algorithm is summarized below in Algorithm 3. In the algorithm, Y_h refers to the hypothesized compression for the training example X_t, using the weights from the previous iteration. $S(X_t, Y_t)$ refers to the score for the true compression Y_t in the training data. $L(Y_t, Y_h)$ represents a loss function that represents the margin in this algorithm. The training data is represented as a set of pairs $\{(X_j, Y_j)\}_1^N$.

Algorithm 3 Learning weights using MIRA

R = maxiter; i = 0; $\mathbf{v} = 0$; $\mathbf{w}_0 = 0$
for $r = 1$ to R **do**
  for $j = 1$ to N **do**
    $\mathbf{w}^{(i+1)} = \min_w ||w - w^i||$
    s.t. $S(X_t, Y_t) - S(X_t, Y_h) \geq L(Y_t, Y_h)$
    where $Y_h = best(X_t; \mathbf{w}^i)$
    $\mathbf{v} = \mathbf{v} + \mathbf{w}^{(i+1)}$
    i = i + 1
$\mathbf{w} = \mathbf{v}/(N \times R)$

As one can see from the algorithm outlined, each iteration considers only one datapoint in the training set, and adjusts the weights so the score of the correct compression is better than the score of the best compression as hypothesized by the previous set of weights by a margin that is greater than a pre-defined loss function. In our experiments, we define loss as the Levenshtein distance between the correct sequence in the true compression, and the hypothesized compression, with a uniform penalty of 1 for each deletion, insertion and substitution.

Experimental Results

We report our results as precision and recall over whether each frame was classified as *relevant* or *not relevant*. As a baseline for comparison, we use a GMM classifier that classifies each frame in the test data as relevant or not relevant, based on the models it learns from the training data. In our setting, suppose we have c_1 frames that belong to class 1, and c_2 frames that belong to class 2. Suppose, we have a model that predicts class 1 for pc_1 frames, and c of these are correct, then for class 1:

$$Precision = \frac{c}{pc_1}; Recall = \frac{c}{c_1} \tag{3.15}$$

While the interpretations will vary depending on the task that this method is applied to, it appears in this case that false negatives are more harmful than false positives, since if something that is relevant is marked as not relevant, then that segment is removed and the AUDs corresponding to it cannot be used in subsequent processing for classification or retrieval. Thus, high recall for the *relevant* class is a desirable property. Naturally, precision and recall trade off against each other, but a low precision would imply that

System	Prec-Rel	Recall-Rel	Accuracy
Compression-64 AUDs	62.6%	63.8%	63.8%
32 gaussian GMM	63.4%	55.8%	62.7%
64 gaussian GMM	58.8%	50.8%	58.6%
128 gaussian GMM	63.0%	41.4%	59.5%

Table 3.6: *Results on MED10 test data*

either we do a poor job of identifying the relevant segments, or that we are not especially selective in choosing the *relevant* segments. As a result, we will report precision and recall for the *relevant* class (*Prec-Rel* and *Recall-Rel*, respectively) as well as an overall accuracy for the 2 classes combined. Accuracy considers both *relevant* and *not relevant* classes, and the number represents how many frames were correctly labeled over the entire test data. Table 3.6 presents the results of our AUD-based compression system, and compares it with GMM-based classifiers.

The improvement in recall of the *relevant* class using our approach over the baseline GMMs is statistically significant. The performance of the AUD-based and GMM-based approaches are fairly similar in terms of precision, as well as overall accuracy. There is a common trend we observed from our preliminary experiments using both our AUD-compression approach, as well as the GMM approach– as the number of AUDs or GMM components is increased, the recall of the *relevant* class worsens, which indicates that the models start to overfit to the data. It is not immediately clear what the reason for this is, although it could be that the data is coherent enough that a small set of acoustic descriptors (or Gaussian components for the GMM) suffice to model it fairly well. Alternately, perhaps improving the initial segmentation in our AUD-based approach will lead to even better performance.

As described earlier, the key advantage that the AUD-based approach provides in extracting segments is that contiguous chunks of audio are either retained or dropped. Thus, on synthesizing the segments described as relevant by the AUD-based compression approach, the disfluencies are not as apparent as they are in the GMM-classification based approach, where each frame is separately analyzed for potential retention or deletion. It is worth noting that the dropped chunks do occasionally include relevant segments resulting in abrupt, undesirable changes in context. The effect that these errors will have on perception depend heavily on the application. For instance, in generating a condensed version of the game, dropping relevant segments is likely to be quite aggravating for a user. However, if the identification of relevant segments is to be used as a tool to help the user navigate audio files better, then identifying relevant content within small errors can be acceptable

as the user can manually change the boundaries to suit himself.

The technique described for detection of audio could be applied to perform domain specific extraction of relevant segments, useful for generating highlights from sports videos, or to help users of multimodal search systems navigate through search results. We note that the concept of relevance as used in this work, even in limited domains, may not be easy to clearly define since it involves subjective judgments. For instance, users could have obtained the same set of files by searching for *baseball crowd* or *baseball plays*, but the notion of relevance is almost exactly complementary in the two cases, and there could be other such conflicting instances. Our intent was to demonstrate that, given certain accepted definitions of segments of interest, an AUDs–based characterization could be employed successfully to identify such segments in new recordings.

3.3 Discussion

In the various applications we described in this chapter, we discussed in detail the process of learning the lowest layer in the hierarchical framework proposed in this dissertation. We discussed various observations that were made during the learning process for the AUDs. We would specifically like to draw the reader's attention to the fact that the learnt AUDs were distributed in the corpus according to a power law. While this property was not specifically expected or enforced, it is a property that appears to be a characteristic of many natural distributions and one that we utilize in our process of learning the higher order units.

We demonstrated further that these units could be successfully applied to a large-scale audio retrieval task to outperform state-of-the-art techniques for this task. We note again, as discussed earlier, that the AUDs are the lowest level of the hierarchy, and the one that captures the largest amount of acoustic information in the units, since the notion of semantics when looking over such small spans of audio is very unclear. Our experiments also showed that using bag of words representations with higher order bags (bigrams, trigrams, instead of unigrams) were not particularly helpful for our task. We hypothesize 2 reasons that we believe were likely to be responsible for this observation. First, the decodes of the audio are likely to be rather noisy with errors potentially introduced due to the presence of background sounds, random channel effects, or simply decoding errors, and that the sequences obtained by decode were likely to be rather different from the ground truth sequences if such information were available. Second, for semantic tasks, even though the distributions contain the semantics, the specific sequences are not very constrained,

and the same semantic notion may be conveyed with various different orderings of the individual semantic units. Consider movies from the same genre as an example, where the specific sequence of events (and therefore, their acoustic manifestation via the low-level acoustic units) are a manifestation of artistic expression where the director is free to choose how the characteristic event sequences will unfold. A simple n-gram-based model is likely to not be powerful enough to capture such variations.

We note here that the AUDs layer is the only layer of our hierarchy that can be directly compared to the kind of work presented in prior work in the literature, since it analyzes the observed audio data at a shallow level, similar to the prior work. In subsequent chapters, we will explore techniques that will attempt to infer semantics via a deeper analysis. While it is fair to compare the numbers on any specific task (which, in this case, is the audio retrieval task), we note that the kind of structure the underlying models that we shall describe are trying to capture a very different kind of information from those that previous work in this area has attempted to capture.

There are two main directions in which one can proceed with the task of learning higher layers over the low-level acoustic units layer. The first attempts to build up the hierarchy one layer at a time, while the second attempts to infer the full hierarchical structure jointly. Chapter 4 takes the first approach, while Chapter 5 takes the second approach. Each of these differing approaches do have their own advantages and disadvantages and we shall describe them in the discussion section of Chapter 5.

Chapter 4

Layer-wise Training of Higher Levels

In prior chapters, we introduced our hierarchical framework for increasingly higher level semantic analysis of audio content. We showed, in Chapter 3, an approach to learning the low level acoustic units automatically from the audio corpus. We discussed, further, two main paradigms that could be used to induce higher level semantic structure on top of the low-level acoustic units. The first involved building up the hierarchy one layer at a time, while the second involved a joint learning of the entire hierarchical structure. In this chapter, we will present our experiments with the first paradigm, that of building up the hierarchy one layer at a time.

We note here, again, that the training process for the higher layers (both in this chapter and the next) employ formulations that can work unsupervised. Unlike the case for AUDs, where even though ground truth is not available, we can employ various tasks to test how well the AUDs learning process works, such as detection of sound types in the audio stream, the units at the higher levels are expected to be far more semantically-oriented. As such, the importance of being able to test the inferred segments directly is even greater. Unfortunately, since such information is not available in the current datasets, and the development of such datasets would require a significant undertaking, we continue to test the models using the task of audio retrieval based on semantic categories. In this context, however, the process of modeling the higher layers in order to hypothesize semantically coherent segments has an auxiliary benefit in addition to being useful for semantic audio tasks– the hypothesized segments can potentially be used for annotations by human annotators, thus removing the need for the annotator to scan the audio to identify and mark segment boundaries, making the annotation process much faster.

In Section 4.1, we introduce the problem of modeling higher levels of this hierarchy. We note that while current approaches in audio analysis have not really worked on this

problem before, it does bear certain similarities with tasks in text processing, especially because the modeling of the higher layers uses the AUDs layer as input, which is a discrete representation of the observed audio. Following this, in Section 4.2, we present a pair of different models for inducing higher layers in this hierarchy. The first model adopts an approach very similar to the one used for learning AUDs, but its use in characterization of audio for semantic tasks does not prove to be particularly successful. The second model employs a power-law prior in learning the units at a higher level, and proves more successful at the audio retrieval task, when used in conjunction with the low-level AUDs to characterize the recording. We describe our experiments and experimental results in Section 4.3. We conclude with a discussion in Section 4.4.

Part of the work described in this chapter has been published in Chaudhuri and Raj [2012].

4.1 Beyond Acoustic Unit Descriptors

In previous chapters, we discussed the different approaches used in the literature to obtain a dictionary of sound concepts or units (either by supervised or unsupervised approaches), and use these units to characterize audio recordings. However, characterizing audio data with elements from an audio dictionary for semantic analysis involves an implicit assumption that the acoustics map directly to semantics. In reality, we expect the mapping to be more complex, because acoustically similar sounds can be produced by very different sources. Thus, to accurately identify the underlying semantics, we need to effectively use more (and perhaps, deeper) structure, such as the sound context, while making inferences. We present this framework (and our experiments) in the context of audio, but it should also apply to other modalities (*e.g.* video), that require semantic analysis of information sequences.

Traditional detection-based approaches, that assign each frame or a sequence of frames of pre-specified length to sound categories/clusters, are severely limited in their ability to account for context. In addition to context, we need to consider the possibility of polysemy in sounds– semantically different sounds may be acoustically similar; *e.g.* a dull metallic sound may be produced by a hammer striking an object, a baseball bat hitting a ball, or a car collision. The sound alone doesn't provide us with sufficient information to infer the semantic context. However, if the sound is followed by applause, we guess the context to be baseball, screams or sirens suggest an accident, while monotonic repetitions of the metallic sound suggest someone using a hammer. In order to be able to capture this kind

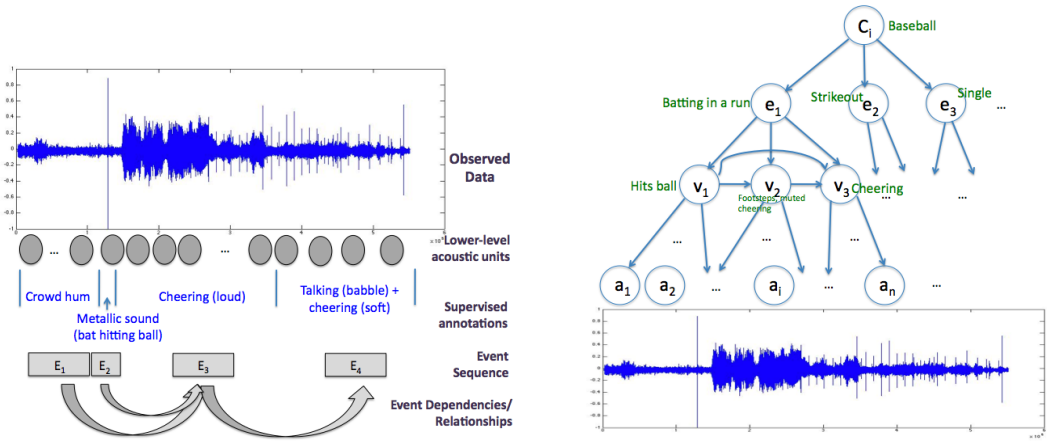


Figure 4.1: The proposed hierarchical framework (a) Left: Conceptualizing increasingly complex semantic analysis; (b) Right: An example semantic parse for baseball .

of temporal context, we need to not only identify the kind of low-level sound events (such as the dull metallic sounds), but also be able to disambiguate the sounds using an analysis of the possible relationship of that sound to other temporally neighboring sounds, and its implication for semantics.

Recall from Chapter 1, our conceptual representation of a hierarchical framework that envisions a system to perform increasingly complex analysis of audio, shown again In Figure 4.1a. The grey circles closest to the observed audio represent short-duration lower-level acoustic units (AUDs) which produce sounds that human ears can perceive, such as the *clink* of glass, *thump* produced by footsteps, etc. These units have acoustic characteristics, but no clear associated semantics since the semantics may be context dependent. Sequences of these units, however, will have interpretable semantics– we refer to these as *events* marked by grey rectangles in Figure 4.1a. The annotations in blue correspond to (usually unavailable) human labels for these events. Further, these events themselves likely influence future events, shown by the arrows, *e.g.* the loud cheering in the audio clip is because a hitter hit a home run.

Figure 4.1b shows the kind of structured information that we envision parsing from the audio. The lowest level, indexed by a, correspond to the lower-level units. The event layer in Figure 4.1b has been further divided into 2, where the lower level (indexed by v) correspond to observable events (*e.g.* hit-ball, cheering), whereas the higher level (e) corresponds to a semantic event (*e.g.* batting-in-run), and the root node represents the semantic category (baseball, in this case). The cost of obtaining such hierarchical annotations would be very high due to the complexity of the annotation task. Typically, audio

datasets contain only a category or genre label for each audio file. As a result, models for learning such structure must be able to operate in an unsupervised framework. Our focus, in this chapter, is on developing such models that will allow us to learn models for the higher order *events*, shown by the grey rectangles, in an unsupervised framework. Note that, just like the process of learning models for AUDs, the approach presented here for modeling higher layers is also task-agnostic, while keeping in mind that the general goal of this process is to be able to model semantics.

The framework described in this chapter for modeling higher layers for semantic analysis of audio is the first effort of its kind to extract deeper semantic structure from audio, to the best of our knowledge. The approach presented in this paper for modeling higher layers is presented in the context of modeling the layer immediately above the AUDs layer. Thus, the input to the system is a dataset of discrete sequences (AUD sequences), and the framework attempts to learn models for units in the layer above it (*events* layer). As one can see, such a framework can be iteratively extended. Once we've learnt units for a layer on top of the AUDs layer, we can in turn decode the AUD sequences using those units to obtain the sequence of units in the higher layer, which can, once again, be an input to the same system to infer a higher layer still.

In the next section, we will present our formal models for deriving higher level structure over local patterns over the AUDs. Before we present these models, however, let us briefly consider the task that we are attempting to accomplish, its relation to prior work in intelligent systems, and its main differences from such work.

4.1.1 Inferring Higher Level Structure

We expect that audio data are composed of a sequence of semantically meaningful *events* which manifest themselves in various acoustic forms, depending on the context. The acoustic unit (AUD) lexicon described in Chapter 3 automatically learns the various acoustic manifestations from a dataset but do not have interpretable semantic meaning. Instead, we expect to find semantics in the local patterns over the AUDs. Thus, given a sequence of AUDs as input, we want to identify the elements of a hidden layer on top of the AUDs layer. These elements can be considered as corresponding to semantically interpretable acoustic events which generate lower level AUDs (and thus, the observed audio).

Therefore, at training time, only the AUD token sequences are observed. We refer to the observed AUD tokens as $\mathcal{X}$, the elements in the *events* layers are modeled as latent variables $\mathbf{Z}$ and the parameters for our processΘ. We would like to estimate the parameters Θ at training time, so that we can then use these parameters to estimate the latent events

present in audio based on an observed AUD stream (the AUD stream is obtained by decoding audio as described in Chapter 3). We will discuss the specifics of the training and decoding algorithms in Section 4.2.

However, first, let us consider the problem of inducing a higher layer at an intuitive level. In the generative model, there is a sequence of conceptual units in the layer above the AUDs. Following the thread of the baseball example, let us imagine that these are play–level units, such as batting in a run, strikeout, etc. Each of these will manifest in some way– the batting in a run element might have the sounds of pitches hitting the catcher's mitt, followed by a bat hitting the ball, running footsteps and then cheering. Each of these can be captured by low level AUDs, which in turn produce the observed audio. Similar sequences exist for the subsequent plays, as well

In such a sequence of AUDs, however, we do not know the *play* boundaries, nor do we know a priori which AUDs correspond to which plays. For instance, the sound of the ball hitting the catcher's glove and cheering might be present in all the plays but not the sound of bat hitting the ball or people running. However, this information has to be inferred from the local patterns over the AUDs. Thus, the task of identifying higher level units, given only the AUD sequences can be considered as a segmentation task, where each of the higher level units correspond to a segment. In the next subsection, we will discuss an intuitive formulation using an example from the text domain, and then explain why the task is different in the case of semantic audio, which has its own challenges.

A Text Processing Analogy for Audio Event Segmentation

As described in the previous section, one can think of the task as analogous to a case where sequences of discrete symbols need to be segmented. A relevant analogous task in the text processing domain would be the identification of a vocabulary of words if the word boundaries (spaces) were removed from any text. Indeed, algorithms have been developed in the text processing domain for automatic vocabulary induction from text to identify tokens of interest, applied to the tasks of discovering the word vocabularies for rare languages, segmentation in languages where spaces are not used (*e.g.* traditional Chinese or Japanese), and morphology induction [Goldwater et al., 2009, Johnson and Goldwater, 2009, Mochihashi et al., 2009, Poon et al., 2009, Schuster and Nakajima, 2012], where a token stream is input to the system and the various approaches learn models that can appropriately segment new sequences.

However, in the semantic audio space, the task of obtaining the higher level units does not correspond to a segmentation task quite as neatly as it does in the text space for

discovery of words and word-like units. There are 2 main differences between the tasks of word segmentation for text and the AUD segmentation for semantic structure discovery in audio– the problem of noisy transcriptions and the problem of a lack of canonical structures.

The Problem of Noisy Transcriptions

When modeling higher layers, the audio is represented using a sequence of AUDs. Although we have mentioned that the mapping between the acoustics of a recording and the underlying sequence of AUDs is stochastic, we have largely treated the AUDs as known. The reality of the situation is otherwise. The AUDs themselves are *latent* – they are not observed and can only be inferred. The process of decoding of the observed audio into AUD sequences can be prone to error due to many reasons.

The process of learning of AUDs was unsupervised, since ground truth in terms of AUD sequences were not provided at the time of learning. As a result, there is no guarantee that the set of AUDs learnt correspond to the optimal set of sounds one would like to capture. Further, as discussed earlier, the AUD units likely contain a mixture of sounds from very different semantic sources, and disambiguation is expected to be handled at higher layers. For context-based disambiguation to work well, the neighboring units hypothesized by an AUD decode in any sequence need to be close to the truth. However, AUD decodes can result in local errors due to a variety of reasons, including channel artifacts, arbitrary recording conditions, presence of background noise in the form of objects not related to the semantic subject (*e.g.* loud car horns in the background in a birthday party), and so forth.

Thus, the input transcriptions are not always ground truth, which is usually the case when dealing with the analogous problem in text, where the character sequences with spaces removed are, in fact, ground truth. [Note: This input setting is, however, similar to the problem of phoneme transcriptions, where the decoded phoneme sequences may not be ground truth, but unlike the case of speech, there is no equivalent of a dictionary.]

The Problem of a Lack of canonical structure

In text, where the units that are sought to be discovered are words, the structure of the words in terms of characters are well defined. Each word is spelled with a canonical sequence of the lower level units (characters of the alphabet). In the case of semantic audio, this differs in 2 respects.

First, the AUD sequences decoded from the raw audio are likely to be noisy, as was just discussed above. Thus, if we assume that the semantic audio events do have a canonical

structure, that structure is not very likely to be present in the sequence of units that are input to this layer.

However, secondly, and perhaps more importantly, semantic events simply cannot be expected to have the kind of canonical structure that words have. Take the example shown in Figure 2.1 in Chapter 2. Ignoring that the difference in the two manifestations may be caused by decoding errors, it should be readily obvious that the recording of the same semantic concepts may manifest very differently due to differences in artistic expression of the people creating the recording, geographical and cultural differences between locations where the same concept was recorded, etc. Indeed, considering the large number of different scene sequences in any movie genre is sufficient to convince the reader that very similar ideas can be communicated in extremely different ways.

As a result, we cannot simply use methods analogous to those used for text segmentation. In Section 4.1.2, we present an intuitive way of conceptualizing the problem while using the thread from text segmentation, since the discrete symbol space of the AUDs is analogous to that of the alphabet in text segmentation.

4.1.2 Modeling Audio Event Segmentation

Given the two main problems discussed, our models for the higher layers will need to tackle them to be successful. In Section 4.2, we will present a pair of generative models for segmentation, one which explicitly tackles the problems, and which does not, instead being based off of a generative model that simply assumes local sequential structure.

Consider the task of segmentation of English text. In the regular text segmentation setting, one would simply remove the spaces from a stream of words, and each word is a sequence of characters corresponding to its correct spelling. In our case, due to the presence of semantic variations as well as noise introduced by decodes, a better analogy would be that the words themselves are badly spelt. Multiple instances of these *semantic* words would contain some of the elements that are truly representative of the semantics, but some of these elements might be missing due to decoding errors. Thus, in the text segmentation analogy, a sequence of words is generated, but the words are badly spelt. Following this, spaces are removed and the sequence of characters is input to the system. We will refer to this as the *Crazy Typist Model* (CTM, henceforth) where the crazy typist is responsible for creating sequence of AUDs by taking the high level semantic events, creating a *mis*spelling in terms of AUDs, removing the spaces to produce the final sequence that generates the observed audio.

In Section 4.2, we will present a formal model for the Crazy Typist Model, as well as a

formulation for training and decoding models. The effectiveness of the various formulations for inducing higher level structure will depend on the structure employed to govern the AUD sequences, and potentially, on the task it is being applied to as well. In general, if we were to impose a higher degree of structure, we can employ simpler formulations, since the structure would guide the solution to a greater degree.

4.2 Generative Models for Inducing Patterns over AUDs

In this section, we present a pair of approaches to modeling patterns over AUD sequences. The first, described briefly in Section 4.2.1, follows a process analogous to the process of learning AUDs from the audio. Due to the problems in modeling the higher, more semantic layers described in the previous section, this model is very limited in its abilities. We discuss these limitations as well, before describing an improved framework that can deal with the Crazy Typist Model in Section 4.2.2.

4.2.1 Structured Sequential Patterns over AUDs

This model is analogous to the model described in Chapter 3 for learning AUDs. Recall the graphical model for AUDs learning that was presented in Figure 3.2. We can employ exactly the same model to induce higher layer structure while modifying the data D to consist of discrete sequences, and our modeling of the parameters so that the output distributions for each of the states would be a discrete multinomial distribution.

We model the higher layer units as we did for AUDs, with a left-to-right Hidden Markov Model with a varying number of states, and discrete multinomial output distributions. We will present results for this framework as well as the Crazy Typist Model in Section 4.3.

We note, however, that this model assumes a similar situation to the regular text segmentation task that does not need to deal with variances in sequences due to semantics or noisy transcriptions. Not surprisingly, then, this model performs significantly worse in capturing higher order structure than the Crazy Typist Model that we will now introduce.

4.2.2 The Crazy Typist and a Power-Law Prior-based Model

We briefly mentioned the Crazy Typist Model in Section 4.1.2. This model is so-called because we introduced the notion of a Crazy Typist to deal with the modifications that had to be introduced to the generative model for text segmentation to deal with the specific differences for semantic audio.

Specifically, the crazy typist model can be conceptualized as follows: For the semantic audio space, there are a number of semantic audio units, each of which contains some true spelling units in terms of AUDs. The actual sequence of these units in the *spelling* are left to the crazy typist's imagination. In order to generate the audio document, the typist decides on a sequence of the semantic audio units. For each of these units, he creates a spelling manifestation including some or all of its true spelling AUD units (modeling absence of some units). This manifestation may include additional AUDs that are not part of the true spelling (modeling noise). Each manifestation, therefore, is a sequence of AUDs. The AUD sequences for each semantic unit are concatenated together with no further markers to denote where each unit ends. The AUD sequence then produces the observed audio, using the individual AUD models.

Note that while this generative model is presented in the context of a higher layer over AUDs, it can be iteratively extended to generate further layers.

For this task, we assume that AUD models have already been learnt and that we are interested in learning the distribution of AUDs in the various higher level semantic units. We noted earlier that the effectiveness of the various formulations for inducing higher level structure will depend on the structure employed to govern the units. We then considered the question of what kind of structure might be appropriate for this task.

Recall our observation, at the time of AUDs learning, that the distribution of AUDs followed by a power-law, which was a property that had been observed in a number of natural distributions. When we applied a method analogous to the AUDs discovery process to learn higher level units, as described in Section 4.2.1, the distribution of the higher level units did not turn out to follow the kind of power law we had observed for AUDs learning. Therefore, we decided to explicitly enforce the power-law prior on the higher level units.

In this section, we will first describe our graphical model using a power law prior that explains the generative process. Then, we will describe how we can perform learning and inference using an Expectation-Maximization algorithm for this model. We note that the model is described using a power-law prior, and the derivation of the learning and inference assume the same. However, if one so wishes, the power-law prior can be replaced by any other prior of choice. Naturally, in that case, the update rules will have to be appropriately modified.

The Generative Crazy Typist Model

We refer to the higher layer units in this section as *events*, since they correspond to units in what was called the *events* layer. Note that these events are different from the query

event types in the dataset for audio retrieval, and that no task-specific assumption is made in the development of this model.

We create a generative model corresponding to the Crazy Typist Model, where we impose a power-law prior on the distribution of events. Events are drawn from this power-law distribution, and then generate lower level acoustic units (AUDs) corresponding to the sounds that are to be produced. Because this process is stochastic, different occurrences of the same event may produce different sequences of AUDs, which are variants of a common underlying pattern.

Formally, we model this with a generative, latent variable model in the following manner. We assume that we know the size of the vocabulary (of higher level semantic units), as well as the size of the character set (AUDs). In our model, we also assume that we know something about the distribution of the unigram probabilities (*e.g.* that the unigram distribution follows a power law). For each document d, we first draw a unigram distribution for the document, based on our beliefs. Then, given the number of higher level semantic unit tokens present in the document (N_d), we draw N_d tokens from our vocabulary as follows: sample a word (semantic unit) from the unigram distribution. For this word, we will generate a character (AUD) sequence as follows: From a negative binomial distribution *for that word*, draw the length of the word (n). Now, draw a character from the character distribution for this word n times to generate a character sequence (c_1^n).

The generative model is shown in Figure 4.2. Our use of notation is as follows:

K: number of words in the vocabulary

M: number of characters in the character set

N_d: number of word tokens in document d

D: number of documents in the corpus

μ: prior beliefs about the unigram distribution

U: the unigram distribution for a document

w: a sampled word

α: Negative binomial distribution with 1 distribution for each word. Each distribution has 2 parameters, r is the number of allowed failures, while p is the probability of success

n: the length of each word

Φ: Probability distribution table for emissions given words

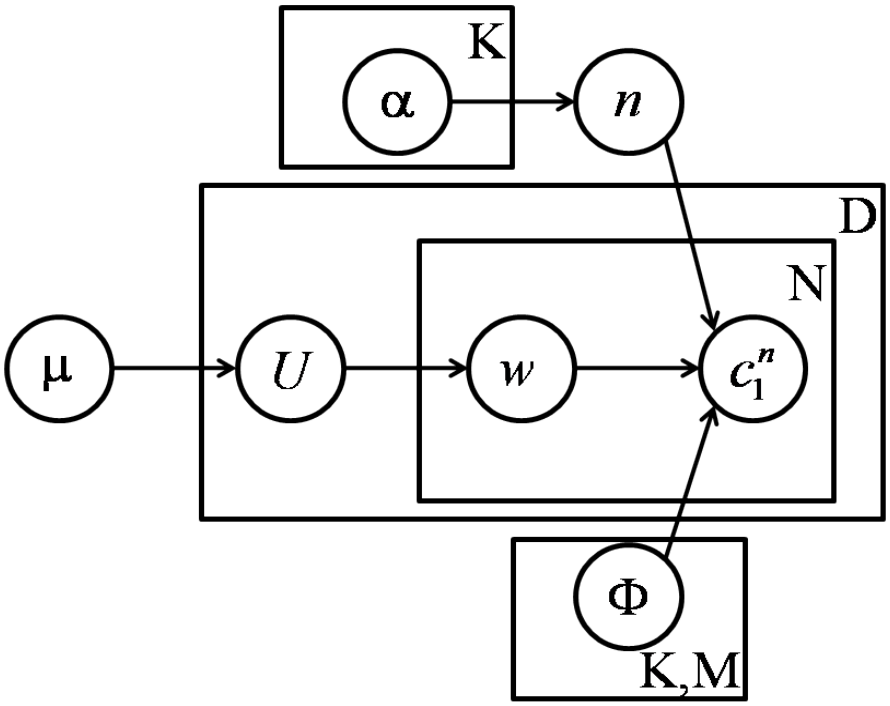


Figure 4.2: The unigram based generative model for segmentation. Only c_1^n is observed.

Thus, in this model, each audio document is a *bag of events* and each occurrence of an event is a *bag of AUDs*; the events themselves are distributions over AUDs, modeled by Φ.

At training time, only the AUD token sequences are observed. We refer to the observed AUD tokens as as $\mathcal{X}$, the latent variables as $\mathbf{Z}$ and the parameters for our process (μ, α and Φ) as Θ. We can write the joint probability of all the variables in this model as shown in Equation 4.1. In the following subsections, we will outline a framework for training the parameter set for this model. We can then use these parameters to estimate the latent events present in audio based on an observed AUD stream (the AUD stream is obtained by decoding audio, which we shall discuss, as well).

$$P(\mathcal{X}, \mathbf{Z}, \mathbf{\Theta}) = \prod_{\mathbf{d}} \mathbf{P}(\mathbf{U_d}; \mu) \prod_{\mathbf{i}} \mathbf{P}(\mathbf{w_i^d}|\mathbf{U_d}) \mathbf{P}(\mathbf{n_i^d}|\mathbf{w_i^d}; \alpha) \mathbf{P}(\mathbf{c_1^n}|\mathbf{w_i^d}, \mathbf{n_i^d}; \mathbf{\Phi}) \tag{4.1}$$

We chose the 2-parameter (r, p) Negative Binomial (Equation 4.2) distribution for α, which approaches the Poisson distribution as r tends to infinity, and the r controls deviation from the Poisson.

$$n \sim NB(r, p), s.t.\ P(n = k) = \binom{k + r - 1}{k} p^k (1 - p)^r \tag{4.2}$$

The power law prior is imposed by a 1-parameter (s) distribution shown in Equation 4.3 ($w^{(k)}$ represents the k-th most frequent word), where the parameter s is drawn from $\mathcal{N}(\mu, \sigma^2)$. For English text, the value of s has been observed to be very close to 1.

$$P(w^{(k)}; s, n) = \frac{\frac{1}{k^s}}{\sum_{i=1}^{i=n} \frac{1}{i^s}} \tag{4.3}$$

Various methods can be used for parameter learning. We present a Hidden Markov Model based model that is used to estimate the parameters in an Expectation-Maximization (EM) framework [Dempster et al., 1977]. We now describe the learning framework, and how the parameter update equations are obtained.

Parameter Updates for our model

We will now describe how to estimate the parameters of this model in order to maximize the likelihood of the data. The various parameters we want to estimate in this case are μ, Φ and α. Recall, from Chapter 2, our brief guide on using EM for learning parameters in a latent variable framework. EM is an iterative framework where we begin with a set of initial guesses for the set of parameters Θ, and iteratively improve our estimate by updating the parameters such that the likelihood of observing the data improves. In our case, since we have priors over some parameters, this is done in an *a posteriori* setting.

[Note, as discussed earlier, that the use of terminology here is motivated by the text segmentation problem– *word* refers to the higher-level semantic units for our purposes, and *character* refers to the lower level acoustic units or AUDs.]

Specifically, recall Equation 2.3, which was the starting point for deriving parameter updates. We will use this equation to derive update rules for the parameters of our system.

We can optimize these parameters individually, using only the terms from Equation 4.1 that contain these parameters since the other terms would not affect the estimation of the current parameter being optimized.

Optimizing character emission probabilities Φ

$$\Phi^* = \arg\max_{\Phi} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_d \prod_i P(\bar{c}|w_i^d, n_i^d; \Phi)\Big] \tag{4.4}$$

We can rewrite the term above and rearrange the terms as shown by the sequence of steps below ($ct_{ij}(z)$ represents the count of the number of times character j was emitted by word i in the latent variable z):

$$\begin{aligned}
\Phi^* &= \arg\max_{\Phi} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_{d} \prod_{i=1}^{K} \prod_{j=1}^{M} \Phi_{ij}^{ct_{ij}(z)}\Big] && (4.5)\\
&= \arg\max_{\Phi} E_{Z|\mathcal{X};\Theta^r}\Big[\sum_{d} \sum_{i=1}^{K} \sum_{j=1}^{M} ct_{ij}(z) log\Phi_{ij}\Big] && (4.6)\\
&= \arg\max_{\Phi} \sum_{Z} P(Z|\mathcal{X};\Theta^r)\Big[\sum_{d} \sum_{i=1}^{K} \sum_{j=1}^{M} ct_{ij}(z) log\Phi_{ij}\Big] && (4.7)\\
&= \arg\max_{\Phi} \sum_{i=1}^{K} \sum_{j=1}^{M} \sum_{d} \sum_{Z} P(Z|\mathcal{X};\Theta^r) ct_{ij}(z) log\Phi_{ij} && (4.8)
\end{aligned}$$

Now, the character emission probabilities for different words are independent of each other and the character emission probabilities for a given word sum to 1. So, we can estimate the probabilities for each word independently. We rearrange and write the terms in Equation 4.8 as follows:

$$\Phi_i^* = \arg\max_{\Phi} \sum_{j=1}^{M} \sum_{d} \Big[\sum_{Z} P(Z|\mathcal{X};\Theta^r) ct_{ij}(z)\Big] log\Phi_{ij} \tag{4.9}$$

We can differentiate the equation above to obtain the optimal solution. However, in this case, we can use Gibbs' inequality which states that for 2 probability distributions p and q, $\sum_i p_i log(q_i) \leq \sum_i p_i log(p_i)$, and equality holds when $p = q$. Thus, we can obtain a solution to Equation 4.9 by setting the following:

$$\begin{aligned}
q &= \Phi_{ij}\\
p &= \frac{\sum_d \Big[\sum_Z P(Z|\mathcal{X};\Theta^r) ct_{ij}(z)\Big]}{\sum_{j=1}^{M} \sum_d \Big[\sum_Z P(Z|\mathcal{X};\Theta^r) ct_{ij}(z)\Big]}\\
\Phi_{ij} &= \frac{\sum_d \Big[\sum_Z P(Z|\mathcal{X};\Theta^r) ct_{ij}(z)\Big]}{\sum_{j=1}^{M} \sum_d \Big[\sum_Z P(Z|\mathcal{X};\Theta^r) ct_{ij}(z)\Big]}
\end{aligned} \tag{4.10}$$

In order to estimate the parameters now, we still need to be able to estimate the numerator in Equation 4.10.

$$\begin{aligned}
\sum_{Z} P(Z|\mathcal{X};\Theta^r)ct_{ij}(z) &= E_{Z|\mathcal{X};\Theta^r}\Big[ct_{ij}(z)\Big] \\
&= E_{Z|\mathcal{X};\Theta^r}\Big[\sum_{t=1}^{T} ct_t(c_j|w_i)\Big] \\
&= \sum_{t=1}^{T} E_{Z|\mathcal{X};\Theta^r}\Big[ct_t(c_j|w_i)\Big] \\
&= \sum_{t=1}^{T} P(w_t = i)\mathcal{I}(c_t = j) \qquad (4.11)
\end{aligned}$$

In the above, $\mathcal{I}(c_t = j)$ represents an indicator function which has value 1 if the argument is true and 0 if it is not true. Thus, we see that in the E-step we will need to estimate the $P(w_t = i)$, which is the probability of being in word i at timestep t.

Required to estimate in E-step: **Probability of being in hidden word i at timestep t**

Optimizing length of words parameter α

In our model, we've chosen to model the distribution over lengths of words as a negative binomial distribution. The negative binomial is parameterized by two parameters – r which is the number of failures after which generation is stopped, and p which is the probability of success.

$P(X = n|r, p)$ represents a draw from the negative binomial distribution parameterized by r and p. n, which is the result of the draw is the number of successes observed before generation was stopped due to r failures. The distribution is represented as follows:

$$P(X = n|r, p) = \binom{n + r - 1}{n} p^n (1 - p)^r \qquad (4.12)$$

As before, we begin with the equation obtained in 2.3 and rewrite the joint probability within the logarithm using the terms from the joint representation of our graphical model that contain α. Thus, we have:

$$\alpha^* = \arg\max_{\alpha} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_{d} \prod_{i} P(n_i^d|w_i^d;\alpha)\Big] \qquad (4.13)$$

We note here that the presence of the word i as a conditioning variable is to indicate a procedural detail, since it indicates the correct α parameter to be used to generate the length n since there is an α corresponding to each of the hidden words. Let us also assume

that word lengths can vary in a range between n_lo to n_hi. We can rewrite the equation as before:

$$\begin{aligned}
\alpha^* &= \arg\max_{\alpha} E_{Z|\mathcal{X};\Theta^r}\Big[log\prod_d\prod_i\prod_{j=1}^{k}\prod_{n=n_{lo}}^{n=n_{hi}} P(n_i^d|w_i^d;\alpha)^{ct_{jn}(Z)}\Big] \\
&= \arg\max_{\alpha} E_{Z|\mathcal{X};\Theta^r}\Big[\sum_d\sum_i\sum_{j=1}^{k}\sum_{n=n_{lo}}^{n=n_{hi}} ct_{jn}(Z)logP(n_i^d|w_i^d;\alpha)\Big] \\
&= \arg\max_{\alpha} \sum_{j=1}^{k}\Big[\sum_d\sum_i\sum_{n=n_{lo}}^{n=n_{hi}}\sum_Z P(Z|\mathcal{X};\Theta^r)ct_{jn}(Z)logP(n_i^d|w_i^d;\alpha)\Big] \\
&= \arg\max_{\alpha} \sum_{j=1}^{k}\sum_{n=n_{lo}}^{n=n_{hi}}\sum_d\sum_i E_{Z|\mathcal{X};\Theta^r}[ct_{jn}(Z)]logP(n_i^d|w_i^d;\alpha) \qquad (4.14)
\end{aligned}$$

The term $P(n_i^d|w_i^d;\alpha)$ is the probability of the observed n given the negative binomial distribution. As shown before in the estimation of character emission probabilities, this can be solved using the Gibbs' inequality. In this case, however, the distribution has 2 parameters, and we need an additional step to solve for those parameters. For a given word, we can obtain the expected count of the word-length n as $\sum_d\sum_i E_{Z|\mathcal{X};\Theta^r}[ct_{jn}(Z)]$ from Equation 4.14. Let ct_i be the count of length i, and let ct_n be the sum of the counts of all lengths.

Thus, if for word i, we have a set of m occurrences in these paths of lengths $n_1, n_2, ..., n_m$, we can estimate r and p using Equation 4.15. p has a closed form solution (Eqn 4.16) but Eqn 4.17 for r needs an iterative numerical solution[1]. $\psi()$ is the digamma function)

$$L = \prod_{i=1}^{m} NB(x = n_i; r, p) \qquad (4.15)$$

$$p = \frac{\sum_{i=1}^{m}\frac{n_i}{m}}{r + \sum_{i=1}^{m}\frac{n_i}{m}} \qquad (4.16)$$

$$\sum_{i=1}^{m}\psi(n_i + r) - m \times \psi(r) + m \times ln(\frac{r}{r + \sum_{i=1}^{m}\frac{n_i}{m}}) = 0 \qquad (4.17)$$

Required to estimate in E-step: **Count of word i and length n,** $\forall n \in \{n_{lo}, n_{hi}\}$

[1]Note that MATLAB provides a function called *negbinfit* that can be used to obtain solutions for the equations below

We note that, for computational tractability, we are forced to limit the lengths of the words between a range (n_{lo}, n_{hi}).

Optimization of the super-unigram distribution parameter μ

Again, as before, we can obtain the equation for optimizing μ by retaining the terms in Equation 2.3 that contain it. This results in:

$$\mu^* = \arg\max_{\mu} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_{d} P(U_d;\mu)\Big] \tag{4.18}$$

Using the previous method of derivation, with $ct_i(Z)$ representing the count of the i-th observation relevant to the unigram distribution drawn, we can obtain an update rule as follows:

$$\begin{aligned}
\mu^* &= \arg\max_{\mu} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_{d} P(U_d;\mu)\Big] && (4.19)\\
&= \arg\max_{\mu} E_{Z|\mathcal{X};\Theta^r}\Big[log \prod_{i=1}^{|U|} P(U_i;\mu)^c t_i(Z)\Big] && (4.20)\\
&= \arg\max_{\mu} E_{Z|\mathcal{X};\Theta^r}\Big[ct_i(Z) log P(U_i;\mu)\Big] && (4.21)\\
&= \arg\max_{\mu} \sum_{i=1}^{|U|}\sum_{Z} P(Z|\mathcal{X};\Theta^r) ct_i(Z) log P(U_i;\mu) && (4.22)
\end{aligned}$$

As before, using Gibbs' equation we can obtain:

$$\begin{aligned}
q &= P(U_i;\mu) && (4.23)\\
p &= \frac{\sum_Z P(Z|\mathcal{X};\Theta^r) ct_i(Z)}{\sum_j \sum_Z P(Z|\mathcal{X};\Theta^r) ct_i(Z)} && (4.24)
\end{aligned}$$

It's not clear at this point what the best selection of μ is. If we consider μ to be a multinomial over unigram distributions, then the implication of U_i is obvious as the ith element of the multinomial. A multinomial over unigram distributions is not, however, a very intuitive choice (even though it is a perfectly reasonable choice). In our work, we've used a unigram distribution as following power-law properties with one parameter s drawn from $\mathcal{N}(\mu, \sigma^2)$.

To estimate the $\mathcal{N}(\mu, \sigma^2)$ for the power-law parameter s, we compute expected word

(event) frequencies E_{f_i} for all events for each AUD stream. This can be done using the forward-backward table as shown in Equation 4.25 and 4.26. The Zipf parameter is estimated as the slope of the best-fit line between the log-expected-frequencies (Y) and log-rank ($X = [log\_rank \;\; 1]^T$). The set of s values in the corpus are used to estimate the μ and σ^2.

$$E - Step \;\; : \;\; count(w_i) = \sum_{t=1}^{T} P(state = \mathcal{S}_i, t) \tag{4.25}$$

$$E_{f_i} = \frac{count(w_i)}{\sum_j count(w_j)} \tag{4.26}$$

$$M - step \;\; : \;\; s_d = (YX^+)_0, \forall d \in D \tag{4.27}$$

$$\mu^*, \sigma^{2*} = \arg\max_{\mu,\sigma^2} \prod_{i=1}^{i=D} P(s_i | \mathcal{N}(\mu, \sigma^2)) \tag{4.28}$$

Without loss of generality, we could have imposed any other distribution on s as well.

Required to estimate in E-step: **Counts of** E_{f_i}

Latent Variable Estimation in the Learning Framework

Our discussion so far has wished away the computation of expected counts as a task to be carried out in the E-step. Based on the derivations of the update rules, we can compile the list of required counts to be obtained for updation of parameters. These are:

- Probability of being in hidden word i at timestep t
- Count of word i and length n, $\forall n \in \{n_{lo}, n_{hi}\}$
- Counts of parameter s or relevant terms

Given our parameters from the previous iteration of the EM algorithm, we would like estimate the above terms. The process of estimation of these required counts actually turns out to be fairly straightforward due to the modeling choices we make, that allow us to model the variable length of events in terms of AUDs with an HMM model efficiently. Once we do this, the process of estimation of the sufficient counts fall into the HMM paradigm where the Forward-Backward tables can be constructed and used to efficiently estimate them.

First, we construct an automaton for each of the K events in the higher level unit vocabulary– an example is shown in Figure 4.3. This example allows a maximum length of 3, and has 4 states for lengths 0 to 3 and a fifth dummy terminal state. The state for length

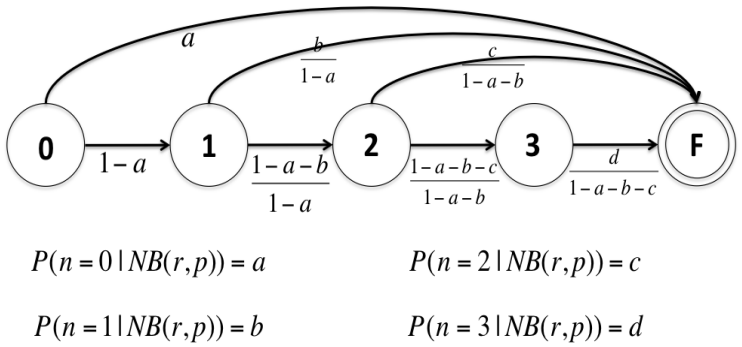


Figure 4.3: An example automaton for a word of maximum length 3. a, b, c and d represent the probabilities of lengths 0 to 3 given the parameters r and p for the negative binomial distribution.

0[2]behaves as the start state, while F is the terminal state. An AUD is emitted whenever the automaton enters any non-final state. The transition probabilities in the automaton are governed by the negative binomial parameters for that event. Based on these, states can skip to the final state, thus accounting for variable lengths of events in terms of number of AUDs. We define $\mathcal{S}$ as the set of all start states for events, so that $\mathcal{S}_i$=start state of event i. Since we model event occurrences as bags of AUDs, AUD emission probabilities are shared by all states for a given event.

Each of the states that correspond to a length between the minimum and the maximum can transition directly to the final state for that event (word), thereby allowing modeling of the variable length property.

Now, in order to decode an AUD (character) sequence in terms of the events (words), the automatons for the events are now put together as shown in Figure 4.4– the black circle represents a dummy start state, and terminal states for each event can transition to this start state. $P_d(w_i)$ represents the probability of the event w_i given the unigram distribution for the document d. Given a sequence of observed tokens, we can use the automaton in Figure 4.4 to compute a forward table and backward table, in exactly the same manner as used in HMMs. At training time, we combine the forward and backward tables to obtain our expected counts, while at test time, we can use the Viterbi algorithm to simply obtain the most likely decode for the observation sequence in terms of the latent events.

Let us refer to the forward table as α. α is parameterized by 2 arguments, a state i and a timestep t, where the corresponding entry in the table provides the probability of being

[2]We do not permit length 0 in our experiments, instead forcing a minimum length n_{lo}.

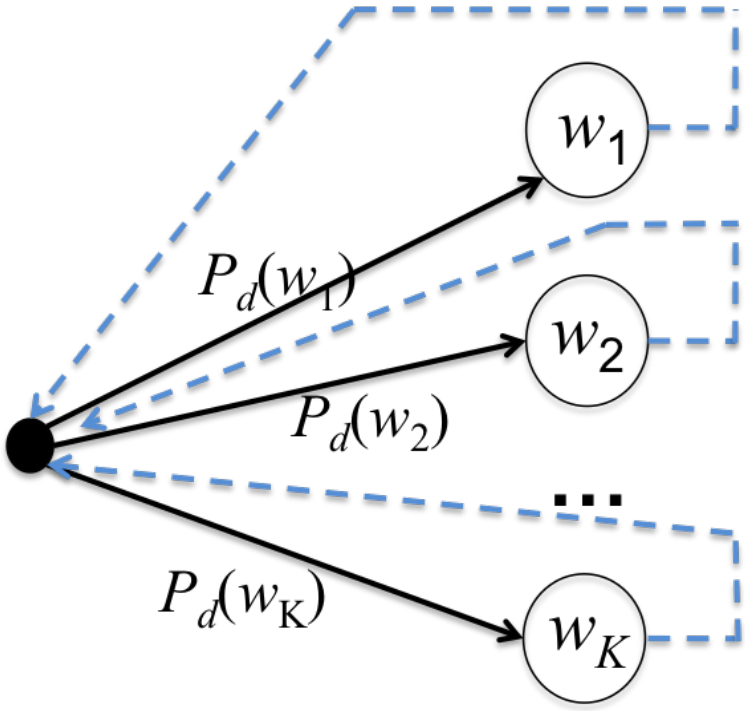


Figure 4.4: An automaton with the K word automatons in parallel for decoding a token stream

in state i at timestep t having seen the observed AUD sequence c_1^t. Thus:

$$\alpha(i,t) = P(state = i, t | c_1^t) \tag{4.29}$$

Let β refer to the backward table, which is similarly parameterized, and $\beta(i,t)$ is the probability of being in state i at timestep t having observed the sequence c_{t+1}^n.

$$\beta(i,t) = P(state = i, t | c_{t+1}^n) \tag{4.30}$$

We can compute the likelihood of being in state i (and extend that to being in word i) at time-step t given the entire observation sequence:

$$P(state = i, t) = \frac{\alpha(i,t) \times \beta(i,t)}{\sum_j \alpha(j,t) \times \beta(j,t)} \tag{4.31}$$

$$P(w_i, t) = \frac{\sum_{k \in w_i} \alpha(k,t) \times \beta(k,t)}{\sum_j \alpha(j,t) \times \beta(j,t)} \tag{4.32}$$

The forward-backward tables are constructed with our current estimates and the sufficient expected counts are obtained using these estimates, which are then used to update the parameters for the next iteration.

4.3 Experimental Results

In this section, we first present a pair of oracle experiments to verify that the joint segmentation and vocabulary learning model performs as expected. Then, we present results using the 2-level hierarchical model on the audio retrieval task.

Oracle Experiment 1: We picked five words {the, cat, ate, blue, man} (the sequence of words in the set is shown in order of their frequency ranks) and used our model to generate 100 documents ($s \sim \mathcal{N}(1.5, 0.2)$) with a 100 tokens each, and use this as the data to learn words unsupervised. After running the learning algorithm over this dataset for 25 iterations, the likelihood appears to not change very much in subsequent iterations. We terminate the training process after 25 iterations and use the learnt parameters to decode the training data.

We analyzed the training process based not only on how well the learnt parameters are able to decode the training data to recover the true word boundaries, but also on how good the learnt parameters were compared to the true parameters. Figure 4.5 compares the true distribution of the character emission probabilities to the learned distributions. We note that the results of learning are sensitive to initialization as is expected with EM-based algorithms, but end up fairly close to the true parameters. Since the true distributions are set with awareness of the characters present in the word, the distributions for each word are sparse. On the other hand, for the learnt distributions, there is a finite, non-zero probability for each character being emitted by each of the words, so the matrix isn't nearly as sparse. However, we do notice that the higher-probability characters in both matrices are very similar. This shows that our learning process does result in nearly the correct solution.

The segmentation results are harder to interpret since the segment boundaries are not always correct, but often missing by one character to either side. It appears that the initialization is significantly to blame for this, in the general case. This is because if the most frequent word is "the", as in our case, but no word is initialized to have relative probabilities that similar to the characters "e", "h" and "t" for the true word "the", then the correct segmentation will not be retrieved very often. However, in terms of parameter estimates, on average, the system seems to perform quite well. The average error per

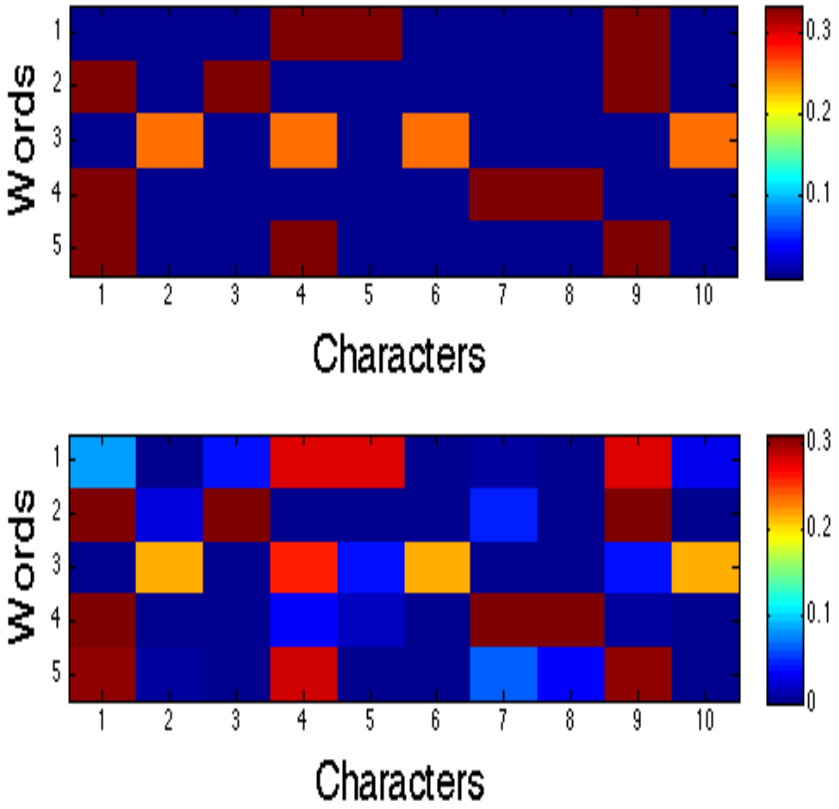


Figure 4.5: Oracle Experiment 1 character emission distribution for the 5 words (Top) True distribution; (Bottom) Learnt distribution

parameter of the character emission matrix is 0.03 averaged over 5 random initializations.

Oracle Experiment 2: We performed a similar experiment with web URLs concatenated together, since web URLs have a clear structure beginning with "http://" or "www" and containing ".com". Again, the learner automatically identified the most frequent word to be one which had highest emission probabilities for {'.', 'c', 'o', 'm'} and the second most frequent word with {'h', 't', 'p', '/', ':', 'w'} characters having high probabilities. The respective segments identified for those words while decoding conforms to our expectations and usually corresponded to ".com" and "http://www".

Audio Retrieval Task:

The entire MED11 and BBC data were decoded using the best performing AUDs model, and the sequences of AUDs were then used to learn models for the higher level semantic units for each of the datasets. Based on the learnt models, we can decode the data in terms of these *events*. Since there are no annotations available at these levels, the discovered events are not assigned any specific semantics, but listening to multiple instances

System	Average AUC in BBC	Average AUC in MED11
PHONE	0.3011	0.2614
FOLEY	0.2872	0.2921
VQ	0.2143	0.2339
AUDs	0.1744	0.2174
EVENT	0.1911	0.2297
DISC–AUD–2	0.1961	0.2512
EVENT–COMB	**0.1729**	**0.1842**
DISC–AUD–COMB	0.1732	0.2214

Table 4.1: Performance of the events layer, individually, and in combination with the AUDs, compared to the baseline systems for audio retrieval

concatenated together shows similar phenomena being captured. One such event consists of sequences of sounds that relate to crowds with loud cheering and a babble of voices in a party being subsumed within the same *event*.

We use the decoded AUDs and event sequences for each file to characterize the data again using the unigram *bag of words* approach, and evaluate the effect of using the the event layer individually (EVENT, respectively) as well as when combined with the AUDs layer (EVENT–COMB). The setting EVENT–COMB simply concatenates the feature vectors from the AUDs layer and the EVENTs layer, and this characterization is used for training and testing of the category specific retrieval systems.

We also compare these results to the results obtained when using the AUDs-like learning process to learn the higher layer units, without the use of a power-law prior, as described in Section 4.2.1. We refer to this setting as DISC–AUD–2, referring to the second layer of discrete AUD sequences. We also combine the units learned in this layer to the original AUD units, and refer to that setting as DISC–AUD–COMB.

Table 4.1 augments the performance table with the performance of the various approaches to learning the events layer and using the events layer for characterization of the data, individually and in combination with the lower-level AUDs on the datasets of interest (lower AUC is better). The numbers in this table simply represent the best number obtained for each of the settings– the best numbers for the EVENT setting was obtained when using 128 events on MED11, and using 32 on BBC. The EVENT–COMB setting improves on the AUDs only or EVENT–only setting on both datasets, but the improvement on the MED11 dataset is statistically significant ($p < 0.05$).

While there is considerable improvement when using the information from the events layer in conjunction with the AUDs layer in both datasets, the effect of the events layer in improving performance over the AUDs layer alone is far more pronounced on the MED11

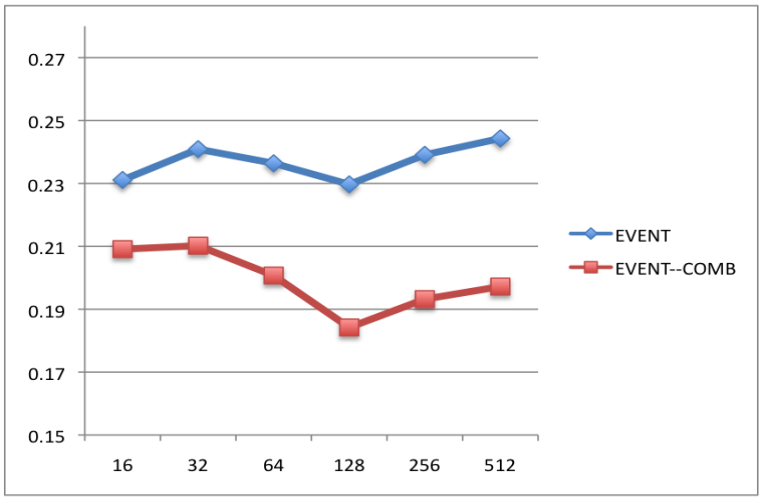


Figure 4.6: Effect of changing the size of the *event* vocabulary over 1024 AUDs on using the events layer only for characterization of recordings as well as in combination with the AUDs layer, on the MED11 dataset. x-axis represents event layer vocabulary size, while y-axis represents the AUC (lower is better)

dataset as opposed to the BBC dataset. Our initial expectation was that, on the BBC dataset, this was due to having selected the wrong vocabulary size and that optimizing the hyperparameter for the size of the vocabulary would lead to a significant improvement at the right vocabulary size. This, however, turned out to not be the case, and it appeared that there was very little change with varying the size of the event vocabulary for the higher layer, although performance degraded really quickly above 128 units. We suspect that this may be due to the fact that the MED11 dataset contains categories that are far more strongly semantic that the categories in the BBC dataset. Indeed, looking at the category-wise performance on the BBC dataset shows that there is almost no change in the numbers for the categories of *animal*, *warfare* and *water*. These categories, while broadly semantic, do not contain enough variation in terms of content of individual recordings that the AUDs, which look for sequential local patterns cannot already characterize well.

On the other hand, performance on the MED11 dataset improves rather significantly when adding this additional layer. Figure 4.6 shows how changing the size of the event vocabulary affects the performance on the audio retrieval task for both the events layer alone used to characterize the file, as well when it is used in conjunction with the AUDs layer. Note that the performance of the AUDs layer alone would be a line parallel to the x-axis with y=0.2174. We make 2 observations:

First, we see that for all settings of event layer vocabulary size, performance when combining the event layer with the AUDs layer improves over the AUDs layer alone. This

is encouraging in that it suggests that there is additional semantic information to be captured over what the AUDs capture, and that the power-law driven events layer can capture it to some degree, even at its weakest. We note that this is not the case for DISC–AUDs–COMB setting which, even at its best, is worse than the AUDs alone. This shows that while there is significant structure available to be captured, looking for canonical local patterns, which is what the left-to-right topology does for the DISC–AUDs settings, is not a good idea, at all.

Second, we note that the event layer alone never outperforms the AUDs layer alone. This is a slight cause for disappointment, since there is reason to believe that the right amount of higher level structure should provide sufficient semantic information to overcome the more specific acoustically-based semantic information available in the AUDs layer. This turns out to not be the case, at least when using 1 layer on top of the AUDs.

We mentioned earlier that this formulation would enable us to continue building the hierarchy iteratively on top of the newly induced units. Our next set of experiments investigated the benefits of adding additional layers, iteratively. In this description, we will refer to the higher layers simply by their height above the AUDs, which form layer 0, since they are the input to the system. The layer that we referred to as the Events layer will now be referred to as Higher–Semantic–Layer–1 (HSL1, henceforth), and the layer above it will be referred to as HSL2, and so on.

We note, first, that using HSL4 or above actively hurts both datasets. For the BBC dataset, the best results were obtained using HSL2 with 32 units over HSL1 with 32 units over 64 AUDs, whereas for the MED11 dataset, the best results were obtained with using HSL3 with 32 units over HSL2 with 32 units over HSL1 with 128 units, over 1024 AUDs. For both datasets, the best performance was obtained when using the characterizations from all of these layers combined. The improvement in performance on MED11 with HSL3 was 0.0015, which may not be worth the rather expensive training.

We observe, also, that on both datasets, the settings in terms of number of units for the hidden higher layers for which optimal retrieval performance is attained appears to favor a rapid narrowing in terms of the number of units in the vocabulary. It is not clear, at this point, what the specific reason for this is, but there are a few potential reasons that might be the cause of this.

First, the higher one goes in the semantic hierarchy, the more high-level the inference becomes. It is possible that the number of concepts at that level drops off quickly as one goes higher. Consider, for instance, the level at which one thinks of movie scenes. At this level, the number of different types of scenes can probably be counted on one's fingers. In

fact, once we move beyond the 2 lowest levels, it is conceivable that the size of the concept space is rather small. This is because the lowest level is that of identifying acoustically consistent sound types, and it is easy to envision there being a large number of different sounds. Similarly the layer just above this could be considered as one that disambiguates the specific object that produced the low-level sound (*e.g.* an ambulance producing a siren sound unit, or a hammer producing a *thwack* sound unit). Once we move above this, however, the space of higher-level concepts could be seen as narrowing much quicker.

Second, as we go higher in the hierarchy, the problem of noisy transcriptions propagates through the levels. The decodes over the noisy AUD transcripts might potentially propagate the transcription errors higher, and since the length of the sequence as one goes higher will be shorter for a given recording (each higher level unit will span several lower level units), the overall training data size decreases quickly. As such, accurately identifying semantic units in the presence of noise with fewer samples available becomes increasingly harder.

Finally, it is possible that the patterns that one is looking for are not always temporally neighboring. Due to computational demands, we forced the maximum length of any Higher–Semantic–Layer to not be greater than 10 units of the lower semantic layer. Thus, at the lower levels, HSL 1 can, at best, span about 3-4 seconds of audio. While this may often be enough for low-level semantics, it may not be able to deal with cases where the semantic (disambiguation) requires longer-range processing. This can be considered analogous to the case of non-projectivity in natural language parsing of text, when parse constituents can have crossing brackets. One of the limitations of our algorithmic framework is that it assumes projectivity in the semantic space, and cannot deal with this kind of potential long-range semantic dependencies.

4.4 Discussion

Although the results above only show gains obtained in objective evaluations on a standard large-scale retrieval tasks, the "events" discovered by the learning algorithm have deeper significance– they represent automatically learned characterizations of longer-scale acoustic phenomena with semantic import. This work presents an initial approach to extracting such deeper semantic features from audio based on local patterns of low-level acoustic units, building up the hierarchy incrementally by inducing one layer at a time. While this approach certainly has benefits (significant reduction in error on a standard task!), it also appear to saturate quickly (very small gains after about 2 higher layers).

While the reasons for retrieval performance saturation are not immediately clear, it is possible that the propagation of errors as each layer is individually learnt is responsible for this. In Chapter 5, we present an approach to learning the entire hierarchy jointly, that can be expected to alleviate concerns about error propagations by maximizing likelihood of the observations given the entire hierarchy. We will then compare the benefits and limitations of the 2 different paradigms in the discussion section of Chapter 5.

The effect of the development of an algorithm for inducing higher level structure immediately raises a few possible directions of future work to consider. First, since the discovered latent events and acoustic units do not have true labels, it would be interesting to explore the process of leveraging the hypothesized segments to generate labels, either via asking annotators employed explicitly for this purpose, or via crowdsourcing.

Second, the current approach described in this chapter to learning higher units requires the optimization of a hyperparameter that governs the size of the vocabulary at each level. While this is the standard approach for parametric systems, it is not quite the natural process that humans use in learning, in general, where the observation of new data results in increasing the size of our conceptual vocabulary to explain the new observations. Such a system is inherently non-parametric, growing its lexicon in a completely data driven manner, and it would be ideal to develop a framework that can perform non-parametric learning.

Chapter 5

A Full Hierarchy Induction Approach

Chapter 4 described our approach to a paradigm for inducing higher level structure over the low-level acoustic unit descriptors, using a layer-wise training approach where individual layers of the hierarchy are incrementally learnt one layer at a time. We start with the AUDs as the lowest layer, and the learn a set of units for the layer above it. In our generative model, the units in this layer generate the AUDs, which in turn, generate the observed audio.

The approach presented, however, makes no assumptions about the kinds of units in the lower or higher layers, except in terms of modeling semantics in general using the assumption that the higher layer represents a higher semantic abstraction from the observed sound data. Thus, we can use that approach iteratively to use the higher semantic layer learnt over the AUDs to now be a discrete sequence input to train an even higher layer, and so on. However, as noted in the discussion of Chapter 4, this might result in one important problem. Since each of the individual layers are learnt independently of the other layers, modeling or decoding errors from the lower levels might propagate as we go higher in the hierarchy, and recovery from these errors might become increasingly difficult. Similarly, the discrete symbol sequence that is input to any higher layer learning algorithm is the best decode obtained from the layer below it. While this is a reasonable assumption, we've noted earlier that in the audio space where there might be a number of different semantic sources producing sound, it is often the case that a decoder will not pick up on the specific source that provides the most *semantic* information about the recording content.

In this chapter, we explore an alternate paradigm for learning the higher layers of our framework, where each of the higher layers is not learnt individually. Instead, the learning process is so designed that the entire hierarchy that we seek to induce over the low-level AUDs are jointly learned. Again, for this work, we assume that the decoding of the low-

level acoustic units from the audio (in our case, we use the AUDs as the low-level units, but the approach is general enough to be applied to any discrete unit sequences) is done as a pre-processing step.

As we shall discuss in the subsequent sections, there are 2 key differences to the learning process used here, as opposed to the one used in Chapter 4. The first is that the entire hierarchy is learnt jointly here, as opposed to the approach in Chapter 4, where the hierarchy was induced layer by layer. The second difference is that some of the assumptions about the hierarchy and the semantic interpretation of the units are modified slightly in this work (we shall discuss this in detail, later in this chapter) and the set of semantic units are shared across layers, whereas in the previous chapter, each layer had its own unique vocabulary.

The rest of this chapter is organized as follows: in Section 5.1, we introduce our approach and the guiding principles behind the modeling paradigm chosen. In Section 5.2, we discuss some prior work to motivate this formulation. We present our learning framework in Section 5.3, and finally our experimental results in Section 5.4. We conclude with a discussion in Section 5.5, where we discuss not just the main lessons from this approach, but also a discussion of the relative strengths and weaknesses of the 2 paradigms for learning hierarchichal structure presented in this chapter and the previous one.

We note that part of the work described in this chapter has been published in Chaudhuri and Raj [2013].

5.1 Hierarchical Structure Induction over Acoustic Units

In this section, we introduce the problem of inducing a full hierarchical structure starting with low-level acoustic units. In our work, we use the acoustic unit descriptors, described in Chapter 3 as the low level units, but one could instead use any other low-level units, as long as the audio can be represented with a sequence of such units. While traditional audio content analysis has relied primarily on shallow analysis of the observed acoustics, based on detection or single-level latent variable models, in this paper, we present a hierarchical paradigm for content analysis that can be used for deeper analysis of the audio content. We note again that this kind of deep structure induction is a novel paradigm for semantic audio tasks, and that in past work, most approaches have worked directly off of the observed acoustic unit layers Chang et al. [2007], Chaudhuri et al. [2011], Pancoast and Akbacak [2011], Slaney [2002], Zhuang et al. [2011]. In that sense, a comparison of performance with the baselines described earlier (Phone, Foley, VQ, AUDs) are not a directly valid

comparison since the information that those models seek to capture is largely acoustic whereas the information the higher level structure attempts to capture is largely semantic. On the other hand, the goal of all of these models is to capture semantic information, and using acoustically derived units for semantic tasks assumes that the observed acoustics map directly to the semantics, whereas our models for deeper analysis claim that the semantic information available at the acoustic levels is limited and that better information can be obtained by extracting more semantically focussed units. Thus, it is fair to compare these models for deeper analysis with the various acoustically derived baselines, since the goal they seek to accomplish is the same.

In modeling the higher levels of a hierarchical structure, we posit that the audio content contains a wealth of semantic information in its structure and sequence that can be used for analysis for various semantic tasks. The analysis of the structure and sequence can be performed independently of any specific audio processing task, and (as discussed in the various examples in earlier chapters) we believe that such an analysis process is similar to the kind of analysis humans perform in understanding audio content. Further, we believe that this kind of task-independent analysis is necessary for the development of a truly intelligent system that is able to continuously stay aware of its context, so that it can make use of this context to respond to any specific situations that may arise.

Recall the kind of analysis that we described in Chapter 1 as being the goal of our framework, as shown in Figure 5.1, using an example from baseball audio. The lowest level of this tree structure corresponds to low-level, generalized acoustic units (AUDs), which may not carry discernible semantic information individually, but the sequences or distribution of the local patterns of these units should capture higher-level semantic information– we refer to these higher-level patterns as *events*[1]. These *event* units themselves might contain certain higher-level patterns, corresponding to still more complex events. In most natural audio, these events themselves do not occur in isolation. They are related to each other in different ways, and event context provides cues for possible future events (event dependencies are indicated by arrows in the figure). Further, the event sequences themselves should carry information about the overall semantic content or class of the audio.

The primary issue in estimating such structures for audio is a scarcity of richly annotated data with information at the various hierarchical levels that could be used to provide supervision for learning the hierarchy in a supervised setting. To address this issue, the various algorithms proposed in this dissertation all deal with unsupervised models for struc-

[1]Note that our reference to *events* refers to higher level units in the hierarchical structure that are referred to as being part of the *events* layer, as distinct from query event categories in the BBC and MED11 datasets.

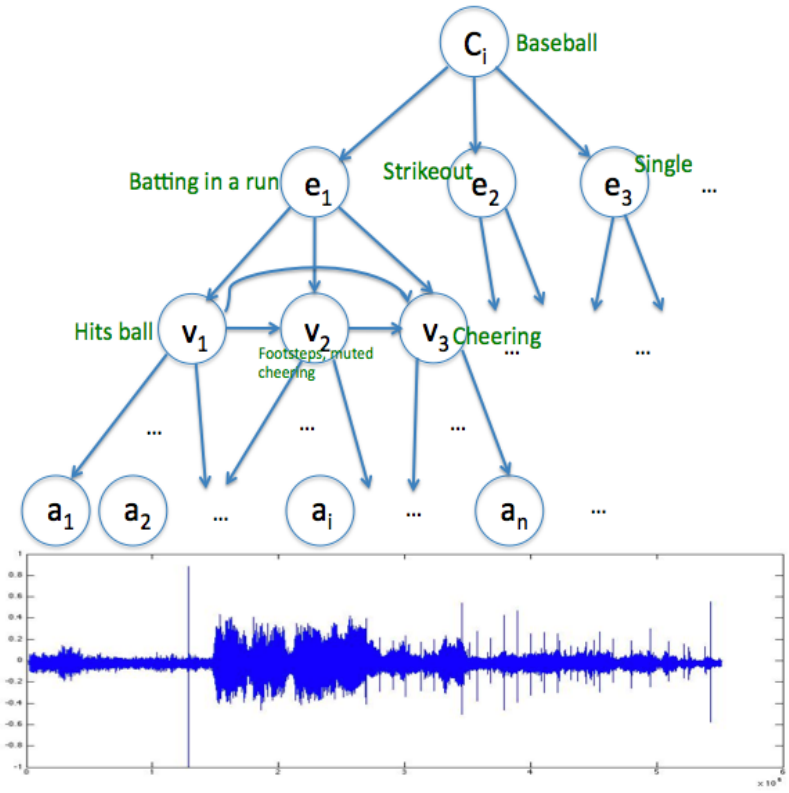


Figure 5.1: An instance of hierarchical analysis for audio.

ture induction that can leverage easily available, but unlabeled, data. The work that we present in this chapter is no different– given simply an audio corpus, we first decode the audio recordings to obtain sequences of AUDs. The algorithmic framework presented in this chapter for a full hierarchy induction takes these AUD sequences as input and estimates corresponding, hierarchical tree structures, unsupervised.

Whether the structure induced by such unsupervised models would be consistently semantically coherent or human-interpretable is unknown, at this point in time, due to the lack of available annotations to use as ground-truth, or the availability of a framework for obtaining human annotations quickly to indicate the semantic coherence of the induced structures. Nonetheless, there are compelling motivations behind such approaches. The process of building richly annotated, hierarchically labeled data sets would normally be an expensive and time-consuming one, but the output of unsupervised approaches can be used both to obtain labels for and verify for coherent semantic units, as well as use them to seed semi-supervised approaches, thus building up labeled resources. Besides, while the presented approach makes no claims toward modeling human approaches to scene understanding, the extracted co-occurrence information and contextual cues provide

a potential basis for comparisons to human reasoning processes in future work.

While an ideal evaluation framework would directly measure the generated structure by comparing it with ground truth structures (generated by annotators), such data is currently unavailable and expensive to obtain. We propose, instead, to use the structured information generated by our models as features for characterization of audio for a semantic audio retrieval task, and we use this proxy measure as an estimate of how well the structure induction process works.

While semantic audio tasks are an obvious application of the deeper analysis obtained from the techniques described in the previous chapter and this one, we believe that the hierarchical structures have considerable significance beyond the specific applications and improvements obtained on standard tasks. These hierarchical analysis structures are important for automated systems to develop a better understanding of the world around it, and the information available in the evolution of acoustics about various aspects of the world, including (but not limited to) understanding the various manifestations of individual objects via acoustics, and what the presence of certain objects in specific contexts might tell us about future events.

An instance of this has been discussed before– consider the various different kinds of actual acoustics that a gunshot might produce. In fact, there are a number of different gunshot sounds in the *warfare* category of the BBC dataset. However, a monitoring system, for instance, does not need to limit itself to the identification of the gunshot, instead actively monitoring the acoustics around it to determine if an alarm should be raised. If the gunshot sound is followed by people screaming, and general sounds of panic, then it should be able to learn that such scenarios are usually followed by the arrival of law-enforcement authorities, as evident from the sounds of sirens. Thus, when it encounters a new instance of a similar situation, it could directly raise an alarm to alert the authorities to such an incident. On the other hand, gunshot sounds on shooting ranges are likely to not be a cause for alarm, and an automatic system should be able to learn that from data as well. Designing intelligent monitoring systems in this manner would be extremely beneficial in avoiding many hours of domain specific adaptation of such systems.

We note that while the hierarchical structure induction task is novel for audio, analogous tasks have been tackled by researchers for text parsing. We discuss some of these approaches in the next section.

5.2 Related Work

Various models can be developed for the individual higher order layers for the framework in Figure 5.1, such as the one described in Chapter 4 Chaudhuri and Raj [2012]. Unlike that approach, which estimates higher levels in the hierarchy one layer at a time, the method outlined in this chapter estimates the entire tree structure jointly following an approach similar to one used in text parsing.

As we discussed briefly in Section 2.2.1 of Chapter 2, the task of hierarchical structure induction for audio bears a number of similarities to the syntactic text parsing task. There are 2 critical differences, however, between these 2 tasks. First, the task of learning a syntactic hierarchy for text lends itself to structured models in a simpler manner than the task of learning a semantic hierarchy for audio, because the low level units for text are ground truth words, whereas low-level units for audio have to be inferred by a decoding process, likely leading to noise in the transcripts. In addition, the same semantic theme might manifest itself using various artistically guided manifestations, whereas linguistic structure can be expected to be somewhat more strongly governed by the laws of language. Second, in the present day, large treebank corpora have been constructed for the syntactic parsing task that are available for supervised training as well as evaluation of the structures learnt (whether the learning was done in a supervised or unsupervised setting). This kind of feedback in terms of a quantifiable measure of error of the learning algorithm is extremely valuable for the training of models, and is not available for semantic audio structure induction tasks. The specific form of such a corresponding treebank for audio is certainly unclear, at present, and (we hope) will be a topic of significant debate in the future. One possible direction would be strongly semantic with labels for individual sound types at the lower levels (e.g. sounds-like-a hammer, sounds-like-an alarm), and higher semantic concepts at the higher levels of the hierarchy (e.g. batting in run, baseball game, horror scene).

There are 2 somewhat disjoint problems in the syntactic parsing domain. (Or, at least, problems that are tackled independently.) The first is the problem of predicting parts-of-speech (POS) for the words in the text, followed by the second problem of modeling of syntactic structure between the various words, including the identification of various phrases, their boundaries and constituents. This is somewhat analogous to our case, as well, where the observed audio is first decoded as a sequence of AUDs (analogous to the parts-of-speech) and this is followed by using the AUDs to infer a higher level structure.

Much like the approach we follow in inducing higher level structure, similar early approaches for syntactic parsing treated the POS induction problem as solved. Various

different distributional clustering approaches had been used for part-of-speech induction Finch and Chater [1994], Schutze [1995] and had resulted in high-quality clusters, though the clusters did not always accurately resemble what one would expect using their world knowledge of language and linguistic structure. The decision to treat the POS tagging problem as solved has the benefit of reducing sparsity issues over the English vocabulary. On the other hand, the actual word tokens are replaced by their POS, and experimental results have shown that structure induction even in supervised settings has often performed poorly.

Grammar induction approaches use the POS sequences as input and attempt to uncover tree structures unsupervised. Such early unsupervised approaches typically used the assumption that the tree was generated by a context-free grammar, and can typically be grouped into 2 main paradigms. The first typically uses notions of prior linguistic knowledge to attempt to fix the structure of the grammar in advance, Carroll and Charniak [1992], Lari and Young [1990]. They then attempt to find the set of parameters such that the likelihood of observing the sentences given those parameters is maximized. The second, and generally more successful paradigm incorporate a structural search to hypothesize grammar modification favoring shorter analyses and penalizing length of the analysis result. The primary weakness of the latter approach was that they tend to grow the structures bottom up (similar to the approach that we adopted in Chapter 4), and how well the system performed overall depended heavily upon how good a job it did at predicting local structures in the intermediate steps.

In subsequent work, improved models were developed for generating tree structures, that were based on the hypothesis that significant chunks of the induced hierarchy corresponded to coherent categories (such as noun or verb phrases) that occurred in distinctive environments (*e.g.* a noun phrase occurs typically between the beginning of a sentence and a verb phrase). The inside-outside algorithm Baker [1979] has been extensively used to model this intuition for EM estimation of probabilistic context-free grammars and is used in estimation of discriminative models for context-free parsing. In the inside-outside algorithm, the product of inside and outside probabilities is the probability of generating the sentence with a j constituent spanning words p through q: the outside probability captures the environment, and the inside probability the coherent category. Thus, if the system has a good idea of what a specific coherent category looks like, and the contexts it appears in, then if a new manifestation of that category were to occur in a typical context, the context should guide the learning system to identify this novel manifestation as an instance of the same category.

However, in the learning process, especially in the earlier iterations, there is only limited clarity of the various contexts in which different structural categories occur. Thus, the context does not force the system enough to correctly identify instances of such local structure. Since the EM is prone to local optima, if the early iterations lead the learning algorithm in the wrong direction, it is unlikely to recover from it.

To deal with this kind of setting, the constituent-context model (CCM) was introduced for unsupervised syntactic parsing of text Klein and Manning [2001, 2002]. We experiment with using this model for audio analysis with some modifications to deal with audio modality. We will now explain the CCM and its application to the task of semantic audio structure induction in the next section.

5.3 Proposed Model and Learning Framework

We described the similarities of various aspects of the task of semantic structure induction for audio to the task of syntactic structure induction for text. In this section, we describe the adaptation of one of the successful unsupervised syntactic structure induction algorithms (the constituent context model, Klein and Manning [2001] to our task.

This algorithm proposes to overcome the problem introduced by weak supervision in the early iterations of the inside-outside algorithm by introducing slight modifications to the way the environmental context guided the identification of different instances of a particular category. Before we describe the specific model, however, we remind the reader of the discussion in Section 4.1.1 of the significant differences between the audio and text domains, introduced by noisy decodes and a lack of canonical structure. The first is still a problem in this setting, whereas we note that syntactically coherent categories of text do not have clear canonical structure either. However, we do believe that the degree of inherent variation in the semantic task is significantly greater than for the syntactic text analogue. As a result, our adaptation of the CCM should be able to deal with these problems.

We begin by first estimating the lowest level acoustic units (indexed by a in Figure 5.1), to convert the continuous audio sequence to a discrete representation. The process of inducing the higher-level structure works on top of this representation, and we show in our experiments in Section 5.4 that features derived from this structure prove effective on the semantic audio retrieval task.

Given the low-level acoustic units for an audio recording, the task of inducing a tree structure can be divided into two tasks– first, we need to decide constituent identity,

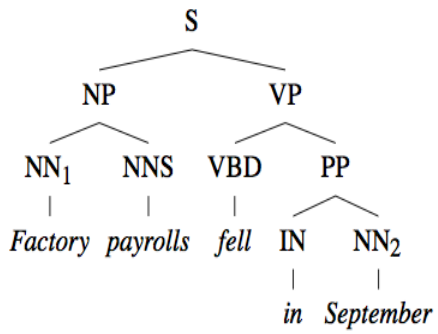
i.e. which and how many of the consecutive low-level units should belong to the same higher-level constituent unit; and second, we need to decide the label for the constituent. This second step is distinct from the challenges that are part of the syntactic text parsing framework. Since syntactic parses are evaluated on large treebank corpora with sentence structures with various chunk boundaries (start and end locations are marked in the corpora for all the various sub-sentence level units of interest), evaluation is done by calculating the number of units that correspond to the correct start and end markers, as well as error metrics defined over the degree of error in identifying boundaries.

While the two tasks are correlated, the task of labeling the constituents is the easier of the two, since the distribution of the lower-level acoustic units within the constituents can be used to cluster the various constituents. The task of deciding constituent identity is significantly harder in our case, because the process of estimation the lower-level acoustic units is typically noisy; unlike text, where the observed surface forms typically correspond to ground truth, the observed audio can contain noise both in terms of innate variations in semantic content, as well as additive background noise that can cause errors in estimation of the lower-level units.

We do not explicitly deal with the problem of noisy decodes in the first stage of our model which works on estimating constituent boundaries. We assume, instead, that a sufficient number of constituent cues are present to allow the identification of constituents over a large enough window, where the window size behaves as an extensible buffer within which enough of the characteristic AUD units should occur to enable identification of appropriate boundaries of the constituent. Thus, the process of identification of boundaries is expected to result in correct, but noisy, constituents, for which we then need to obtain labels so that we can characterize the recording in terms of the units in the structure for any task of interest. (Of course, this would not be necessary if we could directly evaluate the structure using boundaries provided in the ground truth annotations within a corpus.) This second step of obtaining labels corresponds to a clustering task that we shall shortly describe, where we present an approach to deal with the presence of noise in the constituent units to accurately identify constituent sequences that represent the same semantic unit.

As is typically assumed in text parsing applications, our proposed model also relies on two key assumptions:

1. The assumption of projectivity, *i.e.* that constituents of a parse do not cross each other
2. The assumption of strength of contextual cues, *i.e.* that constituents occur in constituent contexts.



Node	Constituent	Context
S	NN NNS VBD IN NN	◇ – ◇
NP	NN NNS	◇ – VBD
VP	VBD IN NN	NNS – ◇
PP	IN NN	VBD – ◇
NN_1	NN	◇ – NNS
NNS	NNS	NN – VBD
VBD	VBD	NNS – IN
IN	IN	VBD – NN
NN_2	NNS	IN – ◇

Empty	Context
ϵ_0	◇ – NN
ϵ_1	NN – NNS
ϵ_2	NNS – VBD
ϵ_3	VBD – IN
ϵ_4	IN – NN
ϵ_5	NN – ◇

Figure 5.2: An illustration of constituents and contexts from Klein and Manning [2001]

In the following subsections, we present our approaches to modeling the 2 tasks– first, we present a log-linear model for identifying constituent boundaries within a tree structure from the data in Section 5.3.1, followed by a description of how label identities are associated to these constituents in Section 5.3.2.

5.3.1 Tree induction using the CCM

Let $\mathcal{A}$ be a sequence of the estimated lower-level acoustic units (AUDs), such that for any given recording, $\mathcal{A} = a_1 a_2 ... a_{n_i}$. Every subsequence of $\mathcal{A}_i$ occurs in some linear context c, $c(\mathcal{A}_j^k) = a_{j-1}\mathcal{A}_j^k a_{k+1}$, where the context elements correspond to the adjacent acoustic units for the subsequence and the AUD units corresponding to $\mathcal{A}_j^k$ are the constituents. To understand what is specifically meant by constituents and contexts, we refer to Figure 5.2 from Klein and Manning [2001], which shows the set of various constituents and contexts for a English sentence.

Unlike the way constituents are modeled in Klein and Manning [2001], we model the constituents as a bag of units, using only presence information. This is because the hard sequence based constituent characterization requires one specific sequence in terms of the AUDs. This is problematic in the semantic audio space for 2 reasons– first, the size of the vocabulary of the POS tags is fairly small (about 40) whereas the size of the vocabulary of AUDs is quite large (1024 for MED11). Thus, having a constituent of size 3 would already imply a constituent feature set size of 1 billion units, which would surely lead to extremely high variance. Second, the noise in terms of decodes would introduce random AUDs within the sequence that would add further variance to the process of estimating weights. To avoid these issues, we model the constituent as a bag instead of a canonical sequence. Note that

there is still noise present in the form of AUDs introduced in the decode that should not be part of the sequence; we do not deal with this at this stage, instead leaving that to the second stage where clusters are induced over the set of constituents.

Thus, we can view any tree t over a sequence $\mathcal{A}$ s as a collection of sequences and contexts. Good trees will include nodes whose yields [2] frequently occur as constituents and these constituents are frequently surrounded by expected contexts. To formally model this, we use a log-linear model with the form for the conditional distribution being as shown below:

$$P(t|\mathcal{A},\Theta) = \frac{\exp(\sum_{\{\mathcal{A}_j^k,c\}\in t} \lambda_{\mathcal{A}_j^k} f_{\mathcal{A}_j^k} + \lambda_c f_c)}{\sum_{t:yield(t):\mathcal{A}} \exp(\sum_{\{\mathcal{A}_j^k,c\}\in t} \lambda_{\mathcal{A}_j^k} f_{\mathcal{A}_j^k} + \lambda_c f_c)} \tag{5.1}$$

Thus, for each tree, we have one feature $f_{\mathcal{A}_j^k}$ for each constituent subsequence $\mathcal{A}_j^k$ in the tree, and its value is the number of nodes in t with yield $\mathcal{A}_j^k$, and one feature f_c for each context c representing the number of times c is the context of the yield of some node in the tree. Joint features over the context and the yield are not used, and no distinction is made between the constituent types at this point.

We model the conditional likelihood of a tree t as $P(t|\mathcal{A},\Theta)$, where $\Theta = \{\lambda_{\mathcal{A}_j^k}, \lambda_c\}$, $\forall\{j,k\}$ that form constituent subsequences. We use an iterative EM-like procedure to find the best parameter estimate given the observed acoustic unit sequences for the given data. The parameter set Θ is initialized and each audio recording is initialized with a random tree structure over the observed acoustic unit sequence for each recording. In alternating steps, then, we find the the best parameter update Θ^* and the best guess for the 1-best tree structure for each of the acoustic unit sequences, given the updated parameters, using a dynamic program. For any Θ, this produces the set of tree structures T^* that maximizes $P(T|\{\mathcal{A}\},\Theta)$. Thus, $P(T^*|\{\mathcal{A}\},\Theta) \geq P(T'|\{\mathcal{A}\},\Theta)$ (where, T' refers to the prior estimate of the set of trees for the set of audio recordings $\{\mathcal{A}\}$). The iterative process then fixes these estimated tree structures to update the parameters. Given the choice of exponential family in Equation 5.1, we do not have a closed-form update rule for the parameters, and will need to adopt a numerical solution for updation, such as conjugate gradient.

Due to the varied linear contexts that can occur in the lower-level acoustic unit sequences, smoothing plays an important role in determining the quality of the induced tree structures. The current system can model arbitrarily long yields, which occur infrequently. The corresponding parameters for these yields may not significantly change from their

[2]The yield of any non-terminal node in a tree structure refers to the sequence of terminals produced by the subtree rooted at the non-terminal node.

initial choices, in spite of multiple learning iterations. Ideally, we would like the weights for unlikely occurrences to have very little influence, by making them as close to zero as possible, thus skewing the distribution of values in $\lambda_{\mathcal{A}_j^k}$ towards low values.

In the conjugate gradient setting, parameter estimates are slow to converge and difficult to smooth with desired priors. Thus, we adopted a different approach that proved to work quite well using the simple smoothed relative frequency estimates, where $\lambda_k = \frac{count(f_k)}{count(k)+M}$. This estimation process ensures that the parameter values lie between 0 and 1, providing a bias toward non-constituency for long subsequences using high values for M.

Once the underlying *most-likely* tree structures have been computed for all the audio recordings given their representation as sequences of low-level acoustic units, we then move to the second stage of the process– that of labeling the induced constituents using a clustering technique. Recall from earlier that we mentioned that the induced constituents were likely to contain considerable noise due to the presence of noise from the decodes. We assumed, nonetheless, that in spite of the noise, the constituent boundaries would be appropriately detected due to our use of bags to represent the constituents. It is in the next stage that we attempt to deal with any noise that may have been introduced at the time of decoding or while identifying constituent boundaries.

5.3.2 Identifying Constituent Labels

This section deals with our approach for identifying constituent labels once the process of inducing a tree structure over the AUD sequences is completed. In syntactic text parsing tasks, this stage is not necessary since the current corpora for that task already contain sentences marked with the boundaries for the various structural elements. For the semantic audio task, however, no such boundaries exist and we need to evaluate the *goodness* of the induced structures on an external task. In order to characterize the recordings for any external task, we need to be able to characterize each of the induced segments structures. We do so by assigning a label to each segment, where 2 segments that have the same label can be thought of as belonging to the same (higher-level) semantic concept.

Besides its being an essential component in order to be able to create characterizations of audio recordings, the task of identifying constituent labels is important on its own for one specific reason. As discussed earlier, if we assume that the constituent identification process does a perfect job at identifying constituent boundaries, we note that the constituent sequences of AUDs are, nonetheless, likely to contain noise due to errors in modeling, decoding, or simply the presence of background noise. Thus, intuitively, this stage has to be robust to the noise while clustering the large number of segments into the appropriate

clusters such that their true semantic identities are retained.

In addition to the tree structures from the previous stage being input to this system, the only other external input at this stage is the hyperparameter K for the number of clusters. Unlike our layer-wise training approach in Chapter 4, where each layer had its own unique set of unit labels, in this work, the set of cluster labels are not dependent on the layers at all. Thus, the same unit might appear at different levels of the hierarchy, and its label is only governed by the semantics it contains as can be interpreted by analyzing its *yield*, as opposed to an artificial semantic controlled by the layer.

From the tree structures hypothesized by the log-linear model, we obtain a set of segments corresponding to every node in the tree. Each of this nodes is the root of a subtree, which in turn, spans a sequence of low-level acoustic units, which are referred to as the yield of the node. While trying to cluster the entire set of segment nodes in the training set, the yields are used to compute similarity, as described below.

The clustering procedure on the set of nodes (each of these nodes spans a constituent) obtained from the set of tree structures, each corresponding to one recording, uses a modified K-means procedure, where we first select a set of K cluster centers. The distance of each induced constituent ($\mathcal{A}_j^k$) to each of these cluster centers ($\mathcal{C}_i$) were computed using a combination of 2 factors– temporal sequence of the constituent acoustic units, as well as distribution of the units in the constituent.

The intuition behind using the temporal sequence is apparent, since we would expect similar higher order units to contain similarities in their constituent sequences. However, since the lower level acoustic units are not ground truth, but in fact estimates from noisy decodes, different manifestations of the same higher-order unit might be quite different due to insertions, deletions and substitutions in the true sequence. To account for this, we consider the bag of words distribution of the acoustic units as well, where we compute the cosine distance between the distributions of the constituent and cluster centers.

We use the Levenshtein edit-distance ($\mathcal{L}(\mathcal{A}_j^k, \mathcal{C}_i)$) to compute temporal sequence similarity for each constituent to each cluster center using the actual acoustic unit sequences that occurs in the constituent and the one for the cluster center to model temporal similarity. For each constituent subsequence, the distances to various centers are normalized by the maximum distance to lie between 0 and 1. To compute the distance between the distributions, we compute the cosine divergence between the 2 distributions ($Cos(\mathcal{A}_j^k, \mathcal{C}_i)$. The final computed distance is a product of the 2 individual distances as follows:

$$\mathcal{D}(\mathcal{A}_j^k, \mathcal{C}_i) = \mathcal{L}(\mathcal{A}_j^k, \mathcal{C}_i) \times (1 - Cos(\mathcal{A}_j^k, \mathcal{C}_i)) \tag{5.2}$$

The constituent $\mathcal{A}_j^k$ is assigned to the cluster center $\mathcal{C}^*$ chosen as:

$$\mathcal{C}^* = \arg\min_{\mathcal{C}_i} \mathcal{D}(\mathcal{A}_j^k, \mathcal{C}_i) \tag{5.3}$$

While the semantic import of the tree structures and the induced constituent labels cannot be understood directly from this process in the absence of extensive studies using humans in the loop to understand if the constituents consistently capture human-interpretable semantics, we hypothesize that these will provide positive improvements on semantically defined tasks, if they do indeed capture some underlying higher-level semantics. We present results with using characterizations derived from the tree structures on an audio retrieval task in Section 5.4.

5.4 Experimental Results

Since we do not have labeled data that can be used to directly analyze the accuracy of the estimation process, we evaluate our framework on the audio retrieval task, described in Section 2.4. Recall from Section 2.4.2 the pipeline used for audio retrieval. As with all the previous systems described in this dissertation, the analysis technique described in this chapter also focuses on content analysis, and the analysis framework is plugged into the pipeline from Figure 2.4 to work as the content analysis technique. Note that the AUDs framework is also part of the content analysis technique box, since the hierarchy induction described in this chapter works on top of the AUDs decodes.

Once a 2-stage tree structure induction process has been completed, including the clustering and labeling of the segment spans, we can now compute a bag of words feature vector in the same manner as before, with one difference. In all the previous cases, the units that were used to compute a bag of words feature vector belonged to the same layer in the hierarchy. In this case, the bag of words feature vector is computed using the set of units from the full hierarchical structure derived jointly instead of layer-wise.

As before, we report results using the average Area Under MD-v/s-FA Curves (AUC) for all the categories for both datasets. Since the curve measures error of the system being evaluated, the lower the area under the curve, the better the performance.

Our hypothesis in testing the induced tree structures on an audio retrieval task was that the induced structures over the lower-level units should improve over the performance of retrieval systems based on the lower-level units alone. We note that there are no restrictions on what kind of units can be used at the lower level to represent the audio. Thus, we can use any lower-level unit representation that can represent the audio as a sequence of discrete

System	Average AUC in BBC	Average AUC in MED11
PHONE	0.3011	0.2614
FOLEY	0.2872	0.2921
VQ	0.2143	0.2339
VQ–Tree	0.2461	0.2572
VQ–Tree–COMB	0.1943	0.2382
AUDs	0.1744	0.2174
AUD–Tree	0.1810	0.2311
AUD–Tree–COMB	**0.1693**	0.2226
EVENT	0.1911	0.2297
EVENT–COMB	0.1729	**0.1842**

Table 5.1: Performance of the full tree structured hierarchy including results of inducing the hierarchy on top of both AUD and VQ units as the lower level of representation, compared to the baseline systems for audio retrieval

symbols.

In the experiments reported in this chapter, we actually experiment with using both the AUDs and the VQ units as the lower level units and tree structures are induced on top of each of those representations. The results are summarized in Table 5.1.

In this table, the systems with results obtained when using the higher level structure only induced over the VQ units and the AUD units are referred to as **VQ-Trees** and **AUD-Trees** respectively. Finally, we can create an additional pair of systems that combines the lower-level units with the structure induction process, by concatenating the pair of feature vectors (we refer to these as **VQ–Tree–Comb** and **AUD–Tree–Comb** respectively). We also retain the rows corresponding to the EVENT and EVENT–COMB settings from Chapter 4, where the EVENT setting represents the system with one higher layer induced over the AUD sequences, and the COMB setting refers to using it in conjunction with the low level AUDs.

We note that the best results on the BBC data was obtained with 64 cluster centers used to cluster the constituent blocks of audio units. The best results on the MED11 dataset were obtained with 256 cluster centers. We see from the table of results that the induced trees over the lower-level acoustic units do not outperform the low level acoustic units alone on any of the datasets. On the BBC dataset, the combination of the units from the induced structure along with the low-level units outperforms the low-level units alone when using both VQ and AUDs as the low-level units (although the improvement is not statistically significant). However, on the MED11 dataset, for both the VQ and the AUD setting, the induced structure (alone, or in combination with the low-level units) does not

outperform the low-level units alone, although the performance of the combination of the low-level and higher level units are very similar to the low level units alone. (We note that the EVENT–COMB setting is still statistically significantly better than the other settings, with a p-value of 0.04, or $p < 0.05$.)

While we do provide tree-only results for our experiments, we note that retrieval based on features from the induced tree alone is not expected to be better for 2 reasons. First, tree structures induced in an unsupervised manner will contain considerable noise. Secondly, and possibly more importantly, even if ground truth labels were available, the higher a node is in the tree hierarchy, the longer is its yield, and information available for such nodes in the training set decreases. The higher level clusters are likely to be broader since we represent the wide range of possibilities by a mapping onto a limited, finite set of clusters, thus collapsing a varied set of concepts together and reducing the discriminative properties of the true higher-level concept. Nonetheless, we do expect that they would provide *additional* information, and the results are consistent with this expectation. While the systems using the induced tree structure information only do not outperform the systems using lower level units only, the combination of the two systems outperforms both the individual systems significantly.

We expect that the primary reason that the tree structure based systems have limited success is due to the fact that they work with the information provided by the lower level units and any error in the estimation of those units is propagated, resulting in the semantics captured being weaker than in a model that can jointly utilize both the observed audio and the estimated units jointly to induce structure. Developing such models remains a focus of our future work.

We note that estimating the parameters of the constituent-context model with the VQ units as the lower layer results in a significantly slower training process, since there is one VQ unit per frame of audio, whereas the AUDs typically span about 30 frames on average. Further, while for syntactic parsing of short sentences (rarely longer than 20 words), it is feasible to have as many layers as required to create a full hierarchy till a root node that spans the entire sentence is reached. On the other hand, a 30 second segment of audio will contain about a 100 AUD units in the sequence, and therefore, the height of a tree that has a number of binary nodes can be quite high. To avoid this, our training process is guided by a rule that requires units at the 2 lowest levels to span at least 8 units of the level below them.

The fact that the combined systems outperform the individual unit-based or tree-based systems (with consistent trends for both baseline systems) is promising, and shows that

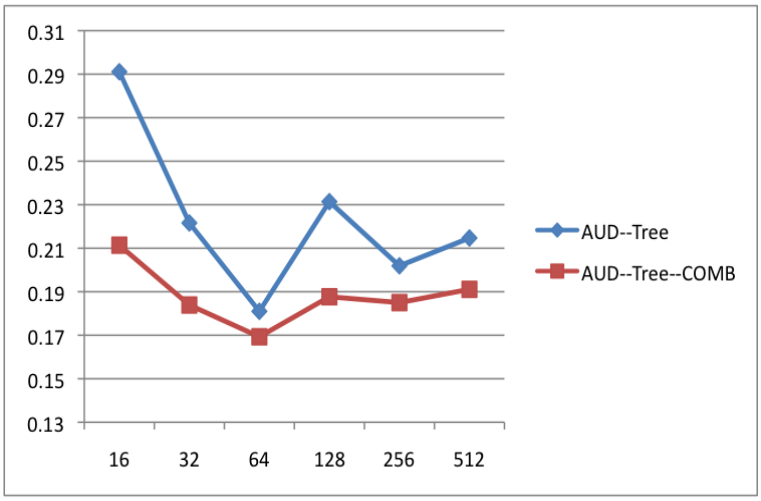


Figure 5.3: Effect of changing the size of the tree unit vocabulary over 64 AUDs on using the events layer only for characterization of recordings as well as in combination with the AUDs layer, on the BBC dataset. x-axis represents tree unit vocabulary size, while y-axis represents the AUC (lower is better)

the induced structure does capture additional semantics over the low-level acoustic units.

Figure 5.3 shows the performance on the BBC dataset (which was the dataset on which the tree structure produced greater improvements over the acoustic units) when varying the number of cluster units used for clustering the constituents. We observe that the performance of the COMB setting outperforms the induced structure across the board. On the other hand, the AUDs– only setting on the BBC dataset results in an AUC of 0.174, which is better than the COMB setting for all the various higher level unit vocabulary sizes except when using 64 units.

The performance clearly peaks at 64 units used to represent higher level units in the tree structure on the BBC dataset, which is the one where a significant performance improvement is obtained over using the low-level units alone. One possible reason that the settings with fewer numbers of units does not perform well is likely that when using 16 or 32 units, these units are quite generalized and the entire vocabulary is not large enough to be able to capture the unique patterns on a consistent basis. On the other hand, having a significantly higher number of units may not work because the system tries to be too specific in capturing local patterns. In addition, due to the noise present, some of the units that map to any individual cluster may be mapped purely by chance due to the presence of certain local noisy segments. It is hard to say what vocabulary size would be too small or too large in general. Our empirical observation is that using 64 units appears to work the best on the BBC dataset.

On the MED11 dataset, on the other hand, performance of the COMB setting is worse than the AUDs only setting across the board, although only slightly worse. We observe, empirically, that the best performance is obtained when using a vocabulary of 256 units for the higher level units.

5.5 Discussion

In this chapter, we presented a novel unsupervised approach to inducing tree structures for modeling higher-level semantic information that can be applied for different tasks. We presented a unified framework that hypothesizes that the observed acoustics map hierarchically to higher-level semantics, and that estimation of these semantics directly from the audio in a task-agnostic manner could be used to derive characterizations that could be appropriately utilized for the specific task at hand.

We leveraged previous work in unsupervised text parsing as well as acoustic unit estimation to generate hierarchical structures for audio in an unsupervised setting. Presently, the semantic import of the derived structures is unclear, since we do not have labeled data for analysis of the estimated constituent boundaries or parse structures.

The approach presented here is in direct contrast to the one employed in Chapter 4, where the higher layers of the hierarchy were learnt one layer at a time, whereas in this work, the entire hierarchical structure was jointly induced. The main points of comparison between the tree induction approach presented in this chapter and the layer-wise training approach from Chapter 4 are discussed below.

The total time required for training is typically faster for the layer-wise approach when using only 1 layer only and 64 or fewer units, but is pretty similar to the tree induction approach when using 3-4 layers, with parameters learned iteratively. However, as the number of units in the layers increases, the size of the parameter set being estimated increases more rapidly for the layer-wise approach than in the tree induction approach.

On MED11, the layer-wise training approach outperforms the tree induction approach quite comfortably, whereas on the BBC data, the tree induction approach is the best performing system (using AUDs as the low level unit). Surprisingly, on MED11, the tree induction approach does not outperform even the acoustic unit only characterization on the audio retrieval task.

The number of layers that can be used is flexible in case of the layer-wise training approach, and can be tuned to the specific task at hand, whereas the number of layers is fixed for the tree induction approach, and is dependent on the length of the audio; a 5 min

audio clip will typically have 4 or 5 layers when using the tree induction approach.

The variation in performance using different numbers of layers with the layer-wise structure induction shows that improvements in performance fall off very rapidly with more than 2 layers. There is, naturally, no direct control over the number of layers when using the tree induction, although changing the parameter that governs how many lower-level units can be spanned by a higher-level unit serves as a proxy switch. Changing this value from a minimum of 4 to a minimum of 12 does not seem to affect overall performance much at all. This may be because the same semantics that are captured by the system with a minimum of 12 units are captured by the system with a minimum of 4 units, except at a higher level, whereas the additional units at the lower level are too noisy to provide significantly meaningful semantic guidance. However, using a fewer number of minimum units significantly increases the training time of the framework.

Neither of the 2 approaches, in their present formulations, are capable of dealing with non-projective semantic dependencies, and assume projectivity.

The evaluation of the semantic units thus obtained is tricky for both cases, since no easy-to-use framework currently exists for their analysis. However, if such a framework were to exist, one can imagine that evaluating the consistency in the semantic units for the layer-wise training might be easier since these units are not shared across layers, and the length of individual manifestations of the different semantic units do not vary by much. On the other hand, since the vocabulary of units is shared across all the levels in the tree induction paradigm, different instances of semantics that map to the same cluster may be vastly different in length (often, double or triple the length). As a result, comparing the 2 instances to decide if they capture similar semantics requires a greater overhead on the part of the annotator.

The development of a framework that models the semantics as a concept space that cannot be completely captured by the low-level acoustically derived units alone is one of the primary contributions of this dissertation. In this chapter, and the previous Chapter 4, we described the 2 main paradigms for modeling higher structure over the low-level acoustic units, and showed empirically using a semantic audio task that using this framework can help in obtaining improvements on a semantic task, thereby validating the notion that there exist semantic information in higher level patterns over the acoustic units.

In the next chapter, we will present a few directions of future work that we feel are particularly important for research in this field. We will also present some preliminary approaches that we have explored in these directions.

Chapter 6

Preliminary Approaches Toward Important Future Directions

In this dissertation so far, we presented our novel, hierarchical framework for mapping the observed acoustics to semantics with the higher levels of the hierarchy corresponding to a higher level of semantic inference. We presented an approach to learn low-level acoustic units (which we refer to as Acoustic Unit Descriptors (AUDs)) unsupervised from the audio corpora, and then described two main paradigms for the discovery of higher level structure– one following a layer-wise training approach to build the higher layers iteratively, while the other attempts to induce a full tree structure directly. Each of these paradigms works on top of the Acoustic Unit Descriptors layer, instead of working directly on the observed audio.

Ideally, we would have liked to evaluate the structures obtained at each of these levels directly by comparing it to ground truth segments, or using some alternate framework for validation. However, current audio datasets are not annotated with the kind of rich, hierarchical structure that we would need to be able to evaluate the induced structures directly. As a result, we evaluated the semantics of the audio segments discovered on a proxy semantic audio retrieval task.

This dissertation represents (to the best of our knowledge) the first efforts in exploring the presence of semantic information in general audio via deep analysis of the observed acoustics, using structured models that work in unsupervised settings. While we've attempted to understand as much as possible about what our models are learning, and the import of the units at the various layers, this understanding is limited largely due to the scarcity of rich, annotated datasets.

Besides the need for development of such datasets, however, the models employed at the

various levels of the hierarchy made some assumptions that were necessary for preserving model simplicity and computational tractability. In this chapter, we will discuss some of the assumptions made by our models, and some of the limitations of the current modeling approaches. We've also worked on some approaches that address such problems, and we present our formulations for the same in this chapter.

In Section 6.1, we discuss one of the limitations of current approaches to learning the low-level acoustic units vocabulary, and present an alternate paradigm for such acoustic unit learning. In Section 6.2, we discuss the issue of modeling semantic information in audio recordings where the semantic content itself is noisy and multiple semantic concepts may be legitimately present. We explore the use of the multiple instance learning paradigm in this context. In prior chapters, we've mentioned that one of the limitations of the current paradigm for learning of units at the various levels is that the number of units is a hyperparameter that needs to be optimized for the various tasks, which is computationally expensive. We would rather design these models so that the learning process of new units is driven by the data itself, where the model has the ability to add additional units to explain data that is significantly different from the characteristics of the data captured by the currently existing vocabulary. To this end, we discuss some recent work in non-parametric learning, and explore its application to our task in Section 6.3.

We note here that the preliminary approaches to these different tasks explored in our work in this chapter have met with limited success thus far. However, we believe that these are directions that ought to be explored by the community, in general, and we present an initial exploration of these ideas and the models tested on the audio retrieval task correspond to the establishment of baseline performances when using these approaches.

Finally, we've mentioned earlier that the approaches to the learning of units at the various levels are applied to the audio processing task, but the principle behind the models are general, such that we can envision applying them to any semantically motivated analysis modality. In Section 6.4, we attempt to learn units analogous to the AUDs to characterize video content.

6.1 Block-Sparse Approach to Learning Atomic Low-level Units

In the past chapters (specifically, Chapter 2 and Chapter 3, we discussed a number of different approaches to learning low-level acoustic units, including the framework for learning the AUDs used in this dissertation. A number of past approaches to analyzing audio in

the wild were built around detecting specific sounds in audio streams such as gunshots, laughter, music, crowd sounds etc. Chang et al. [2007], or mapping words to acoustic phenomenon Slaney [2002] using known vocabularies of sounds. In unconstrained audio, the set of such sounds is large, and supervised data is required to build such detectors.

Instead of using audio libraries to build detectors, current approaches attempt to *learn* a lexicon of sounds from the audio data Chaudhuri et al. [2011], Zhuang et al. [2011] in an unsupervised manner. A common paradigm employed in using these lexicons for characterizing audio data is to decode the recordings using models for the various units in the lexicon, and represent the recording as a sequence of these lexical units. This process typically assumes *source uniqueness* at any given point in time; *i.e.* it assumes either that only one lexical unit may be active at any instant (hard quantization) or avoids making a decision by distributing the uncertainty in its estimate among all the different units (soft quantization). Hard quantization will select the dominant source alone, while soft quantization will estimate the likelihood of presence of each of the lexical units or sound sources independently instead of jointly.

Typically, multiple different sources may produce sound concurrently, which combine to produce the observed audio. Thus, learning the lexicon from the data would result in the units modeling mixtures of sources, instead of the atomic sources themselves. When the number of unique sources present is small, such a method might be reasonably feasible in adequately capturing the different likely acoustics. However, the number of possible mixture will grow exponentially with the number of true atomic sources, whereas learning and estimation techniques can only use limited vocabularies for computational efficiency. Thus, the very large mixture space will be mapped down to the finite vocabulary of a much smaller size, collapsing mixtures from different sources together.

In this section, we describe an approach to extend the current assumption of source uniqueness to its logical next step, and present a framework for explicit estimation of mixtures of lexical units in such settings. The units learnt in the approach described here would be analogous in terms of functionality to the AUDs layer of our hierarchical framework, although they work off of somewhat different underlying assumptions.

In the approach described in this section, we model each concept in the lexicon with a set of basis vectors– using such a set allows us to account for various acoustic manifestations of the same concept, by identifying a subspace from which sound is produced for that concept. Each concept source, when active, produces sound using a weighted combination of its basis vectors. The observed audio is assumed to be generated by the additive combination of the sounds produced by the active concepts. We assume, further, that even though there may

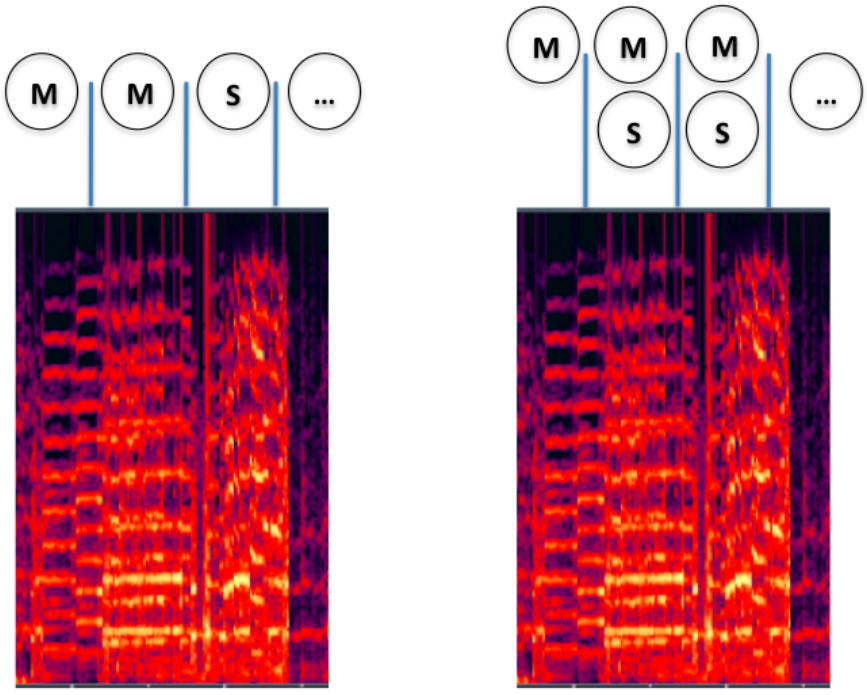


Figure 6.1: Audio analysis (L) Only one unit can be active at any time (R) Proposed approach, where a sparse subset of possible concepts can be active concurrently.

be many such concepts, only a sparse subset will be active at any given instant. However, since there are no constraints on the number of concept-specific basis vectors that may be active when that concept is active, the weight vector at any instant will be block-sparse.

To illustrate the difference, consider an example audio from a *birthday party* scene where music is playing and people are talking (Fig. 6.1). Units corresponding to *music* (represented by **M** in the figure) and *speech* (represented by **S**) should both be active, but a hard-quantization-based system might choose the dominant music (left panel, Figure 6.1), while a soft-quantization system would attribute some speech (as well as music) audio to bases for other concepts. This can lead to loss of discriminative information, and makes further analysis using the lexical unit-based representation harder. The proposed framework would be able to recover the co-occurrence of the different units, resulting in a better interpretation of the audio content, as shown in the right panel of Figure 6.1.

While we would ideally evaluate our proposed framework on the accuracy with which it estimates source presence, no such datasets currently exist for audio in-the-wild. We evaluated the proposed analysis on an audio retrieval task, where the learnt lexicon and the estimated presence of the units is used to characterize the audio file for retrieval,

and obtained significant improvements over standard baselines. However, we expect such models to be useful for various other applications (audio recounting, for instance), since the concurrent estimation of sources allow a finer-level analysis of the audio than current systems would permit.

We briefly present prior related work in Section 6.1.1, and describe our proposed framework and the various models in Section 6.1.2. In Section 6.1.3, we experiment with this model on the BBC data, and discuss our findings in Section 6.1.4.

6.1.1 Related Work

Our framework builds on initial past work in estimating concurrent concept presence in audio content using PLSA Mesaros et al. [2011]. Unlike Mesaros et al. [2011], our approach does not use annotated concept data, since such data is expensive to obtain for audio-in-the-wild, and learns the set of concepts unsupervised. It also incorporates the assumption that only a sparse subset of concepts can be active at any given instant. The ability to estimate these multiple sources allows us to expect more robust performance from this framework. One can think of this proposed framework as imposing a layer of signal separation-based auditory scene analysis Bregman [2004], Cho and Saul [2009], Ellis [1996] with sparsity constraints on top of any of the standard frameworks for audio content estimation. As described in Section 6.1.2, the representation of each of the lexical units using a set of bases means that we need to impose sparsity not directly on the weights for the bases but on the sets, using techniques for block-sparse weight estimation Ben-Haim and Eldar [2011], Eldar et al. [2010]. Similar group sparsity-based techniques have been employed for speaker identification Hurmalainen et al. [2012].

The procedure for estimating the weights of these bases relate to past literature on sparse recovery techniques Becker et al. [2011], Garg and Khandekar [2009], Needell and Tropp [2009]. It also relates to *dictionary learning* techniques Chaudhuri et al. [2011], Pancoast and Akbacak [2011], Yaghoobi et al. [2009], with the difference that we require the learned dictionary to permit block-sparse characterization of data. The process of learning outlined in this paper is similar to techniques used for data decomposition, such as NMF Lee and Seung [2001] and semi-NMFDing et al. [2010] (where the latter permits the use of negative data and bases), our proposed approach additionally imposes sparsity constraints on the estimation process. Unlike sparse variants of the NMF formulation Hoyer [2004], however, our model requires concept-level sparsity and estimates a block-sparse weight vector, instead.

6.1.2 Proposed Block-Sparse Model

The framework proposed in this section is designed to improve upon one of the limitations of current audio content analysis systems, by allowing multiple sources to be concurrently active. Our assumption for the underlying process states, however, that only a sparse subset of all the possible concept sources (dictionary elements) could combine to produce the audio. In this section, we present a novel framework for representing these sources and estimating their presence in the audio. We note that the modeling approach presented in this section does not employ the kind of local structure that was used in the AUDs learning framework, where each AUD unit was a 5-state left-to-right Hidden Markov Model that allowed it to model local temporal structure. Instead, the current implementation of this model works at the level employed in prior approaches, over individual frames or fixed frame windows. The extension of this paradigm to an AUDs-like structured framework is left for future work.

Our proposed model is designed to improve upon the existing techniques mentioned earlier Chaudhuri et al. [2011], Pancoast and Akbacak [2011], Zhuang et al. [2011], and begins by using a standard K-means algorithm to learn a dictionary of K units. This dictionary can be used to assign each audio frame to one of the dictionary elements, using Vector Quantization (VQ). Thus, for each dictionary element, a set of audio frames is assigned to it from the VQ estimation step. In our framework, we refer to each of the dictionary elements as an *atomic concept* and model each concept with a set of basis vectors (as opposed to a mean vector for K-means). For each concept, its basis vector set consists of M basis vectors that can be obtained either by randomly sampling exemplar frames (from the set of frames assigned to that concept) or by using an iterative learning process that we will shortly describe.

Let us first introduce the notation used in this section. We assume that the observed data $\mathbf{D}$ (N audio frames, of dimensionality F each; thus, an N x F matrix) has been generated by a non-negative weighted combination of a sparse subset of concepts. We refer to the set of bases for all concepts collectively as $\mathbf{B}$, and that for each concept as $\mathbf{B}_i$ ($i \in [1, K]$, for K concepts). $\mathbf{W}$ refers to the weight matrix of size (KM) x N, with a weight vector for the entire basis set at each time step. The weight for the j-th basis in the i-th concept bag at the t-th time step is indexed by $w_{ij}^{(t)}$.

We first describe the process of estimation of weights given the set of basis vectors for each concept, while imposing concept-level sparsity. Typically, algorithms for sparse estimation apply L_0 norm minimization on the vector being estimated. These include greedy algorithms such as Iterative Hard Thresholding (IHT) Blumensath and Davies

[2008] and Compressive Sampling Matching Pursuit (CoSaMP) Needell and Tropp [2009]. Alternatively, other approaches relax the NP-hard L_0 minimization problem by using an L_1 penalty instead on the vector, as in the Lasso algorithm Tibshirani [1994]. In this paper, we work off of a generalized definition of sparsity for a vector discussed later, which can be shown to be analogous to the L_1 formulation. This generalized definition allows us to measure sparsity on a bounded scale between 0 and 1.

As mentioned before, our approach enforces sparsity at the concept level instead of the individual weights for each basis vector, leading to a block-sparse weight estimation process. To model this, we introduce a coefficient α to measure the activation level of the individual concepts:

$$\alpha_i^{(t)} = \sum_{j=1}^{M} w_{ij}^{(t)}, \forall i \in [1, 2...K] \tag{6.1}$$

Since the weights are constrained to be non-negative, the activation level is always non-negative. We measure sparsity at the concept level using α as in Equation 6.2. ϕ represents the concept level of sparsity, and lies between 0 and 1. A higher value for ϕ indicates higher sparsity; ϕ is 1 when only one element in α is non-zero, and is 0 when all elements are equal and non-zero.

$$\phi(\alpha^{(t)}) = \frac{\sqrt{K} - \frac{\sum_i \alpha_i^{(t)}}{\sqrt{\sum_i \alpha_i^{(t)2}}}}{\sqrt{K} - 1} \tag{6.2}$$

Given a concept dictionary (which includes the set of basis vectors for the concepts), we can estimate a concept-sparse set of weights for the data by optimizing the following objective function (S represents the desired degree of sparsity):

$$\begin{aligned} & \min_W ||D - W^T B||^2 \\ s.t. \quad & \phi \geq S \\ & W_i \geq 0, \forall i \end{aligned} \tag{6.3}$$

The objective function above does not have a closed-form solution, but a solution can be obtained using an iterative procedure shown in Algorithm 4. Step 6 in Algorithm 4 requires the projection of the α onto a non-negative space such that the projected vector meets the desired sparsity constraints Hoyer [2004]. The projection operation is described in Algorithm 5.

Algorithm 4 Obtaining an optimal set of weights to satisfy the objective function above, given a set of basis vector bags

Step 1: Initialize $\mathbf{W}$ randomly
Step 2: Compute α for each observation
Step 3:Project each α vector to be non-negative, have unchanged L2 norm with L1 norm set to achieve desired sparseness
Step 4:$\mathbf{W} \leftarrow \mathbf{W}. * (B^T D)./(B^T BW)$
Step 5:Recompute α based on the new $\mathbf{W}$
Step 6:Project each α vector to be non-negative, have unchanged L2 norm with L1 norm set to achieve desired sparseness
Step 7: Go to Step 4, till maximum iterations are reached

Algorithm 5 Projecting a vector ($\mathbf{x}$) onto the non-negative space with desired L_1 norm, and unchanged L_2 norm

Step 1: $p_i \leftarrow x_i + (L_1 - \sum_i x_i)/dim(\mathbf{x})$
Step 2: $Z \leftarrow \{\}$
Step 3: If $i \notin Z$, $m_i \leftarrow L_1/(dim(\mathbf{x}) - size(Z))$
Step 4: If $i \in Z$, $m_i \leftarrow 0$
Step 5: $\mathbf{p} \leftarrow \mathbf{m} + \gamma(\mathbf{p} - \mathbf{m})$, where $\gamma \geq 0$ is selected so the resulting $\mathbf{p}$ satisfies the L_2 norm constraint
Step 6: If $p_i \geq 0, \forall i$, return $\mathbf{p}$, end
Step 7: $Z \leftarrow Z \cup \{i : p_i < 0\}$
Step 8: $p_i \leftarrow 0, \forall i \in Z$
Step 9: $c \leftarrow (\sum p_i - L_1)/(dim(\mathbf{x}) - size(Z))$
Step 10: $p_i \leftarrow p_i - c, \forall i \notin Z$
Step 11: Go to Step 3

At training time, the estimated weights can be used to re-estimate the bases. Thus, for the j-th basis for concept i:

$$B_{ij} = \frac{\sum_t w_{ij}^{(t)} \times D^{(t)}}{\sum_t w_{ij}^{(t)}} \tag{6.4}$$

At test time, the weight vector obtained can be used to estimate the occurrence (F) of the individual concepts in an audio file with T frames:

$$F_i = \sum_{j=1}^{j=M} \sum_{t=1}^{t=T} w_{ij}^{(t)} \tag{6.5}$$

While the basis vectors for our experiments should ideally be in the spectral domain, the high dimensionality (typically, 257-513) often result in poor basis estimation– indeed, using

exemplar-based spectra as an overcomplete basis set is common Raj et al. [2010], Sainath et al. [2011]– and the domain of audio in-the-wild exacerbates this problem. Instead, we work in the dimensionality reduced Mel-Frequency Cepstral Coefficients (MFCC) domain, which has been used in past work to explain MFCCs for speech recognition Sainath et al. [2010, 2011], and we empirically find that this representation proves effective.

At this point, we note that the algorithmic solution outlined above holds in principle for the MFCC coefficients as well. However, the weight update rule in Step 4 of Algorithm 4 has been developed under the assumption that the set of bases consist of non-negative elements. When using MFCC vectors, this is not the case (as opposed to using spectra, which are non-negative. Our update rules for weights in this case are derived using a gradient descent approach obtained by updating the current estimate of the weights with an offset that is obtained from differentiating the objective function shown in Equation 6.3, scaled with a learning rate.

$$W^{(i+1)} = \max(W^{(i)} + 2\eta B^T(D - BW^{(i)}), 0) \tag{6.6}$$

In the above equation, $W^{(i+1)}$ refers to the updated weights in the next iteration, from $W^{(i)}$, the current estimate of weights in the i-th iteration. The learning rate, η, is typically quite small (of the order of 10^{-3}) and prevents sharp changes in parameter estimates that can make the new estimate go past the optimal point. As a result, typically, the weight estimates do not become negative after being initialized with positive numbers. However, we do check for negative weights, and negatives that occur are subjected to a floor introduced by the max in Equation 6.6 and set to 0.

6.1.3 Experiments

In this section, we employ our block-sparse estimation framework on the BBC dataset on the audio retrieval task, that has been used as the semantic audio task of interest throughout this dissertation. As noted earlier, we present results of our experiments using the MFCC characterization for audio frames.

We use the Vector Quantization technique at the frame level to initialize the set of basis vectors for each concept in our system. The approach presented in Section 6.1.2 was then used to compute a concept-sparse estimate of the weights for each audio frame. We then use Equation 6.5 to compute the relative occurrences of each concept (i) as shown below:

$$\mathcal{F}_i = \frac{F_i}{\sum_i F_i}, \forall i \in [1, 2...K] \tag{6.7}$$

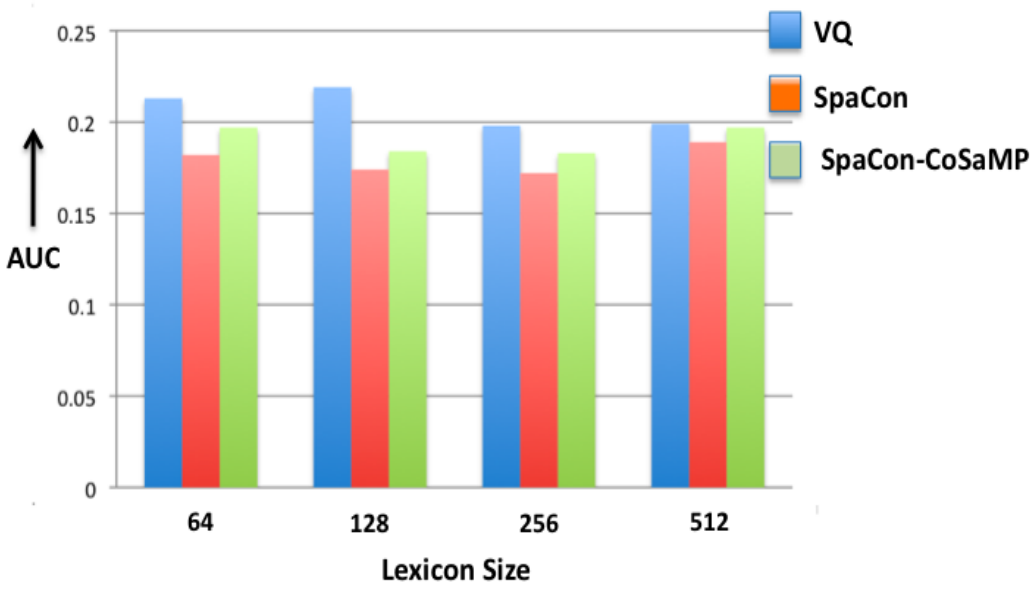


Figure 6.2: Comparison of the various systems using average AUC (y-axis) with varying lexicon size on the audio retrieval task on the BBC dataset (**lower is better**)

Now, the audio file can be represented as a K-dimensional feature vector for the retrieval task, with one feature for each concept where the feature value is the relative occurrence ($\mathcal{F}$) for the concept. We refer to this system using sparse concept representation as **SpaCon**.

The current state-of-the-art in audio retrieval systems currently use "bag-of-words" representations of recordings generated using the Vector Quantization technique Pancoast and Akbacak [2011], Zhuang et al. [2011]. We use the VQ-based system, described as one of the baselines in Chapter 2 as the baseline for comparison with our proposed framework.

In the estimation process outlined in Algorithm 4, the projection of the concept activation vector (α) to the non-negative space with desired sparsity results in some of the concepts being set to 0 early in the iterative process. The update rule for the weights in Step 4 can no longer recover non-zero weights for those concepts in further iterations. To avoid erroneous concept selection at an early stage, we implement a CoSamp-style approach Needell and Tropp [2009] where the projected vector is augmented with a support set consisting of the $2s$ concepts (for an s-sparse projected vector) with the highest gradient values in each iteration. At the end of the iterations for weight estimation, this augmented vector is finally projected down to the desired sparsity level to obtain the final sparse estimate. Again, audio files are represented using the relative occurrence feature representation, as in the SpaCon system. We refer to this system as **SpaCon-CoSaMP**.

Figure 6.2 compares areas under the curve (AUC) of Missed Detection vs False Alarm

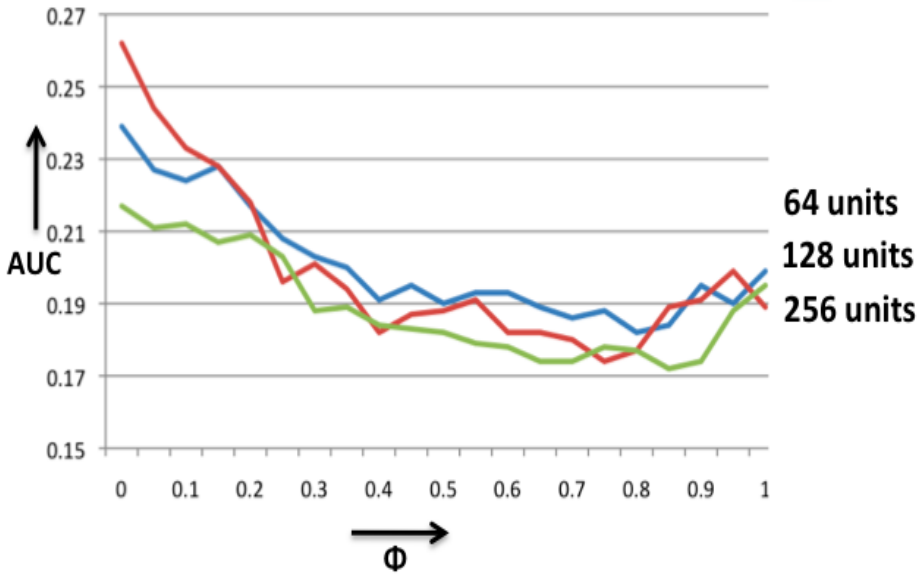


Figure 6.3: Effect of changing the desired sparsity on average AUC (y-axis) (**lower is better**) in the SpaCon system.

rates for the 3 systems described above for varying sizes of the concept lexicon. Recall that since the curve plots missed detections against false alarms, a lower AUC is better.

The sparse concept estimation-based systems provide significant improvement over the VQ baseline. Since the desired degree of sparsity can be modified by the ϕ parameter, this improvement is expected, since the sparsity can be relaxed to create the equivalent of the VQ system. We note that the use of the CoSaMP-style estimation with an augmented support does not appear to improve performance on this retrieval task, resulting in slightly deteriorated performance, in terms of AUC. However, we do believe that this direction needs to be further explored and that such research will improve our understanding of audio content, and how we can analyze it better.

The degree of sparsity (ϕ) imposed on the estimation process outlined earlier has a significant effect on the performance of the system. Figure 6.3 tracks the change in AUC for the SpaCon system with changing the degree of desired sparsity (ϕ) for different lexicon sizes. This plot shows an optimal operating point for ϕ values between 0.75 and 0.9 for the different lexicon sizes.

6.1.4 Discussion

In this section, we presented a novel, signal-separation based approach to audio content analysis, and demonstrated significant improvement in performance over the commonly

used Vector Quantization based technique for audio retrieval on a dataset containing complex audio tracks. While this improvement is exciting by itself, we believe that the importance of this work lies in the fact that it presents a more natural model for the understanding of audio content, due to the assumptions of sparsity among the very large space of natural audio concepts. The improvements in the audio retrieval task suggest that this technique recovers a better estimate of the concept occurrences.

The objective function being optimized in Equation 6.3 could be modified in appropriate settings to add further constraints. For instance, given prior external knowledge about the relations between the various concepts in the lexicon or about the domain of data, the estimation process could make use of expected structure in estimating the presence of the different concepts.

The improved estimate of concept co-occurrence itself could be used in various ways in the future. Specifically, in the work described earlier, we developed a model for extracting patterns over the low-level units in order to understand how lower-level acoustics (units) combine to produce higher-level semantics Chaudhuri and Raj [2012]. The framework proposed here could be used in conjunction with the one in Chaudhuri and Raj [2012], to better leverage the concurrent occurrence structure for improved semantic analysis. We continue to actively explore these directions.

6.2 Multiple Instance Learning for Semantic Analysis

When analyzing semantics in an audio recording, especially reasonably long recordings set in natural circumstances such as the ones usually contained in user-generated content, one is faced with legitimate conundrum. In such recordings, while the focus may have been on capturing a particular semantic event, other events with conflicting semantic imports may be naturally present. As such, these recordings are rarely semantically pure.

To illustrate this issue, consider a case of a Youtube video of *dog barking*. As one can imagine, such audio often consists of other acoustic events (people talking, traffic sounds, etc) before or after the actual barking. Learning algorithms, however, cannot distinguish between segments that include and don't include barking within a single file, and assumes that the entire audio represents a positive example of barking. This assumption will reduce the discriminative ability of the model.

The negative class of the *dog barking* data however contains purely negative samples. Thus, we have a set of pure data belonging to one class, but a mixed set in the other class. In the example above, however, we do have the knowledge that the *dog barking* labeled

audio must have at least one segment in it that corresponds to a dog barking. Any learning algorithm employed for this task needs to be suitably modified to take advantage of this knowledge to train stronger classifiers.

This framework, however, can be generalized to a task of learning from weak supervision, where the system can deduce finer level information from coarse labels– we have *bags* of data, and a few of those bags are labeled as containing data belonging to class c_1, and some as not containing data from c_1. We need to learn a model to detect c_1, so that we can detect other bags that contain c_1, as well as locate the instances. The ability to do this would be useful for a variety of tasks since coarse labels are easier to assign for human annotators, and the system could automatically extract finer level information from them, by locating segments in the larger *bag* that correspond to the label class. This paradigm is known in the literature as multiple-instance learning Dietterich et al. [1997] (MIL).

The ability to infer sub-file level labels or associations from data using only the labels provided at the file level, as well as understanding the granularities at which such labels apply automatically, would make audio analysis systems more powerful. Consider an example from the MajorMiner dataset Maj [2007]. The dataset contains labels for clips of music where the labels correspond to various kinds of information relevant to the music, including genre, mood, tempo, etc. Musical pieces will often capture multiple moods in the same piece. As a result, the annotations might contain multiple mood labels for the same piece. Typically, supervised approaches treat the entire audio file as a positive instance for each mood label associated with the file, resulting in a weakened ability of the learnt models to discriminate between positive and negative instances. This issue sheds some light on the level of difficulty in the annotation process– asking human users to navigate through clips to label individual segments to generate richer annotations results in a harder annotation task, which is more expensive in terms of time and cognitive effort. On the other hand, asking them to listen to an entire audio clip and annotating it with keywords they think apply to the clip is significantly easier.

One way of handling this issue would be to attempt to automatically assign the labels to their correct granularity, where they are directly applicable. In such a setting, we can treat the audio file as a bag, and divide the audio into segments of audio. These segments are treated as individual data points contained in the bag. Let us consider the case of building a detector for a specific label: various audio files have been annotated with that label and are positive instances, while the remaining files in the dataset are negative instances. Each instance can be represented as a bag of data points, where each datapoint is a segment of the audio.

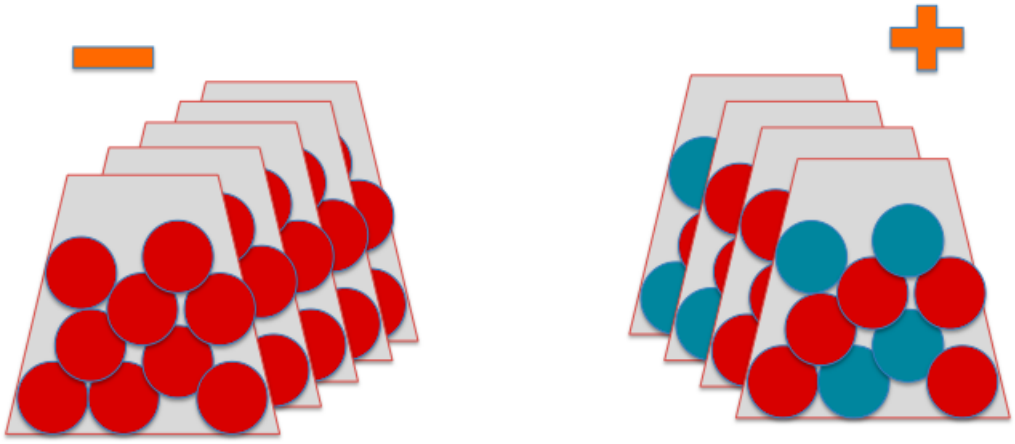


Figure 6.4: An example of a dataset for multi-instance learning

Thus, the training set up will appear to be as shown in Fig 6.4. The quadrilaterals represent bags of data– red balls represent the audio segments that are negative instances of the label, and blue balls represent positive instance. Thus, the positive bags are guaranteed to contain at least one positive data point, while the negative bags contain no positive data points. Such a learning setting is known as multi-instance learning in the literature Dietterich et al. [1997]. A standard approach to solving this problem utilizes a mixed-integer linear program in a Support Vector Machine framework, where the problem can be represented as follows:

$$\min_{y_{ij},w,b,\xi} \frac{1}{2}||w||_2^2 + C\sum_{ij}\xi_{ij} \tag{6.8}$$

subject to

$$y_{ij}(w.x_{ij} + b) \geq 1 - \xi_{ij} \tag{6.9}$$

$$y_{ij} \in \{-1, 1\} \tag{6.10}$$

$$\xi_{ij} \geq 0 \tag{6.11}$$

$$\sum_{j=1}^{l_i} \frac{1}{2}(y_{ij} + 1) \geq 1, \forall i \in I^+ \tag{6.12}$$

$$y_{ij} = -1, \forall i \in I^- \tag{6.13}$$

In the above, i indexes the bags of data, while j indexes the data points in the bag. I^+ indicates the positive bags while I^- indicate negative bags. The number of instances in the i-th bag is assumed to be l_i.

The approach described above is one instance of a solution for this learning setting. Various other frameworks, including logistic regression, boosting, MCMC sampling, etc, have been developed for multi-instance learning. The advantages of using this approach is two-fold: first, it would enable us to leverage coarse annotations to automatically infer their correct segments of applicability. Second, the ability to accurately label segments of audio pieces provides a natural mechanism for providing enhanced navigability to users.

In the following section, we present some brief initial experiments with using some of the standard multiple instance learning approaches on the audio retrieval task, and discuss some of our observations.

6.2.1 Experiments with multiple instance learning

We experimented with multiple instance learning using the SVM formulation (MI-SVM) described above, as well as with a logistic regression (MILR) formulation. While such formulations can be used at each level of the hierarchy, in this preliminary work, we only investigated using them at the level of the low-level acoustic units descriptors (AUDs). The entire audio data were decoded using the learnt AUDs, as described in Chapter 3.

In order to characterize each of the recordings as a bag of individual exemplars (whether positive or negative), the audio was split evenly into 10-second chunks (if the audio were shorter than 10 seconds, the *bag* had only one exemplar segment within it) and each bag had multiple such 10 second segment units as the set of exemplars within the bag. The 2 standard multiple instance learning formulations (MI-SVM and MILR) were then applied on top of this representation of the dataset. For our experiments, we used the implementations of MI-SVM and MILR available in the open source, machine learning software toolkit Weka Hall et al. [2009].

The results of using these formulations for our audio retrieval task are summarized in Table 6.1. For this preliminary experiment in multiple instance learning, we only compare performance with the AUDs baselines reported earlier. On these datasets, for the audio retrieval task, we do not observe a significant benefit from using multiple instance learning approaches. The logistic regression based approach appears to outperform the SVM based approach on both datasets, and it is slightly better than the simple AUDs characterization for the BBC data and slightly worse for the MED11 data.

While our preliminary experiments with using the MIL approach did not produce significant improvements for the audio retrieval task, we do believe that it should still be appropriate for modeling certain audio tasks. Specifically, one of the directions that we believe should be investigated involves tying the learning of the low-level units directly

System	Average AUC in BBC	Average AUC in MED11
AUDs	0.1744	**0.2174**
MI-SVM	0.1861	0.2211
MILR	**0.1733**	0.2192

Table 6.1: Performance comparison using AUC of multiple instance learning approaches over AUDs-based characterizations compared to using simply the AUDs-based characterization on the audio retrieval datasets. (Lower is better)

with an MIL framework, such that the low-level acoustic unit lexicon learning algorithm works taking into account the constraint that not all segments of audio relate to the specific semantic label provided, and the learning process directly makes an effort to identify semantic-topic-relevant units jointly with the learning process. Further, while our preliminary experiments involved using MIL with the AUDs layer, the assumptions made in applying MIL are general, and can be applied to any of the layers in the hierarchy.

6.3 Non-parametric Learning for Audio

The various algorithms and learning techniques used in this dissertation all use *parametric* techniques. The word parametric implies that the number of parameters used in the learning system is fixed when the training process begins. The size of the set of parameters for each of these algorithms is typically governed by a hyperparameter that is input to the system when the training process is started, *e.g.* number of AUDs in the lexicon, number of events in each of the layers built on top of the AUDs.

Such a system has 2 potential drawbacks. First, in order to decide on the optimal value of the hyperparameter, the learning process has to be repeated a few times for different values of the hyperparameter, and tested on a held out development set to find the best value. If the learning process is time-intensive, as is the case for most large-scale datasets (such as the MED11), this can significantly increase training time. Second, in designing generative processes that try to explain the process of generation of data, we typically attempt to mimic the cognitive processing of humans. We believe there is good merit to the argument that a parametric process where the number of units has been decided before the data is investigated is not a natural process that humans employ. Instead, as they encounter more data, the size of the set of units (whatever the specific units may be) can keep increasing to explain the observed data, especially if the prior units in the lexicon cannot do a good job of explaining new data. Such a process is referred to as non-parametric, where the size of the model grows to account for the complexity

of the data. In these techniques, individual variables are typically assumed to belong to parametric distributions, and assumptions are often made about the relationships between the variables.

The concept of applying non-parametric techniques is relevant to the entire hierarchy, and each of the individual layers, depending on the modeling assumptions being made. In this section, we will discuss it in the context of the induction of *events* on top of the AUDs, analogous to the work presented in Chapter 4, where instead of assuming a finite event vocabulary size, we grow it using a non-parametric scheme.

In this section, we first describe such a scheme in the context of our task, using prior work in non-parametric learning. We then present experimental results of using such a model on the audio retrieval task, with a brief discussion of the results.

6.3.1 Higher-order Non-Parametric Structure Induction

Recall from Chapter 4, our comparison of the task of discovering higher units over the AUDs with the text segmentation task, where the AUDs can be considered analogous to the characters of an alphabet, and the events above the AUDs layer as analogous to words. In this section, we use the same terminology, and discuss the unsupervised word discovery task over character (AUD) sequences. We employ a word segmentation algorithm which simultaneously develops a lexicon, i.e., the transcription of a word in terms of an AUD sequence, learns an n-gram language model describing word sequence probabilities, and can use this model to perform segmentation on any new AUD sequence. The underlying statistical model used is referred to in the literature as a Pitman-Yor process, which allows the learning of a word vocabulary with *a priori* specification of the vocabulary size.

As discussed in Chapter 4, various approaches have been used for segmentation of character streams with spaces removed for text processing applications Goldwater et al. [2009], Johnson and Goldwater [2009], Mochihashi et al. [2009], Poon et al. [2009], Schuster and Nakajima [2012]. We explore applying one of these systems Mochihashi et al. [2009] to the audio content analysis task. This system presents a Bayesian approach where both the lexicon (the transcription of a word as a sequence of characters) and the language model are simultaneously estimated. The language model is based on the Pitman-Yor process, which provides a random distribution over discrete probability distributions over infinite sample spaces (in our case, this corresponds to the *event* vocabulary, which may be infinitely large) Teh [2006]. This model has an added advantage in that it focuses on the temporal sequence information [1].

[1] As noted in Section 4.1.1 of Chapter 4, modeling of canonical sequences over noisy decodes will lead

For the event segmentation task, a hierarchical Pitman-Yor process (HPY) is employed, where the hierarchy represents different values of the history size n of the n-gram language model. An AUDs-level language model is nested within the event language model such that the unigram event model backs off to an m-gram model at the AUD level.

The nested Pitman-Yor language model used in Mochihashi et al. [2009] is an extension to hierarchical Pitman Yor language model, which is based on the Pitman-Yor process Pitman and Yor [1997]. The Pitman-Yor process provides a random distribution over discrete probability distributions over innite sample spaces and is a generalization of the Dirichlet process:

$$G \sim PY(d, \theta, G_0) \tag{6.14}$$

A draw form the Pitman-Yor process results in a discrete distribution G. The Pitman-Yor process has three parameters: a base distribution G_0, a discount parameter $0 \leq d < 1$ and a strength parameter $\theta > -d$ which controls the variability around the base distribution. The distribution G sampled from the Pitman-Yor process result in a *rich-get-richer* effect, thereby simulating a distribution that follows a power law property. In practice, we cannot observe G directly because it will be an infinite dimensional distribution over the possible words.

Here, we encounter a significant issue in deciding what we should use for G_0, the prior probabilities over words. If a lexicon is finite, we can use a uniform prior $G_0(w) = \frac{1}{|V|}$ for every word w in the lexicon. However, with word segmentation every substring is a candidate to be a word, thus the lexicon is countably infinite. Building an accurate G_0 is crucial for word segmentation, since it determines the set of possible words. Prior work in text segmentation has used Dirichlet processes with a relatively simple prior of uniform distributions over characters or a word length dependent Poisson priors. The work in Mochihashi et al. [2009] uses a nested Pitman Yor language model, where the character level HPY is embedded within the word-level HPY, and the base measure over the character level HPY is a uniform distribution over the characters (AUDs). For details of the training and inference schemes, the reader is referred to Mochihashi et al. [2009] and an adaptation of this approach for word discovery from phonetic input Walter et al. [2013].

While we do not discuss this work here, Walter et al. [2013] describes an adaptation of the nested Pitman Yor language model to using phonetic outputs, produced from raw input speech data processed by an automatic speech recognition system, as analogous to

to issues. We ignore this problem, in this preliminary discussion, using instead a post-processing step to account for some of the issues introduced.

Training data:
POWERFINANCIALISAFINANCIALSERVICESCONCERNTHATISSIXTYNINEPE
RCENTHELDBYPOWERCORPORATIONOFCANADAAMONTREALBASEDHOL
DINGCOMPANY

$m = 4$:
PO WER F INANCIAL IS A FINAN CIAL SERVICES C ONCERN THAT IS
SIXTY NINE PERCENT HELD BY PO WER CORP ORATION OF C ANADA A
MON TREAL B ASED HOLDING COMPANY

$m = 6$:
POWER FINANCIAL IS A FINANCIAL SERVICES CONCERN THAT IS
SIXTY NINE PERCENT HELD BY POWER CORPORATIONOF CANADA A
MONTREAL BASED HOLDING COMPANY

$m = 8$:
POWER FINANCIAL IS A FINANCIAL SERVICES CONCERN THAT IS SIXTY
NINE PERCENT HELDBY POWER CORPORATION OF CANADA A MONTRE-
ALBASED HOLDINGCOMPANY

Figure 6.5: Example of improvement in segmentation accuracy with character-level context size

characters and using it to discover words. Such a system can be used in unsupervised training of an automatic speech recognizer directly from audio recordings, without using labeled data, i.e., without knowing the text that has been spoken. Such an approach would allow a significantly reduced investment in the tedious and expensive task of obtaining good speech transcriptions for a number of languages.

6.3.2 Experiments

While results using this approach on text have already been reported, we wanted to test our implementation on text data to understand the various aspects of the model. One of the significant observations of our experiments showed that the character level language model and the size of the context m that it models has significant influence on the performance of the model (on data from the Wall Street Journal corpus). An example showing how progressively more refined character-level language models improve accuracy of the discovered words is presented in Figure 6.5.

As discussed earlier, we applied this approach to the task of unsupervised word discovery from phonetic input, and the reader is referred to Walter et al. [2013] for details of the

System	Average AUC in BBC	Average AUC in MED11
AUDs	0.1744	0.2174
EVENT	0.1911	0.2297
EVENT–COMB	**0.1729**	**0.1842**
NP–Moc	0.1979	0.2515
AUDs+NP–Moc	0.1792	0.2213
NP–Moc–clust	0.1964	0.2339
AUDs+NP–Moc–clust	0.1794	0.2165

Table 6.2: Performance comparison for the non-parametric system on the audio retrieval tasks

work, and results. Relevant to the work presented in this dissertation, we applied this model to the audio retrieval task, and the results relative to the AUDs only and AUDs and events setting are summarized in Table 6.2. The non parametric setting, based on Mochihashi *et al*'s prior work is referred to as NP–Moc. Recall that *EVENT* refers to the power-law based induction of higher level units, while *EVENT–COMB* refers to combining the information from the event and the AUDs layers.

The results for the non-parametric systems are the best numbers obtained for $n = 3$, which is the language model context used. First, we note that this is significantly smaller than the size 8 that worked best for the text segmentation task as shown in Figure 6.5. Second, we note that systems using the non-parametric system perform worse than the corresponding parametric process that we described in Chapter 4, based on a power law assumption for a fixed number of units, both individually and in combination with the AUDs layer. This is likely due to the differences in modalities that was discussed in Chapter 4, since the current implementation of the non-parametric approach looks for canonical sequences. We can alleviate this problem to some extent, by introducing a post-processing step, where all unit sequences within a certain edit-distance are clustered together. This improves the results somewhat for the NP–Moc setting and the results are shown in the NP–Moc–clust, where best results are obtained using an edit distance of 2.

While we do still believe that a non-parametric process is theoretically a better paradigm for performing learning of semantic concepts at higher levels, we note that adapting some of the standard formulations to semantic audio processing tasks is a significant undertaking. While the edit-distance-based post-processing does improve performance somewhat, we are skeptical that this improvement can be expected to generalize, since the best edit distance of 2 appears to be smaller than expected, given the noise likely present in the decodes. Further, the improvement does not seem to carry over when those units are combined with the AUDs units. This indicates that even if they do a slightly better job at the events

layer, they capture very little novel information that the AUDs don't already capture.

An ideal approach here would be to integrate the ideas incorporated in our higher unit learning model of Figure 4.2 to tackle the issues of noise and semantic ordering and non-parametric learning, to learn the parameters of the events layer in a non-parametric setting, where the number of event units is allowed to grow with the data. We hope to accomplish the development of such a model in future work.

6.4 Video analysis

While in the experiments reported throughout this dissertation, we focus solely on audio, we noted that our formulation makes no assumptions that are specific to the audio modality. In fact, the TRECVID task provides multimodal data, and the various semantically motivated assumptions that we make are largely applicable to video data as well. In this section, we present a short description of a preliminary application to video analysis using such a framework. The efforts that we describe here are applied to the video to discover units at a level analogous to the AUDs for audio. The further higher level structure induction approaches that we described for audio in Chapter 4 and Chapter 5 can be applied in the same manner as over AUDs. The most important piece in being able to port our approach to video is to find a good way of learning the lower level units, which we refer to as VIsual Descriptors (VIDs, in short).

The hardest task in being able learn good VIDs is in finding a representation for the video frames that can compactly represent the semantic content. The main difference in the case of video, as opposed to audio, is that while the MFCC representation is a standard way of representing the audio, the semantics in the visual frames lie in two dimensions from which various different kinds of semantic inference are made. This is, in fact, a hard challenge that requires careful analysis and design of the low-level feature space (analogous to the MFCC representations for audio). In this work, we don't address that problem. Instead, we adopt the simpler approach of using several different visual concept detectors to find a representation of each frame using a known set of concepts. The values corresponding to each of these concepts can be concatenated to create a feature vector for each frame, where the feature value is the number of times each concept appears in the image. This 1-dimensional vector is analogous in our system to the MFCC features for audio, and the VIDs learning is done over these features, where the video has been processed to a frame rate of 30 frames per second. The features are extracted for each of the video frames in the data using 8 different object detectors that detect the number

Algorithm 6 d-dim discrete space –> continuous space

$D \leftarrow$ d-dimensional discrete representation (n datapts x d)
$P \leftarrow PCA(D, n)$; P represents the d principal components
for $i = 1 \rightarrow n$ **do**
  $\mathcal{F}_i \leftarrow$ Component Loadings for D_i
for $j = 1 \rightarrow d$ **do**
  Let continuous distribution of choice be $\mathcal{N}(\mu_j, \sigma_j^2)$
  $\mathcal{G}_j \leftarrow CDF(\mathcal{N}(\mu_j, \sigma_j^2))$;
  $\mathcal{C}_j \leftarrow CDF(\mathcal{F}(:, j))$
  $\mathcal{T} \leftarrow$ Function to transform $\mathcal{C}_j$ to $\mathcal{G}_j$
  $F(:, j) \leftarrow \mathcal{T}(\mathcal{F}(:, j))$
$\mathcal{P} \leftarrow PCA(F, n)$; $\mathcal{P}$ represents the d principal components
for $i = 1 \rightarrow n$ **do**
  $\mathcal{F}_i \leftarrow$ Component Loadings for $\mathcal{P}_i$
$\mathcal{F} \leftarrow$ Continuous representation for further processing

of people, faces, hands, vehicles, tires, crouching poses, wood and fur textures. These are used to construct the frame-specific feature vector, and then used to learn VIDs.

The algorithm used for learning is exactly the same as the one used for learning the AUDs, as described in Section 3.1.1 of Chapter 3. Since these features are discrete integer valued, we transform them to a continuous space, using the process illustrated in Algorithm 6. This is done since continuous spaces are no less expressive than discrete ones, with the continuous space permitting more operations, if required. We chose to model the transformed space with gaussians, although any other appropriate density functions may be employed.

Table 6.3 shows a comparison of using the VIDs learnt in this manner to a baseline VQ–based system for learning visual units on the audio retrieval task for MED11. As we did when reporting results with AUDs, we use a characterization of recordings based on the presence of the VIDs (VIDs–Binary) and a characterization based on the frequency of occurrence of the VIDs in the recording (VIDs–Count). We obtain the best performance when using a lexicon size of 256 units for the VQ and VIDs. We observe from the table that simply presence information over these visual units is sufficient to outperform the VQ based system which does not used structured information in the learning process. The VIDs–Count based system is the best performing system for this characterization which is not surprising, since as discussed in Chapter 3, the actual frequency of occurrence of a concept in a recording should provide more information about the semantic topic than simply information about its presence.

We note that the results presented here are simply a proof-of-concept to show that

System	Average AUC in MED11
VQ-Video	0.2832
VIDs–Binary	0.2591
VIDs–Count	0.2447

Table 6.3: Performance comparison using AUC of multiple instance learning approaches over AUDs-based characterizations compared to using simply the AUDs-based characterization on the MED11 dataset. (Lower is better)

the iterative learning using structured models can be used to identify improved semantic information as opposed to simply using a fixed frame-level characterization. These are not currently competitive with video analysis based results on the same task, since we use a very simple semantically motivated low-level feature set to characterize audio frames. These features are themselves the outputs of various detectors designed to detect certain objects, and therefore already contain some error in the object detection phase, resulting in noisy features.

Nonetheless, we feel that future research in the video analysis area would benefit from applying structured models on top of standard low-level feature descriptors to identify higher level semantics, as well. The models that we presented in this dissertation should be equally applicable to identifying semantics from a video stream, with low-level modifications to appropriately incorporate improved low-level feature representations for the video.

6.5 Discussion of Future Work

In this chapter, we discussed some directions of future work in the area of semantic content analysis that we believe are important directions for research in the near future. The various approaches and results presented in this chapter are fairly preliminary and should be treated as such. Their true importance lies in presenting the community with a concrete formulation to consider, and debate the benefits and drawbacks of, and use them as an initial step in these directions. The results reported should serve as early baselines that future work can build on.

We note that while the area of semantic content analysis for audio has been explored by researchers for over a decade, it has been only in the past 5 years or so that interest in this area has really spiked, in part due to the IARPA-sponsored ALADDIN program. As such, this might be the first phase of a large-scale, significant investment (in terms of time and money) of concerted research focus in this area, which is a great development for a

very important research area.

While we discussed a few challenges that we believe research efforts should attempt to tackle in the near term, there are a large number of potentially interesting directions for this area, at this stage. In the next chapter, we will conclude this dissertation by discussing the bigger picture of semantic audio content analysis, where this dissertation fits into the bigger picture, a discussion of the main lessons learned from our work in this discussion, and an alternate direction of investigation that will be critical to this area.

Chapter 7

Conclusions

There has been a recent spike in research interest in semantic analysis of multimedia content. In our work in this dissertation, we focus on the audio modality only. Whereas most of the past work in this area so far has approached the analysis task with shallow analysis techniques, based off of analyzing the acoustics directly, there has been an increased interest in deeper analysis for various audio processing tasks recently Deng et al. [2013], Hamel and Eck [2011], Lee et al. [2009], Lu et al. [2013]. There appear to be 2 main issues with deep learning approaches– first, interpretation of the learnt models is not easy. Second, like a number of approaches in the prior work, the input to the deep networks typically work off of a pre-specified window of frames.

This dissertation presents an approach to deeper analysis of audio for semantic tasks that attempts to address the limitations of prior work in this area. We present a framework that maps acoustics hierarchically to higher level semantics, and the higher layers in the hierarchy correspond to increasingly higher levels of semantic inference. In this dissertation, we developed techniques for learning the various layers of this hierarchical framework, and demonstrated that the units learned at the various levels capture semantic information by applying characterizations based on these units to a semantic audio retrieval task, which showed significant improvements over state-of-the-art techniques.

7.1 The need for datasets

When we first considered the various options in terms of directions that we could go in for this dissertation, we felt that there were 2 principal directions for consideration. The first was the direction that we actually decided to go in– namely, the development of a framework and algorithms that would allow the extraction of semantics from audio.

The second direction was the development of Treebank-style datasets that would establish commonly agreed upon high-level semantic annotation schema, and would result in the development of a large annotated semantic dataset that could be used for learning of semantic units from data in a supervised manner. Alternately, such data could be used for evaluation of semantic units learnt in unsupervised settings, as well.

As one can see, these two directions are not independent. One of the main reasons that we chose the former direction was that the latter direction would require a significant investment of time from the community as a whole in order to converge to an agreed upon schema for such annotations. Besides, once such a schema has been agreed upon, the process of development an annotated semantic audio treebank would require significant investments of time and effort, employment of annotators as well as various other significant logistical challenges.

In the current age, however, the popularity of crowdsourcing systems provides an alternate paradigm for generating annotations for audio, leading to a somewhat different challenge of being able to normalize such annotations, since multiple untrained annotators would likely not organically converge to a small set of keywords. Thus, multiple annotators capturing similar semantics with different literal annotations would require a subsequent step of standardization that identifies when they intended to use the same annotations, and collapse those.

Naturally, annotations generated from crowdsourced scenarios have their own unique challenges that need to be addressed. Nonetheless, it might turn out to be a relatively inexpensive way of getting started with small datasets, while the community, as a whole, converges to a commonly accepted annotation schema.

Another reason that we were drawn to the first approach of developing algorithms that automatically identify semantically coherent segments in audio (at various levels) was that such algorithms, by design, provide us with hypothesized semantic segment boundaries. These segments represent sets of shorter audio segments grouped together in various ways (learning of a lexicon at the various levels, *e.g.*) that can be used to catalyze the process of generation of annotations. Further, a small amount of supervision obtained using these segments can be used to refine our models for semantic sense extraction from audio using approaches to learning in weakly supervised settings.

All these various directions of thought lead, however, to the same place, which is our belief that the next big improvement in performance of systems attempting to analyze semantics will require richly annotated datasets that can then be used for developing improved models. Thus, the need for development of such datasets, whether it be done

iteratively using learning algorithms that can bootstrap from small amounts of data, or by an organized effort to generate a large Treebank-style dataset, is paramount, and we hope that such directions will be explored in future work.

The development of even small amounts of hierarchically annotated data would be tremendously useful. The most important benefit of such data would be that researchers could now directly compare the induced structure using their models to the structure in the ground truth annotated dataset. As mentioned earlier, there are significant challenges to overcome in terms of standardizing the sets of annotations used, etc. Nonetheless, even though the induced structures by algorithms such as the ones we presented in this dissertation do not map (at this point, in time) to human interpretable concepts, we can still use the ground truth annotations to evaluate the accuracy of the induced segmentation, in a similar manner as is done to evaluate parse trees in text analysis.

7.2 Where might hierarchical analysis help?

In this dissertation, we developed a hierarchical analysis framework for audio semantics. We believe that this is a natural model for analyzing audio content, and that such approaches will lead to a better understanding of the semantics and how they relate to acoustics. Such understanding would likely, in the future, be an important part of systems that perform retrieval, classification, event recounting, monitoring and surveillance systems. Nonetheless, in today's context, one must ask the question– when can we expect hierarchical analysis systems, such as the one presented in this dissertation, be expected to help?

In our experiments, we presented experiments on an audio retrieval task, with semantic queries. The MED11 dataset consisted exclusively of semantic queries, so differences between different learning approaches are harder to characterize. The data in BBC dataset are somewhat more varied, and gives us some insight into the differences between the different approaches using an analysis of the category-specific performance.

The 10 data categories in the BBC dataset can be characterized under 2 larger umbrellas. The first one can be thought of as *object concepts*, while the second one can be thought of as *abstract concepts*. Examples of the former include water, animals, birds, transport, while those of the latter include household, exterior atmospheres, interior backgrounds. By *object concepts*, we mean that these are tangible, tactile concepts, and while they may manifest in various acoustically different forms, they are clearly interpretable and not context-dependent. *Abstract concepts*, on the other hand, include sounds from a wide variety of objects, and can only be understood in context. For example, a recording

from the exterior atmospheres category is described as follows: *Open air swimming pool, with children splashing and shouting. Recording near London.* This recording represents an instance of an *abstract concept*, which contains sounds from various *object concepts*, such as water, children, etc. While individually they are instances of object concepts, the recording as a whole is an abstract concept (note the hierarchies!).

A category-wise performance of the *object concept* and *abstract concept* categories tells us that the baseline systems, such as the Vector Quantization based systems that map the acoustics directly to the semantic category associated with a recording do a much better job on the *object concept* categories than the *abstract concept* categories. This is not very surprising since the individual units are better at capturing acoustics than semantics, and the acoustics do not always generalize to the semantics. The hierarchical layers induced by our algorithm specifically attempt to capture information in the higher layers and thus do a significantly better job than VQ-based systems on the abstract concept categories. Since the object categories are fairly pure, in terms of the different concept types they contain, the higher layers do not capture significant meaningful information over what is already modeled by the lower level acoustic units.

Finally, we ask the question if there might be certain types of categories where such systems might do better than we expect humans to do. While this is not an easy question to answer, in the absence of extensive experiments with humans in comparison to learning algorithms, one can imagine a particular class of categories where this might be the case. This class would contain various categories or sets of concepts that humans are unfamiliar with. An example that springs to mind is that of identifying geographical location, given an audio stream, but not significant amounts of clear, discernible speech (when humans might be able to use knowledge of accents to perform well at the task), and a small training dataset. In such settings, automatic systems might be able to pick up on cues from certain kinds of sounds in the background that might be characteristic of specific geographical locations that humans do not typically look for. We note that recently published research has started to look specifically at the geo-tagging task Gottlieb et al. [2012].

7.3 Future Extensions of Semantic Analysis in Audio

Given the success of the various deep neural network approaches for speech recognition tasks, we expect that they will soon be applied to large-scale audio content analysis tasks. In accordance with the approach taken in this dissertation work, we expect that the community will focus not simply on the performance of such models on specific tasks, but also

on the units learned, their semantic coherence and import, as well, and that they will continue to focus not just in optimizing performance on standard tasks, but also on the larger vision of identifying semantics with a view to developing truly intelligent systems that can not only detect specific patterns of interest but also identify and understand them in context. Such systems would be extremely useful for a variety of tasks not only in web-based systems such as audio and music retrieval but also tasks in the field such as monitoring and surveillance of both public and private properties, monitoring health of large mechanical systems where the audio signal is one of the main indicators of the health of such systems and many others.

With respect to specific directions of future research, we identified some in Chapter 6 that we believe will be important future directions in developing more refined systems for semantic analysis of audio content. Each of them represents a significant undertaking in this research area, and a deeper understanding of any of those approaches should significantly advance our understanding in this domain.

In this dissertation, while we presented a few approaches to learning higher levels of the hierarchical framework, we believe that there are significant advances to be made still in the specific learning approaches employed. The message that we would encourage readers to take from this dissertation is the idea of mapping acoustics to semantics in a hierarchical manner. The framework, using the learning approaches presented in this dissertation, shows a great deal of promise for future research, and the specific models can be extended in various ways to take other assumptions that may be useful into account. The techniques presented in this dissertation, working in minimally supervised settings, show encouraging improvements over the current, commonly used approaches, and we believe that (along with semantic audio retrieval) they will prove useful in a number of tasks that could benefit from semantic analysis of audio content.

Bibliography

Bbc sound effects library original series, http://www.sound-ideas.com/bbc.html. 2.4.1

http://www.sound-ideas.com/artfoley.html. 2005. 2.4.2

http://majorminer.org/info/intro. 2007. 6.2

TRECVid 2010 Multimedia Event Detection Evaluation, 2010. URL `http://www.nist.gov/itl/iad/mig/med10.cfm,2010`. 3.2.3

http://www.nist.gov/itl/iad/mig/med11.cfm, 2011. 2.4.1

S. Abney. Parsing by chunks. *Principle-Based Parsing*, 1991. 2.2.1

B. Babenko, M. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Proceedings of CVPR*, 2009. 2.1.4

M. Bacchiani. Speech Recognition System Design Based on Automatically Derived Units. *PhD Thesis*, 1999. 3.1

J. Baker. Trainable grammars for speech recognition. *Speech communication papers presented at the 97th meeting of the Acoustical Society of America*, pages 547–550, 1979. 5.2

S. Becker, J. Bobin, and E.J. Candes. Nesta: A fast and accurate first-order method for sparse recovery. *SIAM Journal for Imaging Sciences*, 4:1–39, 2011. 6.1.1

Z. Ben-Haim and Y. C. Eldar. Near-oracle performance of greedy block-sparse estimation techniques from noisy measurements. *IEEE Journal of Selected Topics in Signal Processing*, 5:1032–1047, 2011. 6.1.1

J.F. Bernabeu, J. Calera-Rubio, and J.M. Iesta. Classifying Melodies Using Tree Grammars. In *Pattern Recognition and Image Analysis*, 2011. 2.1.1

S. Berrani, G. Manson, and P. Lechat. A non-supervised approach for repeated sequence detection in TV broadcast streams. In *Signal Processing: Image Communication*, volume 23, pages 525–537, 2008. 2.1

D. Bikel. A Distributional Analysis of a Lexicalized Statistical Parsing Model. In *Proceedings of Empirical Methods in Natural Language Processing*, 2004. 2.2.1

D.M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, pages 993–1022, 2003. 2.2.2

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, 1998. 2.1.4

T. Blumensath and M.E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14:629654, 2008. 6.1.2

A.S. Bregman. Auditory scene analysis. *International Encyclopedia of the Social and Behavioral Sciences.*, 2004. 6.1.1

L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001. 2.4.2

G. Carroll and E. Charniak. Two experiments on learning probabilistic dependency grammars from corpora. *Working Notes of the Workshop Statistically-Based NLP Techniques*, pages 1–13, 1992. 5.2

S.F. Chang, D. Ellis, W. Jiang, K. Lee, A. Yanagawa, A. Loui, and J. Luo. Large-scale multimodal semantic concept detection for consumer video. In *MIR workshop, ACM-Multimedia*, 2007. 2.1, 5.1, 6.1

E. Charniak. Statistical Techniques for Natural Language Parsing. *Artificial Intelligence*, 18, 1997. (document), 2.2.1, 2.2

S. Chaudhuri and B. Raj. Unsupervised structure discovery for semantic analysis of audio. In *Neural Information Processing Systems*, 2012. 4, 5.2, 6.1.4

S. Chaudhuri and B. Raj. Unsupervised Hierarchical Structure Induction For Deeper Semantic Analysis of Audio. In *Proceedings of ICASSP*, 2013. 5

S. Chaudhuri, M. Harvilla, and B. Raj. Unsupervised learning of acoustic unit descriptors for audio content representation and classification. In *Interspeech*, pages 717–720, 2011. 3.1.1, 5.1, 6.1, 6.1.1, 6.1.2

Y. Cho and L.K. Saul. Learning dictionaries of stable autoregressive models for audio scene analysis. In *International Conference on Machine Learning*, 2009. 6.1.1

H.I. Christensen, J. Matas, and J. Kittler. Using Grammars for Scene Interpretation. In *International Conference on Image Processing*, 1996. 2.1.2

J. Clarke and M. Lapata. Constraint-based Sentence Compression: An Integer Programming Approach. In *Proceedings of the Coling/ACL*, 2006. 3.2.3

K. Crammer and Y. Singer. Ultraconservative Online Algorithms for Multiclass Problems. *Journal of Machine Learning Research*, 2003. 3.2.2, 3.2.3

P.P. Cruz-Alcazar and E. Vidal. Two Grammatical Inference Applications in Music Processing. *Applied Artificial Intelligence*, 2008. 2.1.1

A. P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977. 2.3, 4.2.2

L. Deng, G.E. Hinton, and B. Kingsbury. New Types of Deep Neural Network Learning for Speech Recognition and Related Applications: An Overview. In *Proceedings of ICASSP*, 2013. 7

T.G. Dietterich, R.H. Lathrop, and T. Lozano-Perez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89:31–71, 1997. 2.1.4, 6.2, 6.2

C.H.Q. Ding, T. Li, and M. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Transactions on Pattern Analysis and Machine Intelligences*, 32:45–55, 2010. 6.1.1

A. Divakaran, K.A. Peker, R. Radharkishnan, Z. Xiong, and R. Cabasson. Video summarization using MPEG-7 motion activity and audio descriptors. *Video Mining*, 91, 2003. 2.1

Y. C. Eldar, P. Kuppinger, and H. Bolcskei. Block-sparse signals: Uncertainty relations and efficient recovery. *IEEE Transactions on Signal Processing*, 58:3042–3054, 2010. 6.1.1

D. Ellis. Predication-driven computational auditory scene analysis. *PhD Thesis*, 1996. 6.1.1

D. Elworthy. Does Baum-Welch re-estimation help taggers? In *Proceedings of the 4th Conference on Applied Natural Language Processing*, 1994. 2.1.4

P.F. Felzenszwalb and D. McAllester. Computer Science Technical Report-2010-02,, 2010. 2.1.2

P.F. Felzenszwalb, R. Girshick, and D. McAllester. Cascade Object Detection with Deformable Part Models. In *Proceedings of Computer Vision and Pattern Recognition Conference*, 2010. 2.1.2

S. Finch and N. Chater. Distributional bootstrapping: From word class to proto-sentence. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, 1994. 5.2

J. Foote. Content-based Retrieval of Music and Audio. In *Multimedia Storage and Archiving Systems*, 1997. 1.1

G. Friedland, L. Gottlieb, and A. Janin. Using Artistic Markers and Speaker Identification for Narrative-Theme Navigation of Seinfeld Episodes. In *Workshop on Content-Based Audio/Video Analysis for Novel TV Services, 11th IEEE International Symposium on Multimedia*, 2009. 2.1

X. Gabaix. Zipf's Law for Cities: An Explanation. *Quarterly Journal of Economics*, 114: 739–767, 1999. 3.1.1

R. Garg and R. Khandekar. Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. In *International Conference on Machine Learning*, 2009. 6.1.1

J. S. Garofolo, E. M. Voorhees, V. M. Stanford, and K. S. Jones. TREC-6 1997 Spoken Document Retrieval Track Overview and Results. In *Proceedings of TREC-6 Conference*, 1997. 2.1

S. Goldwater, T.L. Griffiths, and M. Johnson. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112:21–54, 2009. 4.1.1, 6.3.1

L. Gottlieb, J. Choi, G. Friedland, P. Kelm, and T. Sikora. Pushing the Limits of Mechanical Turk: Qualifying the Crowd for Video Geo-Location. In *Proceedings of the 2012 ACM international workshop on Crowdsourcing for Multimedia*, 2012. 7.2

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11, 2009. 6.2.1

P. Hamel and D. Eck. Learning Features from Music Audio with Deep Belief Networks, booktitle = Proceedings of the International Society for Music Information Retrieval conference. pages 339–344, 2011. 7

M. Hill, G. Hua, A. Natsev, J.R. Smith, L. Xie, B. Huang, M. Merler, H. Ouyang, and M. Zhou. IBM Research TRECVID-2010 Video Copy Detection and Multimedia Event Detection System. In *Proceedings of TRECVID*, 2010. 2.4.1, 3.2.2

P. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004. 6.1.1, 6.1.2

A. Hurmalainen, R. Saeidi, and T. Virtanen. Group sparsity for speaker identity discrimination in factorisation-based speech recognition. In *Proceedings of Interspeech*, 2012. 6.1.1

W. Jiang, C. Cotton, S.-F. Chang, D. Ellis, and A. Loui. Short-Term Audio-Visual Atoms

for Generic Video Concept Classification. In *Proceedings of ACM MultiMedia*, 2009. 2.1

M. Johnson and S. Goldwater. Improving nonparametric bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: North American Chapter of the Association for Computational Linguistics*, 2009. 4.1.1, 6.3.1

S. Kim, P.G. Georgiou, and S. Narayanan. Supervised acoustic topic model for unstructured audio information retrieval. In *Proceedings of Asia Pacific Signal and Information Processing Association (APSIPA) Annual Summit and Conference*, 2010. 3.2.3

D. Klein and C. Manning. Natural language grammar induction using a constituent-context model. In *Advances in Neural Information Processing Systems*, 2001. (document), 2.2.1, 5.2, 5.3, 5.2, 5.3.1, 5.3.1

D. Klein and C. Manning. A generative constituentcontext model for improved grammar induction. In *Proceedings of the Association for Computational Linguistics*, 2002. 5.2

D. Klein and C.D. Manning. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 2003. 2.2.1

K. Knight and D. Marcu. Statistics-based summarization - step one: Sentence compression. In *Proceedings of AAAI*, 2000. 3.2.3

J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of International Conference on Machine Learning*, 2001. 2.2.1

K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, pages 35–56, 1990. 5.2

C.H. Lee, F.K. Soong, and B.H. Juang. A segment model based approach to speech recognition. In *Proceedings of ICASSP*, 1988. 3.1.1

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Neural Information Processing Systems (NIPS)*, 2001. 6.1.1

H. Lee, Y. Largman, P. Pham, and A.Y. Ng. Unsupervised feature learning for audio classication using convolutional deep belief networks. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2009. 7

K. Lee and D. Ellis. Audio-Based Semantic Concept Classification for Consumer Video. *IEEE Transactions on Audio, Speech and Language Processing*, 18:1406–1416, 2010. 2.1, 2.1.1

H. Li, L. Bao, Z. Gao, A. Overwijk, W. Liu, L. Zhang, S. Yu, M. Chen, F. Metze, and A. Hauptmann. Informedia @ TRECVID2010. In *Proceedings of TRECVID*, 2010a. 2.4.1, 3.2.2

L.J. Li, H. Su, E.P. Xing, and F.F. Li. Object Bank: A High-Level Image Representation for Scene Classification and Semantic Feature Sparsification. In *Proceedings of NIPS*, 2010b. 2.1.2

L.J. Li, C. Wang, Y. Lim, D. Blei, and F.F. Li. Building and Using a Semantivisual Image Hierarchy. In *Proceedings of CVPR*, 2010c. 2.1.2

W. Li. Random Texts Exhibit Zipf's-Law-Like Word Frequency Distribution. *IEEE Transactions on Information Theory*, 38:1842–1845, 1992. 3.1.1

Z. Liu, J. Huang, Y. Wang, and T. Chen. Audio Feature Extraction and Analysis for Scene Classification. In *Workshop on Multimedia Signal Processing*, 1997. 2.1

X. Lu, Y. Tsao, S. Matsuda, and C. Hori. Speech Enhancement based on Deep Denoising Autoencoder. In *Proceedings of Interspeech*, 2013. 7

M. Mandel and D. Ellis. Multiple-instance learning for music information retrieval. In *Proceedings of ISMIR*, 2008. 2.1.4

R. McDonald. Discriminative Sentence Compression with Soft Syntactic Evidence. In *Proceedings of European Association for Computational Linguistics*, 2006. 3.2.3

M. McKinney and J. Breebaart. Features for Audio and Music Classification. In *International Symposium on Music Information Retrieval*, 2003. 2.4.2

A. Mesaros, T. Heittola, and T. Virtanen. Latent semantic analysis in sound event detection. In *Proceedings of the European Signal Processing Conference*, 2011. 6.1.1

D. Mochihashi, T. Yamada, and N. Ueda. Bayesian unsupervised word segmentation with nested pitman-yor language modeling. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics*, 2009. 4.1.1, 6.3.1, 6.3.1

H.H. Nagel. From image sequences towards conceptual descriptions. *Image and Vision Computing*, 6:59–74, 1988. 2.1.2

D. Needell and J.A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 2009. 6.1.1, 6.1.2, 6.1.3

K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of 9th International Conference on Information and Knowledge Manage-*

ment, 2000. 2.1.4

D. Pallett, J. Fiscus, J. Garofolo, and M. Przybocki. Hub-4 'dry run' broadcast materials benchmark tests. In *Speech Recognition Workshop*, 1996. 2.4.2

S. Pancoast and M. Akbacak. Bag-of-audio-words approach for multimedia event classification. In *Interspeech*, 2011. 5.1, 6.1.1, 6.1.2, 6.1.3

J. Paulus and A. Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech and Language Processing*, 2009. 2.1.1

X.H. Phan. CRFTagger: CRF English POS Tagger, 2006. URL `http://crftagger.sourceforge.net/,2006`. 2.2.1

J. Pitman and M. Yor. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900, 1997. 6.3.1

H. Poon, C. Cherry, and K. Toutanova. Unsupervised morphological segmentation with log-linear models. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics*, 2009. 4.1.1, 6.3.1

X. Qi and B.D. Davison. Web page classification: Features and algorithms. *ACM Computing Surveys*, 41, 2009. 2.4.2

L. Rabiner and B. Juang. An introduction to hidden Markov models . *ASSP Magazine, IEEE*, 3:4–16, 1986. 2.1.4, 2.2.1

B. Raj, T. Virtanen, S. Chaudhuri, and R. Singh. Non-negative matrix factorization based compensation of music for automatic speech recognition. In *Proceedings of Interspeech*, 2010. 6.1.2

L. Ramshaw and M. Marcus. Text chunking using transformation based learning. In *Proceedings of the 3rd workshop on very large corpora*, 1995. 2.2.1

C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *Proceedings of the 7th IEEE Workshop on Applications of Computer Vision*, 1995. 2.1.4

Y. Rui, A. Gupta, and A. Acero. Automatically extracting highlights for tv baseball programs. In *Proceedings of the eighth ACM international conference on Multimedia*, 2000. 2.1

T. N. Sainath, B. Ramabhadran, D. Nahamoo, D. Kanevsky, and A. Sethy. Sparse representation features for speech recognition. In *Proceedings of Interspeech*, 2010. 6.1.2

T. N. Sainath, D. Nahamoo, D. Kanevsky, B. Ramabhadran, and P. M. Shah. Exemplar-based sparse representation phone identification features. In *Proceedings of ICASSP*, 2011. 6.1.2

M. Schuster and K. Nakajima. Japanese and korean voice search. In *Proceedings of ICASSP*, 2012. 4.1.1, 6.3.1

H. Schutze. Distributional part-of-speech tagging. In *Proceedings of the European Association for Computational Linguistics*, 1995. 5.2

B. Settles. Active Learning Literature survey, 1995. 2.1.4

R. Singh, B. Raj, and R. Stern. Automatic Generation of sub-word units for Speech Recognition Systems. *IEEE Transactions on Speech and Audio Processing*, 2002. 3.1

M.-h. Siu, H. Gish, S. Lowe, and A. Chan. Unsupervised Audio Patterns Discovery using HMM-based Self-Organized Units. In *Interspeech*, pages 2333 – 2336, 2011. 3.1.1

M. Slaney. Mixture of probability experts for audio retrieval and indexing. In *ICME*, 2002. 2.1, 5.1, 6.1

M.J. Steedman. A Generative Grammar for Jazz Chord Sequences. *Music Perception*, 1989. 2.1.1

I. Szoke, P. Schwarz, and P. Matejka. Comparison of keyword spotting approaches for informal continuous speech. In *Proceedings of Eurospeech*, 2005. 2.1

Y.W. Teh. A hierarchical bayesian language model based on pitmanyor processes. In *Proceedings of the 44th annual meeting of the Association for Computational Linguistics*, 2006. 6.3.1

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1994. 6.1.2

K. Toutanova and C.D. Manning. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 2000. 2.2.1

K. Toutanova, D. Klein, C.D. Manning, and Y. Singer. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL*, 2003. 2.2.1

G. Tzanetakis and P. Cook. MARSYAS: A framework for Audio Analysis. *Organised Sound*, 4:169–175, 1999. 2.1.3

A. Velivelli, C. Zhai, and T.S. Huang. Audio segment retrieval using a short duration example query. In *Proceedings of International Conference on Multimedia and Expo*,

2004. 2.1

O. Walter, R. Haeb-Umbach, S. Chaudhuri, and B. Raj. Unsupervised word discovery from phonetic input using nested pitman-yor language modeling. In *Proceedings of the Autonomous Learning Workshop, IEEE International Conference on Robotics and Automation*, 2013. 6.3.1, 6.3.2

E. Wold, T. Blum, D. Keislar, and J. Wheaton. Content-based Classification, Search and Retrieval of Audio. In *IEEE Multimedia Magazine*, 1996. 1.1, 2.1

M. Yaghoobi, T. Blumensath, and M.E. Davies. Dictionary learning for sparse approximations with the majorization method. *IEEE Transactions on Signal Processing*, 57: 2178–2191, 2009. 6.1.1

D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 1995. 2.1.4

Z.H. Zhou. Multi-instance learning: A survey, 2004. 2.1.4

X. Zhu. Semi-supervised learning literature survey, 2004. 2.1.3, 2.1.4

X. Zhuang, S. Tsakalidis, S. Wu, P. Natarajan, R. Prasad, and P. Natarajan. Compact audio representation for event detection in consumer media. In *Interspeech*, 2011. 5.1, 6.1, 6.1.2, 6.1.3

G. Zipf. *Selective Studies and the Principle of Relative Frequency in Language*. MIT Press, 1932. 3.1.1