Grounding Language in Events

by

Michael Ben Fleischman

B.A. Philosophy, Psychology, Columbia University, 1999
M.S. Computational Linguistics, University of Southern California, 2002

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2008

Author……..…….……
Department of Electrical Engineering and Computer Science
May 23, 2008

Certified by…..…….……
Deb Roy
Associate Professor of Media Arts and Sciences
Thesis Supervisor

Accepted by…..…….……..
Terry P. Orlando
Chairman, Department Committee on Graduate Students

Grounding Language in Events

By

Michael Ben Fleischman

Submitted to the Program in Electrical Engineering and Computer Science
on May 23rd, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Broadcast video and virtual environments are just two of the growing number of domains in which language is embedded in multiple modalities of rich non-linguistic information. Applications for such multimodal domains are often based on traditional natural language processing techniques that ignore the connection between words and the non-linguistic context in which they are used. This thesis describes a methodology for representing these connections in models which ground the meaning of words in representations of events. Incorporating these *grounded language models* with text-based techniques significantly improves the performance of three multimodal applications: natural language understanding in videogames, sports video search and automatic speech recognition.

Two approaches to representing the structure of events are presented and used to model the meaning of words. In the domain of virtual game worlds, a hand-designed hierarchical *behavior grammar* is used to explicitly represent all the various actions that an agent can take in a virtual world. This grammar is used to interpret events by parsing sequences of observed actions in order to generate hierarchical event structures. In the noisier and more open-ended domain of broadcast sports video, hierarchical temporal patterns are automatically mined from large corpora of unlabeled video data. The structure of events in video is represented by vectors of these hierarchical patterns.

Grounded language models are encoded using Hierarchical Bayesian models to represent the probability of words given elements of these event structures. These grounded language models are used to incorporate non-linguistic information into text-based approaches to multimodal applications. In the virtual game domain, this non-linguistic information improves natural language understanding for a virtual agent by nearly 10% and cuts in half the negative effects of noise caused by automatic speech recognition. For broadcast video of baseball and American football, video search systems that incorporate grounded language models are shown to perform up to 33% better than text-based systems. Further, systems for recognizing speech in baseball video that use grounded language models show 25% greater word accuracy than traditional systems.

Thesis Supervisor: Deb Roy
Title: Associate Professor of Media Arts and Sciences

Acknowledgements

I would like to start by thanking my advisor, Deb Roy, for his support and guidance over these last five years. Deb has given me both the direction I needed to succeed, and the freedom I wanted to explore new ideas (even when he wasn't fully convinced they would work). I have come to believe that Deb can do anything he sets his mind to, and I am looking forward to continuing to learn from him in the years to come.

I would also like to thank my committee members Eric Grimson and Jim Glass, both of whom were extremely generous with their time and always glad to give me advice on what I needed to do, and how I could actually do it. I could not have chosen a better committee in Eric and Jim, not only because of their expertise, but also because of their patience and kindness.

There are a number of professors from whom I have had the privilege to learn since I started studying computer science. Regina Barzilay, Mike Collins, Roz Picard, and Cynthia Breazeal have helped make my MIT experience rich and memorable. At USC's Information Sciences Institute I worked with an amazing group of people including Kevin Knight, Daniel Marcu, Chin-Yew Lin, and Yolanda Gil.

But I would not be here today if it were not for my first true mentor, Ed Hovy. Ed took a chance on me when I was first beginning to study Artificial Intelligence. He taught me how to do research, and more importantly, how to do research with other people.

Perhaps the best thing about being a graduate student at MIT is the UROPs. If it were not for my UROPs, I would still not be finished with this thesis. They put in an incredible effort (often putting me to shame). Humberto Evans, Stephen Oney, Yuan Wei, Victor Wang, Daniel Gerber… thank you for all the help and for tackling some pretty grueling tasks.

The Cognitive Machines group is the kind of group that I had dreamed about finding when I first applied to MIT. Passionate and interested, we have spent many hours arguing about everything form philosophy to psychology and from linux to LaTeX. Rony Kubat, Branden Roy, Phil DeCamp, Jeff Orkin, Stephanie Tellex, Kai-yuh Hsiao, Peter Gorniak, Nick Mavridis, Andre Robeiero, Ian Eslick, Philipp Robel, Leo Tsourides, Soroush Vosoughi, Michael Levit, Jethran Guiness, Amber Frid-Jimenez, Josh Juster, Chris Lucas, Yair Gitzah, Tamara Harris, Matsumi Sullivan and Tanya Schalchlin… without you guys I would not have been able to

make it through. I cannot thank you enough for the time you put in helping me, especially in these last few months leading up to my thesis. You have all made my time here at the Lab incredible and I am proud to call you friends.

While in graduate school, I have had a unique opportunity to travel the world in order to attend scientific conferences. This has been perhaps the greatest part of my education, not because of what I have seen, but because of who I have seen it with. For this, I would like to thank my friends: Deepak Ravichandran (my officemate for life), Hal Daume, Patrick Pantel, Alex Frasier, Jimmy Lin, Elena Filatova, Jon Henderson, Michelle Banko, and Nicola Stokes. Thinking of you always puts a smile on my face.

I am also extremely fortunate to have the support and friendship of some truly great people. Adam Checchi, Eli Wainman, Will Uyesugi, Chi-Chien Hou, Uri Nachshon, Damien Clark, Eben Wight, Ethan Wainman, David Asarnow, Jeff Sherwin, Felix Gillette, Carli Paine and Aristotle Socrates. Not everyone is lucky enough to have friends they have known for so long. I look forward to celebrating with each of you soon.

From every member of my family, I have received nothing but support and encouragement throughout the many (many) years that I have been in school. I want to thank my Aunts Judy, Jill, Nechama, and Marika; my Uncles Max, Vlado and Shi; my cousins, Doron, Sharon, Robin, and Nancy; Andy, Sammy, Elvis and Oscar; and my Grandma, Kayla, who is an inspiration to us all.

And a special thanks to my big sister, Shelli, who I still aspire to be like (even though she tortures me for it); to my Mother, Vera, who is the reason I have gotten this far (even though she may not believe it); and to my Father, Jerry, who has taught me the meaning of life (even though I would never admit it).

And finally, to Allison, who has stood by me, who has put up with me, and who makes every day better…. just by being with me. Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Creating machines that can understand natural language has been a goal of Artificial Intelligence (AI) since its inception. However, even today's most successful search engines and speech recognizers have only a partial knowledge of the meaning of words. While they may contain sophisticated models of how words relate to each other, these systems are ignorant of how these words relate to the non-linguistic world. For applications that operate only on text, such information may not be necessary for building successful systems. But for the growing number of applications where language is embedded within rich non-linguistic context (e.g., video search), exploiting the connection between words and the world has significant advantages.

This thesis describes a methodology for representing these connections in order to model the meaning of words for events. Understanding events in a contextually rich domain is extremely challenging for an automatic system. Events cannot be described in purely perceptual terms, but rather, unfold over time, are hierarchical, and can only be interpreted with knowledge of the larger context in which they occur. In this thesis, two approaches to representing the structure of events are presented and used to model the meaning of words.

In the domain of virtual environments, a hand-designed hierarchical *behavior grammar* is used to explicitly represent all the various actions that an agent can take in a virtual world. This grammar is used to interpret events by parsing sequences of observed actions in order to generate hierarchical event structures. This top-down approach to representing events is contrasted with a bottom-up method designed for the noisier and more open-ended domain of broadcast video. In this approach, hierarchical temporal patterns are automatically mined from large corpora of unlabeled video data. The structure of events in video are then represented as vectors of these hierarchical patterns.

A probabilistic model of meaning, called a *grounded language model*, is generated by associating words with elements of these event structures. These grounded language models are used to incorporate non-linguistic information into text-based approaches to multimodal applications. In the virtual domain, this non-linguistic information improves natural language understanding for a virtual agent by nearly 10% and cuts in half the negative effects of noise caused by automatic speech recognition. For broadcast video, video search systems that incorporate grounded language models are shown to perform 33% better than text-based systems. Further, systems for recognizing speech in video that use grounded language models show 25% greater word accuracy than traditional systems.

The contributions of this thesis are twofold: first, methodologies for representing events are introduced that are rich enough for grounding language use in contextually rich domains; second, incorporating these grounded language models with text-based techniques is shown to significantly improve performance for a variety of multimodal applications.

1.1 Challenge of Events

In order for grounded language models to support multimodal applications in which language is embedded within a rich context, event representations must be designed that are both powerful enough to support language use, and robust enough to operate in noisy domains (such as broadcast video). The nature of events, however, makes designing such representations extremely challenging, particularly for the purposes of grounding language.

Unlike most properties and objects, events are temporal entities that defy description in purely perceptual terms. While colors and shapes are largely defined by their immediate impact on the senses, events unfold over time and generate no persistent sensory impressions. This temporal characteristic gives events a hierarchical internal structure; such that, individual events are decomposable into sequences of lower level sub-events (e.g. *entering a room* requires *unlocking the door*, *opening the door*, *walking through the door*, etc.) Whereas objects can be decomposed into observable sub-objects (e.g. a tree is composed of a trunk, branches, leaves, etc.), the hierarchical structure of events is largely hidden and highly context dependent. So, while one often *opens a door* in order to *enter a room*, in other contexts *opening a door* may have very different implications (e.g. *asking someone to leave*, *inviting someone in*, *ventilating the room*, etc.).

The hierarchical and temporal nature of events makes them difficult to interpret in isolation and without knowledge of the larger context in which they occur. Modeling this contextual knowledge, however, can quickly become computationally infeasible, particularly in extremely noisy domains such as broadcast video (where identifying even the lowest level events is unreliable). In order to get a wedge into this extremely challenging problem, this thesis follows a strategy, first introduced in the philosophy of language, which focuses on the interpretation of events in games.

1.2 Language Games

Long before the advent of computer science, philosophers of language also struggled with ways to analyze the meaning of words. In an attempt to examine the functional aspects of words, one such philosopher suggested analyzing meaning by focusing on the simple linguistic interactions that surround constrained tasks (Wittgenstein, 1954). These simple interactions, which he called language games, have the advantage of representing real-world linguistic phenomena, while abstracting away many of the complexities that obscure more detailed analysis. Focusing on such language games gives the philosopher a more manageable means of accessing the complexities of language use. Further, by incrementally increasing the complexity of the language games under examination, the philosopher can develop an increasingly rich model of the meaning of words.

This thesis takes inspiration from this philosophical approach in order to tackle the complexities of representing events for grounded language models. However, while philosophers have largely focused on imaginary interactions between people in contrived tasks (e.g. builders passing each other blocks), this thesis takes the notion of games more literally. In this work, language is modeled using actual games played by real people. We focus on two types of games from very different domains: immersive videogames and broadcast video of professional sports.

1.2.1 Videogames

Videogames provide an ideal environment for examining how to design representations rich enough for language grounding. The current state of the art in virtual environments allow for the creation of exceptionally rich virtual worlds in which human participants interact through the use of virtual characters. These virtual environments offer a very high level of control to designers both in their ability to generate tasks with controllable complexity, and to record the behaviors of their subjects. Further, without the need for any computer vision technology, all actions taken by subjects in a virtual world can be automatically recorded with no perceptual noise.

Most importantly, however, videogames make it possible to explicitly encode all the various actions that an agent can take in a virtual world. As designers, one is given unique access not only to the rules of the games that are created, but also to the deterministic effects of actions on the environment (e.g. mouse-clicking a door makes the door open). By encoding these rules and actions, the context of the game can be fully modeled and rich representations of events can be generated for language grounding.

Unfortunately, many of the advantages of virtual worlds disappear when moving to the domain of broadcast video. Unlike videogames, events in broadcast video are more open-ended and not constrained by the limitations of a virtual world. However, by continuing our literal interpretation of the philosophical method, and focusing on video of humans playing actual games (i.e. sports), different but analogous advantages can be exploited.

1.2.2 Broadcast Sports Video

Even in the domain of sports video, computer vision technology has not been developed to accurately represent events. The ability to identify objects (such as players, balls, etc.) and track their movements is an open problem and the subject of much research. However, focusing on the domain of sports video does afford a number of advantages in terms of visual processing algorithms.

First, the separation of sports games into plays (e.g., "at bats," "downs," "possessions," etc.) gives each game a defined structure and simplifies the segmentation of events from continuous video. Further, the use of similar playing fields and camera angles greatly simplifies the identification of characteristic shots (even across different teams and stadiums) that can be used as the basis for representing actions in a game.

Finally, although many types of events occur in any particular sport, most everything that happens in a game conforms to a strict set of rules of conduct. Thus, although varied and nondeterministic, the events that occur within a game often follow stable patterns (e.g., a double play usually involves a hit, followed by a catch, followed by a throw, etc.). Data mining algorithms can be used to automatically learn these patterns directly from data in order to build up representations of events. By building up more and more complex patterns, robust representations of context can be learned without the need for explicit encoding (as was done in the virtual world).

In addition to its advantages for representing events, sports video offers a number of other practical advantages for learning grounded language models. In terms of size, the popularity and frequent schedule of sports games make it easy to collect the large datasets necessary for learning grounded language models. Over the course of a typical season hundreds of hours of sports video can easily be recorded from broadcast television. These games are particularly well suited for modeling language learning because they are always shown with running commentary by announcers who provide play-by-play descriptions that correlate with the events in the video.[1] Further, this commentary is almost always presented in closed captioning,

[1] Although this correlation is not perfect (sometimes the announcers discuss events in the past, future, or on different topics), there is enough regularity in the data such that, as the amount of data increases, significant patterns can be detected.

thus alleviating the need for human transcription of speech.[2] Finally, any progress in the automatic analysis of sports video is likely to have practical and commercial relevance as evidenced by its wide spread popularity, the existence of multi-billion dollar related businesses, and the large body of academic research that focuses on the topic.

The next section describes our methodology for representing these events and how we train grounded language models on such data.

1.3 Grounded Language Models

Grounded language models represent the relationship between words and the non-linguistic context in which they are used. Our methodology for learning grounded language models operates in two phases: event recognition and linguistic mapping. During event recognition, we generate representations to encode the non-linguistic context that surrounds language use. Then, during linguistic mapping, we model the association between these event representations and the words that are uttered as they occur[3]. We outline this two phase approach below, highlighting the differences between the top-down approach used to represent events in videogames and the bottom-up approach used for sports video.

1.3.1 Event Recognition

As described above, representations of events are dependent on the larger situational context for proper interpretation. For applications in virtual game worlds, we make the assumption that all events in a videogame stem from a generative model of behavior that encodes the strategies and goals of the players of the game. We call this generative model a *behavior grammar* (Miller et al., 1960) and formalize it using a hand-designed probabilistic context free grammar (PCFG). Individual events are represented as hierarchical structures which are

[2] Closed captioning is a service provided by broadcasters primarily for hearing impaired viewers, in which human transcriptions of what is being said is embedded in the video stream. These transcriptions can be displayed during viewing or extracted to a text file for later processing.

[3] This thesis is concerned primarily with descriptive and directive utterances. Other speech acts (e.g. questions) may not ground out in co-occurring context. However, no limitations are placed on the types of utterances used to train grounded language models, and it is left to the system to learn which utterances ground out in context and which do not.

generated by parsing sequences of the players' actions; just as syntactic structures are generated by parsing sequences of words in a sentence. Although this top-down approach has many interesting properties (such as its ability to model the asymmetry in noun and verb learning), it is best suited to deterministic domains with very little noise. In more complex domains, data-driven methods become more effective.

To represent events in video, we introduce a bottom-up approach in which the structure of events is automatically learned from large corpora of unannotated video data. This approach operates over noisy features extracted from multiple video streams. It uses temporal data mining to automatically discover hierarchical temporal patterns that exist within low level features of a video. These hierarchical patterns form a codebook, which acts as an analogue to the behavior grammars described above. Like the behavior grammar, the codebook is a representation of the larger situational context of a game. However, the temporal patterns in the codebook are more flexible than the rigid production rules that make up a behavior grammar. By using these more robust temporal patterns, and learning them directly from data, we produce event representations that are effective for video applications.

1.3.2 Linguistic Mapping

Grounded language models represent the association between words and the non-linguistic context in which they are used. In this thesis, models encode this association using conditional probability distributions of the likelihood of a word being uttered given a representation of its non-linguistic context. These probability distributions are trained using unsupervised statistical techniques that exploit large corpora of event representations paired with the words uttered during those events. By encoding grounded language models in this way, a wide variety of previous work on designing and training probabilistic models can be exploited (e.g., work on Machine Translation models and Hierarchical Bayesian models). Further, the use of probability distributions enables grounded language models to be easily applied to nearly any application with a probabilistic framework (particularly those that use traditional text-based language models). In this thesis, we focus on how context improves performance on three such applications: natural language understanding, video search, and speech recognition.

1.4 Applications

There are a growing number of application domains in which language is embedded in other modalities that contain rich non-linguistic information. Often times such multimodal applications are based on traditional natural language processing techniques that are designed to exploit information only from text. In video search applications, for example, many approaches (both academic and commercial) simply apply techniques for finding documents on the internet, to the transcriptions of speech in a video. Such approaches become easily confused, however, by situations in which people talk about things that are not actually occurring. For example, searching for a "home run" in baseball video using only transcripts of speech often returns false positive results where people are talking about a home run that happened in the past (or might happen in the future).

A central focus of this thesis is to explore how grounded language models can improve such systems by incorporating non-linguistic information into traditionally text-based techniques. We evaluate these models (and the event representations on which they are based) using three applications: natural language understanding, video search, and speech recognition.

Natural language understanding is evaluated in the context of virtual agents interacting with a human in a virtual game world. Grounded language models are used to convert linguistic input into semantic frame representations, and behavior grammars are used to bias understanding towards representations that fit the context of the game. Evaluations show that systems which incorporate context perform nearly 10% better than systems without context and further reduce the error caused by noisy automatic speech recognition by 50%.

Video search is evaluated on video of broadcast baseball and American football games. A grounded language model is used to extend a text-based language modeling approach, combining information from the audio/visual stream of video with speech transcriptions. This non-linguistic context helps avoid the false positive results inherent in using text based methods (as described above). Evaluations show that systems which incorporate grounded language models are over 33% more precise than the traditional text only approach.

Finally, automatic recognition of speech is evaluated on video of broadcast baseball games. Grounded language models are combined with text-based unigram, bigram, and trigram language models to bias recognition towards phrases that are related to the events that occur

during an utterance. (e.g., the occurrence of a ground ball during an utterance increases the probability of the bigram "ground ball"). Evaluations show that a speech recognition system that uses grounded language models has 25% greater word accuracy than a traditional system.

1.5 Roadmap

In the remainder of this thesis is organized as follows:

- In Chapter 2, previous work related to this thesis is described.
- In Chapter 3, a top-down approach to event representation is described. Grammars of behavior are introduced, and their design and implementation is presented.
- In Chapter 4, grounded language models based on top-down event representations are evaluated on a natural language understanding task in two virtual environments: a multiplayer online videogame and a military training simulation.
- In Chapter 5, a bottom-up approach to event representation is described. The approach uses temporal data mining to automatically learn event structure from unlabeled video data.
- In Chapter 6, Grounded language models are evaluated using a sports video search application.
- In Chapter 7, Grounded language models are evaluated on the task of automatically recognizing speech in sports video.
- In Chapter 8, we discuss the contributions of this work and implications for future directions.

Chapter 2

Background

The work in this thesis touches on many areas of computer science. This chapter focuses on work related to grounded language models, and the applications to which they are applied in the thesis. Other related topics such as such as image classification, developmental psychology, machine translation, etc. will be discussed as they are introduced within the following chapters.

2.1 Symbolic Models of Meaning

Traditional approaches to computational semantics define the meaning of a word strictly in terms of its relationship to other words, or word-like symbols. These symbolic models can be categorized based on the purpose and manner in which these relationships are discovered. Much of the initial work in this area comes from researchers in computational linguistics and knowledge representation who sought to build more intelligent machines by formalizing all linguistic knowledge. These early efforts depended on the use of human experts to encode semantic relationships using a variety of techniques including semantic networks (Quillian,

1967), formal logics (Lenat, 1995), semantic lexicons (Pustejovsky, 1991) and ontologies (Miller et al., 1990). In general, designing such models is a very time consuming process, taking many years to complete, and with often controversial claims regarding the completeness and applicability of the final output.

Researchers in cognitive psychology also focus on designing symbolic models of meaning, although with very different goals and techniques than those above. These cognitive models focus on how children develop their linguistic abilities, and thus, are models that learn from data meant to mimic what children actually observe during development. Such learning models take many forms, such as neural networks and Bayesian models (for a review, see Regier, 2003). However, the vast majority of this work operates only on human coded representations of what children actually observe (e.g. the symbol 'CAT' for a real world cat), and thus, also require a great deal of human effort to create. Further, because they are designed strictly to model human word learning, their applicability to real world tasks is highly limited.

More recently, the fields of cognitive psychology and computational linguistics have converged on methods in which symbolic models of meaning are learned automatically from large corpora of text data. These models define the meaning of a word entirely in terms of its correlation with other words that are used in similar linguistic contexts. Thus, the meaning of the word "cat" is not defined by its relation to the symbol 'CAT', but rather, as a correlation with other words, such as "dog," "fur," "milk," etc. Such correlations are learned using a variety of statistical techniques (e.g. Latent Semantic Analysis: Landaur et al., 1997; Latent Dirichlet Allocation: Blei et al., 2003) from large corpora of text documents (news articles, etc.) without the need for human expertise. These models not only correspond well with various aspects of human language development (Landaur, 1997), but also are useful in text-based real world applications such as document retrieval (Deerwester et al., 1990).

Although research into symbolic models of meaning has produced many important results, such models are limited both in their ability to model psychological phenomena, as well as, in their applicability to real world tasks. By abstracting away to a world made up entirely of symbols, cognitive models of word learning ignore a key element of semantics, the relationship between words and the environment of the language user (Harnad, 1990). Similarly, by defining the meaning of a word strictly in terms of other words, the applicability of symbolic models is limited to those tasks that operate only on words (e.g., document retrieval, text

summarization, text-to-text machine translation, etc.). For more realistic models of word learning, as well as, for applications situated in the real world (e.g., robotics, video processing, multiplayer videogames), a model of meaning is required that is grounded in the physical environment.

2.2 Grounded Models of Meaning

Recent work in the cognitive sciences addresses the limitations of symbolic models of meaning by defining the meaning of a word not strictly in terms of other words (or word-like symbols), but rather by *grounding* the meaning of a word in representations of the context in which they are used (for review see Roy, 2005, and Roy and Reiter, 2005). Importantly, these representations of context are generated, not by some human in the loop, but rather, are automatically constructed by the machine based on its perception of the environment.

The form of these representations varies dramatically depending upon the class of words to be grounded and the communication task at hand. In Roy (2002), a grounded model of meaning was learned for words that describe simple shapes and colors in order to support a conversational robot. The robot existed in a shared environment with the human user; it heard words through a microphone and could see the world using a camera. The model learned to ground words for colors and shapes into representations output by a computer vision system using a mutual information technique that exploited recurrent patterns in the speech and video stream. In other words, because people talked about colors and shapes more often in the presence of those same colors and shapes, a grounded model of meaning could be learned to support human interaction.

Although the majority of work on language grounding has focused on more concrete words, such as words for objects and colors (Roy, 2005; Yu et al., 2003), there have been some efforts to ground words for simple movements. This work has focused primarily on grounding event terms in representations of the physical environment. Fern, Givan, and Siskind (2002) describe a system which grounds words for simple physical actions. They use temporal logic representations to encode the force dynamic relationships that change between objects as they

are moved (Siskind, 2001). Thus, "stacking" involves the forces of one object on another, while "lifting" involves removing the force of one object using another, etc.

Bailey (1997) describes a very different method of representation to ground the meaning of low level physical movements (e.g. "push" and "pull"). He represents such movements using the control system of a simulated arm, such that movements are represented by the values of parameters that control forces and joint angles of that arm performing that action. Narayanan (1999) describe how similar underlying representations, called x-schemas, can also be used to understand more metaphorical uses of words for physical motion. By mapping domains such as the economy into representations of physical movement, he develops a system for interpreting phrases from news articles, such as "the rising economy."

2.3 Representing Events

In addition to work on grounding language, a great deal of research in AI has examined ways to represent events. Seminal work by Minsky (1974) and Schank & Abelson (1977) introduced the idea of frames and scripts, respectively, as formalizations of stereotyped information and episodic knowledge about events. Early work on automatic planning systems in AI model events using STRIPS representations, which encode the preconditions and post conditions of actions and the goals that they achieve (Russell and Norvig, 1995). More recently, probabilistic models, such as partially observable Markov decision processes (POMDP), have been proposed to account for the uncertainty that exists when planning in noisy environments (Kaelbling et al., 1998).

Probabilistic methods have also been examined for the inverse planning problem, i.e., recognizing the plans of others. Charniak and Goldman, (1993) describe a plan recognition system based on Bayesian belief networks that is used to understand natural language stories. Pynadath (1999) uses probabilistic context free grammars (PCFG) to recognize events for applications such as highway traffic modeling. Baker et al. (2005) use a Bayesian model to infer the intentions behind an agent's action in a cognitive model of human action understanding. Ivanov and Bobick (2000) recognize events in video (e.g., conducting music) by combining

PCFGs with lower level hidden Markov models (HMMs) to capture both observational uncertainty , as well as, the hierarchical nature of complex events.

Hongeng and Nevatia (2001) and Intille and Bobick (2001) combine low level probabilistic models with higher level temporal constraints to recognize multi-agent events in video. Hongeng and Nevatia (2001) combine HMMs with temporal logic in order to model events in surveillance video (such as the theft of a briefcase). Similarly, Intille and Bobick (2001), insert temporal constraints into Bayesian belief networks to model events in American football (based on manually identified player tracks).

A great deal of additional research has examined representing events specifically in sports video. Li and Sezan (2001) use an HMM to identify "plays" in a baseball game based on visual features that correlate with characteristic scenes of the pitcher. Rui et al. (2000) focus on audio features (e.g., pitch, amplitude, etc.) and use an HMM to recognize highlight events in baseball video. Gong et al. (2004) use a discriminative classification approach to model specific types of events (e.g. home runs, strikeouts, etc.). Events are represented as vectors of features in which each feature corresponds to visual or audio information from individual frames of the video.

2.4 Cross-Modal Learning

There exists a large body of research on algorithms for learning relations across information in multiple modalities. In the domain of word learning, Siskind (1996) describes a model that uses deductive inference in order to find mappings between words and formal logic representations of meaning. Roy (2003) examines word learning in more natural environments, looking at speech and visual data of mothers and children at play. Mutual information is used to measure the independence of visual and acoustic features in order to learn mappings between clusters of phonemes and representations of shapes and colors. Coen (2006) introduces a technique to cluster phonemes based on acoustic and visual features related to the movement of speakers' mouths. His multi-modal clustering algorithm clusters phonemes based on their inter-modal similarity as well as temporal correlations across modalities.

Yu and Ballard (2003) examine the effects of speakers' focus of attention on learning word-object mappings. They use algorithms from Machine Translation, specifically IBM model 1

(Brown et al., 1993) to estimate conditional probability distributions of words given hand coded representations of visible objects. Barnard et al. (2003), also use Machine Translation algorithms for word learning. They focus on a corpus of photographs manually annotated with words describing their contents, and automatically extract visual "blob" features from the images. They also introduce a Hierarchical Bayesian algorithm which extends Latent Dirichlet Allocation for multi-modal data (Blei and Jordan, 2003). This model is similar to models introduced for text processing which encode the relationship between words in a document and their authors (Steyvers et al., 2004).

2.5 Applications

2.5.1 Natural Language Understanding

Natural language understanding describes the task of automatically converting utterances of natural language into computer readable semantic formalizations. While early work in this area used hand designed rules to understand language (e.g. Winograd, 1972), more recent work examines how such language understanding systems can be learned automatically.

Recent work on learning natural language interfaces to databases falls within this category. Such natural language interfaces treat the translation of utterances into SQL queries as a problem of semantic parsing. Using training sets of utterances annotated with semantic representations, machine learning techniques are used to train parsers that infer semantic representations from well-formed utterances (e.g., Zettlemoyer and Collins, 2005; Ge and Mooney, 2005; Epstein, 1996).

In addition to such interfaces, a growing body of work focusing on semantic tagging has employed learning techniques. In such work, large corpora of annotated text are annotated by hand with information about the semantic roles of various words or phrases within sentences (Baker et al., 1998; Palmer et al., 2005). These corpora are then used to train semantic taggers to identify and classify these roles using linguistic and syntactic information extracted from the sentence (Gildea and Jurafsky, 2002; Fleischman et al., 2003).

Finally, there has been some recent work looking specifically at natural language understanding in virtual environments. Bhagat et al. (2005) learn a probabilistic model to select the most likely frame elements given a natural language utterance. This shallow model uses little training data and operates on spoken language (unlike the previous approaches which assume well formed sentences). Gorniak and Roy (2005) present a more sophisticated model of language understanding based on plan recognition and a manually designed lexicon of word to meaning mappings. Although not a learning methodology, their approach is novel in its use of information about the non-linguistic context of a virtual world in order to improve speech recognition.

2.5.2 Multimodal Information Retrieval

Research on multimodal information retrieval (IR) focuses on designing systems that enable searching for video and/or images from large corpora. The majority of work on multimedia IR can be classified into one of two approaches: supervised and unsupervised. Supervised approaches are particularly popular for video IR, and generally operate by first extracting individual frames from the video which are then classified into a number of pre-defined categories (called concepts) based on low level color and texture features (for a review, see Worring and Snoek, 2005). A user's query is then translated into one or more of these concepts and video clips that contain frames of that concept are returned. Such supervised approaches can perform well for certain types of queries, but often require a great deal of human effort both for the system designers (who must hand label example images to train the classifiers) and for the system users (who often must translate their queries by hand into the predefined concept terms).

Unsupervised approaches to multimedia IR can also be divided into two approaches: those that incorporate non-linguistic data and those that do not. Approaches that do not incorporate non-linguistic information are popular for video search because of their simplicity and generality. Such systems generally employ automatic speech recognition (ASR) to generate a transcript of what is said in a video, which can then be used to index the video as if it were a text document, thus turning the problem of video search into one of text search (Wactlar et al., 1996). Although useful for more general applications, such as topic spotting in news

broadcasts, this approach is often unsatisfactory for more specific types of search, particularly in sports video.

The occurrence of a query term in a video is often not enough to assume the video's relevance to that query. For example, when searching through video of baseball games, returning all clips in which the phrase "home run" occurs, results primarily in video of events where a home run does not actually occur. This follows from the fact that in sports, as in life, people often talk not about what is currently happening, but rather, they talk about what did, might, or will happen in the future.

Unsupervised approaches that do incorporate non-linguistic information have primarily been used for searching large image databases (Barnard et al., 2003; Blei and Jordan., 2003). Such image search techniques generally operate by modeling the association between low level features of an image (such as color blobs) and hand annotated labels for that image. Although these approaches employ similar techniques to those used in this thesis, two important distinctions must be noted. First, these image IR systems do not employ representations of dynamic events, but focus only on features from individual images. Second, such systems are trained using clean hand-labeled annotations, not natural speech transcriptions that do not necessarily describe the co-occurring video.

2.5.3 Automatic Speech Recognition

Automatic speech recognition (ASR) is a well studied problem in which acoustic speech signals are transcribed automatically into text. State of the art techniques employ a noisy channel framework in which the correct transcription for a speech signal is found by means of a heuristic search (or decoding) through the space of possible text outputs. This search is based on the input speech signal and two probabilistic models: the channel model (also called the acoustic model); and the source model (also called the language model).

Acoustic models represent the relationship between the written and spoken forms of a word. They are trained using large parallel corpora of transcribed utterances paired with audio samples of people saying those utterances. The audio samples are converted into time-series of spectral features, most typically mel-frequency cepstral coefficients (MFCCs) and the transcribed utterances are converted into sequences of phonemes, based on a pronunciation

dictionary (Rabiner and Juang, 1993). The Expectation-Maximization (EM) algorithm is then used to train hidden Markov model (HMM) representations of the probabilistic relationship between acoustic features and phonemes.

Language models are used to assign prior probabilities to sequences of words in a language, in order to bias a system toward outputs that are more likely to be uttered in a particular language. Language modeling is a widely studied area of research used in many other natural language applications, such as information retrieval, machine translation, and part of speech tagging (Manning and Schutze, 2001).

Standard approaches for generating language models collect frequency statistics for words and phrases in a large corpus of language-specific text documents (e.g., multiple years of the Wall Street Journal). These statistics are used to generate multiple n-gram probability distributions which represent the probability of a word given the n-1 previous words in an utterance. Because many sequences of words may not be observed in a given training corpus, smoothing is used to improve the probability estimates (Manning and Schutze, 2001).

Very little work has been done to in ASR to exploit information from the non-linguistic context that surrounds language use. A notable exception to this is Roy and Mukherjee (2005), in which the language model of an ASR system is extended to take advantage of the visual context of the speaker. Here the language model dynamically updated the likelihoods of words based upon the output of a computer vision system in an extremely limited visual/linguistic domain (utterances were limited to the type: "the large green block beneath the yellow and red blocks").[4]

In this thesis, we present grounded language models as a more principled and general method of integrating non-linguistic context into natural language processing tasks. Like traditional text-based language models, grounded language models encode the prior probability of words using conditional probability distributions. Unlike text-based language models, though, grounded language models represent the probability of a word conditioned not only on the previous word(s), but also on features of the non-linguistic context in which the word was uttered. In the following chapters, we describe in detail how grounded language

[4] See Qu and Chai (2006) for a related approach based on deictic gestures.

models are generated and show how they are applied with significant benefit to the applications described here.

Chapter 3

Events in the Virtual World

In this chapter we describe a top-down approach to representing event structure for a grounded language model. The approach is based on the creation of behavior grammars which explicitly model all possible actions that agents can take in an environment. Behavior grammars enable the design of rich models of meaning that capture the ambiguity inherent in language about events. We highlight two distinct types of ambiguities modeled by behavior grammars. We then describe a two phase approach for learning grounded language models, in which behavior grammars are used to generate hierarchical event structures that are then associated with natural language.

3.1.1 Ambiguity of Events

A key concern when representing the structure of events is to account for the ambiguities that arise when such actions are described. While the ambiguity of events has been studied extensively in the psychological literature (Vallecher and Wagner, 1987; Woodward et al., 2001; Gleitman, 1990), little work on event representation has been proposed that accounts for those ambiguities in a computational framework. To motivate our approach to representing events,

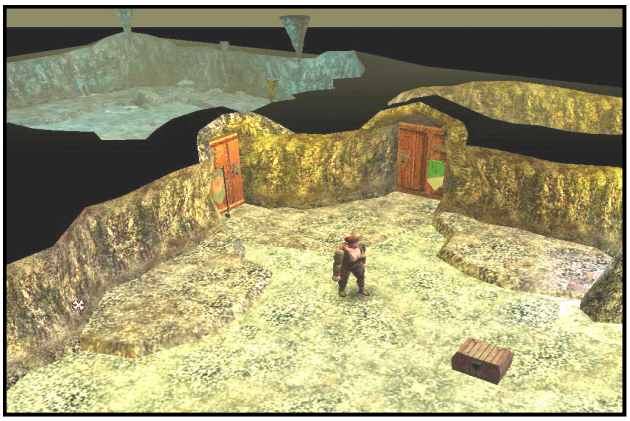

Figure 3-1 Screenshot of virtual game world used in grounded language modeling experiments

we propose a simple example that illustrates some of these ambiguities and suggests a representation to account for them.

Consider a video game world (such as that shown in Figure 3-1) in which, to win the game, a player must respond to spoken requests by their partner to perform various tasks (e.g., if the player is told to "open the door," the player must open the door). Assume that you are watching this game take place and have full access to these interactions (i.e., the verbal requests paired with the player's actions situated in the virtual world). At a certain point in the game, you hear the unknown word "grok" uttered and observe the player click their mouse on the leftmost door. Now, based only on your sensory observation, there are a number of possible interpretations for the word "grok." "Grok" may mean open the door, or alternatively, move to the door. Or "grok" might be a command to let another player into the room, or for the player to get some needed object from the next room (such as an axe).

Such situations demonstrate two distinct types of ambiguity, which we represent graphically as a lattice in Figure 3-2. In this lattice, the leaf nodes represent low level observations of the player's actions (i.e. the mouse clicks), while the root nodes represent the high-level intentions behind those actions (e.g. winning the game).

The first type of ambiguity shown in the lattice, which we refer to as a *vertical ambiguity*, describes the ambiguity between the *find axe* versus *open door* interpretations of "grok." Here the ambiguity is based on the level of description that the speaker intends to convey, not what the speaker wanted the player to do. Even when the speaker's intention is known (e.g. the speaker wanted the player to get the axe), a vertical ambiguity still exists based on whether

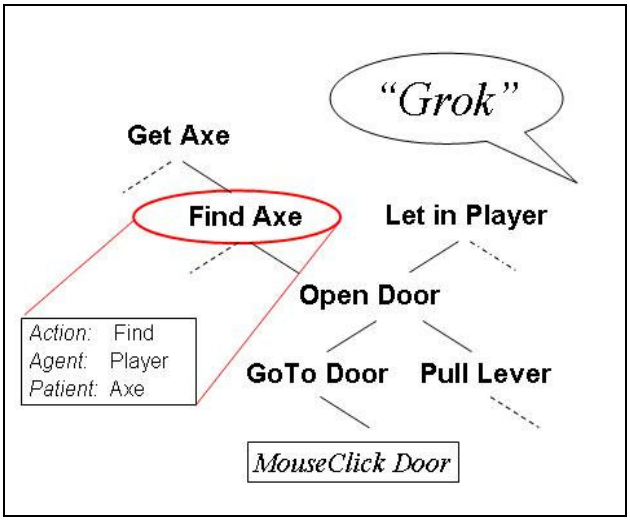

Figure 3-2 Example event structure inferred after observing a subject mouse click a door. The structure provides a graphical representation of two distinct types of ambiguity surrounding word learning.

"grok" means "find the axe" or just the first step of that event, i.e. "open the door." Interestingly, even though "grok" has a fixed meaning, if we were to ask the player "Did you mean to go find the axe?" or "Did you mean to go open the door?" they would answer yes to both questions.

The second type of ambiguity, referred to as *horizontal ambiguity* describes the difference in interpretation between the *find axe* versus *let in player* meanings of "grok." In this case, the high level action behind the sensed action is ambiguous. Unlike with vertical ambiguities, only one of these actions is typically intended. Thus, if the speaker were questioned about their action, they could answer in the affirmative to only one of the questions: "Did you mean to let another player in?" and "Did you mean to go find the axe?"[5]

By representing events as a lattice, both vertical and horizontal ambiguities are captured. Such representations encode the hierarchical nature of events, as well as, represent how the interpretation of an event is dependent upon the context of a situation. In the next section we introduce behavior grammars as a way to formalize this context.

[5] While vertical ambiguities may have a parallel in objects (e.g. animal-dog-poodle) (Rosch, 1976) horizontal ambiguities are unique to intentional actions.

1. Win_Game → Get_Axe Let_in_Player

2. Get_Axe → Open_Door Pick_Up_Axe
3. Let_in_Player → Open_Door

4. Open_Door → Click_Door
5. Open_Door → Pull_Lever

Figure 3-3. Example production rules in a behavior grammar that encodes the strategies to win a simple game. Behavior Grammars are used to infer event representations from sequences of observed movements.

3.1.2 Grammars of Behavior

The idea of a *behavior grammar* dates back at least to Miller et al. (1960) and has been re-visited more recently by Baldwin & Baird (2001). We formalize this notion by defining a behavior grammar as a set of rules that describe how an agent's higher level actions (e.g., *find axe*) can expand into sequences of lower level events (e.g. *open door, go through door, open chest*). For any given high level goal (e.g. wining the game in the above example), the rules in a grammar define all the valid strategies that an agent can take to achieve that goal. In this way, a behavior grammar for a task serves as a description of an agent's knowledge about how to complete that task. Also, it serves as a formalization of all the actions that an agent can take within a certain domain.[6]

Figure 3-3 depicts a simple behavior grammar that can be used to interpret the observed movements in the videogame scenario described above. The highest level intention is to win the game, and this is achieved with the completion of two lower level events: getting the axe, and letting in the other player. These two events are themselves achieved through a sequence of lower level actions, both of which begin by opening a door. For someone observing the scenario described above, the horizontal ambiguity surrounding the agent's opening a door (see Figure 3-2) is represented in the behavior grammar by the existence of multiple rules (i.e. rule 2 and 3) that both begin with the same action. The resolution of such ambiguity is achieved

[6] Importantly, the rules in these behavior grammars are created entirely by hand (see Section 3.2). Explicitly modeling this context is facilitated by the deterministic nature of virtual worlds and the relatively simple games we examine. In domains where these assumptions do not hold, such as broadcast video, alternative techniques are applied. The second part of this thesis discusses an approach which learns event structure automatically from data.

through the observation of more movements in order to determine which of these two rules is consistent with all of the observed data.[7]

Modeling the possible actions as a grammar of behavior allows us to recast the process of recognizing events as a problem of parsing a sequence of observations. As players of a game perform new low level actions (i.e. mouse-clicks), the behavior grammar can be used to infer a hierarchical event structure over those actions. In this top down approach, the knowledge of how to interpret high level events encoded in the behavior grammar is used to disambiguate lower level events.

The following sections describe this process in more detail and show how it is used to train a grounded language model. The methodology operates in two phases: Section 3.2 describes the first phase, in which a top-down approach is used to generate event structure representations from sequences of observed actions; Section 3.3 describes the second phase, in which words are mapped onto these inferred structures to produce a grounded language model.

3.2 Representing Events

Our top-down approach to event recognition infers event structure based on sequences of an agent's observed low level actions. As described above, this procedure is cast as a parsing problem, which relies on a behavior grammar that encodes the strategies an agent can take to achieve some goal. In order to capitalize on this analogy with parsing, we represent such grammars of behavior using a probabilistic context free grammar (PCFG) that allows for the building of event structures in much the same way that a PCFG for syntax allows for the parsing of sentences (e.g., Collins, 1999). Treating grammars in this way allows us to treat the rules that make up a behavior grammar as production rules in which an agent's high level intentional actions *produce* sequences of lower level intentional actions with different probabilities (e.g., find_axe → open_door go_through_door open_chest). Such rules mirror syntactic rules in which high level syntactic categories produce lower level categories (e.g. NP → DT ADJ N).

[7] Because sometimes even the same sequence of observations can be described by multiple rules, intention rules are extended with probabilities that can also be used to resolve ambiguities. This is discussed in more detail below.

Unlike syntactic rules, each node of an inferred event structure is not atomic, but rather, represents an action that itself has internal structure. We can capture this internal structure by using a semantic frame representation for each node in the lattice. Similar to the semantic frames described by Fillmore, (1976), these semantic frames encode the participants of an action and their thematic roles (agent, patient, action, etc.). For example, in Figure 3-2 (see insert), the node labeled find axe, comprises a frame with a FIND action, a PLAYER agent, and an AXE patient.

By formalizing the grammar of behavior as a PCFG, event recognition can be recast as a problem of parsing sequences of observed low level actions. Similar to previous work in video event recognition (Ivanov and Bobick, 2000) and plan recognition (Pynadath, 1999), we borrow an established algorithm used for syntactic parsing from work in computational linguistics (Stolcke, 1994). This algorithm is based on dynamic programming techniques, leading to efficient and tractable performance even with complex grammars and input data. The algorithm is adapted to generate event structures by replacing the syntactic grammar used for sentence parsing with a behavior grammar. The parser can then be run over a sequence of observations of players in a game, just as it would be run over a sequence of words in a sentence.

Using PCFGs to model event recognition represents a compromise between simpler models of plan recognition such as Hidden Markov Models (e.g., Horvitz, 1998), that are easier to parameterize, but are not in general hierarchical, and models that rely on full abductive inference (Appelt and Pollack, 1992) that are less efficient because of their greater complexity. Although this compromise gives PCFGs a good balance between efficiency and complexity, they do this at the cost of a number of assumptions. PCFG parsers cannot infer trees in which an agent executes multiple actions concurrently, nor can a single observed action ever be thought to accomplish multiple higher level goals. Further, PCFGs assume that all actions occur in strict sequential order, and therefore, that they do not occur during or overlap with each other.[8] These assumptions, however, allow for the use of very efficient algorithms to model event recognition and provide a natural method for handling the horizontal ambiguities associated with intentional action.

[8] This assumption does not hold in more complex domains, such as in broadcast video.

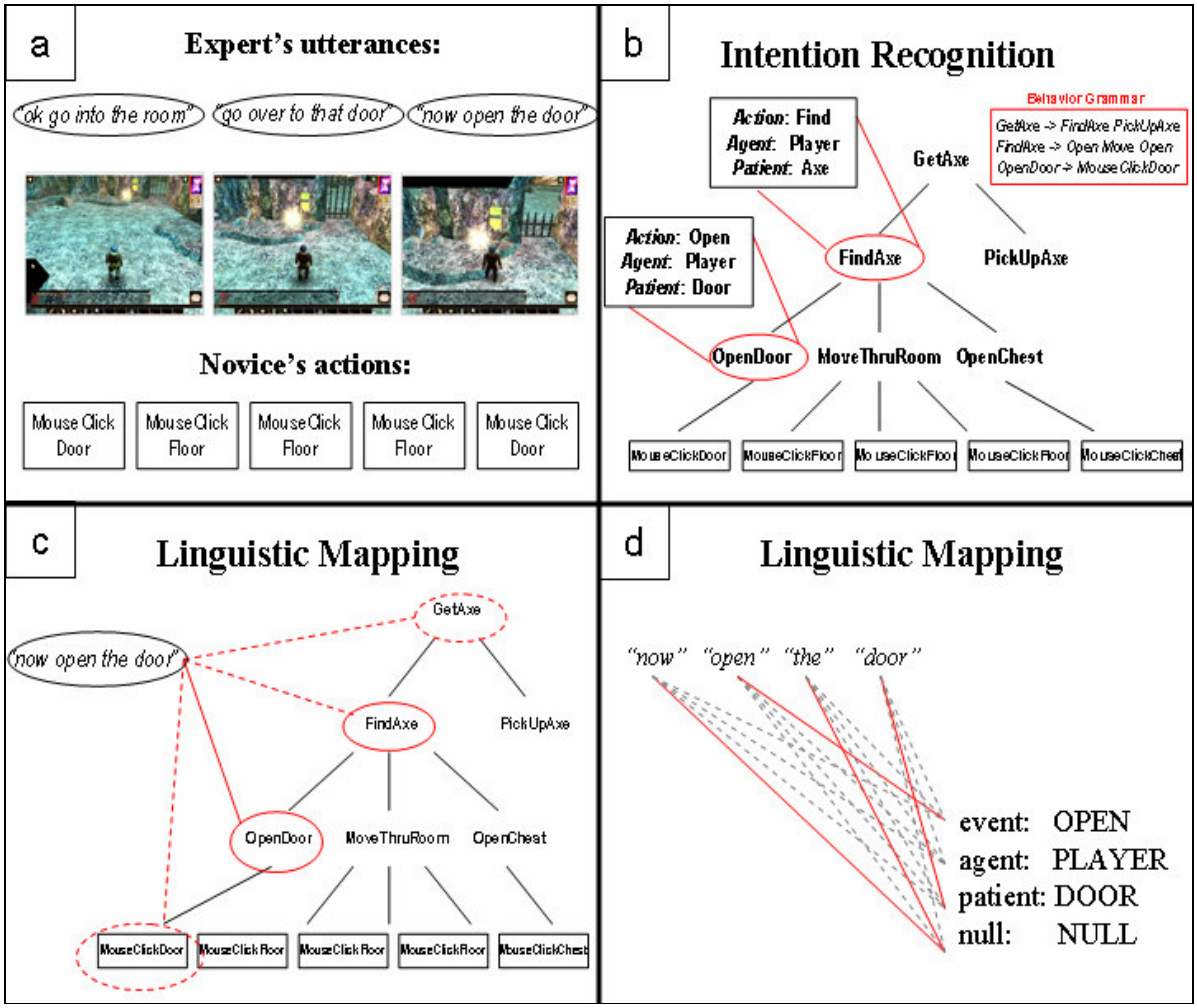

Figure 3-4 a) Parallel sequences of speech and actions are recorded from subjects as the expert guides the novice through a virtual environment. b) A tree is inferred over the novice's sequence of observed actions using a probabilistic context free grammar of behaviors. Each node in the tree is a different level of intentional action and is encoded by a semantic frame. c) The vertical path from a leaf node in the tree (i.e. observed action) to the root (i.e. highest order intentional action) contains multiple possible levels to which an utterance may refer. Linguistic mapping uses d) Expectation Maximization to estimate the conditional probabilities of words given roles to resolve this ambiguity.

As described above, horizontal ambiguities stem from the fact that any given observed action may have been performed to achieve any number of higher level intentions. This ambiguity is captured by the existence of multiple possible paths from the observed leaf action in an event structure to the root intention. By treating a behavior grammar as a PCFG, each of these paths has associated with it a probability that stems from probabilities associated with each rule in the grammar (see Stolcke, 1994, for a discussion of how path probabilities are calculated). As a

1. Set uniform likelihoods for all utterance/frame pairings
2. For each pair, run standard EM
3. Merge output distributions of EM (weighting each by the likelihood of the pairing)
4. Use merged distribution to recalculate likelihoods of all utterance/frame pairings
5. Goto step 2

Figure 3-5. Psuedo-code for Lingistic Mapping algorithm

growing history of movements are observed, the parsing algorithm is able to focus in on the most likely of these paths using the observed sequence as evidence for the agent's high level intentions. After each observation, the algorithm can produce a single tree that represents the most likely path from each of the observed leaf actions to the single root intention (see Figure 3-4b). Thus, in the same way that syntactic ambiguities can be resolved to create a syntax tree using a PCFG of syntax, horizontal ambiguities can be resolved to create a tree using a PCFG of behavior.

Given a tree in which, by definition, horizontal ambiguities have been resolved, the model of language learning is still confronted with the possibility of vertical ambiguities. Such vertical ambiguities, which refer to what level of description a speaker had in mind for their utterance, are represented by the multiple number of nodes that a given utterance could refer to along the *vertical path* from observed action to root goal (see Figure 3-4c). It is an assumption of our model that these vertical paths are manually aligned in time with their appropriate utterance.[9]

3.3 Linguistic Mapping

Having observed a sequence of movements, the output of event recognition is a single tree that represents the model's best guess of the higher order intentions that generated that sequence. This inferred tree can be seen as the conceptual scaffolding onto which utterances describing those intentional actions are to be mapped. The goal of the linguistic mapping algorithm is to learn a grounded language model that encodes the relationship between the words an agent

[9] This assumption is dropped when working with broadcast video.

says and the event structure that describes what an agent does. In this work, a bag of words model is employed and syntax is not exploited (see Fleischman and Roy, 2007c for an extension to this model which incorporates syntactic phrase boundaries).

As described above, each node in an inferred event structure consists of a semantic frame. The linguistic mapping algorithm attempts to learn associations between words in utterances and the elements in these frames. These elements can be either role fillers in a semantic frame (e.g. MOVE, DOOR) or the roles themselves (e.g. EVENT, PATH). This allows a word to map not only to representations of objects and events (e.g. "go"→ MOVE), but also, to the more functional aspects of a semantic frame (e.g. "through" → PATH). Such mappings are represented by the conditional probabilities of words given frame elements [i.e. p(word|element)]. By formalizing mappings in this way, we can equate the problem of learning a grounded language model to one of finding the maximum likelihood estimate of a conditional probability distribution.

Similar to statistical approaches to language translation (Brown et al., 1993), we apply the Expectation Maximization (EM) algorithm to estimate these mappings. EM is a well studied algorithm that finds a locally optimal conditional probability distribution for an utterance given a semantic frame based on the equation:

$$p(W \mid F) = \frac{C}{(l+1)^k} \prod_{j=1}^{k} \sum_{i=0}^{l} p(W_j \mid F_i) \qquad \textbf{3-1}$$

where k is the number of words in the utterance W, W_j is the word in W at position j, l is the number of frame elements in the semantic frame F, F_i is the frame element in F at position i, and C is a constant.

To understand the use of EM for linguistic mapping, it is easiest to first assume that we know which node in the vertical path is associated with each utterance (i.e., there is no vertical ambiguity). EM operates by iterating between an Expectation (E) step and a Maximization (M) step. In the E step, an initial conditional probability distribution is used to collect expected counts of how often a word in an utterance appears with a frame element in its paired semantic frame (Figure 3-4d). In the M step, these expected counts are used to calculate a new conditional probability distribution. By making a one-to-many assumption—that each word in an utterance is generated by only one frame element in the parallel frame (but that each frame

element can generate multiple words)—the iterative algorithm is guaranteed to converge to the maximum likelihood estimate of the conditional distribution. Following Brown et al. (1993), we add a NULL role to each semantic frame which acts as a "garbage collector," accounting for common words that don't conceptually map to objects or actions (e.g., "the," "now," "ok," etc.).

The aforementioned procedure describes an ideal situation in which one knows which semantic frame from the associated vertical path should be paired with a given utterance. As described above, this is not the case for language learners who, even knowing the intention behind an action, are faced with an ambiguity as to what level of description an utterance was meant to refer (Figure 3-4c). To address this ambiguity, an outer processing loop is introduced that iterates over all possible pairings of utterances and semantic frames along the vertical path. For each pairing, a conditional probability distribution is estimated using EM. After all pairings have been examined, their estimated distributions are merged, each weighted by their likelihood. Figure 3-5 describes this procedure, which continues until a cross-validation stopping criterion is reached. The utterance/frame pair with the highest likelihood yields the most probable resolution of the ambiguity.

In this chapter, our top-down approach to representing events for grounded language models was described. Behavior grammars are used to represent the hierarchical structure of events, which can then be used to ground the meaning of words. While event representation is implemented as a parsing procedure, linguistic mapping is cast as a problem of estimating a conditional probability distribution of semantic frame elements given words. In the next Chapter we describe evaluations of this grounded language model using two virtual game environments.

Chapter 4

Application:

Natural Language Understanding

Although multiple strategies for evaluation exist, we examine the performance of grounded language models on a language understanding task. Natural language understanding describes the task of automatically converting natural language utterances into computer readable semantic formalizations. Previous work on learning such systems can be found in many areas of Artificial Intelligence research, such as in the database domain, where systems are trained to convert natural language queries into SQL (e.g., Zettlemoyer and Collins, 2005; Ge and Mooney, 2005), as well as in the natural language processing community, where taggers are built to label sentences with semantic roles (Gildea and Jurafsky, 2002; Fleischman et al., 2003).

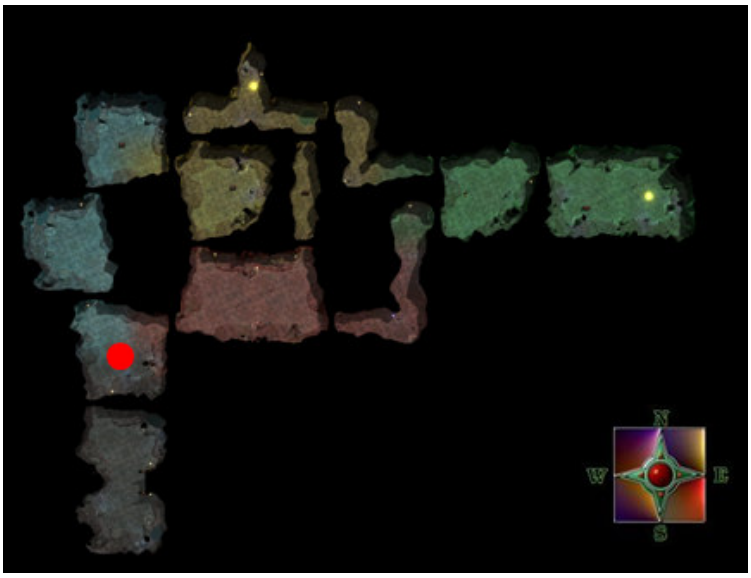

Figure 4-1. Map of virtual world used by experimental subjects. Red circle marks start position.

For these evaluations, we formulate natural language understanding in a Bayesian framework, in which understanding an utterance is equivalent to finding the most likely meaning (i.e. semantic frame) given that utterance (for a related formulation, see Epstein, 1996):

$$p(meaning \mid utterance) \approx p(utterance \mid meaning)^{\alpha} \bullet p(meaning)^{(1-\alpha)} \qquad \textbf{4-1}$$

In this equation, the p(utterance|meaning) is calculated using the grounded language model as in equation (3-1). The prior p(meaning) is approximated by the probability of the most likely inferred tree (i.e. the probability given by the PCFG parser). It is generated online during testing as follows: as each action is performed in the virtual environment, the behavior grammar is used to infer a partial hierarchical event structure that accounts for all actions up to that point. Whenever a new action is performed, the system generates a probability distribution over which action will be performed next. This is done by cycling through each possible action in the grammar, parsing it as though it had actually occurred, and storing the probability of each resulting tree.

In the remainder of this chapter, we describe two sets of experiments, in two different virtual domains, designed to evaluate grounded language models on a natural language understanding task. The first environment, a multiplayer videogame, is designed in order to examine the richness of the event representations generated by the grammars of behavior. Using a natural language understanding task, we examine how the pattern of word learning

demonstrated by the grounded language model is related to psychological findings on human language acquisition.

With the second environment, we examine how the contextual information in the behavior grammar can be used to improve the performance of language understanding. Experiments in an independently designed military training simulation demonstrate the practical benefits of our top-down approach for understanding natural language about events.

4.1 NeverWinter Nights

In order to evaluate the richness of the event representations used to train grounded language models, a virtual environment was designed based on the multiplayer videogame NeverWinter Nights[10] (Figure 3-1). The game was chosen because of its inclusion of an authoring tool that facilitated the creation of novel tasks within the virtual environment. For the purposes of collecting data, a game was designed in which two human players must navigate their way through a cavernous world, collecting specific objects, in order to escape (earlier descriptions of this work appear in Fleischman and Roy, 2005a; Fleischman and Roy, 2005b, and Fleischman and Roy, 2007c).

Subjects were paired such that one, the novice, would control the virtual character moving through the world, while the other, the expert, used a map to guide her through the world via spoken instructions. While the expert could say anything in order to tell the novice where to go and what to do, the novice was instructed not to speak, but only to follow the commands of the expert. The purpose behind these restrictions was to elicit speech that was free and spontaneous, but limited to task specific commands and descriptions (as opposed to other speech acts, such as questions).[11]

For each pair of subjects, the experts were given a map of the environment (see Figure 4-1) and a list of objects that must be retrieved in a specific order (i.e. an axe, a password, and one of two keys) before the cavern could be exited. The task was designed such that subjects could

[10] http://nwn.bioware.com/

[11] Note that in sports video language use is constrained only by the announcers' responsibilities to report on the game being played. This does not preclude discussion of other topics and speech acts, even though they are not explicitly represented in grounded language models.

access these objects in a number of different ways (opening chests, pulling levers, asking for help, bribing officials, etc.). This was done specifically to elicit rich language from the experts (for sample transcript, see Appendix A). The subject pairs were asked to repeat the task (i.e. exit the cavern) indefinitely, but were stopped by the experimenter after five attempts were successful. This process took approximately 30 minutes per pair. For each attempt, the order in which the objects were to be retrieved was randomized.

The subjects in the data collection were university graduate and undergraduate students (8 male, 4 female). Subjects were staggered such that the novice in one pair became the expert in the next. This insured that the experts had time to familiarize themselves with the environment and facilitated completion of the tasks. Only those pairs of subjects that succeeded in completing the task five times were included in experiments. Only eight pairs met this requirement.

The experts wore head-mounted microphones such that all of their speech was recorded, and the game was instrumented so that all of the novices' actions were recorded during game play. Figure 3-5a shows example screen shots of a game in progress along with the two associated parallel sequences of data: the expert's speech and the novice's actions. The expert's speech is then automatically segmented into utterances based on pause structure and manually transcribed (Yoshida, 2002). The resulting transcribed utterances are then manually paired with their temporally appropriate observed actions. These sequences of observed actions are then parsed using a hand designed behavior grammar (see below) to infer a tree representation of the event structure (see Figure 3-5b). For the training phase, but not the testing phase, the entire sequence of movements that resulted in the completion of the task is parsed at once and linguistic mapping is performed using the most likely tree from that parse. This batch processing allows for much more reliable trees for training (since all of the movements in a task have been observed).

In hand building the behavior grammar, two sets of rules were created: one to describe agents' possible paths of locomotion and one to describe other actions necessary for completing the game (see Appendix A for examples of the most frequently used of these rules). The locomotion rules were built semi-automatically, by enumerating all possible paths between target rooms in the game. The non-locomotion rules were designed based on the rules of the game in order to match the actions that players must take to win (e.g. opening doors, taking

objects, interacting with non-player characters, etc.). Rules were built and refined in an iterative manner, in order to insure that all subject trials could be parsed. Because of limited data, generalization of the rules to held-out data was not examined. Probabilities were set using the frequency of occurrence of the rules on the training data.

The output of the collection process is a data set consisting of five trials for each of the eight successful subject pairs. Each one of these 40 trials consists of a stream of the expert's transcribed utterances and a tree representing the most likely parse of the novice's observed actions. In the next section we examine a number of experiments to evaluate grounded language models trained on this dataset.

4.1.1 Experiments

These experiments examine the performance of the grounded language model on a language understanding task. For each subject pair, a grounded language model is trained using only the first four trials of their game play and then tested on the final held out trial. (This gives on average 130 utterances of training data and 30 utterances of testing data per pair.) For each utterance in the test data, the likelihood that it was generated by each possible frame is calculated. In these experiments, we use only the grounded language model to calculate likelihoods and use a uniform prior probability. The effect of a non-uniform prior is examined in Section 4.2.

Given an unseen test utterance, we select the maximum likelihood frame as the system's hypothesized meaning. For all test utterances, we examine both how often the maximum likelihood estimate exactly matches the true frame (frame accuracy), as well as, how many of the frame elements within the estimated frame match the elements of the true frame (role accuracy). For each pair, the number of iterations, beam search, and other initialization parameters (see Moore, 2004) are optimized using cross-validation.

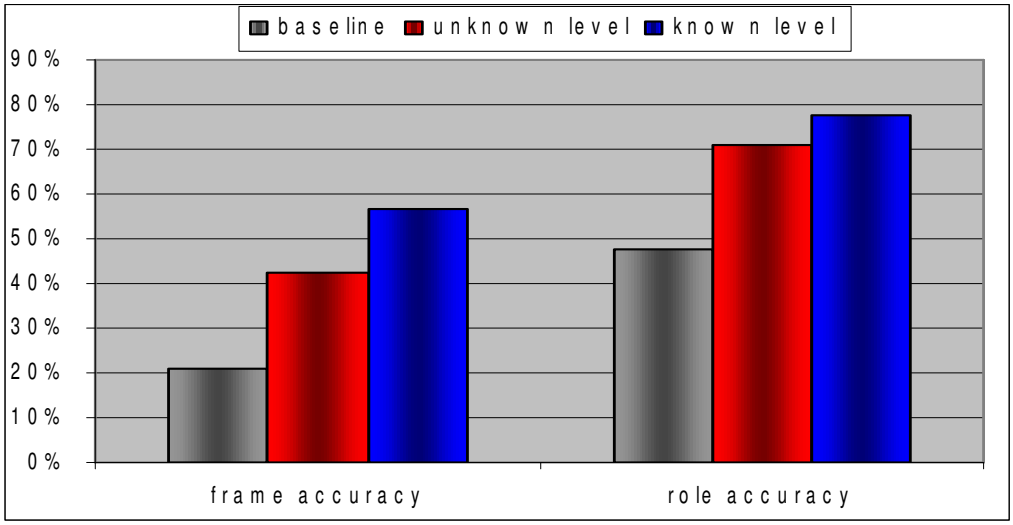

Figure 4-2 Language understanding in NeverWinter Nights. Frame and role accuracy for a grounded a language model is compared when trained both with and without known utterance-to-level alignments (i.e. with vertical ambiguity),

Performance is tested under two conditions: 1) using a grounded language model trained with unknown utterance-to-level alignments (i.e. with vertical ambiguity), and 2) using a grounded language model trained with known utterance-to-level alignments (i.e. with no vertical ambiguity). The performance is compared to a simple baseline of always choosing the most frequent frame from the training data.

Figure 4-2 shows the percentage of maximum likelihood frames chosen by the system that exactly match the intended frame (frame accuracy), as well as, the percentage of roles from the maximum likelihood frame that overlap with roles in the intended frame (role accuracy). The figure shows that either trained with or without vertical ambiguity, the grounded language model is able to understand language better than the baseline. Further, the figure shows that both frame and role accuracy is higher when the system is trained without vertical ambiguity. This is not surprising, for the uncertainty inherent to vertical ambiguity is expected to add noise when training the grounded language model.

More surprising though, is the large difference in how performance changes for roles compared to whole frames. While the performance on frame accuracy declines by 14.3% with the addition of vertical ambiguity, the same ambiguity causes only a 6.4% decline in role accuracy. Because all roles in a frame must be correct for a frame to be considered correct, the

large difference in performance drop suggests that the vertical ambiguity affects learning of words for some types of roles more than others.

This hypothesis is analyzed by examining how often different classes of words (i.e. nouns vs. verbs) are mapped to the correct semantic role. The results of this analysis are presented in Table 4-1, and show the total number of nouns and verbs that are correctly mapped in the test data, as well as, detailed statistics for the 10 most frequent nouns and verbs. The frequency of those nouns and verbs in the training data is also presented. The Table shows that the model's accuracy for nouns is significantly ($p<0.01$) greater than its accuracy for verbs, even though fewer nouns than verbs were present in training.

4.1.2 Discussion

The asymmetry in how well the grounded language model learns nouns compared to verbs mirrors a similar pattern in human language acquisition. Much psychological research on language acquisition has sought explanations for the asymmetry between noun and verb acquisition in the developing cognitive or linguistic abilities of language learners (Gentner, 1982; Snedeker and Gleitman, 2004). The performance of the grounded language model, however, follows directly from the model's formalization of the event structure and the inherent ambiguity of those events.

The key to our noun/verb asymmetric result lies in the fact that, while each node of an event structure (i.e. each semantic frame) has a different action role, often the object roles in different levels are the same. For example, in Figure 3-5b, the actions GET, FIND, and OPEN occur only once along the vertical path from root to leaf. However, the object AXE occurs multiple times along that same path. In a word learning scenario, this means that even if the model misinterprets what level of event an utterance describes, because object roles are repeated at multiple levels, it still has a good chance of mapping the nouns in the utterance to their correct roles. However, because action roles are more specific to their level of description, if the model misinterprets the level, linguistic mapping for the verb cannot succeed.

This pattern is consistent throughout the training data; where, for each vertical path in an intention tree, the same action role is seen on average 1.05 times, while the same object role is

Table 4-1 Word accuracy for nouns and verbs, with frequency in testing and training sets.

	VERBS				NOUNS		
	Frequency		Accuracy		Frequency		Accuracy
Word	Train	Test	Test (%)	Word	Train	Test	Test (%)
Go	342	69	73.9	door	249	47	85.1
Get	96	16	6.3	chest	89	22	54.5
Open	65	17	23.5	portal	49	10	80.0
take	59	14	50.0	key	33	9	100
bash	47	12	66.7	axe	29	11	54.5
follow	31	6	33.3	password	28	7	100
talk	28	7	0.0	lockpick	26	6	100
Turn	22	6	0.0	diamond	25	7	71.4
Ask	19	7	42.9	lever	24	7	85.7
Teleport	10	4	0.0	archway	23	5	100
ALL	719	158	44.0	*ALL*	575	130	65.2

seen 2.05 times. Thus, it is the ambiguity of actions and the recurrence of objects in a vertical path which causes the model to learn verbs more poorly than nouns.[12]

The noun/verb asymmetry learning that we see in our results mirrors the pattern of behavior in children. This of course does not imply that children maintain the particular data structures and algorithms that are used in this model. However, at a functional level, the ambiguities that the model encodes also exist for the children, regardless of the specifics of their representations. This insight suggests that one cause of the noun/verb asymmetry may lie not in the developing cognitive or linguistic abilities of children, but rather are inherent in the nature of events.

The implications of these results are discussed further in Fleischman and Roy (2005a), and extended to examine the role of syntactic information in Fleischman and Roy (2007c). In the next section we continue our examination of the effectiveness of grounded language models for natural language understanding using an additional virtual domain.

[12] While formalizing object ambiguity may dilute this effect, research on "basic level" descriptions (Rosch, 1976) suggests that ambiguity for objects may be different than for actions.

Figure 4-3. Screenshot of Mission Rehearsal Exercise (MRE) military training scenario.

4.2 Mission Rehearsal Exercise (MRE)

The second domain used for testing this top-down approach to representing events for grounded language models is an interactive virtual training environment called the Mission Rehearsal Exercise (MRE) (earlier versions of this work appear in Fleischman and Hovy, 2006). Unlike the NeverWinter Nights videogame environment, which was designed specifically for examining grounded language modeling, the MRE is an ongoing large-scale collaborative research project designed independently of this thesis (Swartout et al, 2005).

The MRE is a fully interactive training simulation modeled after the holodeck in Star Trek. The project brings together researchers working on graphics, 3-D audio, Artificial Intelligence, and Hollywood screenwriters to create a realistic virtual world in which human subjects can interact naturally with simulated agents. The virtual agents communicate through voice and gesture, reason about plans and actions, and incorporate a complex model of emotions. Human users can query and interact with an agent using natural speech as they proceed through scenarios developed for the particular training mission at hand.

Figure 4-3 shows a screen shot of the "peace-keeping mission" scenario employed for these evaluations. The scenario is designed to train army lieutenants for eastern European peace keeping missions. It centers on the trainee, a human lieutenant, who is attempting to move his platoon to support an inspection, when one of his drivers unexpectedly collides with a civilian car. A civilian passenger, a young boy, is critically injured and the trainee must interact with

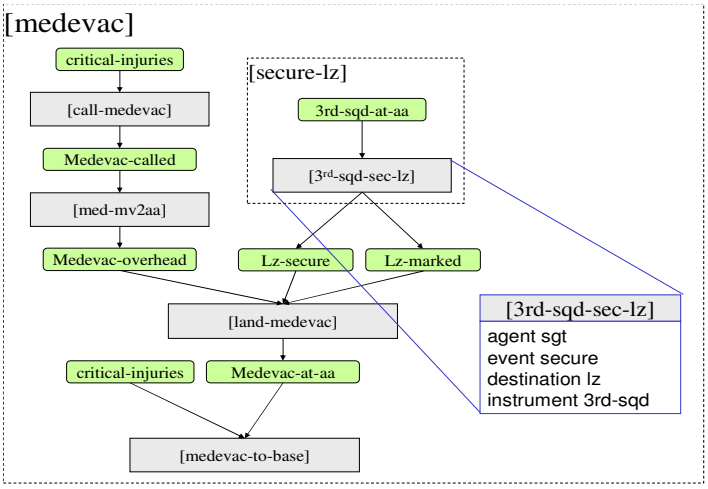

Figure 4-4 Task model of agents in the MRE. Inset shows frame representation of individual action.

his or her virtual platoon sergeant in order to arrange a medical evacuation (i.e. medevac) and stabilize the situation.

In the MRE, all actions and plans that an agent may take are represented as tasks in a task model (Rickel et al., 2002). These tasks are represented using relatively standard plan representations, in which each task is defined by a sequence of (partially ordered) steps, or actions, that may either be primitive (i.e. a physical or sensing action in the virtual world) or abstract (i.e., a task that must be further decomposed into primitive actions). Each action can be related to any other by causal links and/or threat relations that define the pre and post conditions used in planning. Further, because abstract actions are decomposable, the task model maintains a hierarchical structure that can be seen in the graphical representation of the task [medevac], shown in Figure 4-4. Here the large dashed boxes represent abstract actions and the smaller solid boxes represent the primitive actions of which they are composed. (The pre and post conditions are represented by ovals, the relations between states and actions as lines). Further, as seen in the inset, each action has an internal case frame structure, identical to that used in our representations of behavior grammars.

The actions represented in the MRE's task model delimit what the virtual agents in the MRE can do, and thus, encode the contextual situation of the training scenario itself. Although not designed for interpreting language, the task model encodes the same type of information that grammars of behavior are designed to capture. Although not encoded as a probabilistic context free grammar (PCFG), the task model can easily be converted by removing much of the

planner-specific information it contains and attaching probabilities to its various tasks (for more detail, see Fleischman and Hovy, 2006). Appendix A shows a PCFG representation of the information in the task model for the "peace-keeping mission" scenario in the MRE.

By converting the task model to a grammar of behavior, we can evaluate our grounded language model as a natural language interface for trainees in the MRE. Unlike our evaluations using NeverWinter Nights, these evaluations have the advantage of operating on an independently generated dataset, and using a grammar of behavior not hand-designed by the experimenter.[13] In the following section we describe these evaluations in detail.

4.2.1 Experiments

As in the previous set of evaluations, we are interested in the ability of the grounded language model to predict the correct semantic frame representation given a natural language input. Unlike the NeverWinter Nights evaluation, however, these experiments focus on measuring the effect of the context itself on language understanding.

We evaluate our grounded language models on transcripts of eight test runs of the MRE system (totaling 150 individual test utterances). Test runs represent actual interactions between military personnel and the virtual agents in the MRE system. Due to error propagation amongst the modules (e.g. speech recognition, dialogue management, etc.), these test runs are far from the type of clean interactions one sees with Wizard of Oz data. However, evaluating on actual interactions more accurately represents the system's true performance, as such noisy interactions are the norm in interactive applications. (Appendix A shows a portion of one of these test interactions.)

[13] Even though the task model of the MRE is not hand-designed by the author, it is hand-designed by another researcher. The next Chapter describes an approach which avoids hand-designed representations.

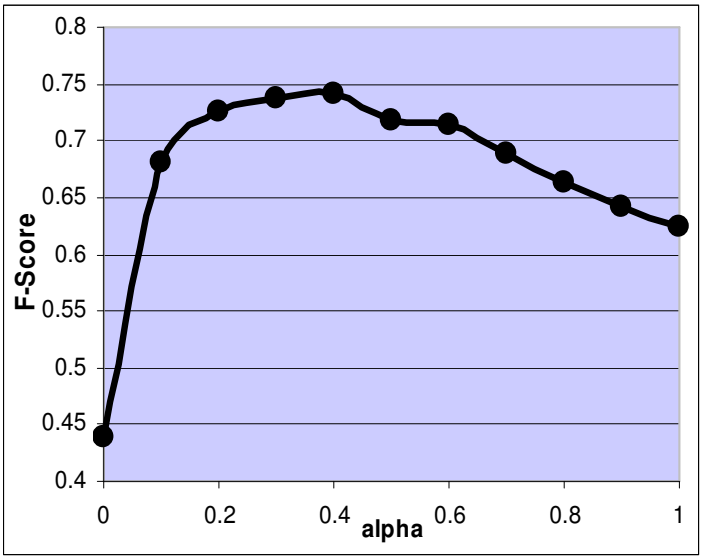

Figure 4-5. Effect of contextual information on natural language understanding in MRE.

Approximately 360 utterances were manually transcribed and paired directly with frame representations from the behavior grammar. A grounded language model was trained on this data as described in Chapter 3. Note that unlike the data used in the NeverWinter Nights evaluations, no vertical ambiguities exist in this training data, i.e. training frames and utterances were hand aligned. This reflects the focus of these experiments on the effect of context on understanding (not ambiguity in the training data).

To parameterize the probabilities used in the behavior grammar, maximum likelihood estimates for the rules were generated using leave one out cross-validation (i.e., probabilities for one run were estimated based on all other runs in the test set).

As described above, language understanding operates using a noisy-channel framework, in which a weighting coefficient alpha is used to adjust the amount of context that is exploited during understanding (see Equation 4-1). In these experiments, alpha=1 corresponds to using only the grounded language model for understanding, while alpha=0 corresponds to using only the prior probability derived from the behavior grammar (i.e. no linguistic evidence at all).

Figure 4-5 shows the results of varying alpha on understanding hand transcribed test utterances. Performance is measured using an F-measure based on the recall and precision of frame elements which the system correctly predicts. Results at each alpha value are above baseline performance of always guessing the most common frame (baseline F-Score = 0.11).

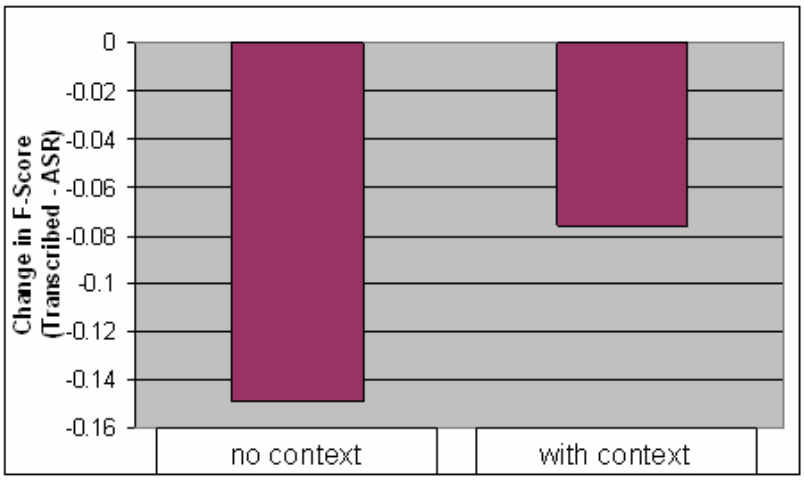

Figure 4-6 Context reduces negative effect of automatic speech recognition (ASR) on language understanding.

While Figure 4-5 shows performance given input from human transcription of speech, Figure 4-6 describes the effect of using automatic speech recognition (ASR) on the performance of the system. Because it introduces noise into the input, using ASR has a negative effect on language understanding performance. Figure 4-6 shows the magnitude of that drop in two different conditions: with and without contextual information from the behavior grammar. Although performance drops in both conditions, the relative drop in the system that does not use context is nearly 50% greater for the system that uses context.

4.2.2 Discussion

The results indicate that exploiting the contextual information encoded in the behavior grammar increases the performance of language understanding in the MRE. In addition, this contextual information makes the language understanding system more robust to noise from automatic speech recognition (ASR).

This follows from the fact that behavior grammars allow the contextual history of an utterance to be exploited during language understanding. Figure 4-7 shows a typical situation in the MRE in which, having completed a series of actions (i.e. collision, secure_area, evacuate_boy), the user produces an utterances that is poorly recognized by the ASR module. Even though the linguistic evidence here is weak, by setting the prior probabilities (in Equation 4-1) of each frame appropriately, the utterance can still be understood. This follows from the

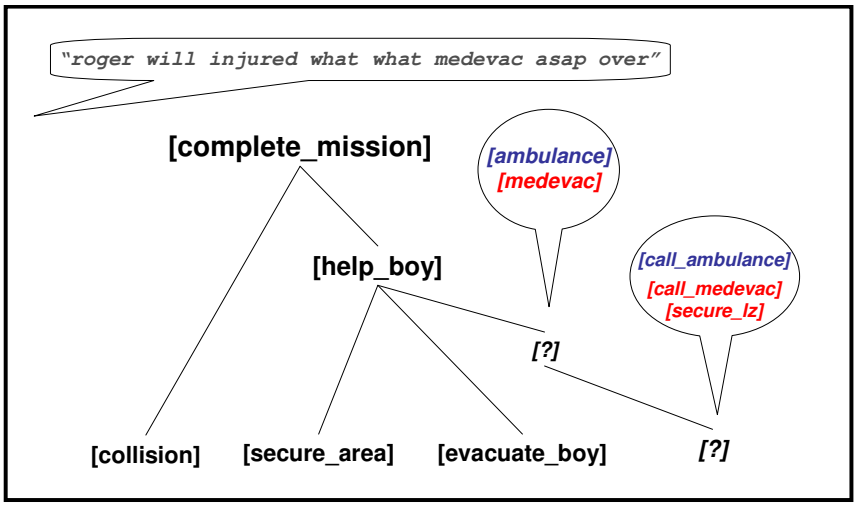

Figure 4-7 Parsing observed actions with a behavior grammar biases understanding toward events that are appropriate given the previous context (even with limited linguistic evidence).

fact that, by encoding the situational context of the task in the behavior grammar, the system is biased to output semantic frames that are appropriate for a given situation.

While encoding context using grammars of behavior has advantages for language understanding, there are a number of limitations to this approach that must be addressed.

4.3 Conclusion

We have described an approach for representing events in a grounded language model. The approach is based on the use of behavior grammars, which explicitly represent the actions an agent may take in an environment. Behavior grammars are used to generate rich representations of events to support natural language understanding in virtual environments. These representations allow for more robust communication with virtual agents. By incorporating contextual information in a principled way, agents can better understand language given limited or noisy linguistic input. Further, the rich event structures generated by this top-down approach provide insights into psychological phenomena in human linguistic development.

Although there are many advantages to explicitly representing context using behavior grammars, the top-down methodology that we introduce has serious limitations for

generalization to other, more complex domains. The primary source of these limitations is the dependence on hand-crafted grammars of behavior to model the situational context of events.

As the complexity of a domain increases, hand crafting grammars of behavior becomes less practical. Although natural in the relatively simple virtual games described here, the situational context that must be modeled for many real world interactions is too complex to model by hand. Even interactions within more advanced videogames (particularly ones that seek to mimic the real-world) may be too complicated to be captured by hand built grammars of behavior.[14] But in domains that represent the real world, such as the domain of broadcast video, such hand crafted models are a practical impossibility.

In hand crafting behavior grammars for virtual worlds, a number of features could be exploited that are unique to such environments. First, virtual worlds are deterministic such that the consequences of every user action can be predicted ahead of time (e.g. clicking on a door opens the door, etc.). Further, all user actions are transparent to the machine and can be recorded with virtually no noise or error. Finally, the artificial nature of virtual worlds itself facilitates the hand crafting of behavior grammars because the context in which events occur is already (in some sense) being explicitly modeled.

Generalizing grounded language models to more complex domains requires substituting our top-down method, with its reliance on hand crafted behavior grammars, for a more robust approach. In the following sections, we describe an alternative methodology in which event representations do not relay on hand crafted behavior grammars, but rather, are learned directly from large amounts of data. We use this bottom-up approach to train grounded language models for the domain of broadcast sports video; and devote particular attention to the benefit of such models on practical video applications such as video search and speech recognition.

[14] See Orkin and Roy (2007) for first steps toward addressing this problem.

Chapter 5

Events in Sports Video

In this section we introduce a methodology for representing events in video for a grounded language model. Unlike the top-down approach introduced for virtual environments, this method does not rely on hand crafted grammars of behavior to interpret events in a domain. Rather, the method automatically learns event structures directly from large corpora of unlabeled data. By avoiding the need for hand built models of context, this bottom-up approach learns event representations that are robust enough for grounding language in the very challenging domain of broadcast sports video.

Broadcast video is a rich and unique data source, encoding both visual and linguistic information about the events that occur in the world. Learning grounded language models that exploit both of these information sources requires robust representations of the structure of these events. Unlike with videogames, the domain of broadcast video does not afford access to clean deterministic events that can be recognized without error or noise. Rather, video is complex and noisy, making it an extremely challenging domain for grounding language.

While the challenges to grounded language models for video are great, so are the potential rewards. Interest in video applications, both academic and commercial, has grown rapidly in the recent past as the availability of digital video has increased. The extreme popularity of sports video in particular is indicative of the great demand for applications in that domain. Such applications would benefit greatly by exploiting the relationship between words and the visual world encoded by grounded language models. In this chapter, we detail our bottom-up approach to representing video events for grounded language models and examine their effectiveness for improving the performance of two such practical applications: video search and speech recognition.

5.1.1 Events in Sports Video

In the previous section, we demonstrated the benefits of using a top-down approach to representing events for grounded language models in virtual environments. As a first attempt to ground language in sports video, one might adopt the same approach by trying to hand craft a grammar to represent events in sports.

Previous work has applied similar methods to recognizing events in surveillance video (e.g., Ivanov and Bobick, 2000; Hongen et al., 2004). In these approaches, complex events (e.g. *stealing a bag*) are recognized using hand crafted models that describe temporal relations between lower level sub-events (e.g., *putting down/picking up a bag*). In order to apply this method to the sports domain, one could imagine an analogous approach in which complex events (such as *home run*) are built up from temporal relations between the basic sub-events of sports video (e.g., hitting, throwing, catching, running, etc.).

Applying this approach to sports video, however, has a number of serious drawbacks. It is not trivial to specify by hand the temporal relations that make up complex events in sports. Such events often occur in a many different ways and under many different conditions, making them difficult for even an expert to manually encode[15]. Even when such relations can be specified by hand, the basic sub-events in sports video must still be identified. Unfortunately, identifying such basic events in video is infeasible given the limitations of current computer

[15] See Intille and Bobick (2001) for an example of a methodology in which temporal relations within an event are hand crafted.

vision algorithms.[16] Although advancing quickly, computer vision is unreliable (at best) for recognizing objects in images, and even less consistent in identifying events in video. It is just not the case that computer vision can recognize events such as *hitting a baseball* (which are defined by the movement of a small white ball moving at velocities so high, they are difficult even for a human to see).

In order to account for the limitations of a top-down approach to sports video, we focus on a method which does not rely on hand crafted relations between difficult to recognize events. Rather, we introduce a bottom-up method in which hierarchical temporal patterns of easy to identify low level features are automatically mined from large corpora of sports video.

5.1.2 From Grammars to Patterns

Our bottom-up approach to representing events in sports is based on the exploitation of the unique characteristics of broadcast video. Unlike home video, broadcast video is highly produced. It is made up of multiple different shots (often from many different cameras) and is brought together into a coherent whole by a human director. Broadcast sports video is no different; it exploits a large number of cameras (sometimes more than 15), pointing at different parts of the field, and relies on a human director to make sure that the audience follows the events of the game. Although each game may have a different director, the styles that each one employs are extremely consistent within a sport. Even across different teams, in different stadiums, and on different broadcast networks, the common directing styles of a sport embody a *language of film* that is highly regular within a particular domain.

For example, in baseball, events generally start with a view from behind the pitchers' mound in which the pitcher, batter, catcher and umpire are all visible (see Figure 5-1). Depending on what happens next, the scene may switch to a view of the field, the stands or to a close-up of the batter or pitcher. When a ball is hit into the air, its trajectory is tracked by the camera; and as it falls into the glove of a player, the camera zooms in to get a clear picture.

The elements that make up the language of film (i.e., the scene selection, the camera motion, etc.) are in place to insure continuity for the viewers. During often fast moving events, such human direction serves to limit the viewers' disorientation and guide their attention towards

[16] Intille and Bobick (2001) operate on manually defined traces of player movements.

the important aspects of the events. In this way, the various devices used to direct the production of a sports video can be seen as standing in for the attentional focus of the director. As events unfold, the director shifts the cameras to where the event is occurring, insuring that the audience and he are attending to the same thing (i.e. the event). By capturing this attentional information, the same techniques that are used to benefit human viewers, can also be used to help the machine represent events for grounding language.

It is often the case that patterns in the language of film can be found that correlate with high level events. Often these patterns are easier for a machine to recognize than the high level events themselves. For example, a very common pattern in baseball is for a sequence of scenes (or shots) in which a view of the pitching mound immediately jumps to a view of the field. Intuitively, this temporal pattern is highly correlated with a player hitting the baseball into the field. Because it is much easier for a machine to recognize these low level features (i.e. pitching scenes and field scenes) than the event itself (i.e. hitting a baseball), we can use the temporal pattern as a kind of "motion signature" for that event. By introducing more low level features (such as camera motion, audio classes, etc.), and finding more complex motion signatures, the machine is able to build up rich representations for many different types of complex events.

The following sections describe our methodology for learning temporal patterns to represent events for grounded language models. These temporal patterns act as analogues to the rules that were hand crafted for the behavior grammars described in Chapter 3. The notion of a formal grammar of behavior, however, is dropped in favor of a *codebook* of automatically learned temporal patterns. This codebook is used to represent events, not as a hierarchical event *tree*, but rather as a vector of temporal patterns that match low level features of a video. Unlike the top-down approach used for virtual environments, this bottom-up method generates events representation robust enough for grounding language in sports video.

Our approach to learning grounded language models operates in two phases. Section 5.2 describes the first phase, in which events that occur in the video are represented using hierarchical temporal patterns automatically mined from low level features. Section 5.3 details the second phase, in which a conditional probability distribution is estimated that describes the likelihood that a word was uttered given such event representations. (Appendix C describes the workflow for collecting and processing data with details of what software was employed).

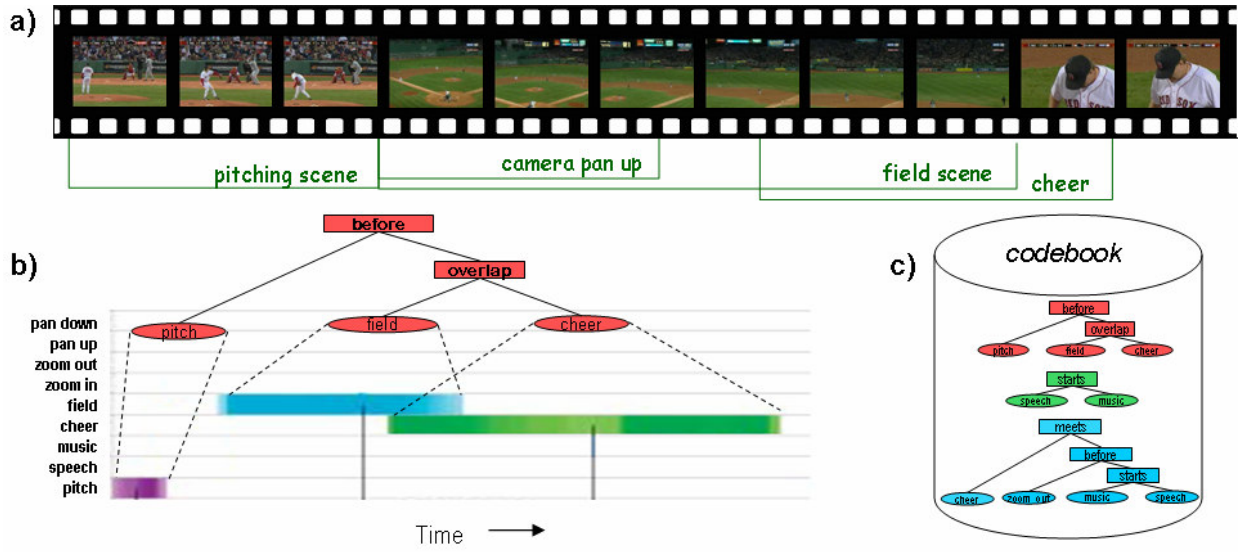

Figure 5-1. Overview of Event Representation in Video. a) Events are represented by first abstracting the raw video into visual context, camera motion, and audio context features. b) Temporal data mining is then used to discover hierarchical temporal patterns in the parallel streams of features. c) Temporal patterns found significant in each iteration are stored in a codebook that is used to represent high level events in video.

For expository reasons, the following sections describe these two aspects of our approach as they are applied to the domain of broadcast baseball games. Importantly, with the exception of the visual context features (see Section 5.2.1), nothing about this methodology is specific to baseball video. Section 6.3 describes and evaluates the methodology as it is applied to American football.

5.2 Representing Events

The following sections describe our method for representing events based on the consistencies exploited in the language of film for the baseball domain (see Section 6.3 for examples from the domain of American football). We describe our method in two phases: first, low level features of the video that can be easily and reliably detected are extracted from the video stream. Then in the second phase, temporal data mining is used to find patterns within these low level feature streams that correlate with higher level events.

5.2.1 Feature Extraction

The first step in representing events in video is to abstract the raw video stream into features that are more semantically meaningful than raw pixels. This step represents the only part of the grounded language modeling process which uses supervised learning, and is the only step which is specific for each domain being modeled. For the sake of simplicity, the next sections focus only on a single domain, i.e. the domain of baseball video.

Determining what features to extract from the video is a critical part of designing a grounded language model. The features must be informative, while at the same time be easily and reliably extractable from the video. Thus, extremely low level properties of the pixels (such as color values) are undesirable as they are not informative enough for language grounding. While at the same time high level features (such as the movement of the ball) cannot be reliably identified. The following sections describe three features types that provide a good balance between being informative and being easy to extract. These feature types are: visual context features, camera motion features, and audio context features.

Visual Context Features

Visual context features encode general properties of the visual scene being displayed in a video. These features correspond to the general type or category of a particular frame of video, e.g. whether the frame shows a view from behind the pitcher's mound or looking out onto the outfield. Figure 5-2 describes the steps for extracting visual context features in the baseball domain.

The first step in classifying visual context features is to segment the video into shots (or scenes) based on changes in the visual scene due to editing (e.g. jumping from a close up to a wide shot of the field). This process, called shot detection, is a well studied problem, and is based on frame by frame comparisons of video in which differences in color histograms are used to find points of sudden change (for a review, see Boreczky and Rowe, 1996). By applying a threshold to the differences in the color histograms of adjacent frames, boundaries can be detected which correspond to shot changes in the video. The set of adjacent frames (greater than 30 frames) that occur between two such sequential boundaries is defined as a shot.

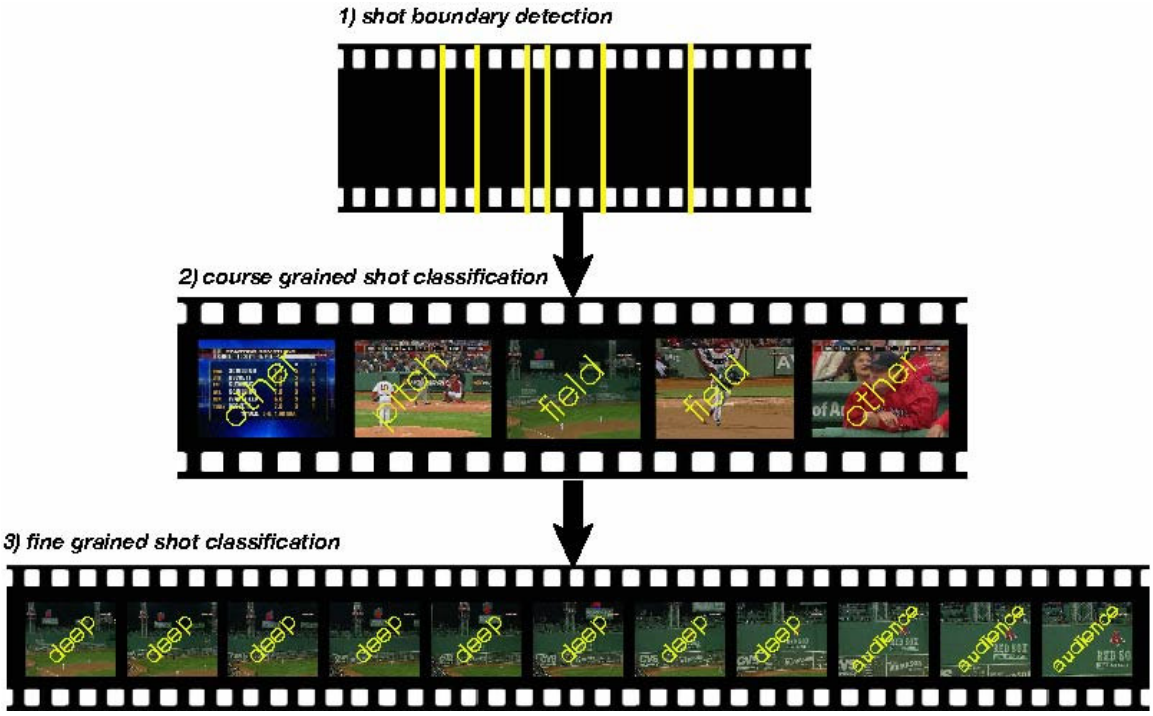

Figure 5-2 Overview of scene classification for baseball games. Color and motion features are used to break raw video into individual shots. Key frames within each shot are then classified into course categories (i.e., field, pitch, other). Shots categorized as *field* are then broken into sub key frames (one for every ten frames) and sub-categorized.

In this work, we use a method of shot detection, developed by Tardini et al. (2005), which focuses not on directly adjacent frames in a video, but rather on windows of frames and the gradient of change that is seen within the window. They provide a fast and efficient open source implementation of their method, which has demonstrated high performance on sports video.

After the video is segmented into shots, the shots are categorized using a two step classification process. First, individual frames are sampled from the beginning, middle and end of each shot. These frames, called key frames, are treated as exemplars of the entire shot and are used as the basis for shot classification. Each key frame is represented as a vector of low level features that describe its various visual properties, such as color distribution, entropy, etc. (see Appendix B for the complete list of features). A set of manually labeled key frames is then used to train a decision tree classifier (with bagging and boosting) using the WEKA machine learning toolkit (Witten and Frank, 2005). Frames are categorized into three high-level domain-dependent shot types: *pitching-frames* which include characteristic shots of the pitching mound;

field-frames which include shots focusing on any part of the field; and *other-frames* which include all other shots from close ups of players, to commercial breaks. Evaluations show very high performance classification of these categories, with f-scores ranging from 0.94 to 0.97 (see Appendix B for more details of the evaluation). [17]

Shots are classified based on the categorization of frames in a hierarchical manner. For each shot, if any key frame within that shot is classified as a *pitching-frame*, then the whole shot is classified as a *pitching-shot*. If no *pitching-frame* is found, but a *field- frame* is found, then the shot is classified as a *field-shot*. Otherwise, the shot is classified as *other-shot*. [18]

Given this first level classification, a second pass is made through the data in order to subcategorize the *field-shots* into more fine grained classifications. Because field shots often contain a great deal of activity, we resample key frames from the shots at a rate of one key frame per 10 frames[19]. Since field shots represent a minority of the total shots in a sports video, sampling at such a fine rate is not prohibitively expensive.[20] Given this new sampling, each key frame is now subcategorized (again using a boosted and bagged decision tree) into one of the following categories: *audience-frame, on-base-frame, infield-frame, outfield-frame, deep-outfield-frame, full-field-frame, misc-frame,* and *running-frame,* with f-scores ranging from 0.80 to 0.94. (see Appendix B for more details on evaluations).

Audio Context

The audio stream of a video can also provide useful information for representing events. For example, in baseball video, the cheering of the crowd is often an indicator of the difference between a base hit and a foul ball. In order to capture such audio information, supervised methods are used to train classifiers to categorize the audio stream from mpeg video into a stream of discrete categories of *speech, cheering*, and *music.*

---

[17] F-score is equal to the harmonic mean of precision and recall.

[18] This method of voting was implemented specifically to address problems of missed shot boundary detection. If the system incorrectly misses a boundary between, for example, a *pitching-shot* and an *other-shot*, it is preferable to label that combined shot as a *pitching-shot*. Biasing classification in this way insures that pitching-shots and field-shots are available for further processing.

[19] Video is recorded at 29.97 frames/sec.

[20] The percentage breakdown of shot types in the baseball corpus is *field-shot:* 12.4%; *pitching-shot:* 17.4%; *other-shot:* 70.2%.

Classification operates on a sequence of overlapping 30 ms frames extracted from the audio stream. For each frame, a feature vector representation is computed using, mel-frequency cepstral coefficients (MFCCs). Such representations have been shown to perform well in speech related tasks (e.g., automatic speech recognition: Rabiner and Juang, 1993). In addition to MFCCs, energy, the number of zero crossings, spectral entropy, and relative power between different frequency bands is included in the vector representation.[21]

A series of binary decision trees is trained (with boosting and bagging) for each of the audio classifications described above using the WEKA machine learning toolkit (Witten and Frank, 2005). Binary classifiers were used to allow for the possibility of multiple overlapping audio classifications (e.g., speech occurring while the crowd is cheering). Each classifier is applied to each frame, producing multiple sequences of class labels. F-scores for these classifiers range from 0.67 (for cheering) to 0.93 (for speech).[22] These labels are then smoothed using a dynamic programming cost minimization algorithm (similar to what is used in Hidden Markov Models) (Fleischman, Roy, and Roy, 2007). Once these streams of discrete audio labels are extracted, they are input along with the visual context and camera motion features, to a temporal data mining algorithm which discovers patterns that will be used to represent events for the grounded language model.

Camera Motion Features

In addition to visual and audio context features, camera motion also provides informative features. Unlike visual and audio context features, which provide information about the global situation that is being observed, camera motion features represent more precise information about the actions occurring in a video. The intuition here is that the camera is a stand in for a viewer's focus of attention. As actions occur in a video, the camera moves to follow it; this camera motion thus mirrors the actions themselves, providing informative features for event representation.

Like shot boundary detection, detecting the motion of the camera in a video (i.e., the amount it pans left to right, tilts up and down, and zooms in and out) is a well-studied problem. Many

[21] These features are adequate for classification. For more accurate results additional features that encode pitch information should also be included. Selecting additional useful audio features is beyond the scope of this thesis.

[22] Appendix B details the performance of the classifiers on a held out test set.

techniques have been proposed using, for example, optical flow (Berger et al., 1992), or mpeg motion vectors (Tan et al., 2000). We use an open source implementation of Bouthemy et al. (1999) in which camera motion is estimated by fitting the parameters of a two-dimensional affine model to differences between each pair of sequential frames in a video.[23] The parameters of this model are converted to a three dimensional feature vector that represents the amount of pan, tilt, and zoom estimated in the video between successive frames.

The output of the above process is a stream of real valued numbers. However, as described in the next section, the temporal data mining algorithm that is employed requires discrete categories of low level features. In order to convert the stream of real valued pan, tilt, zoom features into a stream of discrete category features, a Hidden Markov Model (HMM), implemented with the Graphical Modeling Toolkit is used.[24]

A 15-state, fully connected 1st order HMM is trained on unlabeled camera motion data. [25] This HMM is then used to decode continuous streams of pan/tilt/zoom values from a video clip into a stream of discrete state IDs (one ID for each state in the model). These state IDs represent clusters of characteristic camera behaviors often seen in the video. For example, the HMM trained on unlabeled baseball video clusters together camera motions in which the camera zooms in quickly while panning slowly to the left. This characteristic behavior is interesting because it is often seen (among other times) after a batter is walked, as the cameraman follows him as he goes from home plate to first base. Discritizing camera motion vectors in this way facilitates the mining of temporal patterns between camera motion, visual and audio context features and provides the basis for representing events in the grounded language model.

[23] http://www.irisa.fr/vista/Motion2D/index.html

[24] http://ssli.ee.washington.edu/~bilmes/gmtk/

[25] The choice of 15 states for the HMM was based on informal analysis of the resulting clusters. With more formal tuning, performance of the system may be improved.

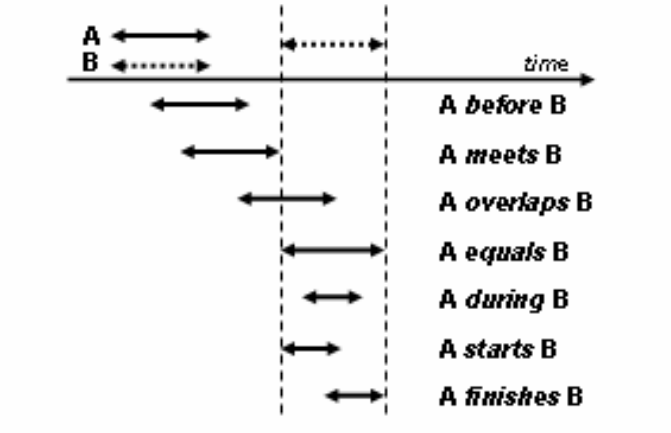

Figure 5-3. Allen relations. Any two events must be in one of these seven asymmetric temporal relations.

5.2.2 Temporal Pattern Mining

As described above, we follow previous work in event recognition (Fleischman et al., 2007), in which high level events are represented using temporal relations between lower level sub-events. Because of the limitations of computer vision technology, detecting ideal sub-events for sports video (e.g. throwing, catching, running, etc…) is not feasible. Thus, this work focuses on using temporal relations between simple low level features that correlate with events in sports video.

Unlike previous work in event recognition (e.g., Intille and Bobick, 2001; Ivanov and Bobick, 2000; Hongen et al., 2005) in which representations of the temporal relations between low level events are built up by hand, this thesis borrows methods from work in temporal data mining to automatically discover such patterns from a large corpus of unannotated video. Importantly, these techniques allow discovery of temporal patterns that are not strictly sequential, but rather, are composed of features that can occur in complex and varied temporal relations to each other.

To find such patterns automatically, we build on previous work in video content classification (Fleischman and Roy, 2006) in which temporal data mining techniques are used to discover event patterns within streams of lower level features (see Figure 5-1). The algorithm we use is based on work by Hoppner (2001) and Cohen (2001) and is fully unsupervised. It proceeds by examining the relations that occur between multiple features streams extracted from the video. The algorithm looks for temporal relations that occur between these low level

features, keeping track of how often each relation is observed. After observing the entire video corpus, it uses statistical significance testing to determine which relations are significant. The algorithm makes multiple passes through the data, and relations between individual features that are found significant in one pass (e.g. [OVERLAP, *field-scene, cheer*]), are themselves treated as individual features in the next pass. This allows the system to build up higher-order nested relations in each iteration (e.g. [BEFORE, [OVERLAP, *field-scene, cheer*], *field scene*]]).

Pseudo-code for the algorithm is presented in Figure 5-4. The algorithm processes the multiple streams of features frame by frame, at each point checking if any low level features have just ended (e.g., music stops playing, a shot of the pitcher switches to a shot of the field, etc.). If this has happened, that low level feature is compared with other features in three different sets: 1) the set of features that have also just ended; 2) the set of features that are still ongoing; and 3) the set of features that have recently ended. We define a time limited Short Term Memory (STM) in which these recently completed features are stored (set to 30 frames)[26]. The size of this STM acts as a windowing length such that only events within this window can be compared.[27]

For each pair of features (i.e., the newly ended feature and the features selected from one of the three sets), the temporal relation that exists between them is calculated. Following Allen (1984), any two features must be in one of seven temporal relations with each other (see Figure 5-3).[28] This relation is now treated as a new composite feature: i.e., a feature composed of the relation and the two features (e.g. [OVERLAP, *field-scene, cheer*]). A record is then updated which keeps track of how often each feature has been observed Further, if the composite feature has already been found significant (i.e., in a previous iteration), it is added to the set of features that just ended or is still active, so that it can be composed with other features (as described above). By recursively adding features that were previously found significant, the system is able to discover higher-order nested relations in each iteration.

---

[26] The STM window parameter was set informally in order to capture long distance relationships while maintaining computational tractability.

[27] This differs from STM in Cohen (2001) which is limited by size, not time, and thus allows comparisons between events that occur arbitrarily far apart in time.

[28] To avoid missing good examples due to small differences in timing, relations are based on soft intervals, not hard constraints. Thus, for example, instead of A meets B iff (end of A)==(start of B), we define A meets B iff (end of A)-α < (start of B) < (end of A)+α. For these experiments, α is set to 5 frames.

```
LEARN-PATTERNS(matrix data)
  significant Events ← ∅
  counts ← ∅
  for i < maxIteration
    foreach timeslice t in data
      events ← FIND-COMPOSITE-EVENTS(t)
      increment counts
    foreach event F in events
      if F is significant
        add to significantEvents
    clear counts
  return significantEvents

FIND-COMPOSITE-EVENTS(vector t)
  candidateCompositeEvents ← ∅
  justFinishedEvents ← list of events ending at t
  stillActiveEvents ← list of events still open at t
  foreach event F in justFinishedEvents
      //find present relations (equals, finishes)
      FIND-RELATIONS(F, justFinishedEvents)
      //find future relations (overlap, meet, during, start)
      FIND-RELATIONS(F, stillActiveEvents)
      //find past relations (before)
      FIND-RELATIONS(F, STM)
  updateSTM()
  return candidateCompositeEvents

FIND-RELATIONS (event F, list eventSet)
  foreach event G in eventSet
  compositeEvent ← temporal relation btw F and G
  if compositeEvent is significant
      push onto eventSet
  else
      push onto candidateCompositeEvents
```

Figure 5-4 . Pseudo-code for mining hierarchical temporal patterns from large unannotated video corpora.

Once the algorithm has examined all the frames in the video it cycles through each observed composite feature and checks if that feature is significant. Similar to Cohen (2001), we use the *phi* statistic to measure the significance of a composite feature. For each composite feature, we create a 2-by-2 contingency table that describes how often different sub-features of the

Table 5-1. Contingency table used to calculate significance of event during(cheer, pitching-shot).

during	**pitching-shot**	**¬ pitching-shot**
cheer	A	B
¬cheer	C	D

composite were observed in that temporal relation. For example, in the contingency table shown in Table 5-1, A represents how often a *cheer* occurred during a *pitching-shot*, B represents how often a *cheer* occurred during any other type of feature, C represents how often a non-*cheer* feature occurred during a *pitching-shot*, and D represents how often any non-*cheer* occurred during any non-*pitching-shot*.

Phi can now be calculated using Equation 5-1, in which χ^2 is the chi square statistic calculated from the contingency table and N is the table's total.[29] The phi statistic provides a measure between 0 and 1 of the strength of the association between the sub-features in a composite feature and can be tested for statistical significance as with a Pearson r. In order for a composite event observed by our system to be considered significant, its phi must both be greater than some value *rho* (set to 0.05) as, as well as, significant above a threshold *alpha* (set to 0.95).

$$\phi = \sqrt{\frac{\chi^2}{N}} \qquad \textbf{5-1}$$

In this work, the algorithm was set such that each iteration produced significant patterns of increasingly higher orders (e.g., the first iteration produced relations one level deep, the second produced relations two levels deep, etc.). After all iterations are completed, the output of the program is a set of significant temporal patterns discovered from the unannotated data. In this work we set the maximum number of iterations to three (see Table 5-2 for example patterns).

These significant temporal patterns make up a codebook which is then used as a basis for representing a video. The term codebook is often used in image analysis to describe a set of

[29] The use of the chi square statistic is somewhat arbitrary. Other measures of variable dependence can be used as well (e.g. Mutual Information). Although not discussed, log likelihood (Dunning, 1993) was tested with no significant difference in performance observed.

Table 5-2. Example hierarchical temporal patterns learned in three iterations of the temporal data mining algorithm. Freq. represents the number of times the pattern was observed in the data and phi represents the strength of the association between features.

Patterns	phi	Freq.
Iteration 1		
starts(field)(cheering)	0.0705	238
during(motion-cluster-14)(pitch)	0.121	48063
meet(pitch)(field)	0.310	7337
equals(field)(outfield)	0.376	226
meet(infield)(on-base)	0.439	1919
Iteration 2		
finishes(meet(pitch)(field))(outfield)	0.060	2158
meet(starts(pitch)(motion-cluster-14))(field)	0.099	4711
during(infield)(meet(pitch)(field))	0.099	6366
starts(pitch)(meet(motion-cluster-14)(motion-cluster-3))	0.125	3080
overlap(meet(pitch)(infield))(field)	0.209	1865
Iteration 3		
starts(overlap(meet(motion-cluster-14)(motion-cluster-3))(cheering))(pitch)	0.180	276
finishes(meet(meet(pitch)(infield))(outfield))(field)	0.186	306
finishes(during(motion-cluster-2)(meet(pitch)(deep-outfield)))(field)	0.203	408
finishes(starts(meet(pitch)(infield))(overlap(motion-cluster-14)(cheering)))(starts(field)(motion-cluster-2))	0.205	36
equals(before(equals(motion-cluster-2)(infield))(outfield))(during(motion-cluster-12)(field))	0.224	32

features (stored in the codebook) that are used to encode raw image data. Such codebooks are used to represent raw video using features that are more easily processed by the computer.

Our framework follows a similar approach, encoding raw video using a codebook of temporal patterns. Encoding proceeds by first segmenting a video into discrete events; "at bats" in the case of baseball (see Section 5.3 for more details). Each event is then abstracted into the visual context, camera motion, and audio context feature streams (as described in Section 5.2.1). These feature streams are then scanned, looking for any temporal patterns (and nested sub-patterns) that match those found in the codebook. A vector representation is generated for each events based on the patterns from the codebook that match the event.

This vector representation is the same length for each event and is equivalent to the number of temporal patterns in the learned codebook (approximately 1900 patterns for the baseball domain, and 1000 for the football domain). Each element in this vector is associated with a specific pattern from the codebook. The value of that element depends on the duration for which that pattern was observed in that particular video event.[30] These values range from 0 (if the pattern was not observed in the video) to m, where m is the total number of frames in the video (if the pattern was observed for the entire video event).

For example, say that temporal data mining generated a codebook with four patterns: A, B, C, and A-*before*-B (where A-*before*-B is a higher order combination of patterns A and B). A video event is now scanned and patterns A and C are observed for 10 frames each. The codebook represents this video event as the vector (10, 0, 0, 10). Now, another video event is observed with pattern A for 10 frames, pattern B for 20 frames, and pattern A-*before*-B for 10 frames. This event is represented as the vector (10, 20, 10, 0). Representing events in this way not only captures long distance dependency information (in the composite patterns) but also encodes duration information; two information sources that are not easily captured in standard dynamic models (e.g., HMMs).

This process of representing video events is analogous to the parsing method described in Section 3.2. However, in that method a behavior grammar was used to generate well formed tree representations for sequences of low level actions. Here, a codebook is used to match multiple, parallel streams of low level features in order to generate a vector of hierarchical matched patterns. Given this method for representing the non-linguistic context of a video, we can now examine how to model the relationship between such context and the words used to describe it.

5.3 Linguistic Mapping

Having described how non-linguistic context is represented in a grounded language model, we can now turn to modeling the relationship between this context and the language used to describe it. Modeling this relationship assumes that the speech uttered in a video refers often

[30] The duration of a composite event is equivalent to the duration of its shortest sub event.

(although not exclusively) to the events being represented by the temporal pattern features. Given such an assumption, we formalize the relationship between words and context as a conditional probability distribution, i.e. as the probability of a word given a vector of temporal patterns. We estimate these conditional distributions using a framework similar to that used for training translation models in research on statistical machine translation (MT).

In statistical machine translation, conditional probability distributions are used to model the likelihood that a word(s) from one language (e.g. English) can be translated into a word(s) from another (e.g., French) (Manning and Schutze, 2001). These conditional distributions (called translation models) are estimated based on a parallel corpus in which a sentence in English is paired with a translation of that sentence in French. To estimate grounded language models we make an analogy between the parallel corpus used for MT, and the utterances and representations extracted from sports video. Just as MT systems are trained using parallel corpora of French translations and the English sentences to which they refer, grounded language models exploit a parallel corpus of natural language utterances and representations of the context in which they were said.

In order to create this parallel corpus, we first segment the raw videos of full sports games into a set of individual video clips, where each clip shows a different event that occurred in a game. The nature of this segmentation differs for each domain; for broadcast baseball, the natural granularity for each event is at the level of individual pitches (i.e. each pitch is treated as an independent event).[31]

The first step in segmenting baseball video is to extract the visual context features from the video and look for scenes classified as *pitching-shots* (see Section 5.2.1). We follow previous work in sports video processing (Gong et al., 2004) and define an event in a baseball video as any sequence of shots starting with a *pitching-shot* and continuing for four subsequent shots. This definition follows from the fact that the vast majority of events in baseball start with a pitch and do not last longer than four shots.[32]

[31] See Section 6.3 for a discussion of segmentation in American football.

[32] Although some events, such as stolen bases, do not involve an actual pitch, even these are very often preceded by a shot classified as a *pitching-shot.*

For each of these events in our corpus, a representation of the event is generated using the temporal pattern features as described in Section 5.2.2. These representations are then paired with all the words from the closed captioning transcription that occur during that event (plus or minus 10 seconds). Because these transcriptions are not necessarily time synched with the audio, we use the method described in Hauptmann and Witbrock (1998) to align the closed captioning to the announcers' speech.

This method uses dynamic programming to find the best possible alignment between the closed captioning transcript and the output of an automatic speech recognition (ASR) system applied to the audio of the game. ASR is performed using the Sphinx 3 speech recognition system with a language model trained specifically on the closed captioning for that segment of video. Unlike the closed captioning transcription, the output of the ASR system contains accurate time codes for exactly when words were uttered. Although formal alignment evaluations were not performed, informal analysis confirms that the dynamic programming algorithm needs relatively few correctly recognized words to use as anchor points in order to estimate reasonable time codes for the closed captioning transcription. However, this alignment process does introduce noise in both training and testing, suggesting more sophisticated approaches (e.g., Hazen, 2006) would lead to improved performance.

Having generated a parallel corpus of utterances and representations of the contexts in which they were said, we are now faced with the question of how to use the corpus to estimate our grounded language model. Our initial approach continued the analogy and adopted a methodology similar to what is used in Machine Translation. Fleischman and Roy (2006) describes a system which follows IBM Model 1 (Brown et al., 1993), in which each word in an utterance is assumed to be generated independently by exactly one temporal pattern feature from its corresponding event representation. The expectation-maximization (EM) algorithm is used to estimate the probability of a word given a video representation based on the equation:

$$p(W \mid P) = \frac{C}{(l+1)^k} \prod_{j=1}^{k} \sum_{i=0}^{l} p(W_j \mid P_i) \qquad \textbf{5-2}$$

where k is the number of words in the utterance W, W_j is the word in W at position j, l is the number of temporal pattern features in the vector representation P, P_i is the feature in P at position i, and C is a constant.

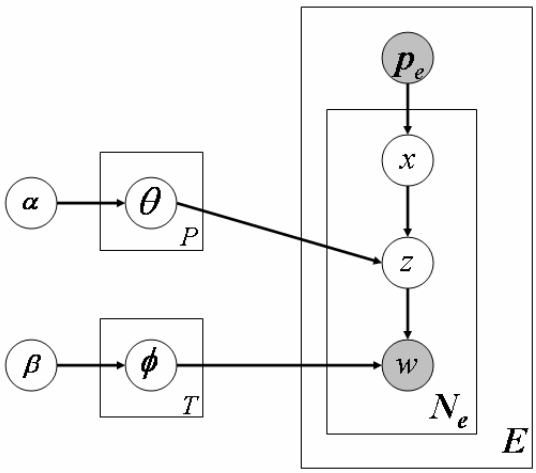

Figure 5-5 Plate notation for a generative process corresponding to an adapted AT model (Steyvers et al., 2004). For each event *e* (of the *E* events in the corpus), each word *w* (of the N_e words paired with *e*) is chosen by first randomly selecting a pattern *x* from the vector of patterns (p_e) paired with *e*. A topic *z* is then randomly chosen from the multinomial distribution in θ associated with *x*. *w* is then randomly chosen from the multinomial distribution in ϕ associated with *z*. α and β are parameters on the dirichlet priors for θ and ϕ.

Although such models are satisfactory (see Fleischman and Roy, 2006 for pilot evaluations), this methodology is not ideal for estimating grounded language models. Practically speaking, because using EM requires examining all possible alignments between words and features, training such MT based models on large corpora can be expensive. A more fundamental concern, however, lies in the basic generative story for MT. While the assumption that words are generated by elements from their parallel representation is reasonable for French translations of English sentences, it is far less reasonable to posit that individual temporal pattern features generate words in an utterance that describes an event. Unlike words in French, temporal pattern features are very abstract representations of non-linguistic context that do not necessarily map directly onto words in English. Instead, a more natural assumption posits that a layer of hidden variables intervenes between the temporal features and the English words.

Informally, we want a generative story that describes the sports announcer as she is watching a game and commenting on it. Whenever some event occurs in the game, the announcer's perceptual system encodes the event as a temporal pattern representation. This representation then activates concepts (i.e., latent variables) in the announcer's mind that have

come to be associated with such representations. These concepts in turn generate words that describe those concepts that have been activated.

In order to model this generative story, we follow recent work in automatic image annotation (Barnard et al., 2003; Blei and Jordan, 2003) and natural language processing (Steyvers et al., 2004), in which hierarchical Bayesian models are used to represent such hidden (or latent) variables. We follow closely a model used to represent text documents, i.e., the Author-Topic (AT) model (Steyvers et al., 2004), which is a generalization of Latent Dirichlet Allocation (LDA) (Blei et al., 2005).[33]

LDA is a technique that was initially developed to model the distribution of topics discussed in a large corpus of text documents. The model assumes that every document in a corpus is made up of a mixture of topics, and that each word in a document is generated from a probability distribution associated with one of those topics. The AT model generalizes LDA, saying that the mixture of topics is not dependent on the document itself, but rather on the authors who wrote it.[34] According to this model, for each word (or words) in a document, an author is chosen uniformly from the set of the authors of the document. Then, a topic is chosen from a distribution of topics associated with that particular author. Finally, the word is generated from a distribution associated with that chosen topic. We can express the probability of the words in a document (W) given its authors (A) as:

$$p(W \mid A) = \prod_{m \in W} \frac{1}{A_d} \sum_{x \in A} \sum_{z \in T} p(m \mid z)\, p(z \mid x) \qquad \textbf{5-3}$$

where m is a word in W, x is an author in A, A_d is the number of authors in A, and z is a topic in the set of latent topics T induced from a large corpus of unannotated training documents.

We use the AT model to estimate our grounded language model by making an analogy between documents and events in video. In our framework, the words in a document correspond to the words in the closed captioning transcript associated with an event. The authors of a document correspond to the temporal patterns representing the non-linguistic

[33] In the discussion that follows, we describe a method for estimating unigram grounded language models. Estimating bigram and trigram models can be done by processing on word pairs or triples, and performing normalization on the resulting conditional distributions. See Section 7.1.1 for more details.

[34] LDA can be seen as a special case of the AT model, in which each document was generated by a single, unique author.

context of that event. We modify the AT model slightly, such that, instead of selecting from a uniform distribution (as is done with authors of documents), we select patterns from a multinomial distribution based upon the duration of the pattern. The intuition here is that patterns that occur for a longer duration are more salient and thus, should be given greater weight in the generative process. We can now rewrite (1) to give the probability of words during an event (W) given the vector of observed temporal patterns (P) as:

$$p(W \mid P) = \prod_{m \in W} \sum_{x \in P} \sum_{z \in T} p(m \mid z)\, p(z \mid x)\, p(x) \qquad \textbf{5-4}$$

where x is now a temporal pattern feature in the vector P.

Figure 5-5 describes this generative process using plate notation. For each event e (of the E events in the training corpus), each word w (of the N_e words paired with e) is chosen by first randomly selecting a pattern x from a multinomial distribution over the patterns in the vector (p_e) paired with e. A topic z is then randomly chosen from the multinomial distribution in θ associated with x. w is then randomly chosen from the multinomial distribution in φ associated with z. α and β are parameters on the Dirichlet priors for θ and φ. We use the Matlab Topic Modeling toolkit[35] to train the parameters θ and φ using Gibbs sampling, a Markov Chain Monte Carlo technique.

Having described the methodology by which grounded language models are trained, in the next sections we describe evaluations run to examine the ability of grounded language models to improve performance on two practical video applications. We describe experiments on video information retrieval and automatic speech recognition.

[35] http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm. The toolkit was modified to enable sampling temporal pattern features from a multinomial distribution, as opposed to sampling authors from a uniform distribution (see Appendix C).

Chapter 6

Application: Video Search

The increasing prevalence of video data on the internet has sparked a growing interest in the field of video search. Like traditional information retrieval (IR), video search focuses on the task of finding specific videos, or types of videos, from within a large collection of data. This similarity has led to a number of approaches (both academic and commercial) that attempt to repurpose techniques that have been successful for finding documents on the internet. Such approaches treat the speech uttered in a video like words written in a document, and then apply traditional text-based IR algorithms to transcriptions of the videos' speech. These approaches are popular because of their scalability and the lack of human supervision required to index large corpora. However, applying such methods to searching sports video faces serious challenges, even when human generated speech transcriptions are available (for example, in the closed captioning stream).

Unlike the case with text documents, the occurrence of a query term in a video's audio track is often not enough to assume the video's relevance to that query. For example, when searching through video of baseball games, returning all clips in which the phrase "home run" is uttered

results primarily in video of events where a home run does not actually occur. This follows from the fact that in sports, as in life, people often talk not about what is currently happening, but rather, they talk about what did, might, could, should, will or won't happen in the future.

Traditional IR techniques cannot address such problems because they model the meaning of a query term strictly by that term's relationship to other terms. To build systems that successfully search video, IR techniques must exploit not just linguistic information but also elements of the non-linguistic context that surrounds language use. A great deal of research has addressed this issue by designing video search techniques that rely on supervised methods to classify events (for a review, see Snoek and Worring, (2005). The majority of these systems do not index events by natural language query terms (as traditional IR approaches do), but rather, categorize events using classifiers trained on hand labeled examples of predefined event types (e.g. *home runs*, *ground balls,* etc.).[36] Although these approaches can be useful, such supervised approaches to video retrieval are labor intensive for the system designers, who must label examples and train the concept classifiers. Further, they limit the usability for the system's users, who cannot query the system for whatever they want, but rather, are given access only to those predefined event types which the designers thought to include.

In this section, we present experiments that compare traditional text-based approaches to video search to a method for content-based indexing of sports video that is based on the grounded language models introduced in this thesis (earlier versions of this work appear in Fleischman and Roy, 2007b and Fleischman and Roy, 2007a). Our method maintains the advantages of traditional IR approaches while incorporating contextual information in an unsupervised manner. In the following sections we examine the effectiveness of this approach in retrieving video events from a held out test set of broadcast baseball and American football games. Results indicate that performance of the system using the grounded language model is significantly better than traditional text based approaches. These results not only validate the ability of grounded language models to learn the meaning of words uttered in sports video, but also, indicate the practical benefits of using grounded language modeling in solving a real world task.

[36] For an interesting exception see Babaguchi et al. (2002) in which hand chosen terms from the closed captioning stream are used to index a limited set of predefined events.

6.1 Grounded video indexing

The majority of work on video search in the sports domain, (e.g. Gong et al., 2004), focuses on retrieving video data using a set of supervised classifiers that categorize events into pre-determined concepts (e.g. *homerun, infield out, outfield hit etc.).* Such supervised systems can be seen as discovering mappings from a **closed** set of query terms to a closed set of events (as in Babaguchi et al., 2002). However, the most successful video search systems, like the most successful text search systems, allow users to use an **open** set of query terms to find any event that they may wish. A supervised system can only perform such an open task with the addition of a function to map (automatically or manually) from an open set of query terms to its pre-defined set of event classes.[37] For example, a query using the natural language term "homer," must explicitly be mapped to a query for the predefined event type: *home run*. Importantly, if no predefined event type matches the users query, the system cannot return a result.

The goal of our approach is to develop a method which maps an open set of query terms to an open set of events without the use of explicit mapping functions to predefined event types. Our method is based on a text-based approach to document retrieval, the language modeling approach of Song and Croft (1999), which allows for open natural language queries and does not rely on supervised indexing methods. By extending this method with grounded language models, we can reap the benefits of traditional methods while avoiding many problems faced when applying text based methods to video search.

In Song and Croft (1999), documents relevant to a query are ranked based on the probability that each document generated each query term. For video search, we make a natural analogy between words in a document and speech uttered in a video. We can now rank the relevance of each event in a video corpus to a user query using the following formula (adopted from Song and Croft (1999) :[38]

[37] This is often done in the news domain in which video is indexed based on hundreds of classifiers and natural language queries are converted (often manually) into Boolean expressions of predefined event types Snoek and Worring, (2005).

[38] In these experiments, only unigram language models are used. However, the general framework can be easily expanded to bigrams and trigrams, as is suggested by the experiments on automatic speech recognition, discussed in Section 7.1.1.

$$p(query \mid event) = \prod_{word}^{query} p(word \mid caption) \quad \textbf{6-1}$$

Here, *query* is defined as a set of independent words, i.e., *word.* Each *event* in the corpus has associated with it a set of terms (i.e., *caption*) uttered in the context of that event which are extracted from the closed captioning transcription (see Section 5.3).

In our experiments, we follow Song and Croft (1999) in treating the probability of a word given a caption as an interpolation between the probability of the query term given the words in the caption and the probability of the word in the entire corpus:

$$p(word \mid caption) = \omega * P_{caption}(word \mid caption) + (1 - \omega) * P_{corpus}(word) \quad \textbf{6-2}$$

Here ω is a weighting coefficient and the $P_{caption}$ and P_{corpus} are maximum likelihood estimates (using add N smoothing) of the probability of the word in the caption and the probability of the word in the corpus, respectively.[39]

In extending the language modeling approach to incorporate contextual information in the video, we make the simplifying assumption that the relevance of an event to a query can be modeled as two independent probabilities: the probability of the query word given the speech of the announcer, and the probability of the word given a representation of non-linguistic context of the event. We formalize this by extending Equation 6-1:

$$p(query \mid event) = \prod_{word}^{query} p(word \mid caption)^{(1-\alpha)} * p(word \mid video)^{\alpha} \quad \textbf{6-3}$$

The *p(word|caption)* is estimated using Equation 6-2, while the *p(word|video)* is estimated as in Equation 5-4. α is a weighting coefficient used to bias the system to the different sources of information.

In the next sections we evaluate the grounded language indexing method on two sports domains: baseball and American football. Although the majority of work focuses on baseball, results from American football serve to reinforce the conclusions drawn from the baseball domain. For each set of experiments, we describe the data sources, generation of artificial query terms, experiments and results. We conclude with a discussion of the results.

---

[39] Song and Croft (1999) use the more sophisticated Good-Turing smoothing. The significance of using such techniques given the small size of the captions used here has not been explicitly examined and is left for future work. In this thesis we set N to 0.001 and ω=0.5 based on an informal tuning.

6.2 Experiments: Baseball

6.2.1 Data

Evaluating the effects of grounded language models on video search performance poses a number of challenges. Traditional evaluation methodologies focus on the use of a standardized corpus from which events can be searched, as well a standardized set of queries for which the relevance of each event has been manually annotated. Performance of various systems can then be compared based on the precision with which relevant results are returned from the corpus for each query.[40]

As standardized corpora are unavailable for the baseball domain, we generated our own corpus by recording games from broadcast television. The corpus is composed of 99 Major League Baseball games from the 2006 season totaling approximately 275 hours and 20,000 distinct events (where events are defined as sequences of shots, as discussed in Section 5.3). These games represent data from 25 teams in 23 stadiums, broadcast on five different television stations. From this set, six games were held out for testing (15 hours, 1200 events, nine teams, four stations).[41] From this test set, 237 highlights were hand annotated for use in evaluation. A highlight is defined as any event that results in a player either *out* or *safe*. These highlights are generally considered the more interesting events in baseball, and include home runs, strikeouts, etc.

Each highlight is annotated with one or more of 13 labels according to the type of the event (e.g., *strikeout vs. homerun)*, the location of the event (*e.g., right field vs. infield*), and the nature of the event (e.g., *fly ball vs. line drive*). See Figure 6-4 for a complete listing of categories. Although only highlights are hand annotated, both highlights and non-highlights are used in the test set. Thus, retrieval operates over the complete set of events in a game (which is significantly more challenging than retrieval from just highlights alone).

---

[40] Although precision is reported throughout these experiments, other measures of performance have been suggested in the literature. Mean Average Precision (MAP), for example, is a measure of weighted precision which gives higher points when results are returned in order of their relevance and gives more weight to queries for infrequent events. We focus on precision in these experiments to insure ease of interpretability.

[41] Although our methodology is unsupervised, and thus, does not require a held out test set for evaluation, we choose to apply this more stringent evaluation procedure both to reduce the amount of hand annotation required for evaluating performance, as well as, to demonstrate the generality of the approach to unseen data.

TOPIC 3-10		TOPIC 3-25		TOPIC 3-34		TOPIC 3-42	
to_third_base	0.02325	in_the_air	0.03162	at_first_base	0.03532	swing_and_miss	0.0226
first_out_of	0.01424	the_air_to	0.02227	to_first_base	0.01669	with_two_outs	0.01795
two_down_in	0.01403	the_first_out	0.01252	in_this_game	0.016	full_count_now	0.0154
on_the_ground	0.01351	left_center_field	0.00962	the_double_play	0.01212	of_the_way	0.00964
the_ground_to	0.01288	the_second_out	0.00909	up_for_the	0.00983	count_now_to	0.0092
to_end_the	0.01184	there's_two_down	0.00685	game_hitting_streak	0.0096	the_ball_game	0.00897
pops_it_up	0.01142	out_number_one	0.00685	first_base_line	0.00789	down_and_away	0.00842

TOPIC 2-9		TOPIC 2-13		TOPIC 2-22		TOPIC 2-40	
center_field	0.06001	makes_the	0.03665	strike_out	0.04514	second_base	0.05463
left_field	0.03657	the_catch	0.03353	swing_and	0.03785	the_inning	0.04036
to_left	0.03215	out_number	0.02135	and_miss	0.03219	double_play	0.02378
to_right	0.0285	it_up	0.01112	at_bat	0.02523	in_time	0.02104
to_center	0.02274	for_out	0.01076	strikes_out	0.02007	up_and	0.01809
the_wall	0.02184	catch_for	0.01041	breaking_ball	0.01806	this_time	0.01376
right_field	0.01857	the_third	0.00961	the_fastball	0.01738	base_line	0.01304

TOPIC 1-6		TOPIC 1-8		TOPIC 1-9		TOPIC 1-38	
down	0.07517	strike	0.05317	base	0.07935	field	0.17031
out	0.04869	out	0.04941	hit	0.04649	center	0.08867
there's	0.04263	pitches	0.04756	double	0.0413	left	0.08826
grounded	0.0369	ball	0.03582	third	0.04058	catch	0.05928
throws	0.03197	pitch	0.03107	field	0.03769	makes	0.03142
base	0.03141	home	0.031	second	0.03517	deep	0.03044
ground	0.03052	strikeouts	0.02447	left	0.03106	air	0.02752

Figure 6-1. Example topic distributions. Most likely words given topics for unigram, bigram, and trigram grounded language models.

Since a standard set of query terms was also unavailable for the baseball domain, we automatically generate queries using a technique similar to that used in Berger & Lafferty (1999). For each of the highlight categories described above, a likelihood ratio is used to generate a measure of how indicative each unigram, bigram, and trigram in the corpus is of a particular category (Dunning, 1993).[42] Query terms are then selected by taking the top 10 ngrams that are most indicative of each category (e.g. "fly ball" for category *flyball*). This gives us a set of queries for each annotated category (130 in all; see Figure 6-4) for which relevant results can easily be determined (e.g., if a returned event for the query "strike out" is of the *strikeout* category, it is marked relevant).

[42] The likelihood ratio is a statistical technique similar to chi-square or mutual information (Dunning, 1993). It compares the probability of two competing hypotheses: h_1: P(word |event) = P(word |¬event) and h_2: P(word |event) >> P(word |¬event). Thus, high values of the likelihood ratio for a given word and event pair indicate that the word is highly indicative of that event.

6.2.2 Model

We train our AT model using the Gibbs sampling method implemented in the Topic Modeling toolkit (Steyvers et al., 2004). We run the sampler on a single chain for 1000 iterations. We set the number of topics to 50, and normalize the pattern durations first by individual pattern across all events, and then for all patterns within an event. The resulting parameter estimates are smoothed using a simple add N smoothing technique, where N=1 for the word by topic counts and N=0.01 for the pattern by topic counts. These parameters settings are based on informal analysis of performance. It is expected that optimization would further improve results and is left for future work.

Figure 6-1 shows examples of 12 topics learned by the grounded language model along with the top seven n-grams most likely to be generated conditioned on each topic.[43] The topics were hand selected to demonstrate how topics in grounded language models converge on event types in the sports domain. Qualitative examination shows that topics range from the very specific, such as a "ground ball outs to third base" (TOPIC 3-10) to the very general, such as "hits" (TOPIC 1-9). More mid-level event types are associated with the bigram models, for example: "field locations" (TOPIC 2-9), "catches" (TOPIC 2-13), "strikes" (TOPIC 2-22), "double plays" (TOPIC 2-40).

6.2.3 Results

Table 6-1 shows example outputs of the system run on all events from the six test games (both highlights and non-highlights). The results are shown for a system that uses an equal interpolation between a traditional language modeling retrieval system and a system using the grounded language model [i.e., using an alpha=0.5 in Equation 6-3]. The top five results are returned for each query, and the relevant classes are displayed. Results are reported using precision which indicates the number of relevant results divided by the number of returned results (set to five in these experiments).

Figure 6-2 shows a comparison between the performance of a traditional language modeling retrieval system that uses only speech information versus a system using only the grounded

[43] Note that in the following information retrieval experiments, only unigram probabilities are used. For more discussion of bigram and trigram models, see the speech recognition evaluations, and in particular Section 7.1.

Table 6-1. Example output of search system with grounded language model (i.e. alpha=0.5). Query terms are displayed with relevant category in parentheses. Query results are presented in ranked order and described by all relevant categories to which they belong (e.g., GROUND_1ST_OUT represents an event where a batter hit a ground ball to 1st base and was called out. NON_HIGH represents a non-highlight event such as a foul ball).

Result rank	Query terms				
	walks (WALK)	strike out (STRIKEOUT)	left field (LEFT)	it's gone (HOMER)	towards the corner (DOUBLE)
1	WALK	NON_HIGH	LINE_LEFT_SINGLE	NON_HIGH	LINE_LEFT_DOUBLE
2	NON_HIGH	STRIKEOUT	LINE_LEFT_DOUBLE	FLY_LEFT_HOMER	NON_HIGH
3	NON_HIGH	STRIKEOUT	NON_HIGH	GROUND_1ST_OUT	LINE_LEFT_DOUBLE
4	WALK	NON_HIGH	FLY_LEFT_OUT	FLY_LEFT_HOMER	NON_HIGH
5	NON_HIGH	STRIKEOUT	FLY_LEFT_HOMER	NON_HIGH	NON_HIGH
Precision.	0.40	0.60	0.80	0.40	0.40

language model and a system that interpolates between the two [using an alpha=0.5 in Equation 6-3].

Results are reported for the 130 automatically generated queries described in Section 6.2.1, run over all events in the test games (both highlights and non-highlights). Comparisons are made for searches within a single corpus made up of all six held out test games, as well as, for searches within each test game individually. For each query, the precision is computed and the mean over all queries is reported. The performance of the combined system for all games, and the system using only words is statistical significant (p=.035; df=649). [44]

Figure 6-3, shows the effect on performance of varying the weighting parameter alpha from Equation 6-3. We report results on two sets of queries generated using the automatic technique described above: one taking the top 10 ngrams per highlight category, and one taking only the top three. This second set of queries represents a smaller and cleaner test set to evaluate the performance of the system. Peak performance is found at the alpha=0.4 level, showing statistically significant differences both for the full query set (p=.0018; df = 649), and for the top three query set (p=0.006; df=194).

[44] Significance is tested using a paired ttest.

Figure 6-4 shows a detailed view of the performance of the combined system (alpha=0.5) for each individual query. Queries are grouped according to their highlight category and precision for each query is reported.

6.2.4 Discussion

Figure 6-2 shows that for all six test games, using the grounded language model improves results over traditional IR techniques. In all but one game, using both text and non-linguistic information produced better performance than either information source on its own. The one exception to this is game 6, in which the text only system performed poorly due to unusually poor time-code alignments of the closed captioning (due to poor ASR results). This suggests that with better alignments, performance would increase both when using text alone and with grounded language models.

The increase in performance due to grounded language models is even more evident when searching the complete set of test games. The more detailed results reported in Figure 6-4 show a large range in performance due in part to the quality of the query term used. Because query terms are generated automatically, as more terms are selected, their quality begins to deteriorate. Thus, by only examining terms with high log-likelihood ratios, we would expect better performance from the system. This is just what is shown in Figure 6-3.

Here we see that results on just the top three queries generated automatically show markedly better performance than the larger set of test queries. Also in Figure 6-3, we see the benefit of varying the weight between the grounded and traditional language models. The increased performance is due to the complementary nature of the grounded language model and the traditional language model for IR. As described above, traditional IR approaches return many false positives because of the tendency of announcers to discuss things that are not currently occurring. On the other hand, grounded language models are poor at distinguishing between visually similar phenomena. This is understandable considering that the difference between a home run and a foul ball can often be only a matter of inches. By combining the two together, the grounded language model buttresses the traditional approach, leading to significant increases in performance when compared to either system on its own.

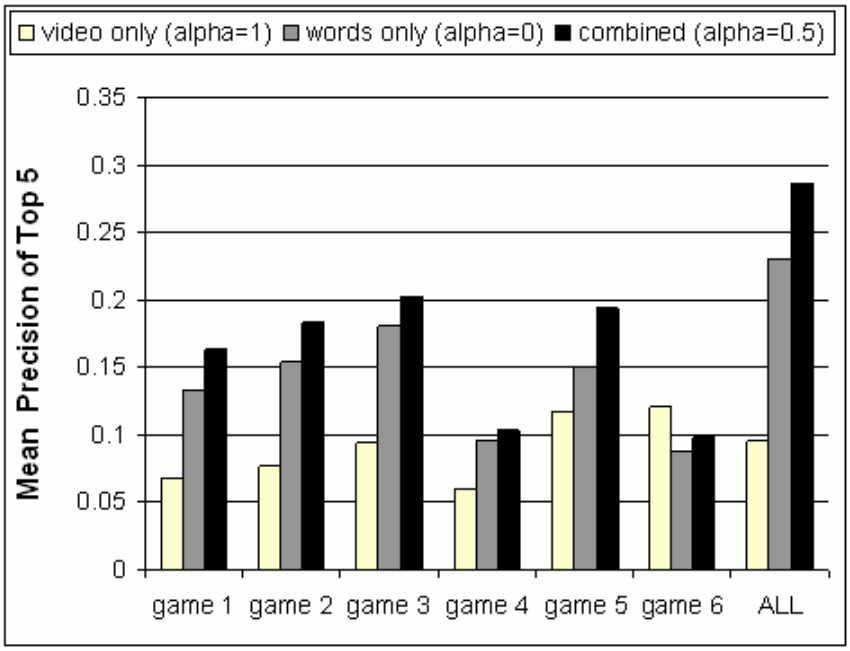

Figure 6-2 Baseball video search performance on individual test games. Performance shown for system using only closed captioning (alpha=0), only video features (alpha=1), and using both together (alpha=0.5).

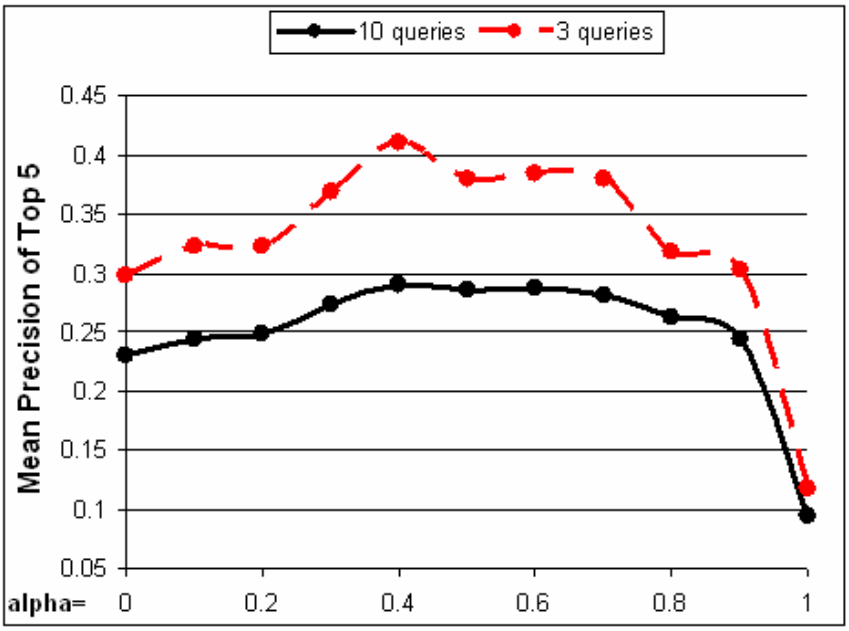

Figure 6-3. Baseball video search precision as a function of non-linguistic context. Alpha=0 represents the system using only the traditional language model. Alpha=1 represents the system using only the grounded language model. Results are presented for automatically generated query sets using top 10 and top 3 most indicative terms.

6.3 Experiments: Football

Although the domain of baseball is the primary implementation focus of this thesis, in order to evaluate the generality of our approach, pilot experiments were conducted in the domain of American football games. American football is a more challenging domain than baseball for a number of reasons. First, the layout of the field and the typical camera angles used are more limited in variety (and informativeness) to what is seen in baseball games. Further, because the ball does not move as far or as fast as it does in baseball, there is generally less camera movement in videos of American football. Despite these challenges, however, football shares with baseball the inherent advantages that sports offer for grounded language modeling (as described in Section 1.2.2). This section describes how such grounded language models for football are trained, and discusses experiments that show their usefulness for video information retrieval.

6.3.1 Model

Training grounded language models for American football operates just like training models for baseball, with two exceptions: extracting visual context features and segmenting events. Visual context features are extracted using the same two phase classification process described in Section 5.2.[45] However, unlike shots in baseball which are classified into 12 categories, we classify football shots into only four categories: *field-shots, goal-post-shots, umpire-shots,* and *other-shots.* The selection of a smaller category set is a reflection of both the relative homogeneity of football video, as well as the smaller dataset used in our pilot experiments. More detailed evaluation of visual shot classification can be found in Appendix B.

[45] See Appendix B for more detailed evaluations of visual context feature extraction in American football.

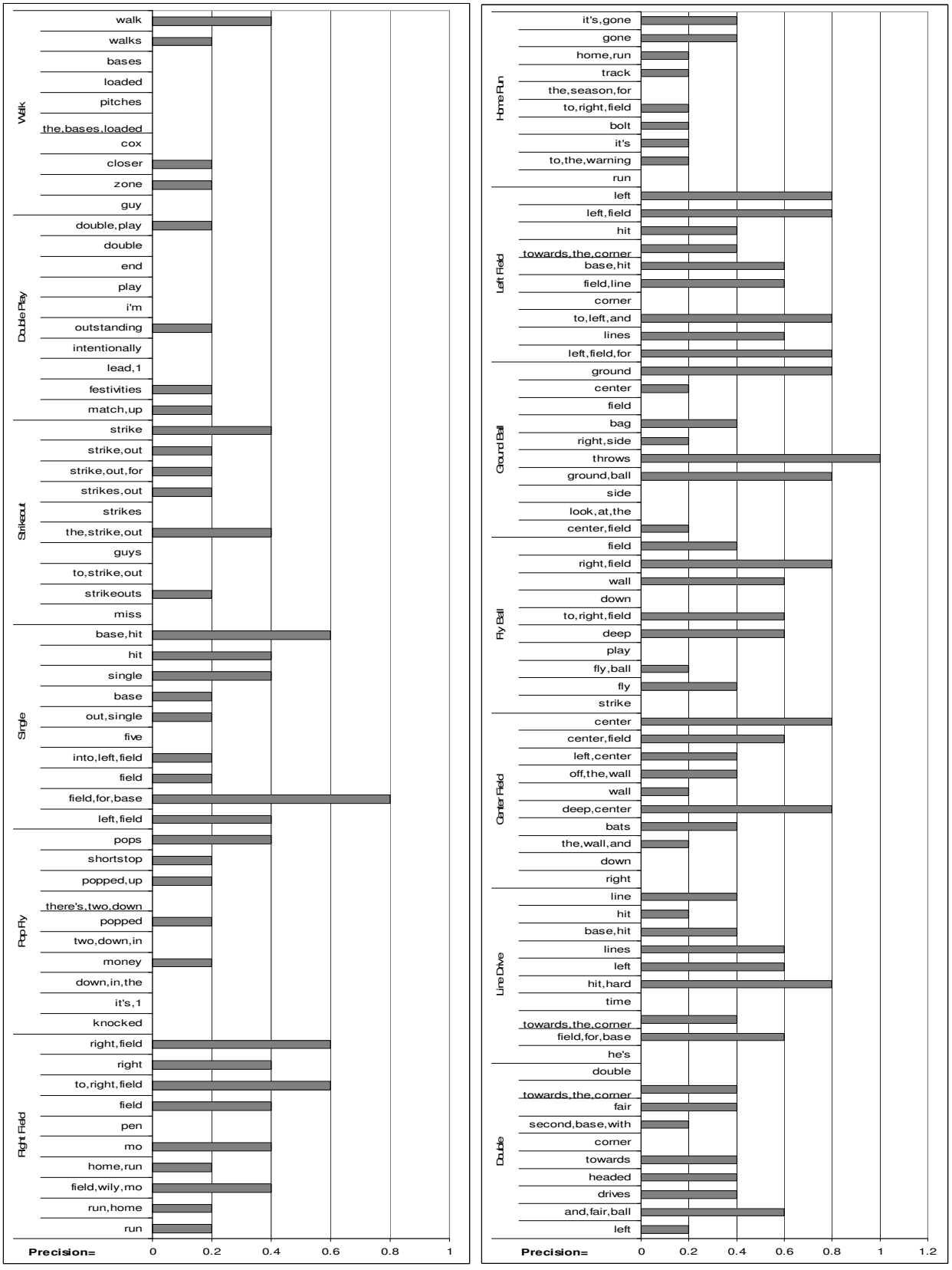

Figure 6-4 Detailed query results of baseball video search using both linguistic and non-linguistic context (alpha=0.5).

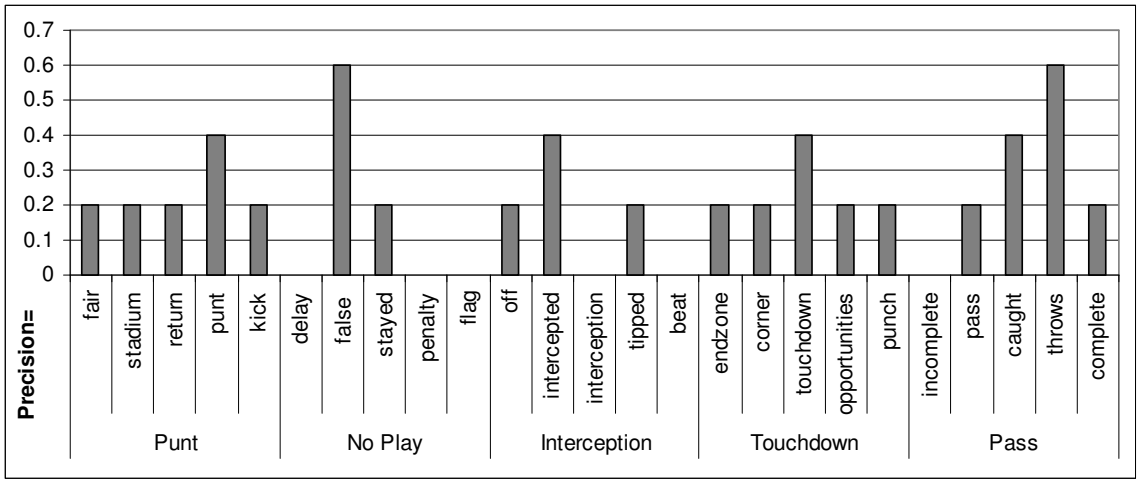

Figure 6-5 Detailed query results of football video search using both linguistic and non-linguistic context (alpha=0.4).

The second difference associated with training grounded language models for football is the method of event segmentation. Unlike baseball where events occur over sequences of shots of many different types, all events in football occur within shots of the field. Thus, unlike baseball in which an event is delimited by four shots starting with a *pitching-shot*, in football each shot classified as *field-shot* is treated as an individual event.

With the exception of visual context features and event segmentation, grounded language models for football are trained just as described in Section 0. Temporal patterns are mined from the multiple streams of features extracted from the raw video and Gibbs sampling is used to estimate the parameters of an AT model given a parallel corpus of event representations and closed captioning text. The AT model is parameterized as for baseball, although with the number of topics set to five due to the more homogenous nature of the feature streams and the significantly smaller training set. Smoothing and normalization are performed as described for baseball.

6.3.2 Data

As with the domain of baseball, standard corpora do not exist for training grounded language models of American football. Thus we created our own corpus by recording broadcast games from the National Football League (NFL). A small corpus of eight games (from 13 teams, in eight stadiums, broadcast on three channels, 1577 individual events) was collected. Events were

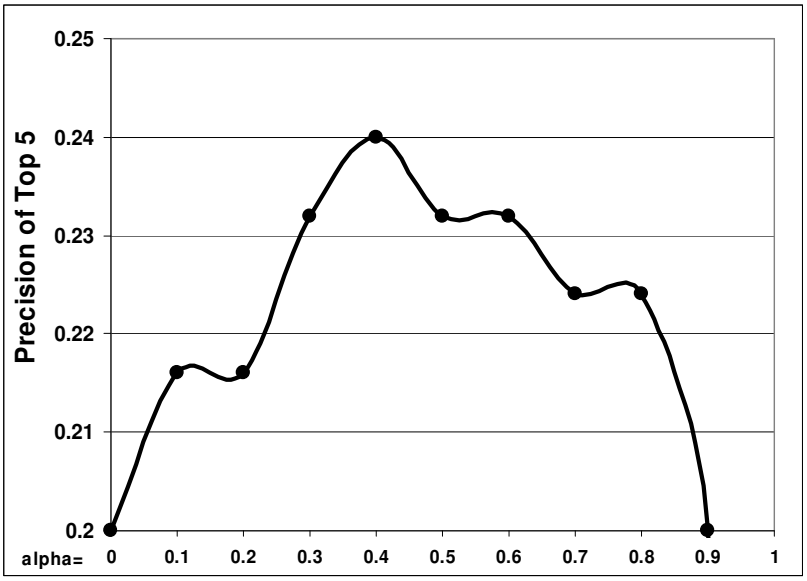

Figure 6-6. Football video search precision as a function of non-linguistic context. Alpha=0 represents the system using only the traditional language model. Alpha=1 represents the system using only the grounded language model. (precision=.08).

segmented as described above and 411 highlights for five categories were labeled by hand as: *touchdown, pass, incomplete-pass, punt,* or *no-play*.[46] Although by no means comprehensive, this set of highlights represents a useful subset given the limited amount of data available for these experiments.

Artificial query terms (five per category) are generated from this corpus using the technique of Berger and Lafferty (1999) (see Figure 6-5 for the specific terms used). Due to the limited amount of data, the corpus was used both to train and test the grounded language model. However, since training is an unsupervised process, this does not bias the results of our evaluations.

6.3.3 Results

Figure 6-6 shows the precision of the grounded language modeling approach, applied to the artificial queries described above, as a function of the weighting parameter alpha. As in Figure 6-3, an alpha=0 indicates the system is using no nonlinguistic information (i.e. only the

[46] *No-play* refers to plays that were stopped before the snap due primarily to a penalty against a team.

traditional language model), while an alpha=1 indicates no closed captioning data is used by the system (precision at this setting is 0.08). Finally, Figure 6-5, show individual precision for each query in each category for the system with parameter alpha=0.4.

6.3.4 Discussion

Although the results of the information retrieval evaluations for American football are lower than that found for the baseball domain, the trend that is shown in Figure 6-6 follows closely that which is seen in the domain of baseball. Just as in baseball, the system which exploits both text and non-linguistic context performs better than systems that use either source of information on their own. Although this trend is qualitatively clear from the figure, the results are not statistically significant. This is not surprising, however, given the dramatically smaller amount of training data used for these pilot experiments (eight training games) compared to what was used in the baseball evaluations (91 training games).

In addition to the limited amount of training data, the lower improvement in American football follows from the more homogenous nature of the features extracted from football video. As discussed above, unlike baseball where different areas of the field are visually distinct, in football, the field has a consistent appearance in most sections (except for the end zones, which we do not identify). Further, while in baseball, players spread across the field and the camera moves dramatically to follow the ball; in football, players often bunch up close together and the camera remains stationary. The limitations of the low level features used in these experiments (which were very similar to those generated for the baseball domain) suggest that if additional features were selected that better suited the domain of football better results could be expected.[47] However, that the trend in American football does indeed follow what was demonstrated in baseball, even given the limited nature of the visual features, further highlights the usefulness of grounded language models for video information retrieval.

In the next chapter, another application domain is examined in which grounded language models are applied to automatic speech recognition (ASR). As with video search, grounded language models allow for non-linguistic context to be incorporated into traditional techniques

[47] For example, Intille and Bobick (2001) obtain good results identifying plays in American football games using human generated traces of player movement.

in a principled way. Experiments show that such context significantly improves the performance of ASR on a number of evaluation metrics.

Chapter 7

Application: Video Speech Recognition

In the previous chapter we described experiments that evaluate the usefulness of grounded language models for the task of video information retrieval. In those experiments, the performance of a text-based language modeling technique was improved by using grounded language models to exploit the non-linguistic context of a video.

This result can be generalized to any methodology that exploits text-based language models by simply replacing those models with their grounded counterparts. This follows from the fact that both text-based and grounded language models encode the prior probability of words using conditional probability distributions. Unlike text-based language models, though, grounded language models represent the probability of a word conditioned not only on the previous word(s), but also on features of the non-linguistic context in which the word was uttered. In this chapter we examine the generality of grounded language models by examining their affect on another real-world task: automatic speech recognition in sports video (an earlier description of this work appears in Fleischman and Roy, 2008).

Automatic speech recognition (ASR) relies heavily on the use of text-based language models and is a common preprocessing step for many multimodal applications such as video search

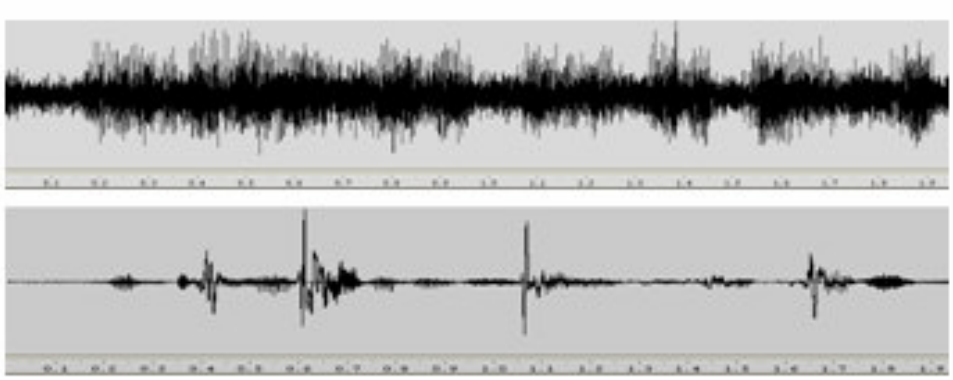

Figure 7-1. Sample waveforms showing the effect of magnitude of background noise in baseball audio noise. Both waveforms are of the same utterance: "into left center field for a base hit." The bottom utterance was spoken in controlled laboratory conditions. The top utterance was taken from the audio stream of a broadcast baseball game.

and summarization (Snoek and Worring, 2005).[48] Although their performance is often reasonable in controlled environments (such as studio news rooms), ASR systems have significant difficulty in noisier settings (Wactlar et al., 1996).

Sports broadcasts are particularly challenging to ASR systems for a number of reasons: first, there are multiple announcers in sports games making it difficult to train speaker dependent models. Also, there are many out of vocabulary words such as player names and sports specific slang. Further, announcers speak at varying speeds (sometimes fast, sometimes slow) and with changing prosody (excited speech, shouting, etc.) which is difficult for many ASR systems to model. Finally, and most challenging, the audience of a sports game generates high levels of background noise that can make it difficult even for humans to understand what is said. Figure 7-1 gives a qualitative demonstration of this noise by showing two waveforms of the same utterance; the first, being said in the audio of a baseball game, the other spoken by the author under ideal laboratory conditions. Figure 7-2 shows an additional measures of the level of background noise in baseball video. Here the distribution of power during just background noise is compared to the distribution of power during speech and background noise. The figure shows that background noise in baseball video has a much more similar power distribution compared to speech uttered in laboratory controlled conditions.

[48] Note that closed captioning transcriptions are available for some, but not all, sports broadcasts.

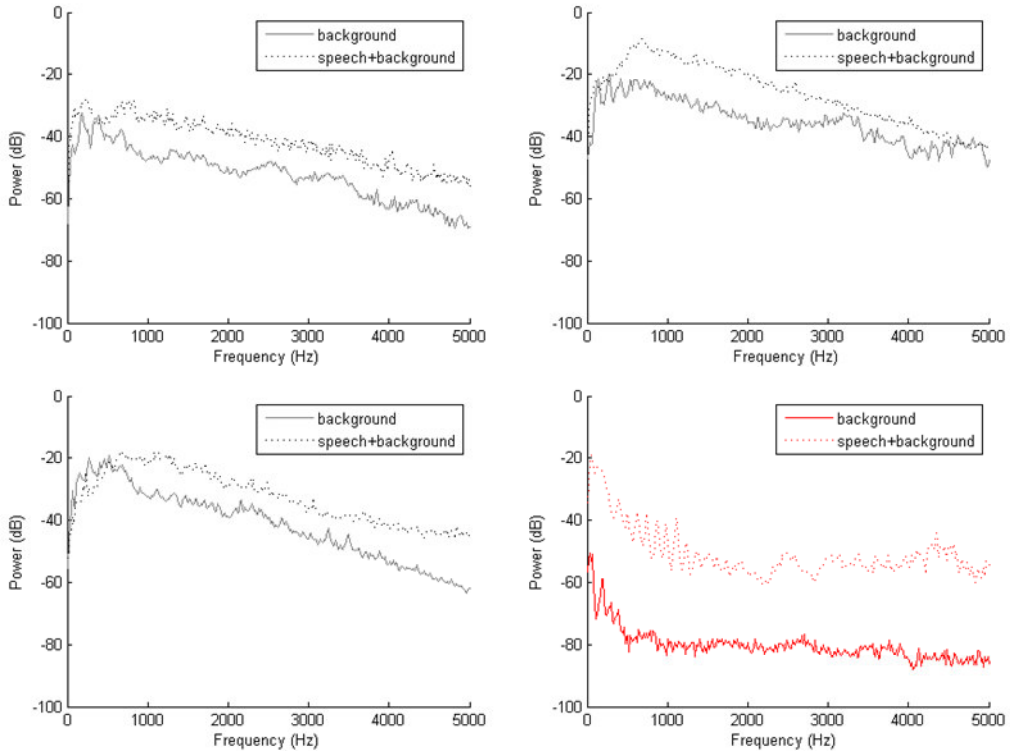

Figure 7-2. Sample power distributions showing effect of background noise in baseball video. Four audio samples compare the distribution of power during just background noise to the distribution of power during background noise plus speech . Top samples and lower left sample extracted from baseball video (in gray). Lower right sample generated under laboratory conditions (in red).

While many researches have examined ways to compensate for these different challenges, few have attempted to leverage information in the visual stream to improve speech recognition performance (for an exception see Roy and Mukherjee, 2005). For many types of video, and in particular sports video, visual context can provide valuable clues as to what has been said. For example, in video of Major League Baseball games, the likelihood of the phrase "ground ball" increases dramatically when a ground ball has actually been hit.

In the following section we examine the generality of grounded language models by showing how they can be used to improve ASR performance. Our experiments focus only on the domain of broadcast Major League Baseball games although, just as with video search, the approach can be generalized to other domains. Even though the amount of noise present in the

audio makes ASR exceptionally challenging, results indicate that grounded language models significantly improve the quality of recognized speech.

7.1 Experiments

7.1.1 Model

As described above, our method for incorporating non-linguistic information into a speech recognition system is based on the learning of grounded language models from unlabeled data. The experiments that follow use the same parameterizations and training and test sets described in Section 6.2. The training set is the paired corpus of event pattern representations and the associated closed captioning transcriptions of the announcers' speech (vocab=17k, word count=1.65M). The test set is a held out paired corpus of 1200 events, 237 of which are hand labeled highlights (vocab=1.8k, word count=12.6k). Speech recognition experiments are evaluated only on these highlights, each of which has been re-transcribed by hand in order to avoid transcription and time code errors present in the closed captioning.

Unigram grounded language models are trained in the same manner as described in Section 6.2.2 Bigram and trigram models are also trained on this data, by treating each two and three word sequence as a single token and training a model on these word sequences. As in Section 6.2.2, all instances of proper names, low frequency phrases, and word sequences composed entirely of stop words are removed from the training set. The resulting models are then normalized to produce bigram and trigram grounded language models of the same form as traditional language models using the following equations:

$$p(w_0 \mid P, w_{-1}) = \frac{p(w_{-1}, w_0 \mid P)}{\sum_{w' \in W} p(w_{-1}, w' \mid P)} \quad \textbf{7-1}$$

$$p(w_0 \mid P, w_{-1}, w_{-2}) = \frac{p(w_{-2}, w_{-1}, w_0 \mid P)}{\sum_{w' \in W} p(w_{-2}, w_{-1}, w' \mid P)} \quad \textbf{7-2}$$

These grounded language models are then combined in a backoff strategy with a traditional unigram, bigram, and trigram language model. This backoff is necessary to account for the words not included in the grounded language model itself (i.e. stop words, proper names, low

frequency words). Although more complicated backoff strategies exist (e.g., Hsu, 2007), we employ a very simple approach: if the n-gram appears in the grounded language model (GLM) then it is used, otherwise the traditional language model (LM) is used.

We generate traditional language models from two datasets: the switchboard corpus, which is a commonly used corpus of transcriptions of open domain telephone speech (vocab=27k, word counts=3.65M); as well as, the closed captioning transcriptions from the training games used to train the grounded language models. The SRI language modeling toolkit (Stolcke, 2002) is used to generate two separate language models (one for each dataset) using Chen and Goodman's modified Kneser-Ney discounting and interpolation (Chen and Goodman, 1998). These two models are then combined using static interpolation (also using the SRI language modeling toolkit), giving equal weight to both models.

We evaluate the effect of grounded language models on ASR performance using 3 metrics: perplexity, word error rate, and precision on an information retrieval task.

7.1.2 Perplexity

Perplexity is an information theoretic measure of how well a model predicts a held out test set. Although not a direct measure of speech recognition performance, perplexity is an often used measure of the fit between a language model and the speech it is designed to recognize. Thus, lower perplexity often correlates with better ASR performance. The perplexity of a language model *p(*)* for a given test set is defined as:

$$ppl = 2^{-\sum_{i=1}^{N} \frac{1}{N} \log p(x_i)} \quad \textbf{7-3}$$

where N is the number of words *x* in the held out corpus. The better a model predicts a test set, the lower the perplexity of the model.

We use perplexity to compare our grounded language model to the traditional language models described above. We calculate perplexity for five distinct models: 1) a language model trained only on the telephone-domain switchboard corpus, 2) a language model trained only on the baseball-domain closed captioning, 3) a language model that interpolates between these two (with equal weight given to both), 4) a grounded language model that backs off to the baseball-domain traditional language model, and 5) a grounded language model that backs off to the

Table 7-1. Perplexity of language models on a held out test set of baseball highlights. Three text-based language models were trained on the switchboard corpus, closed captioning from baseball training games, and an interpolation between the two models. The grounded language model was evaluated with a backoff to the closed captioning text-based model, and with a backoff to the interpolated text-based model.

	Text (Switchboard)	**Text** (Closed Caption)	**Text** (CC+ Switch)	**Grounded** (cc)	**Grounded** (cc + switch)
Ppl	**1404**	**143.54**	**145.27**	**86.37**	**83.88**

interpolated traditional language model.[49] Table 7-1 reports the results of these calculations (lower is better).

Not surprisingly, the switchboard-only model performs far worse than any other language model. This is due to the large discrepancy between both the style and the vocabulary of speech about sports compared to the telephone speech sampled in the switchboard corpus. Even though the much larger switchboard corpus contains information useful for predicting the test set (i.e., there are n-grams in the test set that appear in the switchboard, but not in the closed captioning language model), interpolating between these two models has a non-significant effect on perplexity.[50]

Of more interest, however, is the decrease in perplexity when the grounded language model is used. When the grounded language model backs off to the baseball-only traditional language model a large improvement in perplexity is observed. Note that both these language models are in fact trained on the same speech transcriptions, i.e. the closed captioning from the training games. However, whereas the baseline model remains the same for each of the 237 test highlights, the grounded language model generates different word distributions for each highlight depending on the visual features extracted from the highlight video. This result demonstrates the strong predictive power of visual context for language in sports video.

The benefits of such adaptation are again seen when backing off to the interpolated traditional model. Although perplexity here is not significantly better than when backing off to

[49] All language models use the 17k word vocabulary from the baseball-domain closed captioning training data.

[50] However, this additional information is sufficient to keep the switchboard model from significantly reducing the perplexity of the interpolated model. Further, informal analysis shows that the interpolated model performs better on ASR tests than the closed caption only model.

the baseball-only model, the interpolated model does show improvement. This improvement becomes more pronounced in the next set of experiments which directly examine speech recognition performance.

7.1.3 Word Accuracy and Error Rate

The performance of an ASR system is typically evaluated using two measures: word accuracy, and word error rate. The word accuracy of a system represents the proportion of words in the actual speech stream that were correctly recognized. Word error rate (WER) is a stricter measure which punishes a system for mistakenly adding words not in the speech signal. WER is a normalized measure of the number of word insertions, substitutions, and deletions required to transform the output transcription of an ASR system into a human generated gold standard transcription of the same utterance.

Unlike perplexity which only evaluates the performance of language models, measuring word accuracy and error rate requires running an entire ASR system, i.e. both the language and acoustic models. We use the Sphinx system to train baseball specific acoustic models using parallel acoustic/text data automatically mined from our training set.[51] Following Jang and Hauptman (1999), we use an off the shelf acoustic model (the Hub4 model) to generate an extremely noisy speech transcript of each game in our training set, and use dynamic programming to align these noisy outputs to the closed captioning stream for those same games. Given these two transcriptions, we then generate a paired acoustic/text corpus by sampling the audio at the time codes where the ASR transcription matches the closed captioning transcription.

For example, if the ASR output contains the term sequence "… and farther *home run for David* forty says…" and the closed captioning contains the sequence "…another *home run for David* Ortiz…," the matched phrase *"home run for David"* is assumed a correct transcription for the audio at the time codes given by the ASR system. Only looking at sequences of three words or more, we extract approximately 18 hours of clean paired data from our 275 hour training corpus. A continuous acoustic model with 8 gaussians and 6000 tied states is trained on this

[51] http://cmusphinx.sourceforge.net/

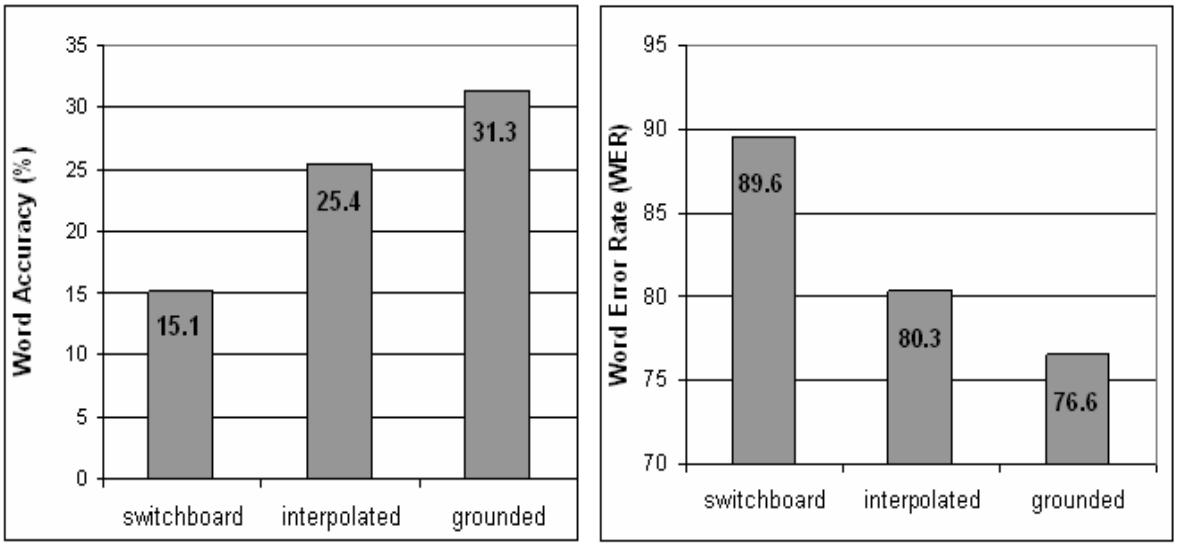

Figure 7-3. Word accuracy and error rates for ASR systems using a text based language model trained on the switchboard corpus (switchboard), and the switchboard model interpolated with a text based model trained on baseball closed captions (interpolated), and a grounded language model which backs off to the interpolated model (grounded),.

data using the Sphinx speech recognizer.[52] Although not done in this work, by repeating this process, and iteratively training new acoustic models that are then used to generate better ASR output, better alignments could be generated and more acoustic data could be extracted. This iterative training is left for future work.

Figure 7-2 shows the word accuracy and WER for an ASR system run using the Sphinx decoder with the acoustic model described above and three language models: the traditional model trained on the switchboard corpus, that model interpolated with the model trained on closed captioning,, and the grounded language model backing off to the interpolated model.

In analyzing the performance of these models it is important to note how challenging speech recognition is in the sports domain. The audio input to the models comes from multiple speakers, speaking quickly and at varying speeds (a particular challenge ASR systems). Further, the speech is recorded in a large stadium full of background noise coming from thousands of fans who are often screaming (particularly during the highlight events that our system is attempting to recognize). The effects of this noise, combined with the limited amount

[52] http://cmusphinx.sourceforge.net/html/cmusphinx.php

of training data used to build the acoustic model, should be taken into account when considering the performance of the speech recognizer.

Even with this noise, however, results indicate that the use of grounded language models significantly improves performance of both word accuracy and WER. WER shows significant absolute reductions of 13% compared to the traditional switchboard language model, and 3.7% compared to the interpolated model ($p<0.001;df=236$).[53] Word accuracy improvement is even more significant, with a 15.2% absolute improvement over the switchboard model and a 5.9% absolute improvement over the interpolated model ($p<0.001;df=236$).

Even though their performance increases are significant, drawing conclusions about the usefulness of grounded language models based on word accuracy and WER is difficult. Both metrics penalize a system that mistakes "a" for "uh" as much as one that mistakes "run" for "rum." When using ASR in real world situations (e.g. to assist in searching video), though, such substitutions are not of equal importance. Further, while visual information may be useful for distinguishing the latter error, it is unlikely to assist with the former. Thus, in the next section we examine an extrinsic evaluation in which grounded language models are judged not directly on their effect on word accuracy and WER, but based on their ability to support video information retrieval. By performing such an extrinsic evaluation, we are able to more carefully analyze the types of errors that are avoided when using grounded language models.

7.1.4 Precision of Information Retrieval

One of the most commonly used applications of ASR for video is to support information retrieval (IR). When closed captioning transcripts are not available, such video search systems must use ASR to generate transcriptions that can be used to index video. In such cases, the output of the ASR system is used in the same way that closed captioning was used in the evaluations presented in Chapter 6. Because the quality of the search results depends upon the accuracy of the speech recognizer, the performance of a video IR system gives an indirect evaluation of the ASR's quality. Further, performing such evaluations provides an error analysis of speech recognition results by highlighting a system's ability to recognize the more relevant content words without being distracted by the more common stop words.

[53] Significance is tested using a paired ttest.

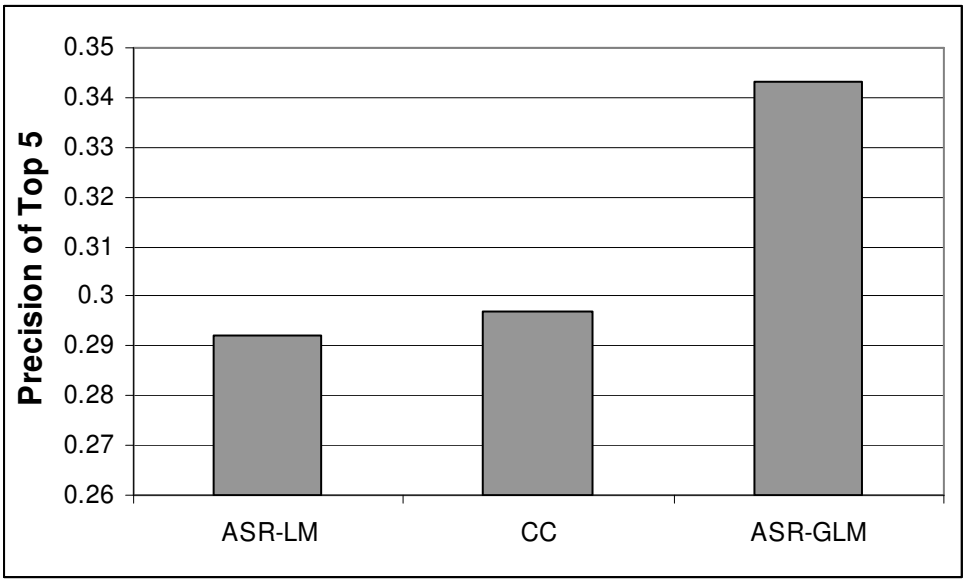

Figure 7-4. Precision of top five results of a video IR system based on speech transcriptions. Three different transcriptions are compared: ASR-LM uses ASR with a text-only interpolated language model (trained on baseball closed captioning and the switchboard corpus); ASR-GLM uses ASR with a grounded language model; CC uses human generated closed captioning transcriptions (i.e., no ASR).

Following Section 6.1, we compare the performance of IR systems that employs the language modeling approach of Song and Croft (1999).[54] Three such systems are compared: one that indexes video based on the output of an ASR system using the traditional interpolated language model described above (ASR-LM), one based on the output of an ASR system using the grounded language model backing off to the interpolated model (ASR-GLM), and one that does not use ASR at all, but rather uses the human generated closed captioning associated with the test set.

For each system, all 1200 events from the test set are indexed. Importantly, for these experiments *all* events from the test set are searched, not just the 237 highlight events used to evaluate perplexity, word accuracy, and WER. The top three artificial queries for each of 13 event types are generated as described in Section 6.2. Performance of the systems is measured using the precision of the top five results; i.e. the proportion of the first five returned results that are events of the correct category.

[54] Note for these experiments, the grounded language model is not integrated directly into the retrieval system, as it is in Chapter 6, but rather, is only employed in generating ASR transcriptions of the audio.

Figure 7-3 shows the precision of the video IR systems based on ASR with the grounded language model (ASR-GLM), ASR with the traditional language model (ASR-LM), and closed captioning transcriptions (CC). As with our previous evaluations, the IR results show that the system using the grounded language model performed better than the one using only traditional language models (17.5% relative improvement: p=0.013; df=194). More notably, though, Figure 7-3 shows that the system using the grounded language model performed significantly better than the system using the hand generated closed captioning transcriptions (15.5% relative improvement: p=0.023; df=194). Although this is somewhat counterintuitive given that the closed captioning transcripts are relatively noise-free, the results follow from the tendency of text based methods to return false positive results.

As discussed in Chapter 6, the occurrence of a query term in a video is often not enough to assume the video's relevance to that query. For example, when searching through video of baseball games, returning all clips in which the phrase "strikeout" is uttered, often results in video of events where a strikeout does not actually occur. This follows from the fact that in sports, as in life, people often talk not about what is currently happening, but rather, they talk about what did, might, or will happen in the future.

By taking into account non-linguistic context during speech recognition, the grounded language model system indirectly circumvents some of these false positive results. This follows from the fact that an effect of using the grounded language model is that when an announcer utters a phrase (e.g., "walk"), the system is more likely to recognize that phrase correctly if the event it refers to is actually occurring (i.e., if someone actually was walked).

Table 7-2 shows a segment of the closed captioning from a video event in the test set, along with a transcription of the same segment as generated by two ASR systems (using grounded and text-based language models, respectively). The event shows a player being struck out; but the announcer is describing the pitcher's statistics for both strikeouts and walks. While both the keywords "walks" and "strikeouts" appear in the closed captioning, only the keyword "strikeouts" is recognized by the ASR system using the grounded language model. Further, neither keyword is recognized by the ASR system using the grounded language model.

This example demonstrates how a grounded language model biases a system to correctly recognize phrases that describe what is currently happening. Similar to the effect seen in

Table 7-2. Transcriptions of speech in a baseball video event. CC is from the closed captioning, ASR-GLM and ASR-LM are output by a speech recognizer using grounded and text-based language models respectively.

Method	*Transcription*
CC	*"No walks and three strikeouts for the Big Unit"*
ASR-GLM	*"No one son and for you strikeouts to begin"*
ASR-LM	*"No one son and breeze I counseling"*

Section 6.2, the grounded language model thus reduces the likelihood of false positive results. However, in Section 6.2 it was shown that video search performance is significantly improved by interpolating a closed caption based system (CC) with a grounded language model. This interpolated model, in fact, performs better than the ASR-GLM system in Figure 7-3. So, an interesting question is how such an interpolation would affect the search results for a system using automatic speech recognition with a grounded language model (ASR-GLM).

Figure 7-4 shows this effect, along with the effect of interpolating CC with a grounded language model (repeated from Figure 6-3). Note that the precision of each condition at alpha=0 (from Equation 6-3) is the same value that is reported in Figure 7-3. Recall that as alpha increases, and context from the grounded language model is added, the closed captioning (CC) system significantly improves. That precision improves beyond what is seen for the ASR-GLM system implies that it is better to search over cleaner transcriptions interpolated with contextual information, than over noisy transcriptions that were generated by exploiting contextual information. This is not surprising given that the word accuracy of the ASR-GLM system is still relatively poor, and thus, many keywords that might be useful to search are incorrectly recognized. More interesting though, is that interpolating the ASR-GLM system with a grounded language model improves the system, but not significantly. This implies that the benefit of interpolating with a grounded language model is largely redundant when using speech recognized with that same model.

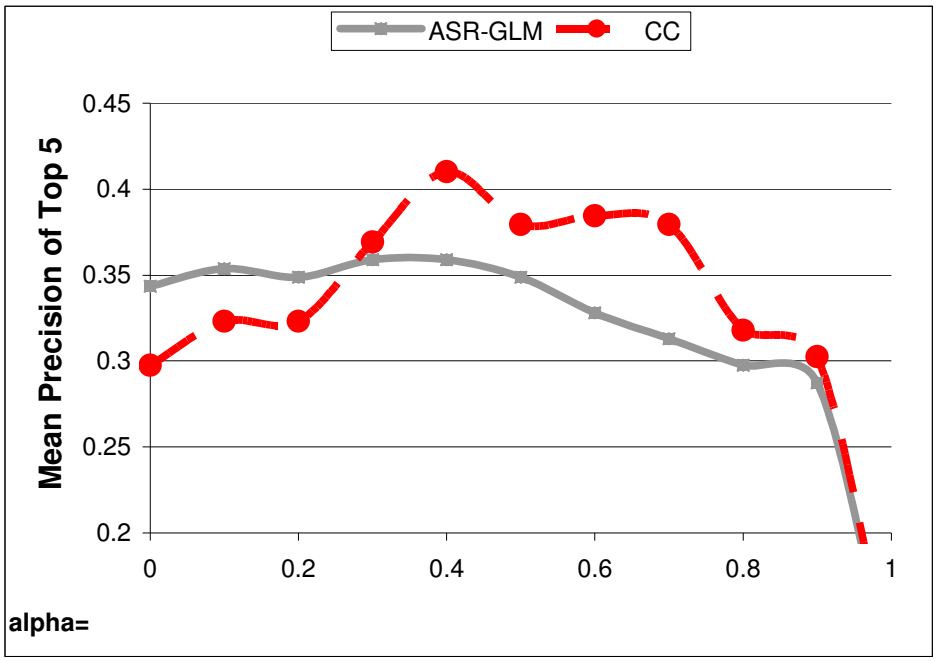

Figure 7-5. Baseball video search using ASR as a function of non-linguistic context. Results are presented using closed captioning (CC; repeated from Figure 6-3) and automatically recognized speech with a grounded language model (ASR-GLM). Higher alpha means more linguistic context.

However, that both ASR and closed captioning based search system improve when exploiting grounded language models highlights the benefit of incorporating non-linguistic context in multimodal applications.

7.2 Discussion

In this section grounded language models were shown to have improved the performance of an automatic speech recognition (ASR) system on the audio from broadcast baseball games. The benefit of the grounded language model lies in ability to bias the ASR system to recognize words that relate to what is actually occurring in the video. This not only improves the overall performance of speech recognition (in terms of word accuracy and error rate), but also improves the performance of applications that rely on ASR performance (such as video search).

The results of these experiments (as well as the results of the experiments in Chapter 6) suggest that grounded language models may benefit many other multimodal applications. The

probabilistic form of the model allows it to be easily integrated into other applications that exploit a probabilistic framework. This is particularly true of natural language applications that use a noisy channel framework (such as ASR). For such applications, the text-based language model can easily be expanded to include information from the grounded language model (as was done here, using a simple backoff strategy). Perhaps the most obvious application where this could be applied is in Machine Translation, where a grounded language model could be used to bias a system to translate words that correspond to the events shown in the video. In the next section, other future directions for using and designing grounded language models will be more fully discussed.

Chapter 8

Conclusions

8.1 Contributions

In this thesis we have presented a methodology for representing events in a grounded language model. The contributions of this work are as follows:

- A top-down approach to modeling events based on the use of grammars of behavior to explicitly represent the context of the situation in which events occur. Behavior grammars are used to parse out event structures used for language grounding. These event structures enable language understanding in virtual domains that is robust to noise and parallels aspects of human language acquisition.
- A bottom-up approach to modeling events in which temporal data mining is used to automatically learn event representations from a large corporus of unlabeled video. Unlike the behavior grammars used in the top-down approach, the mined event structures are not dependent on hand designed production rules, making them flexible enough to be used to model language in broadcast sports video.
- A methodology for learning grounded language models that are designed to facilitate the incorporation of non-linguistic context into practical multimedia applications. This methodology is:

- **Principled**: by encoding grounded language models as conditional probability distributions we can exploit a wide variety of algorithms designed for parameterizing such models. Further, the use of probability distributions enables grounded language models to be easily applied to nearly any application with a probabilistic framework. This thesis demonstrated three such applications: natural language understanding, information retrieval, and automatic speech recognition. Additional applications that operate using similar frameworks (e.g., machine translation) are left for future work.
- **Unsupervised:** the method for learning grounded language models requires no human annotation of the events being represented and learns entirely from paired data of language and the events it describes. This distinguishes the work from previous multimodal research which often requires costly human annotation.
- **Non-trivial:** grounded language models are trained on a real world domain and evaluated on practical multimodal applications. Unlike most of the previous work on grounded language models that learn only from data collected in highly controlled settings, our approach operates on non-toy environments with both academic and commercial implications.

8.2 Future Directions

The research described in this thesis leaves open a number of possible directions for future work. Many of these directions relate to practical issues involved in the generation of grounded language models and their applications. For example, the tuning of parameter settings, the use of more sophisticated speech alignment techniques, and the addition of more low level visual features, are just a few of the many ways in which the models described in this thesis could be improved. Although such practical directions are important, this section focuses on more theoretical directions for exploring future work.

8.2.1 Effect of Syntax

The grounded language modeling approach presented here makes the simplifying assumption that words in an utterance are independent of each other. Although bigrams and trigrams are used, no higher level syntactic information is incorporated into the linguistic mapping approach. This is in spite of psychological evidence that such syntactic information is critical to humans when learning words for events (Snedeker and Gleitman, 2004).

Fleischman and Roy (2007c) describe a simple extension to the linguistic mapping algorithm described in Section 3.3. The extension exploits syntactic phrase boundaries automatically identified in utterances (Daume and Marcu, 2005) to better learn the meaning of verbs in a grounded language model. A promising future direction is to examine ways to incorporate more complex statistical information, as is currently being explored in work on statistical machine translation (e.g., Yamada and Knight, 2001).

8.2.2 Merging Top-Down and Bottom-Up Methods

In addition to exploiting more information from the linguistic stream, another important direction for future work is to examine ways to improve representations of the non-linguistic events for grounded language models. One such promising direction examines the question of whether top-down and bottom-up methods for representing events can be combined into one hybrid approach. Such a hybrid method could have significant advantages for representation of events. As this thesis has shown, learning event structure bottom-up from data provides for robust representations which can be exploited in very noisy domains such as broadcast video. The approach we describe is entirely unsupervised, and learns without assistance from human teachers. This approach, while economical, is perhaps overly conservative regarding the use of human supervision. In fact, there is a great deal of high level knowledge about particular domains (especially sports) which can be exploited without excessive cost in human-hours of labor.

Continuing with the domain of baseball, an ideal method would allow humans to write out some or all of the rules of baseball (much like grammars of behavior are written out) and then use those rules to guide the learning of event structures directly from data. Similar combinations of top-down and bottom-up methods have been suggested to play a role in many aspects of human cognition, for example, word learning (Xu and Tenenbaum, 2007) and perceptual learning (Hinton, 2007).). Embracing such a framework for learning grounded language models represents a promising direction for future research.

8.2.3 Generalizing to Additional Domains

Although this thesis presents evaluations of grounded language models in two very different domains (i.e. virtual worlds and broadcast sports video), the generality of the approach to many other domains is still in question. We posit that the generality of grounded language modeling to additional domains is limited by two factors: how easily low level features of the domain can be represented; and how much situated language use occurs in the domain.

The limiting factor of both the top-down and bottom-up approaches to representing events is the ability to extract low level features from the domains. In order to get a wedge into the very challenging problem of grounding language for events, we purposefully selected domains which facilitated such low level feature extraction. For sports games, exploiting information in how sports are filmed made extracting low level features relatively easy. While, for videogames games, access to near-perfect information about low level actions obviated the need for computer vision entirely.

In moving to additional video domains, the ability of computer vision to identify low level features will limit the ability to represent high level features, and thus, the ability to learn grounded language models. However, computer vision technology is advancing all the time and promises to have a dramatic impact on multimedia applications. For example, improvements in object/person tracking will allow for much finer grained analysis of events in many sports domains such as American football (see Intille and Bobick, 2001). Also, designing large ontologies of image classifiers that can identify commonly observed scenes and objects will enable more detailed representations of elements of news video (Naphade et al., 2006). Finally, advances in affective computing will open the door to recognizing the emotional states of people in video (Picard, 1997). Modeling patterns in how a person's emotions change (and cause changes in others) is essential for moving beyond representations of strictly physical events, and into the world of social interaction.

The second limitation to our grounded language modeling approach lies in the amount of situated language that occurs in the domain. "Situated language" is that subset of language in a domain that refers to, or describes, what is actually happening in the "here and now." Our approach depends on situated language use because of its reliance on

learning algorithms that exploit the recurrence of words and event representations in parallel training data. If a domain does not have enough situated language, no amount of such data will enable the training of a model.

Again, the domains chosen in this thesis purposefully contained sufficient situated language use, but it is an open question as to how much, and how situated, the language must be. Certainly there are other domains with situated language; for example, infants interacting with caregivers, cooperative games where agents communicate about their state, and broadcast television programming, such as news shows, cooking shows, home shopping shows, game shows, etc. If the definition of situated language is loosened, other domains become possibilities; for example, soap operas do not necessarily involve much discussion of the here and now, but the language used (e.g., "love," "jealousy," "revenge," etc.) often corresponds with visual patterns in the video (e.g. music rising, zooming into a close up, fading to black, etc.). By incorporating richer features as they become available (e.g. affective information), we begin to see how the limitations described here can be stretched and how extremely complex domains could be tackled using the methods described in this thesis.

8.3 Concluding Remarks

This thesis introduced methods for learning models of the meaning of words for events. It pushed the boundaries of previous work in two directions. First, by focusing on events, this thesis has tackled a significantly more challenging class of words than previous work on grounded models of meaning. Second, by designing robust representations, this thesis shows the practical advantages of using grounded language models in multimodal applications.

It is true that only a few domains have been addressed in this thesis, and that they are the ones most amenable to our approach. We have no illusions that this thesis presents a final optimal solution to the challenges of modeling meaning. Rather, this thesis represents a proof of concept. It is less about specific representations or models, but rather, about a

general method for teaching machines about the world. This thesis shows that machines can be built to learn about the world by simply watching lots of television.

If machines are to be truly intelligent, they must be designed to learn. Although there is much to learn from books and the internet, these resources have little to say about most of human knowledge. Books cannot teach us about common sense. The web has little to offer on the meaning of words. But television (and video, in general) provides information richer than any text-based medium could ever hope to. Although extremely complex, video can be harvested for building artificially intelligent machines. And while much more work is required for machines to fully take advantage of such information, the hope is that this thesis will guide others toward that goal.

Appendix A

Virtual Environments

A.1 Mission Rehearsal Exercise

Rule	Prob
complete_mission -> collision help_boy support_inspection	.5
complete_mission -> collision support_inspection help_boy	.083
complete_mission -> collision help_boy support_inspection reinforce_lz	.25
complete_mission -> collision support_inspection help_boy reinforce_lz	.083
complete_mission -> collision help_boy reinforce_lz support_inspection	.083
help_boy -> secure_area evaluate_boy medevac	.5
help_boy -> secure_area evaluate_boy ambulance	.083
help_boy -> evaluate_boy secure_area medevac	.25
help_boy -> evaluate_boy secure_area ambulance	.083
help_boy -> evaluate_boy treat_on_scene	.083
medevac -> secure_lz call_medevac	.777
medevac -> call_medevac secure_lz	.222
ambulance -> call_ambulance	1
Support_inspection -> squad_to_celic	1

Figure A-1. Behavior grammar rules converted from MRE task model and their probabilities.

Human Transcribed	ASR Transcribed
[collision]	
sergeant secure the assembly area	i sergeant secure the assembly area
[secure_area]	
[evaluate_boy]	
eagle base this is eagle two six over	out eagle base this is eagle two six over
roger we have an injured boy that's been struck by one of our vehicles we need to get him medevaced asap over	roger will injured what what medevac asap over
[call_medevac]	
sergeant have third squad secure the lz	that sergeant have third squad secure the l z
sergeant have third squad secure the lz	that's sergeant have third squad secure the l z
do it	Do it
[secure_lz]	
sergeant send fourth squad to celic	i sergeant send one squad tucci what
roger eagle one six this is eagle two six we are at a medevac site	roger eagle one six is eagle two six we have a medevac site
roger eagle eagle one six i have one squad inbound over	i route to eagle one six that was squad about
sergeant send fourth squad to celic	sergeant send one squad the child's
[squad_to_celic]	
sergeant let's move the boy to the lz	sergeant what happened to the l z
sergeant move the boy to the lz	sergeant move the boy to the l z
sergeant control the crowd	l sergeant route
sergeant have first	sergeant have third
sergeant have first squad reinforce the lz	that sergeant have third squad reinforce the l z
[reinforce_lz]	

Figure A-2. Human and automatic transcripts of sample run of MRE scenario.

A.2 NeverWinter Nights

Rule	Freq
CHANGE_ROOM -> OPEN_DOOR GO_THROUGH	206
OPEN_CHEST -> MOVE>Chest	151
CLOSE_CHEST -> CLICK_ON>Chest	148
OPEN_DOOR -> ASK_OTHER	72
ATTACK_AT>Door -> ATTACK_AT>Door	61
TRY_DOOR -> TRY_DOOR	48
BREAK_DOWN -> ATTACK_AT>Door	46
GetPassword -> GOTO_Password TakePassword	40
GetBlueKey -> GOTO_BlueKey TakeBlueKey	40
Exit -> GOTO_Exit LEAVE	40
OPEN_DOOR -> PULL_LEVER	37
TakeBlueKey -> OPEN_CHEST TAKE>Blue_Key CLOSE_CHEST	35
ASK_OTHER -> GIVE_GYPSY_DIAMOND	35
CHANGE_ROOM -> TRY_DOOR OPEN_DOOR GO_THROUGH	34
CHANGE_ROOM -> OPEN	34

Figure A-8-3 Most frequent non-motion rules and their frequencies.

Rule	Freq
GOTO_Exit -> MOVE>green_chest CHANGE_ROOM MOVE>enterance	8
GOTO_Password -> MOVE>enterance MOVE>red_man CHANGE_ROOM MOVE>red_corridor CHANGE_ROOM MOVE>green_woman CHANGE_ROOM MOVE>green_chest	7
GOTO_Exit -> MOVE>blue_chest MOVE>blue_lever MOVE>enterance CHANGE_ROOM MOVE>exit	7
GOTO_BlueKey -> MOVE>green_chest CHANGE_ROOM MOVE>enterance CHANGE_ROOM MOVE>blue_lever CHANGE_ROOM MOVE>blue_chest	7
GOTO_Password -> MOVE>blue_chest CHANGE_ROOM MOVE>gold_lockpick CHANGE_ROOM MOVE>gold_portal MOVE>gold_corridor CHANGE_ROOM MOVE>green_woman CHANGE_ROOM MOVE>green_chest	4
GOTO_LockPick -> MOVE>blue_chest CHANGE_ROOM	4
GOTO_BlueKey -> MOVE>enterance CHANGE_ROOM MOVE>blue_lever CHANGE_ROOM	4
GOTO_Axe -> MOVE>enterance MOVE>red_man CHANGE_ROOM MOVE>gold_axe	4
GOTO_Password -> MOVE>gold_axe CHANGE_ROOM MOVE>gold_portal MOVE>gold_corridor CHANGE_ROOM MOVE>green_woman CHANGE_ROOM MOVE>green_chest	3
GOTO_Password -> CHANGE_ROOM MOVE>gold_portal MOVE>gold_corridor CHANGE_ROOM MOVE>green_woman CHANGE_ROOM MOVE>green_chest	3
GOTO_LockPick -> MOVE>enterance MOVE>red_man CHANGE_ROOM MOVE>gold_lockpick	3
GOTO_Exit -> MOVE>gold_lockpick CHANGE_ROOM MOVE>gold_portal CHANGE_ROOM MOVE>enterance	3
GOTO_Exit -> MOVE>blue_chest MOVE>blue_lever MOVE>enterance CHANGE_ROOM	3
GOTO_Exit -> CHANGE_ROOM MOVE>gold_portal CHANGE_ROOM MOVE>enterance	3
GOTO_BlueKey -> MOVE>gold_lockpick CHANGE_ROOM MOVE>gold_portal CHANGE_ROOM MOVE>enterance CHANGE_ROOM MOVE>blue_lever CHANGE_ROOM MOVE>blue_chest	3

Figure A-4 Most frequent motion rules and their frequencies.

BEGIN_TRIAL
there is a red archway on your right
go through that
go straight through that to the other side of the room
go through the door there
go through this hallway to the end
there is a door on your right
get the gem from the chest
talk to the person
go through that door
get the password from the chest
now activate the portal
allright now bash the door directly in front of you
go through it
flip the floor lever to open the door in front of you
go through that
get the key from the chest
there is a door behind you
on your right
get the lockpicks from the chest
the door in front of you that you didnt come through
go through that
turn around
there is a door behind you
bash it
actually it should open
open it
now open that door
EXIT_COMPLETE

Figure A-5. Human transcription of sample NeverWinter Nights trial

Appendix B

Evaluation of Event Representations

Our methodology for representing events in sports video is to extract multiple streams of features that are then mined for temporal patterns which correlate with high level events (see Section 5.2 for more detail). In this appendix, evaluations are presented showing both the performance of the feature extraction techniques, as well as, the informativeness of the temporal patterns mined from the video.

B.1 Evaluating Visual Context Features

Visual context features encode general properties of the visual scenes in a video. These features are generated by first segmenting a raw video into shots, extracting individual (key) frames from those shots, and classifying those key frames into sports-specific categories useful for representing events. The following sections describe experiments run to evaluate the performance of feature extraction in broadcast baseball and American football.

B.1.1 Baseball

Classifying frames of broadcast baseball games is treated as a two step, hierarchical process. In the first step, three key frames are extracted from each shot and classified into one of three categories: *pitch-shot, field-shot,* and *other-shot*. In the next step, all *field-shots* are resampled at finer granularity (one key frame extracted per 10 frames of video) and classified into one of eight categories: *audience-frame, on-base-frame, infield-frame, outfield-frame, deep-outfield-frame, full-field-frame, misc-frame,* and *running-frame.* In both stages, key frames are represented using the set of features described below:

- Duration: the duration (in frames) of the shot from which the frame was extracted.
- Camera motion: the median values of camera motion detected in the pan, tilt, and zoom dimensions of the shot from which the frame was extracted. Motion is detected using the implementation of Bouthemy et al. (1999).
- Grass and soil pixels, the number of pixels with color values characteristic of grass (hue=0.19-0.46; sat.=0.2-0.7; bright >0.3) and soil (hue=0.06-0.15; sat.=0.25-0.8; bright >0.5) in the key frame. Values are reported for the entire frame as well as just the top, bottom, left, and right halves of the frame.
- Grass and soil ratios: the ratio of grass and soil colored pixels by region of key frame (e.g., number of grass pixels in whole frame vs. number of soil pixels in whole frame). Ratios include grass vs. soil, grass vs. grass, and soil vs. soil, in the regions whole frame vs. whole frame, top vs. bottom, and left vs. right.
- Grass histogram stats: statistics measuring the distribution of grass pixels within a frame. This feature is taken from work on *pitch-shot* detection (Pei and Chen, 2003), and exploits the fact that pitching shots have very characteristic distributions of grass colored pixels (see Figure B-1). To create these features, a histogram is generated which bins the number of grass colored pixels along the vertical and horizontal axes of the frame. The features used for classification represent the number of bins in those histograms that have values above the mean bin value in the entire histogram, in the first half of the histogram, and in the first ¾ of the histogram.
- Entropy: the entropy of the intensity values of the image frame in the following regions: the whole frame, the center of the frame, the top, bottom and each quadrant of the frame.
- Entropy ratios: the ratio of entropy values in the top vs. bottom of the frame and the center vs. whole frame.
- Line stats: the number, maximum length and slop of max length line found in the frame. Lines are discovered using Canny edge detection and Hough transforms.
- Color histogram stats: the frame is histogramed by color into 16 bins and the number of pixels in each bin are used.

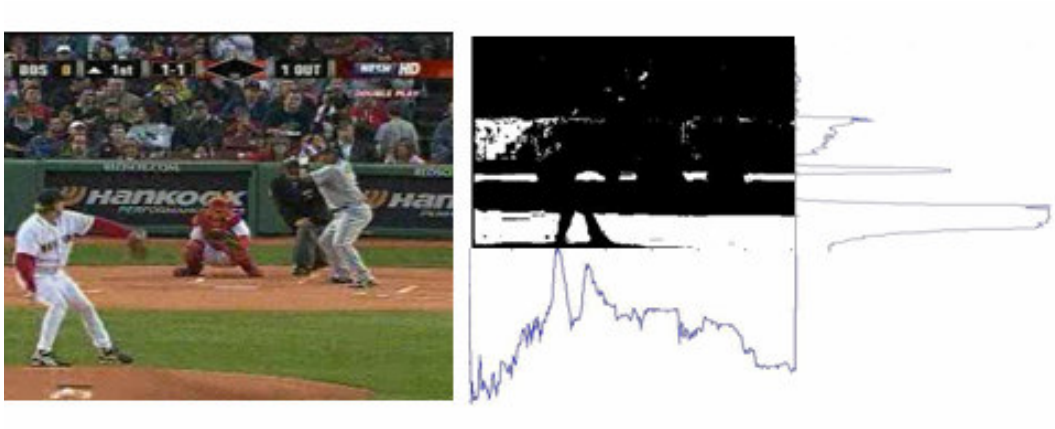

Figure B-1. Characteristic distributions of grass colored pixels in *pitching-frame*.

- Face and uniform colors: connected components of skin colored pixels (hue<0.138; sat.=0.23-0.68; bright=0.35-0.95) are found and features are generated based on the number, maximum size, mean size, and standard deviation of those components. Features are also generated for connected components of the most prominent color in the image (which is often a players' uniforms).

Classification evaluations were run on a corpus of hand-labeled frames from a held-out set of baseball games (i.e. games not used in the grounded language modeling experiments described in Chapter 6 or Chapter 7). The labeling was performed iteratively by first hand labeling frames for part of one game, training a classifier with those labels, running the classifier over a set of new games, correcting the errors and repeating. This procedure allowed a very large set of frames to be labeled (approximately 14,000 frames for each classifier) with relatively little human effort (approximately 5 human hours). Figure B-2 shows results of 10-fold cross validation experiments run on the entire training set for both stages of classification. Classifiers are decision trees trained with boosting and bagging (with five subcommittees, run for 50 iterations) using the WEKA machine learning tookit (Witten and Frank, 2005).

B.1.2 Football

Extracting visual context features from football video operates just as described above for baseball. Classification operates in two stages, first key frames are classified into one of the following categories: *field-shot, goal-post-shot, umpire-shot,* and *misc-shot.* Then all shots not classified as *misc-shot* are resampled (at one key frame per 10 frames). However, unlike

baseball, these resampled shots are classified into one of the same four categories. Thus only one classifier is trained for the football experiments. Also unlike baseball, a different set of features is used to represent each key frame. The set of features is described below:

- Duration: the duration (in frames) of the shot from which the frame was extracted.
- Camera motion: the median values of camera motion detected in the pan, tilt, and zoom dimensions of the shot from which the frame was extracted. Motion is detected using the implementation of Bouthemy et al. (1999).
- Grass pixels: the total number of grass colored pixels.
- Goal-post lines: a mask is used to identify all pixels of the characteristic color associated with goal posts (hue=0.16-0.24; sat.=0.3-0.75; bright>0.65). Canny edge detection and hough transforms are used to identify goal post colored lines, and features are generated representing the number, total length, maximum length, mean length, median length, and standard deviation of lengths for those lines.
- Striped lines: similar to the goal-post line features, these features look for high contrast lines indicative of black and white striping (useful for identifying shots of the umpire). Canny edge detection is used on the intensity image, with a threshold set to 0.15. Features are generated representing the number, total length, maximum length, mean length, median length, and standard deviation of lengths for lines output by the Hough transform.
- Face detection features: a face detection algorithm is used to put bounding boxes on faces in the frame (Kienzle, et al., 2005). These features represent the size of the largest face found, the mean size and standard deviation of all faces, and the number of faces found in the whole/top/bottom/left/right of the image. Features are generated for two threshold settings: 5 and -5.
- Region stats: in addition to the above stats, the frame is split into 9 equal sized regions and further features are generated for each of these individual regions. These features are:
    - Total number of grass/black/white pixels in the region
    - Entropy of the region
    - Stripe features: the number of stripes in region, the ratio of stripes in region to total stripes in frame, the maximum/mean/median/standard deviation of stripes in the region.

Training of the classifier proceeded as with the baseball video. Figure B-2 reports results of 10 fold cross validation on a set of approximately 10,000 frames from games not used in grounded language model evaluations. Classifiers are decision trees trained using boosting and bagging (with three subcommittees, run for 25 iterations) with the WEKA machine learning toolkit (Witten and Frank, 2005).

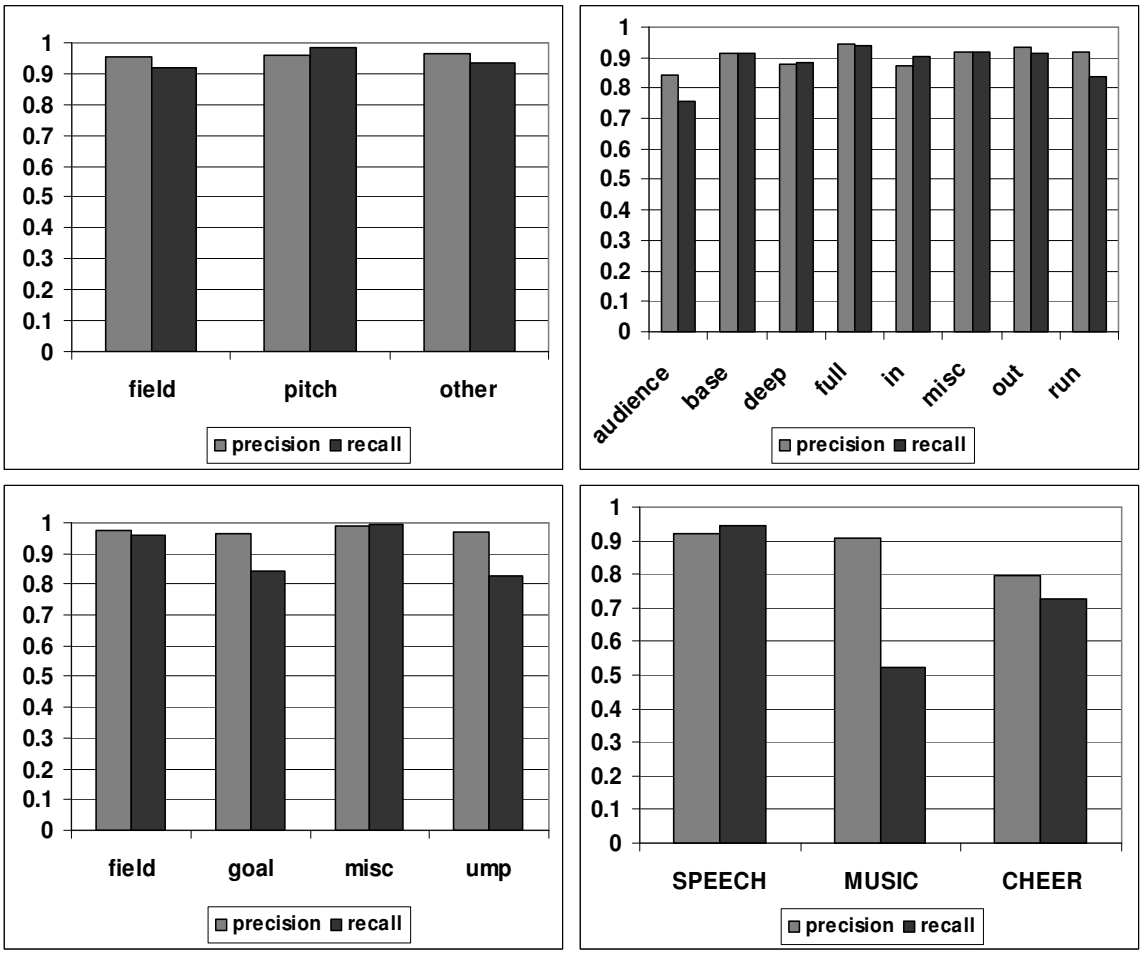

Figure B-2. Precision/recall results of shot classifiers for baseball and American football.

B.2 Evaluating Audio Context Features

Audio context features are generated from video by sampling a sequence of 30ms overlapping frames, representing those frames as a vector of low level features, and then classifying each frame using a series of binary classifiers for the classes: speech, music, and cheering. Each frame is represented using mel-frequency cepstral coefficients (MFCCs), energy in the audio signal, the number of zero crossings, spectral entropy, and relative power between different frequency bands. Training data is generated using a similar technique to that used for visual context features. A small set of audio segments are hand annotated using the Transcriber tool (Barras et al., 2000). Overlapping frames from these segments are extracted and represented as feature vectors. These instances are used to train a decision tree with boosting and bagging

Table B-1. Confusion matrices or baseline classifier (above) and classifier using temporal pattern features of depth 5 (below)

[hyp→]	Home	Out Hit	In Out	Strike	Out Out	In Hit	Walk	*[rec]*
Home	3	2	0	2	0	0	0	.43
OutHit	1	37	5	1	7	0	1	.71
InOut	0	6	60	3	3	1	0	.82
Strike	0	0	5	36	0	0	5	.78
OutOut	0	12	2	0	18	0	0	.56
InHit	0	2	1	0	0	0	0	0
Walk	0	0	0	14	0	0	10	.42
[prec]	.75	.63	.82	.64	.64	0	.63	

[hyp→]	Home	Out Hit	In Out	Strike	Out Out	In Hit	Walk	*[rec]*
Home	3	2	1	1	0	0	0	.43
OutHit	1	31	10	3	6	0	1	.60
InOut	0	5	60	3	5	0	0	.82
Strike	0	2	3	31	0	0	10	.67
OutOut	3	16	2	0	11	0	0	.34
InHit	0	2	1	0	0	0	0	0
Walk	0	0	5	13	0	0	6	.25
[prec]	.43	.53	.73	.61	.5	0	.35	

(with three subcommittees, run for 75 iterations) using WEKA (Witten and Frank, 2005). This classifier is then used to label more held out data, which is corrected by hand, and then the process is repeated. The final training set contained approximately 60,000 training frames (30,000 for speech, 10,000 for music, 20,000 for cheering). Figure B-2 shows the 10-fold cross validation performance for the three binary classifiers used to extract audio content features.

B.3 Evaluating Temporal Patterns

Grounded language models represent the non-linguistic context of sports video using temporal patterns automatically mined from a large corpus of video. The corpus is represented as multiple streams of features that describe the visual context, audio context, and camera motion occurring during each frame of video. By representing events as temporal patterns mined from such features, we are able to encode complex relationships that often co-occur with high level events in sports. The experiments described in Chapter 6 and Chapter 7 implicitly justify the use of such event representations based on extrinsic evaluations of information retrieval and automatic speech recognition. Below we describe a more explicit evaluation in which temporal

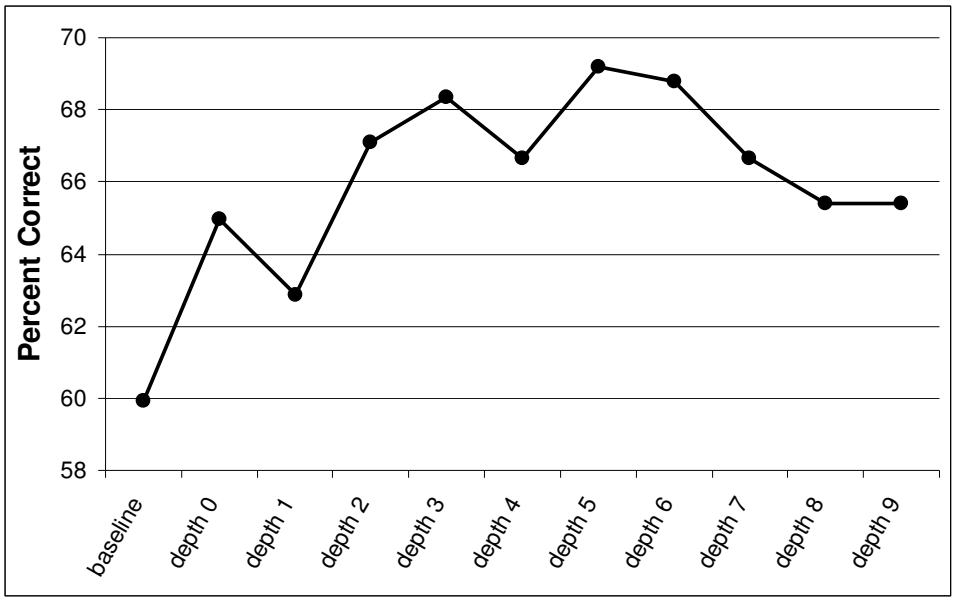

Figure B-3. Comparison of classifiers using temporal pattern features. Depth level corresponds to complexity of temporal features (i.e., number of iterations of algorithm). Baseline uses only low level features.

patterns are used to improve the ability of a system to distinguish between different types of events in baseball video.

A test set was created by extracting 237 events from 6 baseball games in which nine teams played in four stadiums which were broadcast on four US television stations.[55] Following Gong et al (2004), each highlight was hand labeled into one of seven categories: *homerun, outfield hit, infield hit, strikeout, outfield out, infield out,* and *walk.* We train a decision tree with bagging and boosting using leave on out cross-validation (with five subcommittees, run for 50 iterations) using the WEKA machine learning toolkit (Witten and Frank, 2005).

In order to examine the informativeness of temporal patterns, we set up a baseline classification system which follows Gong et al. (2004) and is trained only on the low level features used to train the visual context features. Temporal patterns are evaluated by iteratively adding pattern features mined by the algorithm to these baseline features.

[55] These games are the same as those used in the test set described in Section 6.2.1.

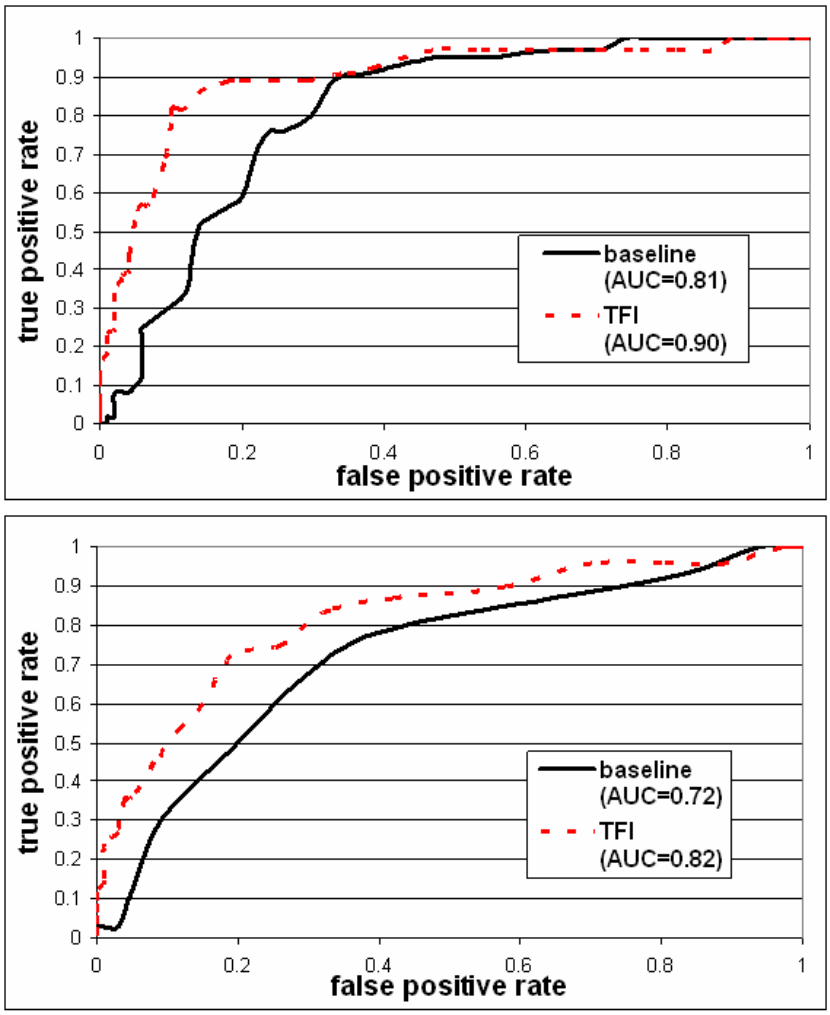

Figure B-4. ROC curve for classification of *left field* highlights (above) and *fly ball* highlights (below). Baseline is compared to classifier using temporal pattern feature. AUC reports area under the curve for each classifier.

Figure B-3 shows the accuracy of the baseline system compared to systems using temporal patterns of increasing levels of complexity. Here the level refers to the maximum depth of the temporal pattern mined (i.e. the number of iterations used by the data mining algorithm), where depth 0 refers to only using the duration of the visual context, audio context, and camera motion features (e.g. *pitching-scene* for 90 frames, *cheering* for 20 frames, etc.), depth 1 refers to relations between two categories (e.g., [BEFORE, *pitching-shot*, *field-shot*] for 90 frames), etc.

These results demonstrate statistically significant improvement ($p<0.05$; $n=237$, one-tail) using temporal patterns of depth two and greater, with a peak performance at depth five. In order to understand the nature of this performance increase, we show the confusion matrices

for these two systems (baseline and depth five) in Table B-1. These tables show that temporal feature induction improves precision and recall for all classes, with the most notable increases coming from the categories *walk* and *outfield out*. As can be seen in the confusion matrices, these two categories are often confused in the baseline system with the visually similar categories *strikeout* and *outfield hit,* respectively. The system using temporal features is less prone to such confusions, because of the finer grained temporal information that it captures.

The benefit of this finer grained information is even more pronounced when finer grained classifications are required. For highlight classes focusing on specific types of hits (e.g. *fly balls*) or specific locations of hits (e.g. *left field*), using temporal features becomes increasingly useful. Figure B-4 show ROC curves for these example classes.[56]

The results of these experiments show that temporal pattern features encode information useful for distinguishing events in sports video. This supports the intuition that temporal patterns between contextual features of video provide a useful basis for representing events in grounded language models of sports video.

[56] ROC curves plot the tradeoff between the true positive and false positive rates as the threshold used for classification is changed.

Appendix C

Workflow for Grounded Language Modeling

The following describes the process workflow for generating grounded language models from the collection of videos from broadcast cable to the modeling of the relationship between event representations and words from the closed captioning. The Figure C-1 illustrates this process with different colored boxes representing the type of tools used: red boxes indicate 3rd party software implementations, dark blue boxes indicate the use of 3rd party toolkits, and light blue boxes indicate code generated entirely by the author. Descriptions of each box are further described below with references to the locations of software used given as footnotes.

- **Raw MPEG video:** raw video is captured from broadcast television (at a rate of 29.97 frames/sec) using the commercially available EyeTV mpeg encoder from Elgato Systems.
- **Remove commercials:** commercials are identified using free software called comskip[57] and ignored throughout the remainder of video processing
- **Extract raw camera motion:** camera motion is computed using free software called Motion2D.[58] The output of this system is converted using equations (Bouthemy et al., 1999) into three value output for pan, tilt, and zoom.

[57] http://www.kaashoek.com/comskip/
[58] http://www.irisa.fr/vista/Motion2D/index.html

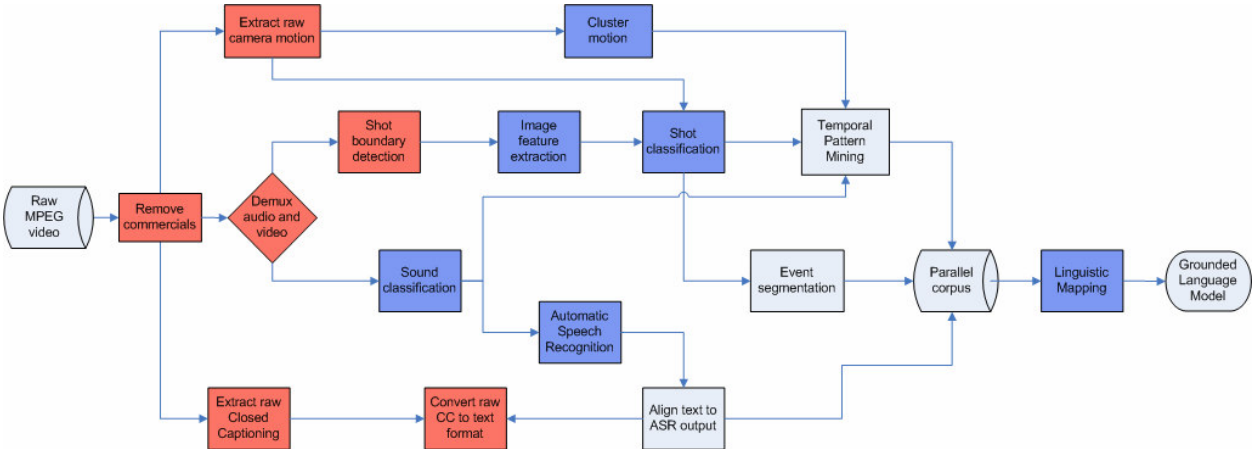

Figure C-1. Workflow of video processing.

- **Cluster motion:** the raw camera motion is clustered using an HMM trained with Jeff Blimes Graphical Modeling Toolkit (GMTK).[59]
- **Demux Audio and Video:** the raw video is demuxed into separate audio and video files using the BBDEMUX.exe tool available with the comclean software utility.[60]
- **Shot boundary detection:** shot boundary detection is performed using an implementation of Tardini et al. (2005) built by the researchers. Thanks to Constantino Grana for help obtaining the software.
- **Image feature extraction:** image features were extracted using scripts written with the Matlab image processing toolkit.[61]
- **Shot classification:** shots were categorized using classifiers trained with the WEKA machine learning toolkit.[62]
- **Event segmentation:** events are segmented based on the output of shot classification. Individual event clips are cut from the larger video using mpgtx.exe available with the comclean software.[63]
- **Sound classification:** sound classification operates on MFCC audio features extracted using sphinx 4 tool.[64] Audio is first normalized using the lame free audio software.[65] Classification was based on classifiers trained using the WEKA machine learning toolkit.[66] Thanks to Brandon Roy for implementing the majority of sound classification.
- **Temporal Pattern Mining:** temporal pattern mining was implemented by the author in the JAVA programming language.[67]
- **Automatic Speech Recognition:** ASR was performed using the sphinx 3 tool.[68]

[59] http://ssli.ee.washington.edu/~bilmes/gmtk/
[60] http://www.kaashoek.com/comskip/
[61] http://www.mathworks.com/
[62] http://www.cs.waikato.ac.nz/ml/weka/
[63] http://www.kaashoek.com/comskip/
[64] http://cmusphinx.sourceforge.net/
[65] http://audacity.sourceforge.net/download/windows
[66] http://www.cs.waikato.ac.nz/ml/weka/
[67] http://www.java.com/en/

- **Extract raw Closed Captioning:** raw closed captioning is extracted using the commercially available software, MPEG STRIP.[69]
- **Convert raw CC to text format:** raw closed captioning is converted to text using the free tools SCC tools.[70]
- **Align text to ASR output:** closed captioning to ASR alignment was implemented by the author in the JAVA programming language. [71]
- **Linguistic mapping:** linguistic mapping is performed using the freely available Topic Modeling Toolbox for Matlab.[72] The toolbox was modified slightly such that the function GibbsSamplerAT in file GibbsSamplerAT.cpp was changed to incorporate the probability of author k when calculating the probability of assigning a word token to a topic j and an author k.

---

[68] http://cmusphinx.sourceforge.net/

[69] http://homepage.mac.com/dvd_sp_helper/

[70] http://www.geocities.com/mcpoodle43/SCC_TOOLS/DOCS/SCC_TOOLS.HTML

[71] http://www.java.com/en/

[72] http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm

Bibliography

Allen, J.F. (1984). A General Model of Action and Time. Artificial Intelligence. 23(2).

Appelt, D. and Pollack, M. (1992). Weighted abduction for plan ascription. *In Proceedings of User Modeling and User-Adapted Interaction*.

Babaguchi, N., Kawai, Y., and Kitahashi, T. (2002). Event Based Indexing of Broadcast Sports Video by Intermodal Collaboration. IEEE Transactions on Multimedia. 4(1): 68-75.

Bailey, D. (1997). When push comes to shove: A computational model of the role of motor control in the acquisition of action verbs. Unpublished Ph.D. Dissertation, University of California, Berkeley.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998): The Berkeley FrameNet project. *In Proceedings of the COLING-ACL*, Montreal, Canada.

Baldwin, D. & Baird, J. (2001). Discerning Intensions in Dynamic Human Action. Trends in Cognitive Science. 5(4): 171-178.

Barras, C., Geoffrois, E., Wu, Z., and Liberman, M. (2000). Transcriber: development and use of a tool for assisting speech corpora production . Speech Communication special issue on Speech Annotation and Corpus Tools. 33(1-2).

Barnard, K., Duygulu, P., de Freitas, N., Forsyth, D., Blei, D., and Jordan, M. (2003). Matching Words and Pictures. Journal of Machine Learning Research, Vol. 3, 1107-1135.

Bergen, J., Anandan, P., Hanna, K. and Hingorani, R. (1992). Hierarchical Model-Based Motion Estimation. *In Proceedings of the Second European Conference on Computer Vision.*

Berger, A. and Lafferty, J. (1999). Information Retrieval as Statistical Translation. In Proceedings of SIGIR-99.

Bhagat, R., Leuski, A., and Hovy, E. (2005). Shallow Semantic Parsing despite Little Training Data. *In the ACL/SIGPARSE 9th International Workshop on Parsing Technologies.* Vancouver, B.C., Canada.

Blei, D. Ng, A., and Jordan, M (2003). Latent Dirichlet allocation. Journal of Machine Learning Research 3:993-1022.

Blei, D. and Jordan, M. (2003). Modeling annotated data. *In proceedings of the 26th International Conference on Research and Development in Information Retrieval.*

Blum , A. and Mitchell, T. (1998). Combining Labeled and Unlabeled Data with Co-Training. *In Proceedings of the Workshop on Computational Learning Theory.*

Bouthemy, P., Gelgon, M., Ganansia, F. (1999). A unified approach to shot change detection and camera motion characterization. IEEE Trans. on Circuits and Systems for Video Technology, 9(7):1030-1044.

Boreczky, J. and Rowe, L. (1996). Comparison of video shot boundary detection techniques. *In Proc. SPIE Storage and Retrieval for Image and Video Databases.*

Brown, P. F. Della Pietra, V. J. Della Pietra S. A. & Mercer., R. L. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics 19(2).

Charniak, E., and Goldman, R.P. (1993). A Bayesian model of plan recognition. *Artificial Intelligence*, pages 53-79.

Chen, S. F. and Goodman, J., (1998). An Empirical Study of Smoothing Techniques for Language Modeling, Tech. Report TR-10-98, Computer Science Group, Harvard U., Cambridge, MA.

Coen, M. (2006). Self-Supervised Acquisition of Vowels in American English. *In Proceedings of the Twenty First National Conference on Artificial Intelligence* (AAAI'06). Boston, MA.

Cohen, P. R. (2001). Fluent Learning: Elucidating the Structure of Episodes. *In Proceedings of the Fourth Symposium on Intelligent Data Analysis*. London, England.

Collins, M. (1999). Head-Driven Statistical Models for Natural Language Parsing. Unpublished Doctoral Dissertation, University of Pennsylvania.

Daume, H. III & D. Marcu, (2005). Approximate Large Margin Learning for Structured Outputs as a Search Optimization Problem. *In Proceedings of the International Conference on Machine Learning* (ICML).

Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W. and Harshman, R. A. (1990). Indexing by latent semantic analysis. Journal of the Society for Information Science. 41(6): 391-407.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. Computational Linguistics. 19:61-74.

Epstein, M. (1996). Statistical Source Channel Models for Natural Language Understanding. Unpublished Doctoral Dissertation, New York University.

Feldman, J. and Narayanan, S. (2004). Embodied Meaning in a Neural Theory of Language. Brain and Language. 8: 385-392.

Fern, A., Givan R., Siskind, J. (2002). Specific-to-General Learning for Temporal Events with Application to Learning Event Definitions from Video. Journal of Artificial Intelligence Research (JAIR) 17: 379-449.

Fillmore, C. (1976). Frame semantics and the nature of language. Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech, Vol. 280, 20 32.

Fleischman M, Roy, D. (2008). Grounded Language Models for Speech Recognition in Sports Video. *In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Columbus, Ohio.

Fleischman M, Roy, D. (2007a). Situated Models of Meaning for Sports Video Retrieval. *In Proceedings of the Conference on Human Language Technology/ North American Association of Computational Linguistic*s (*HLT/NAACL)*. Rochester, NY.

Fleischman, M. and Roy, D. (2007b). Unsupervised Content-Based Indexing of Sports Video Retrieval. *In Proceedings of the 9th ACM Workshop on Multimedia Information Retrieval (MIR)*. Augsburg, Germany.

Fleischman M, Roy, D. (2007c). Representing Intentions in a Cognitive Model of Language Acquisition: Effects of Phrase Structure on Situated Verb Learning. *AAAI Spring Symposium*. Palo Alto, CA.

Fleischman, M., Roy, B., and Roy, D. (2007). Temporal Feature Induction for Sports Highlight Classification. In Proceedings of ACM Multimedia. Augsburg, Germany.

Fleischman, M. B. and Roy, D. (2005a). Why Verbs are Harder to Learn than Nouns: Initial Insights from a Computational Model of Intention Recognition in Situated Word Learning. *In Proceedings of the 27th Annual Meeting of the Cognitive Science Society*. Stresa, Italy.

Fleischman, M. B. and Roy, D. (2005b). Intentional Context in Situated Language Learning. *In Proceedings of the Ninth Conference on Computational Natural Language Learning* . Ann Arbor, MI.

Fleischman, M. and Hovy, E. (2006). Taking Advantage of the Situation: Non-Linguistic Context for Natural Language Interfaces to Interactive Virtual Environments. *In Proceedings of the Conference on Intelligent User Interfaces (IUI).* Sydney, Australia.

Fleischman, M., DeCamp, P. Roy, D. (2006). Mining Temporal Patterns of Movement for Video Content Classification. *The 8th ACM SIGMM International Workshop on Multimedia Information Retrieval*. Santa Barbara, California.

Fleischman, M., Kwon, N., and Hovy, E. (2003). Maximum Entropy Models for FrameNet Classification. *In Proceedings of the Conference on Empirical Methods for Natural Language Processing.* Sapporo, Japan.

Ge, R. and Mooney, R.J. (2005). A Statistical Semantic Parser that Integrates Syntax and Semantics. *In Proceedings of the Ninth Conference on Computational Natural Language Learning*, Ann Arbor, MI.

Gentner, D. (1982). Why nouns are learned before verbs: Linguistic relativity versus natural partitioning. In S. Kuczaj, editor, Language development: Vol. 2. Language, cognition, and culture. Erlbaum, Hillsdale, NJ.

Gildea, D. and Jurafsky, D. (2002). Automatic Labeling of Semantic Roles. Computational Linguistics, 28(3) 245-288 14.

Gleitman, L. (1990). The structural sources of verb meanings. Language Acquisition, 1(1).

Gong, Y., Han, M., Hua, W., and Xu, W. (2004). Maximum entropy model-based baseball highlight detection and classification. Computer Vision and Image Understanding. 96(2).

Gorniak, P. and Roy, D. (2005) Probabilistic Grounding of Situated Speech using Plan Recognition and Reference Resolution. *In Proceedings of International Conference on Multimodal Interfaces (ICMI).*

Harnad, S. (1990). The symbol grounding problem. Physica D, 42, 335-346.

Hauptmann, A. and Witbrock, M., (1998) Story Segmentation and Detection of Commercials in Broadcast News Video. *Advances in Digital Libraries*.

Hazen, T.J. (2006). Automatic Alignment and Error Correction of Human Generated Transcripts for Long Speech Recordings, *In Proceedings. of Interspeech*. Pittsburgh, Pennsylvania.

Hongen, S., Nevatia, R. Bremond, F. (2004). Video-based event recognition: activity representation and probabilistic recognition methods. Computer Vision and Image Understanding. 96(2).

Hongeng, S. and Nevatia, R. (2001). Multi-Agent Event Recognition. *In Proceedings of IEEE International Conference on Computer Vision*. II: 84-91.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D, and Rommelse, K. (1998). The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users. *In Proc. of the 14th Conference on Uncertainty in Artificial Intelligence (UAI).*

Hsu , Bo-June (Paul). (2007). Generalized Linear Interpolation of Language Models. In *Proc. of Automatic Speech Recognition and Understanding (ASRU)*, Kyoto, Japan.

Höppner, F. (2001). Discovery of Temporal Patterns. Learning Rules about the Qualitative Behaviour of Time Series. *In Proceedings of the Conference on Knowledge Discovery in Data.* 192-203.

Intille, S. and Bobick, A.F. (2001). Recognizing planned, multi-person action. Computer Vision and Image Understanding. 81.

Ivanov, Y. and Bobick, A. (2000). Recognition of Visual Activities and Interactions by Stochastic Parsing. IEEE Transactions on Pattern Analisys & Machine Intelligence. 22(8).

Jang, P. and Hauptmann, A. (1999). Learning to Recognize Speech by Watching Television. IEEE Intelligent Systems Magazine, 14(5): 51-58.

Kaelbling, L.P., Littman, M.L. and Cassandra, A.R. (1998). Planning and acting in partially observable stochastic domains. Artificial Intelligence Journal. 101: 99-134.

Kienzle, W., Bakir, G., Franz, M. and Scholkopf, B. (2005). Face Detection-Efficient and Rank Deficient. *In: Advances in Neural Information Processing Systems*. 17: 673-680.

Li, B. and Sezan, M.I. (2001). Event Detection and Summarization in Sports Video. *In Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL)*.

Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: the Latent Semantic Analysis theory of acquisition, induction and representation of knowledge. Psychological Review, 104(2) , 211-240.

Landauer, T. K., Foltz, P. W., and Laham, D. (1998). Introduction to Latent Semantic Analysis. Discourse Processes. 25: 259-284.

Lenat, D.B. (1995). CYC: A Large-Scale Investment in Knowledge Infrastructure. Communications of the ACM, 38(11): 32-38.

Manning, C., Schutze, H. (2001). Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA.

Miller, G. A., Galanter, E. and Pribram K. H. (1960). Plans and the Structure of Behavior. Halt ,New York, NY.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1990). Five papers on Wordnet. International Journal of Lexicology, 3(4).

Minsky, M. (1974). A Framework for Representing Knowledge. MIT-AI Laboratory Memo 306. Reprinted in: They Psychology of Computer Vision. P. Winston (Ed.). McGraw-Hill, 1975.

Moore, R. C. (2004). Improving IBM Word Alignment Model 1. *In Proc. of 42nd Association of Computational Lingusitics (ACL)*. Barcelona, Spain.

Naphade, M, Smith, J., Tesic, J., Chang, S., Hsu, W., Kennedy, L., Hauptmann, A., Curtis, J. (2006). Large-Scale Concept Ontology for Multimedia. IEEE Multimedia. 13(3): pp. 86-91.

Narayanan, S. (1999). Moving right along: A computational model of metaphoric reasoning about events. *In Proceedings of the American Association of Artificial Intelligence (AAAI)*. Orlando, FL.

Orkin, J. and Roy, D. (2007). The Restaurant Game: Learning Social Behavior and Language from Thousands of Players Online. Journal of Game Development. 3(1), 39-60.

Pei, S. and Chen, F. (2003). Semantic Scene Detection and Classification in Sports Video. *In proceedings of 16th Conference on Computer Vision and Image Processing*.

Palmer, M., Gildea, D., Kingsbury, P. (2005). The Proposition Bank: A Corpus Annotated with Semantic Roles. *Computational Linguistics Journal*. 31:1.

Picard, R. W. (1997). Affective Computing**.** MIT Press, Cambridge, MA.

Pustejovsky, J. (1991). The generative lexicon. Computational Linguistics, 17(4), 209-441.

Pynadath, D. (1999). Probabilistic Grammars for Plan Recognition. Unpublished doctoral dissertation, University of Michigan.

Qu, S., and Chai, J. (2006). Salience modeling based on non-verbal modalities for spoken language understanding. *In proceedings of the 8th Internationsl conference on Multimodal Interfaces (ICMI)*. Banff, Alberta, Canada.

Quillian, M. Ross. (1967). Word Concepts: A Theory and Simulation of Some Basic Semantic Capabilities. Behavioral Science. 12: 410-430.

Rabiner, L.R. and Juang, B.H. (1993). Fundamentals of Speech Recognition. Prentice Hall, Saddle River, NJ.

Regier, T. (2003). Emergent constraints on word-learning: A computational review. Trends in Cognitive Science. 7: 263-268.

Rickel, J., Marsella, S., Gratch, J., Hill, R., Traum, D. and Swartout, W. (2002). Towards a New Generation of Virtual Humans for Interactive Experiences. In IEEE Intelligent Systems.

Rosch, E., C.B. Mervis, W. Gray, D. Johnson, and P. Boyes-Braem. (1976). Basic objects in natural categories. Cognitive Psychology. 8: 382-439.

Roy, D. (2005). Grounding Words in Perception and Action: Insights from Computational Models. Trends in Cognitive Science. 9(8), 389-396.

Roy, D. and Mukherjee, N. (2005). Towards Situated Speech Understanding: Visual Context Priming of Language Models. Computer Speech and Language. 19(2): 227-248.

Roy. D. and Reiter, E. (2005). Connecting Language to the World. Artificial Intelligence 67:1-12.

Roy, D. and Pentland, A. (2002) Learning Words from Sights and Sounds: A Computational Model. Cognitive Science. 26(1): 113-146.

Rui, Y., Gupta, A. and Acero, A., (2000) Automatically extracting highlights for TV baseball programs. *In Proceedings of ACM Multimedia*. Marina del Rey, CA.

Russell, S. and Norvig, P. (1995). Artificial Intelligence: A Modern Approach. Prentice Hall, Saddle River, NJ.

Schank, R.C. & Abelson, R. (1977). Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Associates, Hillsdale, NJ.

Siskind, J. (2001). Grounding the Lexical Semantics of Verbs in Visual Perception using Force Dynamics and Event Logic. Journal of Artificial Intelligence Research. 15: 31-90.

Siskind, J.M. (1996). A Computational Study of Cross-Situational Techniques for Learning Word-to-Meaning Mappings. Cognition. 61(1-2): 39-91.

Snedeker, J. & Gleitman, L. (2004). Why it is hard to label our concepts. To appear in Hall & Waxman (eds.), Weaving a Lexicon. MIT Press, Cambridge, MA.

Snoek, C.G.M. and Worring, M. (2005). Multimodal video indexing: A review of the state-of-the-art. Multimedia Tools and Applications. 25(1):5-35.

Song, F. and Croft, W.B. (1999). A General Language Model for Information Retrieval. *In Proceedings of the eighth international conference on Information and Knowledge Management.*

Steyvers, M., Smyth, P., Rosen-Zvi, M., & Griffiths, T. (2004). Probabilistic Author-Topic Models for Information Discovery. *In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Seattle, Washington.

Stolcke, A., (2002). SRILM - An Extensible Language Modeling Toolkit. *In Proc. of the Intl. Conf. Spoken Language Processing*. Denver, Colorado.

Stolcke, A. (1994). Bayesian Learning of Probabilistic Language Models. Unpublished Doctoral Dissertation, University of California, Berkeley.

Swartout, W., Gratch, J., Hill, R.W. Jr., Hovy, E., Lindheim, R., Marsella, S., Rickel, J., Traum, D.R. (2005). Simulation in Hollywood: Integrating Graphics, Sound, Story and Character for Immersive Simulation. In Oliviero Stock and Massimo Zancanaro (Eds), Multimodal Intelligent Information Presentation.

Tan, Y., Saur, D., Kulkami, S. and Ramadge, P. (2000). Rapid estimation of camera motion from compressed video with application to video annotation. IEEE Transactions on Circuits and Systems for Video Technology. 10(1): 133-146.

Tardini, G. Grana C., Marchi, R., Cucchiara, R. (2005). Shot Detection and Motion Analysis for Automatic MPEG-7 Annotation of Sports Videos. *In 13th International Conference on Image Analysis and Processing*.

Vallacher, R. R., & Wegner, D. M. (1987). What do people think they're doing? Action identification and human behavior. Psychological Review. 94: 3-15.

Wactlar, H., Witbrock, M., Hauptmann, A., (1996). Infomedia: News-on-Demand Experiments in Speech Recognition. *ARPA Speech Recognition Workshop*. Arden House, Harriman, NY.

Winograd, T. (1972) Understanding Natural Language. Cognitive Psychology. 3(1): 1-191.

Wittgenstein, L. (1953). Philosophical Investigations. Blackwell, Oxford., England.

Witten, I. and Frank, E. (2005). Data Mining: Practical machine learning tools and techniques. 2nd Edition, Morgan Kaufmann. San Francisco, CA.

Woodward, A., J. Sommerville and J. Guajardo (2001). How infants make sense of intentional action. In Malle, Moses, Baldwin (eds.), Intention and Intentionality. Cambridge, MA: MIT Press.

Xu, F. and Tenenbaum, J.B. (2007). Word learning as Bayesian inference. Psychological Review. 114(2).

Yamada, K. and Knight, K. (2001). A Syntax-Based Statistical Translation Model. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).*

Yoshida, N. (2002). Automatic Utterance Segmentation in Spontaneous Speech. Unpublished Masters Thesis, MIT.

Yu, C., Ballard, D., Aslin, R. (2003). The Role of Embodied Intention in Early Lexical Acquisition. *In Proceedings of the Twenty-Fifth Annual Meeting of Cognitive Science Society*. Boston, MA.

Zettlemoyer, L. and Collins, M. (2005). Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. *In proceedings of Uncertainty in Artificial Intelligence (UAI).*