IDENTIFYING LATENT ATTRIBUTES FROM VIDEO SCENES USING KNOWLEDGE ACQUIRED FROM LARGE COLLECTIONS OF TEXT DOCUMENTS

by

Anh Xuan Tran

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

2014

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by Anh Xuan Tran, titled Identifying Latent Attributes from Video Scenes Using Knowledge Acquired From Large Collections of Text Documents and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

Mihai Surdeanu — Date: 30 April 2014

Kobus Barnard — Date: 30 April 2014

Ken S. McAllister — Date: 30 April 2014

_____ — Date: 30 April 2014

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Dissertation Director: Paul R. Cohen — Date: 30 April 2014

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

SIGNED: Anh Xuan Tran

ACKNOWLEDGEMENTS

First and foremost, I would like to express my heartfelt gratitude to my PhD advisors, professors Paul Cohen and Mihai Surdeanu, for their patient guidance, enthusiastic encouragement, and insightful discussions. While most graduates are lucky to have one great mentor, I was blessed with two. Paul's vast and expansive view of Artificial Intelligence gave me the golden opportunity to explore a wide range of areas and interests, while Mihai's unrivaled understanding of natural language processing was invaluable during the development of my dissertation. I am convinced that they are the among the most intelligent men I have ever met, and it was truly an honor and a privilege to call them my advisors.

I would also like to thank my other committee members, professors Kobus Barnard and Ken McAllister, for their advice and assistance in shaping my dissertation. I have learned a great deal from our illuminating conversations. I would also like to extend my appreciation to Dr. Clayton Morrison for his support during the final finishes of this work.

Along the way, I also had the pleasure of meeting many wonderful labmates and friends. In particular, I am thankful to Wesley Kerr and Daniel Hewlett for paving the way for our lab, and grateful to my fellow partners in crime Jeremy Wright, Mark Tokutomi, Derek Green, Antons Rebguns, and Marco Antonio Valenzuela Escárcega for sharing parts (if not all) of this journey with me. I also greatly appreciate the always-enlightening companies of the computer vision folks from down the hall: Jinyan Guan, Ernesto Brau, Kyle Simek, and Kate Kharitonova.

To all of my friends in the desert, I thank you for keeping me hydrated and sane throughout graduate school. We have shared so many great memories. From board-game nights to Sunday pool leagues and beyond, I will always treasure our times together.

Lastly, and most importantly, none of this work would have been possible without the love and support of my family. From the bottom of my heart, I thank my parents for giving me the opportunity to succeed, my sister for inspiring me to achieve, and my fiancée Naka for her unwavering love and support.

DEDICATION

For my parents. Thank you for the innumerable number of sacrifices you have made for my future.

TABLE OF CONTENTS

TABLE OF CONTENTS – *Continued*

TABLE OF CONTENTS – *Continued*

LIST OF FIGURES

LIST OF FIGURES – *Continued*

LIST OF TABLES

LIST OF TABLES – *Continued*

LIST OF TABLES – *Continued*

LIST OF TABLES – *Continued*

ABSTRACT

Peter Drucker, a well-known influential writer and philosopher in the field of management theory and practice, once claimed that "the most important thing in communication is hearing what isn't said." It is not difficult to see that a similar concept also holds in the context of video scene understanding. In almost every non-trivial video scene, most important elements, such as the motives and intentions of the actors, can never be seen or directly observed, yet the identification of these latent attributes is crucial to our full understanding of the scene. That is to say, latent attributes matter.

In this work, we explore the task of identifying latent attributes in video scenes, focusing on the mental states of participant actors. We propose a novel approach to the problem based on the use of large text collections as background knowledge and minimal information about the videos, such as activity and actor types, as query context. We formalize the task and a measure of merit that accounts for the semantic relatedness of mental state terms, as well as their distribution weights. We develop and test several largely unsupervised information extraction models that identify the mental state labels of human participants in video scenes given some contextual information about the scenes. We show that these models produce complementary information and their combination significantly outperforms the individual models, and improves performance over several baseline methods on two different datasets. We present an extensive analysis of our models and close with a discussion of our findings, along with a roadmap for future research.

CHAPTER 1

INTRODUCTION

Imagine in your mind two young children chasing each other in the playground on a bright sunny day. Are they enjoying themselves? Is the child desperately running for his life, or is he happy and playful? Is it safe or should someone notify the police? The fact that you can answer these questions without ever being informed about the state of mind of each child means that you must know what it's like to be chased by a child in the playground; you've probably experienced it yourself. Our extensive knowledge of past experiences and common sense makes it easy to infer these kinds of unobservable information, such as a person's mental states, from scenes. So easy, in fact, that we often do it automatically without ever thinking about it. Machines, on the other hand, do not have access to the same knowledge and wisdom that humans do. This makes the task of identifying hidden, or latent, attributes from scenes quite challenging for an automated system.

The recognition of activities, participants, and objects in videos has advanced considerably in recent years (Li et al., 2010; Poppe, 2010; Weinland et al., 2011; Yang and Ramanan, 2011; Ng et al., 2012). However, identifying latent attributes of scenes, such as the mental states of human participants, has not been addressed. Latent attributes matter: If a video surveillance system detects one person chasing another, the response from law enforcement should be radically different if the people are jolly and cheerful (e.g., children playing, lovers frolicking), or afraid and angry (e.g., a person fleeing from an assailant).

Our work attempts to solve this higher-level problem: The problem of latent attribute identification. In particular, we do not aim to re-solve the problem of low level video detections (e.g., activity and object recognitions), but instead strive answer the question of what to do next after we have recognized what's happening in the scene.

This dissertation formalizes the task of latent attribute identification from video scenes given some contextual information about the scenes, focusing on the identification of actors' mental states. We propose a novel idea for solving this task based on the use of knowledge extracted from large collections of text documents. Using this approach, we design, develop, and evaluate several largely unsupervised methods for identifying the mental state labels associated with actors in video scenes.

The decision to focus on mental state label identification is purely for academic purposes. Mental state inference is a well-established problem in current literature, which allows us to draw comparison and similarity between our approach and previous research. In fact, we could have chosen any other types of latent attributes as our focus, such as the goals, intentions, motives, and beliefs of the actors. We could also aim to identify latent properties about the scene based on what's happening in the scene, such as whether or not it portrays an anomalous activity or a dangerous location. Or, we could try identifying hidden facts about the scene based on explicit visual cues. For example, consider a scene with a person carrying a bag in the parking lot of a grocery store. Depending on whether or not he is walking towards or away from the store, we could postulate whether or not the bag is empty. We believe that our approach generalizes well to these problems and our solution provides a framework to solve all of them.

Moreover, the idea we propose in this dissertation extends well beyond the video domain of our problem. We believe the main concept of using knowledge extracted from large collections of text documents for problem solving can be applied to many different domains. For example, one could conceivably use our technology to do medical diagnosis. Given a list of symptoms as the contextual information, our approach provides a largely unsupervised method for extracting relevant causes using information from large collections of specialized medical texts.

1.1 Motivation

The correct identification of latent attributes in scenes is pivotal to our understanding of the whole story. Case in point, it is often necessary to know something about the mental states of the actors in the scene (e.g., happy, sad, angry, scared, etc.) in order to correctly interpret their intentions and motives. Currently, applications in video surveillance and video description often have a hard time unravelling the full story of a video, because while they can be very good at recognizing what is happening in the video, they cannot explain why those things happen. Perhaps the former manager of DARPA's Mind's Eye[1] program, James Donlon, said it best: "Labeling a narrowly avoided vehicular manslaughter as *approach(car, person)* is missing something." It is missing the full story, which requires knowledge of latent scene attributes.

Moreover, our own experience from the Mind's Eye program tells us that top-down information can be very valuable for low-level computer vision processes. Knowledge about latent scene attributes can help guide vision-based algorithms by providing good possible explanations for the data. For instance, the attributes could help explain why a person is likely not smiling (e.g., because s/he is in a sad emotional state), or why the track for a particular person is moving faster than normal (e.g., because that person is in a hurry to catch the bus).

1.2 The Problem

Given that latent scene attributes matter and that the identification of these attributes can be beneficial to a wide range of applications, we now formalize the novel task of latent attribute identification from video scenes. In particular, we ground our research to the problem of *identifying the mental state labels of actors participating in a video activity, given that we already know some contextual infor-*

[1]The Mind's Eye program, funded by DARPA, sought to develop the capability for visual intelligence by automating the ability to learn generally applicable and generative representations of action between objects in a scene directly from visual inputs, and then reason over those learned representations.

mation about the video.

As mentioned earlier, our choice to focus on mental state labels is rather arbitrary; we could have selected to concentrate on the identification of any other types of latent attributes in videos. Moreover, we specifically formulated our problem statement to emphasize the focus of our research on identifying the latent elements of the scenes, such mental states, and not the observable information, such as object detections. Thus, the input for our task is whatever can be extracted from video (e.g., detections of activities, actors, objects, and locations) and the output is a description of the mental state labels relevant to the activity and its actors.

1.3 The Approach

Our approach to the problem is based on one key observation: Attributes that are latent in visual representations are often explicit in textual representations. This suggests a novel method for inferring latent attributes: Use explicit features of videos to query text corpora, and from the resulting texts extract attributes that are latent in the videos, such as mental states. Figure 1.1 roughly outlines our idea. The hypothesis is that the kinds of information contained in large text corpora can be applied to explain video scenes.

This insight gives us a way to build largely unsupervised models for acquiring the necessary information to solve this problem. As information is bountiful in existing texts, we can simply mine for relevant knowledge using various information extraction techniques. If it works, this is a drastic improvement over the alternatives of either manually engineering the knowledge into the system, which can get tedious, or learning the knowledge from data, which usually requires a large amount of training samples. Abstractly, the information contained in large collections of text documents is used as a proxy for human knowledge and common sense. This idea is even more enticing when considering the fact that the World Wide Web, which is nothing more than an enormous corpus of text, is one of the largest machine-accessible sources of information out there.

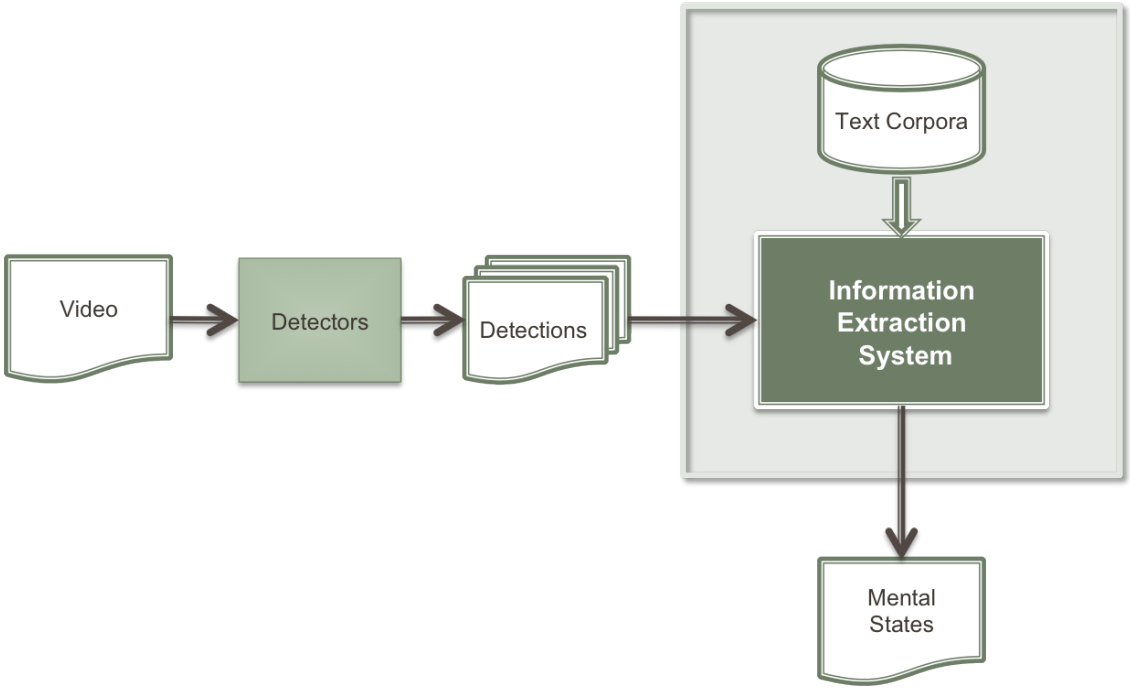


Figure 1.1: High-level schematic of the proposed approach, which is to use explicit features of videos (e.g., detections of activities and actors) to query text corpora, and from the resulting texts extract attributes that are latent in the videos (e.g., mental states of actors). The gray box represents the scope of our proposed system.

Following this idea, we propose several largely unsupervised information extraction (IE) models that identify the mental state labels of human participants in a scene, given some known detections about the scene. Our research investigates several robust models: a lexical semantic (LS) model that extracts mental state terms that are highly similar to the context of the scene in a latent, conceptual space generated by a recurrent neural network language model (RNNLM) (Mikolov et al., 2013a); and an information retrieval (IR) model that identifies mental state labels that commonly appear in sentences related to the explicit scene context. We also investigate an event-centric model that focuses on the mental state terms associated with participants in the relevant event (identified using syntactic patterns and coreference resolution). We design our models to output a normalized distribution over mental state labels for each video.

To test the performance of our models, we use videos from two different datasets.

One contains 26 different chase videos and the other includes 45 different hug videos. For each video, we are interested in how well the output of our models align with human output, generated using crowd sourcing. For this comparison, we investigate and define a novel performance measure based on the standard F_1 measure, called *constrained weighted similarity-aligned* F_1, that accounts for both the differences in shapes between mental state distributions and the semantic relatedness of mental state terms in each distribution (i.e., partial credit is given if a system produces *irate* when the human subject uses *angry*).

1.4 Overview

The remainder of this dissertation is organized as follows: We begin with a literature review of relevant research in Chapter 2, drawing on the fields of computer vision and natural language processing, as well as information extraction and information retrieval. Chapter 3 details the video and text corpora used in our research. In Chapter 4, we present the main contribution of this dissertation: several different information extraction models for identifying latent attributes in video scenes. We then explore several novel performance measures for comparing two different distributions of mental state labels in Chapter 5, and follow with an investigation into the effectiveness of these measures in Chapter 6. Using the most effective measure, we evaluate the performance of our extraction models in Chapter 7. Chapter 8 draws some final conclusions and Chapter 9 closes with possible avenues for future research.

CHAPTER 2

RELATED WORK

As far as we know, the task proposed here is novel. We can, however, review work relevant to each part of the problem and our solution. We divide this chapter into three main sections. The first section discusses research relevant to the problem of mental state inference in general. The next section examines pertinent work in computer vision literature and highlights what is currently feasible with computer vision technologies. The final section presents currently available tools and techniques in natural language processing, and reviews relevant work in the areas of information extraction and information retrieval.

2.1 Mental State Inference

Mental state inference is often formulated as a classification problem, where the goal is to predict target mental state labels based on low-level sensory input data. Most solutions try to learn classification models based on large amounts of training data, while some require human engineering of domain knowledge. We specifically focus on the branch of research that makes inferences based on video data input.

Hidden Markov Models (HMMs) and Dynamic Bayesian Networks (DBNs) are popular representations because they can model the temporal evolution of mental states. For instance, the mental states of students can be inferred from unintentional body gestures using a DBN (Abbasi et al., 2009). Likewise, an HMM can also be used to model the emotional states of humans (Liu and Wang, 2011). Some solutions combine HMMs and DBNs in a Bayesian inference framework to yield a multi-layer representation that can do real-time inference of complex mental and emotional states (El Kaliouby and Robinson, 2004; Baltrusaitis et al., 2011).

Our work differs from these approaches in several ways: It is mostly unsupervised, multi-modal, and requires little training.

2.2 Computer Vision

While development in low-level computer vision detections is not a focus of our work, our models do predicate on the use of vision-based detectors as input. Thus, it is worth reviewing the current state-of-the-art in computer vision technologies to get a better sense of what is feasible.

Computer vision is a matured area of research with a large body of literature covering many application domains. Here, we highlight a few selected solutions pertaining to the areas of object detection and human motion capture. For a more thorough review, we refer the readers to existing surveys, such as (Moeslund and Granum, 2001; Moeslund et al., 2006). Moreover, results from many state-of-the-art vision-based detection systems can be found submitted to the annual PASCAL Visual Object Classes (VOC) Challenge[1] (Everingham et al., 2014).

Object detection is a fundamental challenge in image processing. In their work, Felzenszwalb et al. (2008) described a discriminatively trained, multi-scale, deformable part model that can be trained to detect any types of objects, provided that a sufficient amount of training images is available. They showed the generality of their approach by training detectors for many different ordinary objects, such as: bike, bird, boat, car, chair, table, person, sofa, and others. This so-called "Felzenszwalb-detector" is popular in the vision community for its generality and ease of use. When it comes to performance, however, specialized detectors often outperform the general Felzenszwalb-detector. For example, the work of Ramanan (2012) showed us how to efficiently detect faces and facial poses in daily images, while Yang and Ramanan (2011) gave us a method for estimating the human body pose in 2D images using flexible mixtures-of-parts.

A related task to detection is tracking. The goal of object tracking is to asso-

[1] `http://pascallin.ecs.soton.ac.uk/challenges/VOC/`

ciate the correct identity to detections over a sequence of images. Many tracking algorithms have been developed, such as group tracking (McKenna et al., 2000) or tracking by learning appearance (Ramanan et al., 2007). In general, most tracking techniques operate in the 2D space of the image. However, algorithms exist that attempt to solve the problem in 3D space (Giebel et al., 2004; Brau et al., 2011). These 3D algorithms offer additional advantages over traditional 2D trackers, such as the ability to reason about the height of the object.

Human action recognition is an active and popular domain of research that warrants its own comprehensive surveys (Poppe, 2010; Weinland et al., 2011). Of notable mentions are two current state-of-the-art action recognition methods capable of achieving near perfect performance on the commonly used KTH Actions dataset[2] (Schuldt et al., 2004), as well as high performance rates on other more challenging datasets. The first method is inspired by the *object bank* approach to image representation. Sadanand and Corso (2012) presented an *action bank* representation, which comprises the collected output of many pre-trained template-based action detectors into a high-level representation of actions. O'Hara and Draper (2012) represented *tracklets* of actions as points on Grassmann manifolds, where each track is modeled as a 3-dimensional data cube with axes of width, height, and frame number. The authors used this representation to construct a scalable action recognizer using a subspace forest. It is worth mentioning that this subspace forest approach is capable of running at video frame rate (O'Hara and Draper, 2012).

Many researchers have also attempted to classify the surrounding scene in images, both in 2D (Fei-Fei and Perona, 2005; Lazebnik et al., 2006; Quattoni and Torralba, 2009; Li et al., 2010; Xiao et al., 2010) and 3D space (Herman and Kanade, 1986; Han and Kanade, 2001; Slabaugh and Culbertson, 2001).

[2] `http://www.nada.kth.se/cvap/actions/`

2.3 Natural Language Processing

The field of natural language processing (NLP) offers researchers many tools and techniques for documents parsing, such as part of speech tagging (Collins, 1997; Brants, 2000), syntactic dependency resolution (Marneffe and Manning, 2013), named entity recognition (Zhou and Su, 2002), coreference resolution (Lee et al., 2013), semantic role labeling (Surdeanu et al., 2007), and many more. To extract mental state information from texts, one might use any or all of these technologies, so a complete review of relevant technologies is impossible, here.

We can, however, highlight a few selected pieces of research in information extraction and retrieval that are of immediate relevance, such as the work of Marneffe et al. (2010), which attempted to identify the latent meaning behind scalar adjectives (e.g., which ages people have in mind when talking about "little kids"). The authors learned these meanings by extracting scalars, such as children's ages, that were commonly collocated with phrases, such as "little kids", in web documents. Mohtarami et al. (2011) tried to infer yes/no answers from indirect yes/no question-answer pairs (IQAPs) by predicting the uncertainty of sentiment adjectives in indirect answers. Their method employs antonyms, synonyms, word sense disambiguation as well as the semantic association between the sentiment adjectives that appear in the IQAP to assign a degree of certainty to each answer. Sokolova and Lapalme (2011) further showed how to learn a model for predicting the opinions of users based on their written contents, such as reviews and product descriptions, on the Web. Reschke et al. (2011) presented a framework for generating and ranking fine-grained, highly relevant questions from user-generated reviews, which can then be used to generate recommendation dialogs. On a different thread, Gabbard et al. (2011) found that coreference resolution can significantly improve the recall rate of relations extraction without much expense to the precision rate.

With regards to unsupervised information extraction methods, Chambers et al. showed that *narrative schemas*, with associated participants, and *information templates* can both be automatically extracted from text via an unsupervised infor-

mation extraction methodology (Chambers and Jurafsky, 2009, 2011). Figure 2.1 shows an example narrative schema from (Chambers and Jurafsky, 2009). Riloff and Jones (1999) presented a bootstrapping method for simultaneously learning a dictionary of semantic lexicon and the corresponding information extraction patterns.

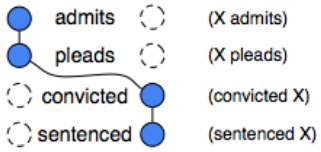


Figure 2.1: An example narrative schema from (Chambers and Jurafsky, 2009).

Our work builds on these efforts by combining information retrieval, lexical semantics, and event extraction to extract latent contextual information from video scenes.

In addition to information extraction, our research also involves the investigation for a good performance measure to our problem. In particular, we draw similarity between our evaluation task to that of coreference resolution and word sense disambiguation. In coreference resolution, an evaluation metric must properly account for the similarity between the chains of entities, as well the similarity between the entities themselves. Luo (2005) proposed a novel metric, called Constrained Entity-Aligned F-Measure (CEAF), that tries to find an optimal one-to-one mapping between subsets of reference and ground-truth entities before computing the F-measure. Cai and Strube (2010) made some modifications to CEAF to account for common edge cases involving *twinless* mentions (a.k.a mentions that exist in either the reference or ground-truth set, but not both). Research in word sense disambiguation have also yielded many interesting results. These findings are very pertinent to our work since our evaluation model must also account for the semantic similarity between mental state labels. In particular, Pedersen et al. (2004) have presented and implemented many novel word similarity and relatedness measures based on the WordNet dictionary (Miller, 1995). Lastly, the Earth Mover's Distance (EMD) presented by Rubner et al. (2000) for image retrieval is especially relevant

to our work as it naturally accounts for correspondences between the weights of elements in different bins (i.e., it accounts for similarity of distribution shapes as well as similarity at the level of distribution elements).

CHAPTER 3

VIDEO & TEXT CORPORA

The research proposed in this dissertation requires the use of both video and text data. Text corpora act as the knowledge source for our information extraction models, and videos are needed to train and test these models.

For our experiments, we focus on two types of videos: videos containing chase scenes and videos containing hug scenes. Chases often invoke clear mental state inferences, and depending on context can suggest very different mental state distributions for the actors. Similarly, hugs can also invoke strong mental state inferences in some cases, but the differences in mental state distributions between different hug scenarios are much more subtle (e.g., a hug between a parent and his/her child versus a hug between two siblings).

3.1 Video Datasets

We compiled two different video datasets using videos found on the Web. The first set contains 26 different chase videos. Of these, five involve police officers, seven involve children, four show sport-related scenes, and twelve describe different chase scenarios involving civilian adults (two videos involve both children and sport scenes).

The second set contains 45 different hug videos. All videos involve some combinations of civilian adults and children. There are no police officers present in these videos. In particular, there are eight videos depicting total strangers hugging, ten show acquaintances hugging, eight contain friends, six portray lover embraces, seven involve a parent and his/her child, and the remaining six show siblings hugging.

Table 3.1 shows some statistics for the length of the videos in our datasets. Indeed chase videos are shorter than hug videos on average, which is as expected

Dataset	Total Videos	Mean Length	Median Length	Range
chase	26	8.8	8	$[4, 18]$
hug	45	16.3	16	$[6, 25]$

Table 3.1: Summary statistics for the length of videos (in seconds) from our datasets.

since chases often happen quickly on camera. This makes good chase videos much harder to find. Figure 3.1 shows screenshots of some videos from our datasets. All videos are available for download[1] and online viewing[2,3].

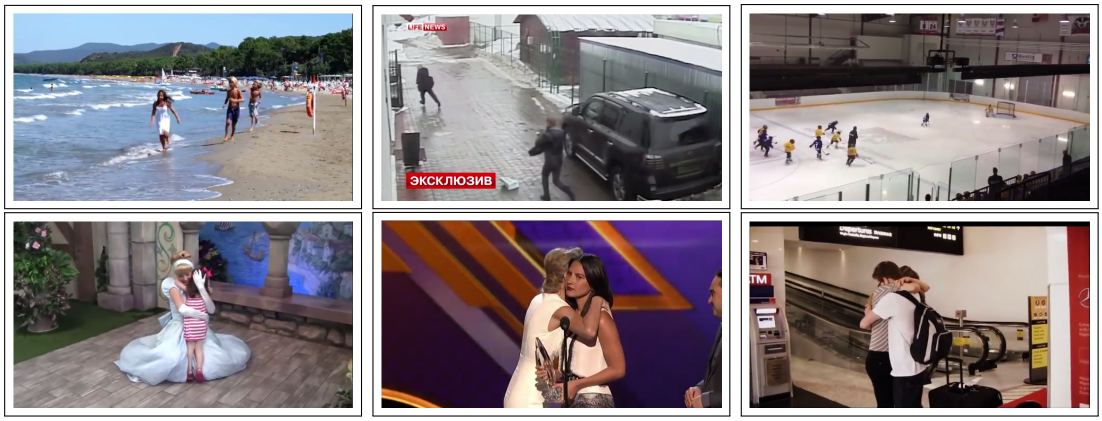


Figure 3.1: Screenshots of chase (top row) and hug (bottom row) videos from our datasets.

3.2 Video Annotations

For each video, we used Amazon Mechanical Turk (MTurk) to generate annotations about both the visual context and the mental states of the actors in the video. The annotated context are used as representative output of an ideal computer vision detection system that operates at human-level performance and are passed as input into our system. The collected mental state labels, on the other hand, are composed into gold standard data to be compared against the system's output. Figure 3.2

[1]All videos and annotations are available for download at: `http://trananh.github.io/vlsa`
[2]chase: `http://www.youtube.com/playlist?list=PLnmTyw7KoDXAX7r6cs53uM3E80FjE-1ZF`
[3]hug: `http://www.youtube.com/playlist?list=PLnmTyw7KoDXAxcjiLOih15DkpU-tXmSXa`

illustrates the contributions of the MTurk workers in the context of our high-level design diagram from Section 1.3.

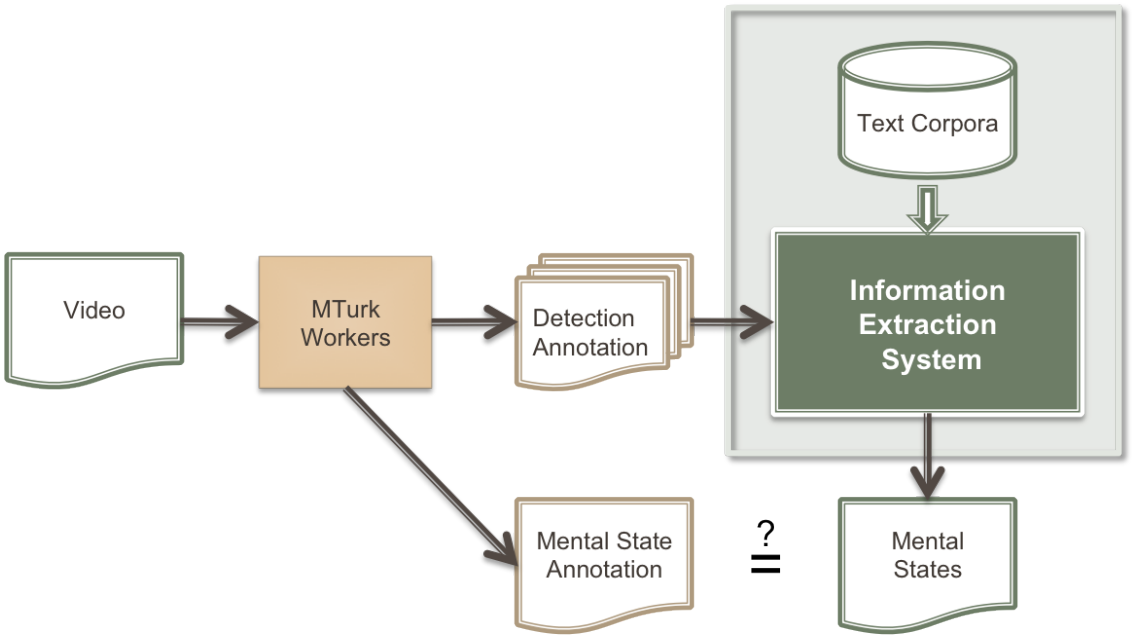


Figure 3.2: High-level schematic of the proposed approach showing the use of data collected from Amazon Mechanical Turk workers.

For each annotation experiment, we asked the workers to view a video in its entirety before answering some questions about the video scene. We give no prior training to the workers. However, we do require each worker to have had at least 50 approved HITs[4] with a 90% approval rating. For consistency and quality control purposes, we additionally require all workers to be from the United States. The questions we asked were carefully phrased to apply to all participants of a particular role, for example all actors (if there are more than one) being chased. We also ask obvious validation questions about the participants in each role (e.g., are the people being chased running towards the camera?) and use the answers to these questions to filter out poor responses. In general, we found that most responses were good and we only needed to reject a few incomplete HITs.

[4]A HIT (Human Intelligence Task) is a task to be completed by a MTurk worker.

Detection Annotation

In the first experiment, we asked MTurk workers to select, from a predefined list of tags, various detections that apply to the video. This labeling task is a proxy for a computer vision detection system that functions at a human level of performance. Indeed, we restricted the types of labels available for selection to a set that can be reasonably expected from automatic detection algorithms. The list of detections range from different types of actions (e.g., *walking*, *running*, *jumping*, etc.) to different types of actors (e.g., *person*, *policeman*, *child*, and (non-human) *object*) and location settings (e.g., *indoor*, *street*, *field*, etc.). The number of tags we used are summarized in Table 3.2.

Tag Type	**No. of Tags**
Location Settings	24
Objects in Scene	16
Actions	12−15
Actor Types	4

Table 3.2: The different types of tags (representing detections) used for the MTurk detection annotation experiment and the size of each set.

Although we asked for a large number of detections in the annotation, we found in our experiments that our models only needed to depend on a small number of detectors to perform reasonably well. In fact, we primarily only required detections about the actor types in our models. These detections can be reasonably expected from automatic detection algorithms. For example, police officers often wear distinctive color uniforms that can be learned using the Felzenszwalb detector (Felzenszwalb et al., 2008), whereas children can be reliably differentiated by their heights under a 3D-tracking model (Brau et al., 2013). Screenshots of an example HIT for this experiment is shown in Figure 3.3.

Each video in our datasets was annotated by three different workers and the union of their annotations was produced. The overall accuracy of the annotation was excellent. The MTurk workers correctly identified the important actor(s) in

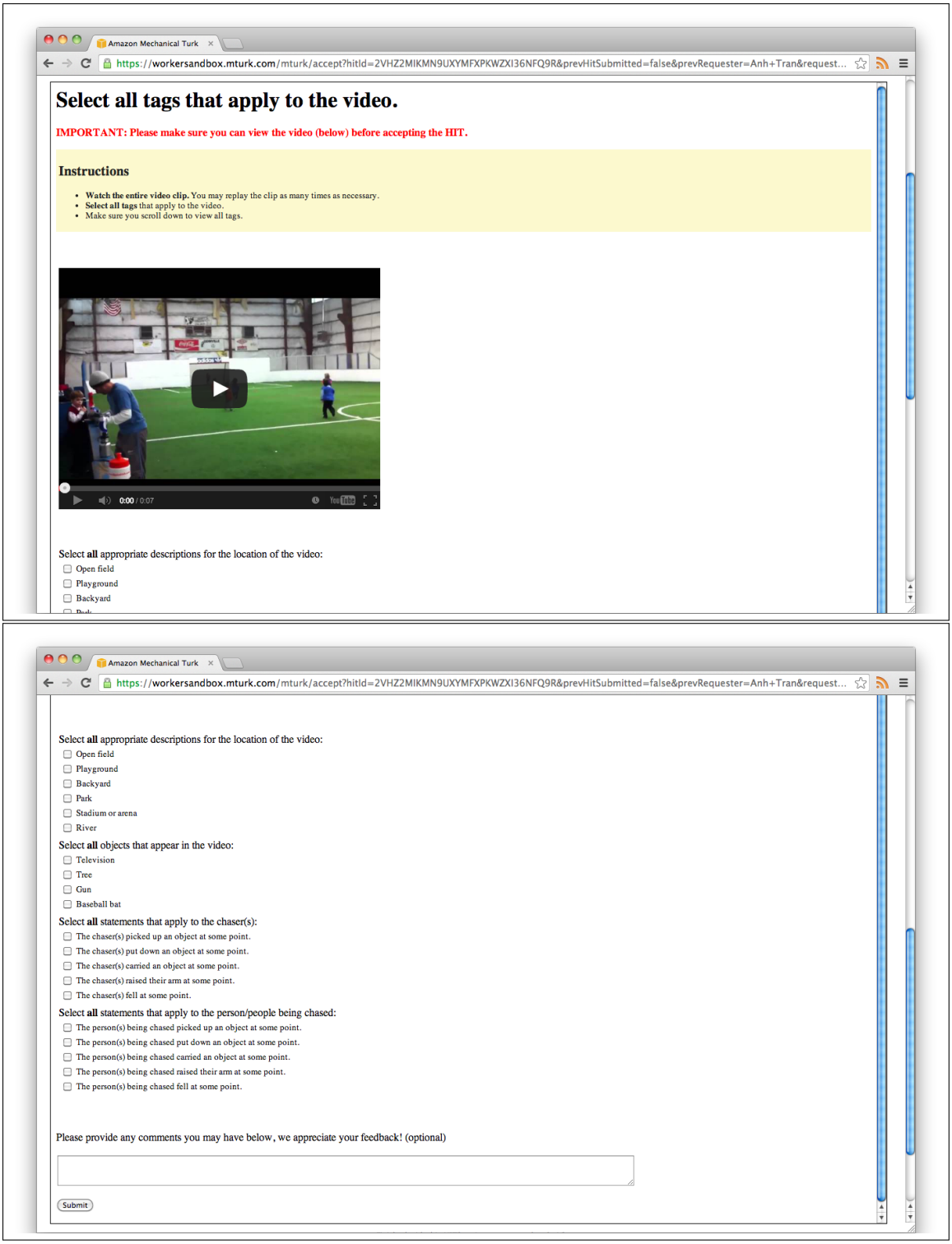


Figure 3.3: Screenshots of a detection annotation HIT on MTurk.

every video.

Mental State Annotation

Next, we collected a gold standard list of mental state labels for each video by asking MTurk workers to identify *all* applicable mental state adjectives for the participants involved (see Figure 3.4). We chose to use a text-box to allow for free-form input. Studies have shown that people of different cultures can perceive emotions very differently, and having forced choice options cannot always capture their true perception (Gendron et al., 2014). Thus, we did not restrict the response of the workers in any way and allowed workers to abstain from answering if they felt the video was too ambiguous. Each video was evaluated by ten different workers.

Some post-processing steps were necessary to clean up the annotations. In particular, we needed to spellcheck and convert each term provided to the closest adjective form if possible. Terms with no equivalent adjective forms were left in place. On rare occasions, workers provided sentence descriptions despite being asked for single-word adjectives. These sentences were either removed, or collapsed into a single word if appropriate.

Dataset	Vocab Size	Total Labels	Mean Frequency	Variance
chase	167	1109	6.6	247.9
hug	344	2045	5.9	487.7
chase + hug	441	3154	7.2	680.7

Table 3.3: Summary of the results retrieved from the mental state MTurk annotation experiment. The mean frequency refers to the average number of times a mental state label was used in the annotation.

The summary statistics for the resulting mental state labels are shown in Table 3.3. While the vocabularies used to describe videos from each dataset do share common labels, it is clear that they also differ. For example, there were 97 different mental state terms used to describe participants of chase videos that were not used to describe any hug scenarios. The large discrepancies between the vocabulary sizes

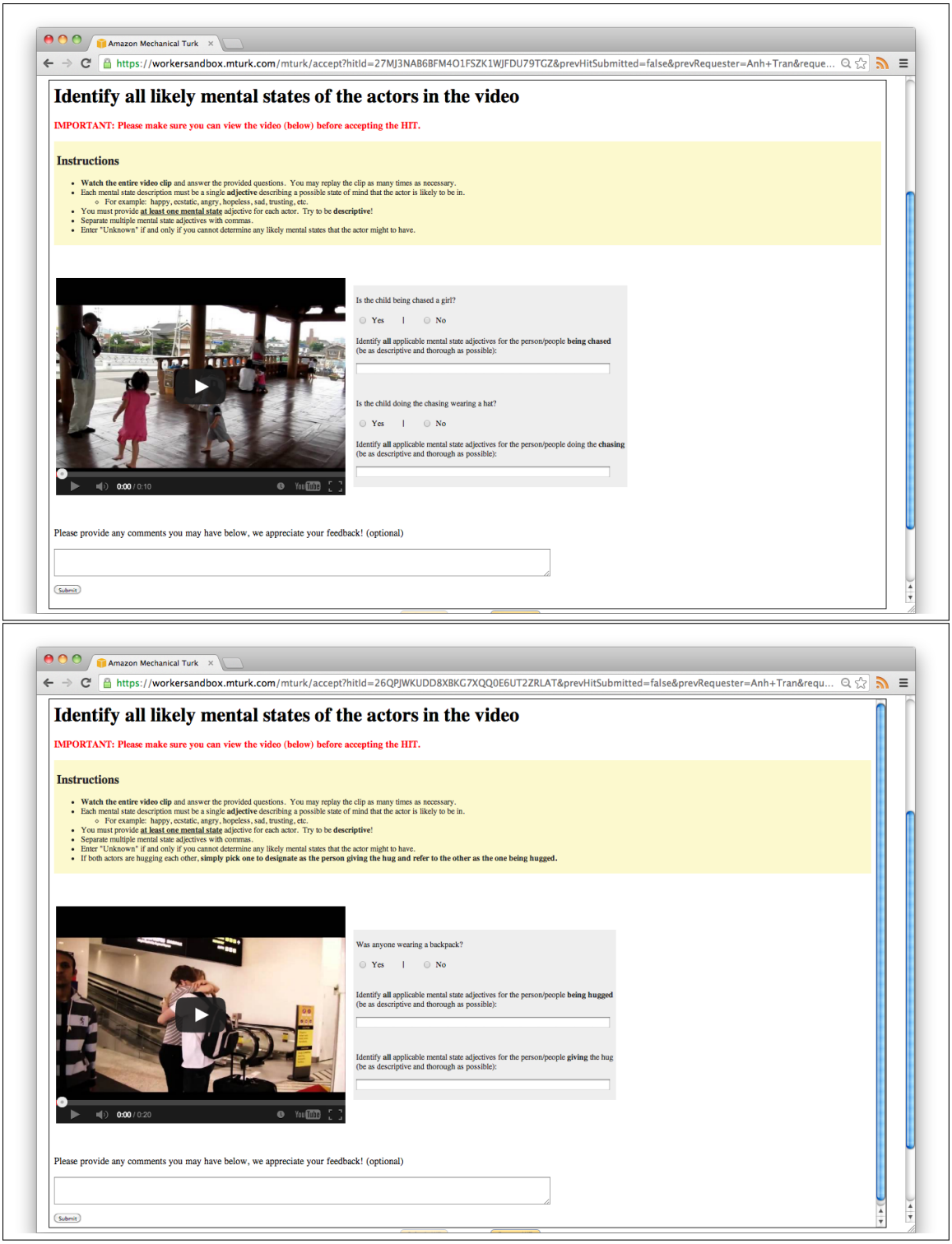


Figure 3.4: Screenshots of two different mental state annotation HITs on MTurk.

and the total number of labels used between the two datasets can mostly be explained by the fact that the *hug* dataset contains many more videos than the *chase* dataset. The rather large population variance reflects the fact that a small portion of terms from the vocabulary were used very frequently while a large number of labels were used only once or twice. Table 3.4 shows the top 5 most frequently used mental state labels to describe each dataset.

Dataset	**Top 5 Most Frequently Used Labels**
chase	(excited, 109), (angry, 108), (happy, 74), (fearful, 61), (scared, 60)
hug	(happy, 341), (loved, 135), (excited, 131), (surprised, 76), (joyful, 74)
chase + hug	(happy, 415), (excited, 240), (loved, 135), (angry,119), (joyful, 91)

Table 3.4: The top 5 mental state labels most frequently used by MTurk workers for each dataset.

The overall quality of the annotations was good and generally followed common intuition. In addition to the expected common labels, we received some very colorful (yet informative) descriptions, like *incredulous* and *vindictive*. In the *chase* dataset, chases involving police scenarios often contained fearful and angry states while chases involving children received more cheerful descriptions. We also received unexpected descriptions, such as *annoy* for a playful chase between two children. Upon review of the video, we agreed that one child did indeed look annoyed in parts of the video. Thus, the resulting descriptions were subjective, but very few were hard to rationalize. By aggregating the answers from the workers, we generated a frequency distribution of mental state terms for each video.

3.3 Text Corpora

The primary text corpus used by our models is the English Gigaword 5th Edition corpus[5], made available by the Linguistics Data Consortium (LDC) and indexed by Lucene[6]. It is a comprehensive archive of newswire text data that has been acquired

[5]Linguistics Data Consortium catalog number LDC2011T07
[6]Apache Lucene: `http://lucene.apache.org`

over several years, totaling approximately 26 GB in size. The Gigaword corpus is ideal for our work because it is large enough to support information extraction, yet small enough to allow for fast development of prototype models. It is in this corpus that we expect to find latent attributes cued by known contextual information.

Before settling on the English Gigaword corpus, however, we also explored other possibilities. The Google Web N-gram Corpus[7], also made available by the LDC, offers English word n-grams (up to 5-grams) and their observed frequency counts. The counts were generated from approximately 1 trillion word tokens of text from publicly accessible Web pages. The (compressed) size of the corpus is approximately 24 GB. Unfortunately for our models, 5-grams offer too little in context to be useful.

We also experimented with the idea of generating a new text corpus by collecting streaming data from Twitter[8] – an online social networking and micro-blogging service that enables users to send and read short 140-character text messages, called "Tweets". We collected all Tweets pertaining to specific subjects of interest, such as *chase* and *hug*. In one month, we collected just over 200 MB worth of Tweets containing the term *chase*. Upon inspection, we found that a large percentage of the messages are repeats of each other (also known as re-Tweets). Most Tweets are irrelevant and do not describe any actual *chase* scenarios (e.g., "Why do I chase the ones who don't want me? #mylife" and "I don't chase nobody.."). Moreover, the style of language used in Tweets often does not conform to standard language models. They typically contain elaborate uses of smiley faces, abbreviations, slangs, and other sequences of characters that typically depict specific meanings. For these reasons, we decided against using Twitter as a data source for our work.

The last corpus we investigated was the World Wide Web. The Web is one of the largest machine-accessible sources of information currently available, which makes it highly desirable for our work. Unfortunately, we currently do not have the means or resources to efficiently query the entire web. The best we can do at the moment is to use free web search API's provided by popular online search engines in order

[7]Linguistics Data Consortium catalog number LDC2006T13

[8]`http://www.twitter.com`

to query the web. This comes with significant disadvantages. First, most search engines limit the number queries we can make, as well as the number of documents returned per query. Secondly, we have no control over the ranking mechanism used by the underlying search engine; and hence, cannot directly influence the kinds of results we want. At the moment, most popular commercial search engines such as Google[9], Bing[10], and Yahoo![11] limit the amount of free queries we can do to a few hundreds documents. This pales in comparison to the number of documents we actually need. There are alternative search engines, such as Faroo[12], that offer much higher limits. However, these non-commercial engines do not provide nearly enough coverage of the entire web as the other options. An alternative to online search is to use an offline version of the Web. The ClueWeb09[13] dataset consists of 1 billion web pages (in ten different languages) that were collected in January and February 2009. Indexing 1 billion web pages is a daunting task and we currently do not have the necessary resources to do so in a timely manner. Moreover, a large dataset, such as ClueWeb09, can hinder development progress. As such, we elected not to use ClueWeb09 as our development corpus. However, this corpus remains as an enticing option for future work.

[9] `https://www.google.com/`
[10] `https://www.bing.com/`
[11] `https://www.yahoo.com/`
[12] `http://www.faroo.com/`
[13] `http://lemurproject.org/clueweb09.php/`

CHAPTER 4

LATENT ATTRIBUTE EXTRACTION

This chapter describes the main contribution of our work: a fully developed system for identifying latent attributes in a scene given some known contextual information about the scene. We apply the system to the task of identifying mental state labels of activity participants in video scenes. The known context is assumed to come from the output of computer vision detectors, but can, in fact, come from any other sources of evidence. Our research investigates several largely unsupervised information extraction (IE) models for this task.

We first give a high-level overview of the system. This includes explicit definitions of the system's input and output. Then, we present several robust models for extraction: a lexical semantic model that extracts mental state terms that are highly similar to the context of the scene in a latent, conceptual space; an information retrieval model that identifies mental state labels that commonly appear in sentences related to the known scene context; and an event-centric model that looks for mental state terms associated with participants of the relevant event (identified using syntactic patterns and coreference resolution). We also discuss how these models can be combined to produce more robust ensemble models that can outperform their respective individual component alone.

4.1 System Overview

The primary goal of our system is to identify the latent attributes of a scene given that we already know some contextual information about the scene. This naturally defines the input and output to our system: the input being the known context, and the expected output is a list of mental state labels describing the scene. Figure 4.1 shows the overall system diagram. Details of the diagram are described in the

following subsections.

4.1.1 Input: Detections / Query Tuples

The known context comes into our system as a list of detections, or tags, extracted from the scene. The context can be detections about the activities in the scene, the types of participants involved, the location of the scene, and possibly others. These detections are then parsed into tuples of detections by their types – we refer to these as *query tuples*. Query tuples follow specific typing patterns, such as an (*activity*, *actor-type*) pattern or an (*activity*, *actor-type*, *location*) pattern. Larger tuples typically contain more context information. For our experiments, we primarily used the (*activity*, *actor-type*) pattern, also referred to as the "AA" query pattern. The AA query pattern requires only a few number of detectors, which lessens the dependency of the model's performance on the amount of known context. For the rest of this chapter, our discussion focuses mainly on input query tuples following the AA pattern. However, it is worth noting that our models are compatible with any other query patterns.

An AA query tuple is simply a two-element list that contains an "activity" detection and an "actor-type" detection (e.g., (*chase*, *policeman*) and (*hug*, *person*)). If multiple actor-types are detected, then multiple tuples will be generated for the scene. In our experiments, which used the detections annotated by MTurk workers (see Section 3.2), the activity participants could be either a person, a policeman, a child, or a (non-human) object. If the list of input detections describes a participant as both a person and a child, or a person and a policeman, we automatically remove the *person* tag as it is a WordNet (Miller, 1995) hypernym of both *child* and *policeman*. For each human type, we further expand the pool of actor-type tags by retrieving the synonym set (synset) of its most frequent sense (i.e., sense #1) from WordNet. We ignore multi-words synonyms. For example, a chase involving a policeman would generate the following query tuples: (*chase*, *policeman*) and (*chase*, *officer*).

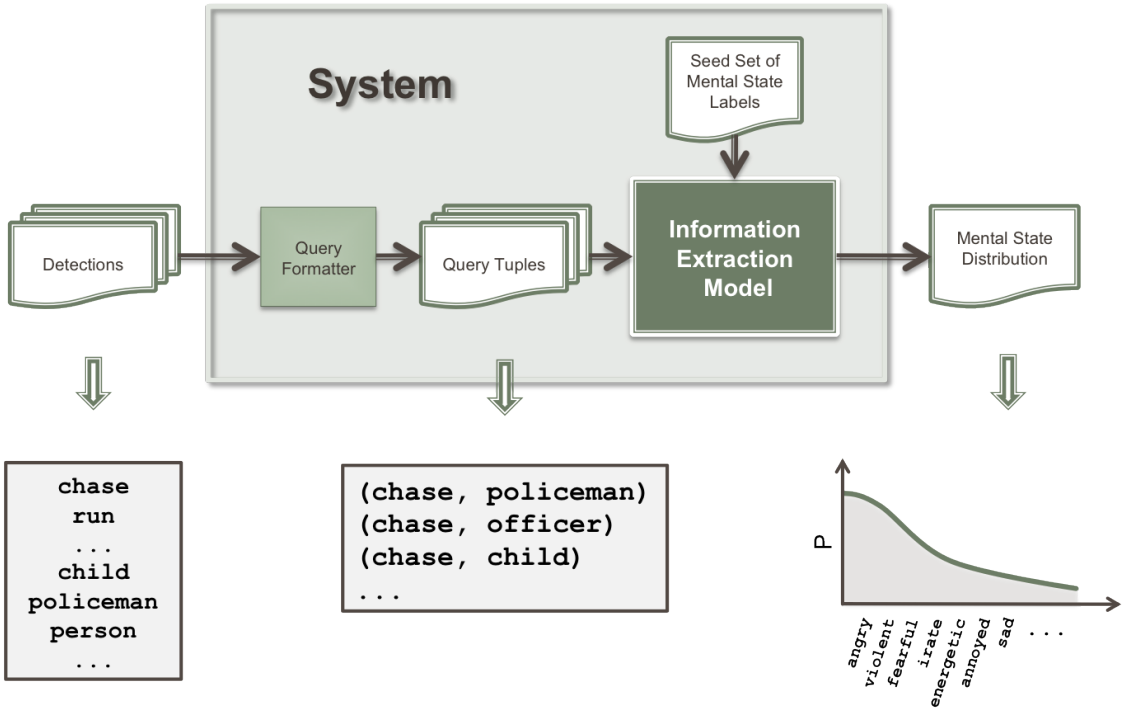


Figure 4.1: The overall system design. The input to the system is a list of detections, or tags, about the scene and the output is a distribution over mental state labels. Detections are combined and formulated into query tuples using specific query patterns, such as the (*activity*, *actor-type*) pattern shown in the diagram. Query tuples, along with an initial seed set of 160 mental state labels are given to the information extraction model to produce the output response distribution. Information about the information extraction models are described in Section 4.2.

4.1.2 Input: Mental State Seed Set

In addition to query tuples, our information extraction models also require an initial seed set of mental state adjectives as input (see Figure 4.1). The seed set used in our experiments was compiled from popular mental and emotional state dictionaries, including the Profile of Mood States (POMS) (McNair et al., 1971) and Plutchik's wheel of emotions (Figure 4.2). We also included frequently used labels gathered from synsets found in WordNet.

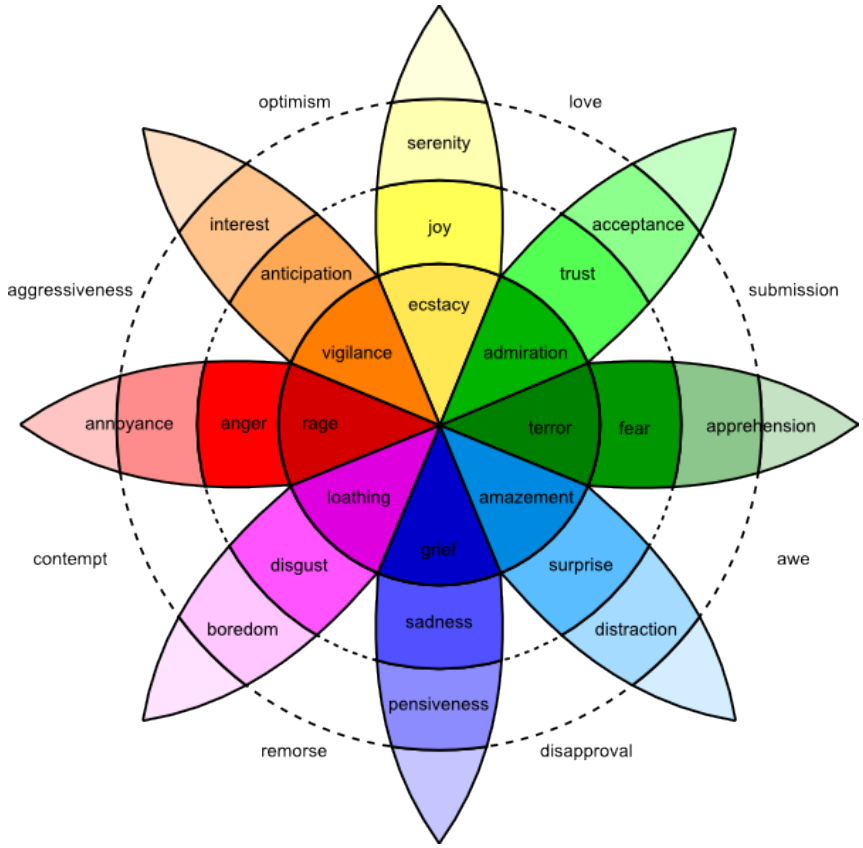


Figure 4.2: Plutchik's wheel of emotions.[1]

Table 4.1 shows a small sample from the set. The full set of all 160 mental state labels can be found in Appendix A. Note that the gold standard set of labels used in the evaluation of the experiments (produced by MTurk workers as described in Section 3.2) was *not* a source for this seed set.

[1]Image source: `http://commons.wikimedia.org/wiki/File:Plutchik-wheel.svg`

Source	Example Mental State Labels
POMS	alert, annoyed, energetic, exhausted, helpful, sad, terrified, unworthy, weary, etc.
Plutchik	angry, disgusted, fearful, joyful/joyous, sad, surprised, trusting, etc.
Others	agitated, competitive, cynical, disappointed, excited, giddy, happy, inebriated, violent, etc.

Table 4.1: The initial seed set contains 160 mental state labels, compiled from different sources like the popular Profile of Mood States dictionary and Plutchik's wheel of emotions.

In general, this initial seed set of mental state labels tells our information extraction models what adjective terms to search for during the extraction phrase, and ultimately, what labels to build a distribution over.

4.1.3 Output: Mental State Distribution

Given query tuples, our information extraction models use the initial seed set of mental state adjectives to output a single distribution over mental state labels, referred to as the *response distribution*, for each scene. The response distribution contains labels from the seed set that best describe the mental states of activity participants in the scene, along with an associated weight for each label.

4.2 Neighborhood Extraction Models

We now present several largely unsupervised models for extracting mental state labels from large collections of text documents given some contextual cues. The models are designed based on the *neighborhood* paradigm, that is the hypothesis that relevant mental state terms will appear "near" the neighborhood of the search context. In this case, the search context contains various contextual information that are known about the scene, such as detections about the activity in the scene and its participants. In our approach, it is necessary to know something about the activity, or at the very least knowing that the activity has occurred, in order to retrieve meaningful descriptions relevant to the activity. With no a priori contextual

information, we can only retrieve mental state descriptions that generally apply to all scenarios, which in practice is not very informative, or interesting.

The search context is passed to our models as *query tuples* following specific query patterns, such as the (*activity*, *actor-type*) pattern, which produces tuples like (*chase*, *policeman*). The models also require an additional seed set of mental state adjectives to guide the extraction process. The output of each model is a single response distribution over mental state labels (refer to Figure 4.1 for clarifications). Below, we describe in detail three different information extraction models for extracting relevant mental state labels from text corpora.

4.2.1 Vector Space with Back-off Linear Interpolation

The first model aims to extract relevant mental state labels by finding terms from the initial seed set that are highly similar to the search context in a latent, conceptual space generated by a recurrent neural network language model (RNNLM) (Mikolov et al., 2013a). We refer to this lexical semantic (LS) model as the Vector Space (or *vector*) model.

Vector Space Representation

The basic strategy of the *vector* model is to train the RNNLM of Mikolov et al. (2013a) on our corpus of text documents, and use it to project both mental state labels and the search context into a latent conceptual space. For all of our experiments, we used a RNNLM computed over the English Gigaword (5th Ed) corpus, using 600-dimensional vectors. The resulting RNNLM has a vocabulary size of $813,908$ words. The training corpus contains approximately 3.89B tokens.

The trained RNNLM allows us to map English words, such as mental state labels, to their equivalent vector representations. For instance, we could map the term *mad* in Figure 4.3 to the vector $\vec{v}_{\text{mad}}$, where $\vec{v}_{\text{mad}}$ is a 600-dimensional vector that belongs to some latent vector space generated by the RNNLM. Similarly we can map any other words to their corresponding vector representations.

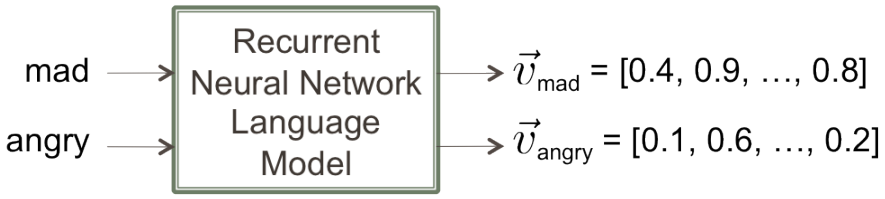


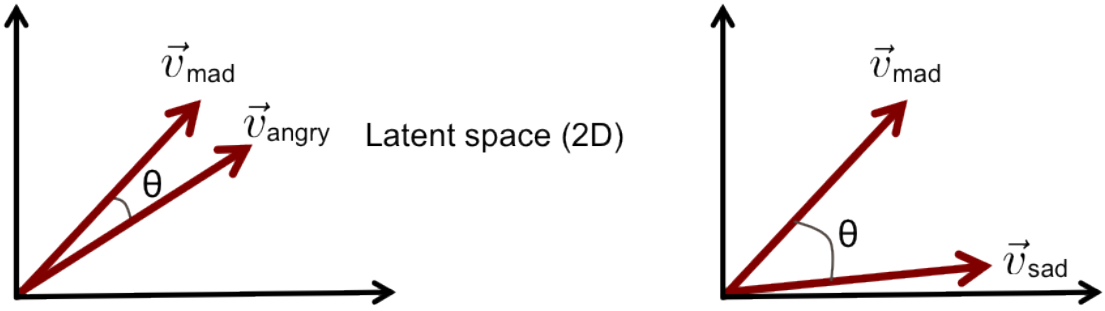
Figure 4.3: Words are mapped to their corresponding 600-dimensional vector representations using a RNNLM trained on the English Gigaword (5th Ed) corpus.

This mapping allows us to estimate similarities of words by comparing their vector representations in the latent space. The hypothesis is that in this latent space, similar words will project to similar vectors and dissimilar words will project to dissimilar vectors, where similarity can be measured as the angle between two vectors. For example, consider a simple 2D space where words are mapped to 2-dimensional vectors, as illustrated by Figure 4.4. In this space, we would expect the angle between the vector representations of *mad* ($\vec{v}_{\text{mad}}$) and *angry* ($\vec{v}_{\text{angry}}$) to be small, while the angle between $\vec{v}_{\text{mad}}$ and $\vec{v}_{\text{sad}}$ should be much bigger (see Figure 4.4a).

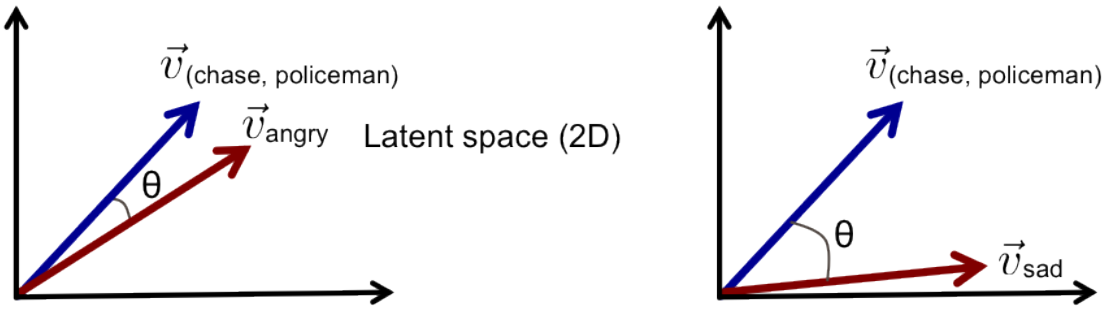
Furthermore, suppose that given a tuple of contextual information, such as (*chase*, *policeman*), we could also map it to a corresponding vector space representation (see Figure 4.4b). Let's call this vector space representation of the tuple a *context vector*. Then clearly, the vector representations of some mental state labels would map closer to this context vector than others, as shown in Figure 4.4b. This gives us a natural way to rank the relevancy of mental state labels based on their similarity score to the context vector. Thus, we need a way to project a tuple of contextual information into a corresponding context vector. In the next few subsections, we discuss how our model constructs these tuples of context, maps them into vector space, and builds a mental state distribution using similarity scores.

From Query Tuples to Back-off Levels of Context Tuples

Before we can map tuples of contextual information into context vectors, we first need to define these tuples. While we can simply use the input query tuples directly, it is not very robust as the number of query tuples may be small. Furthermore, a



(a) Example vector representations of mental state labels.



(b) Example vector representations of mental state labels and context tuples.

Figure 4.4: Mental state labels and context tuples are mapped to example vector representations in a simple 2D latent space. In this space, similar mental state labels should hypothetically project to similar vectors, resulting in a smaller angle between the two vectors, as shown in (a). Likewise, we can also map each context tuple to a corresponding vector representation to find its similarity to each projected mental state label, as in (b).

large query tuple may lead to unreliable results due to the sparse data problem. As with common practices, we instead take a back-off approach by separating query tuples into multiple levels of back-off *context tuples*. In other words, we separate each query tuple into several different tuples of context, each containing an increasing amount of contextual information.

For instance, the first level of context uses only the first term from each query tuple, while the second level uses the first two terms from each query tuple, and so on. Figure 4.5 shows how context tuples are constructed from query tuples. For

query tuples following the AA pattern, only two back-off levels of context apply. The first level includes the set of activities as singleton context tuples, e.g., (*chase*) and/or (*hug*), while the second level contains all (*activity*, *actor-type*) tuples as context tuples. Note that multiple query tuples may generate the same context tuple.

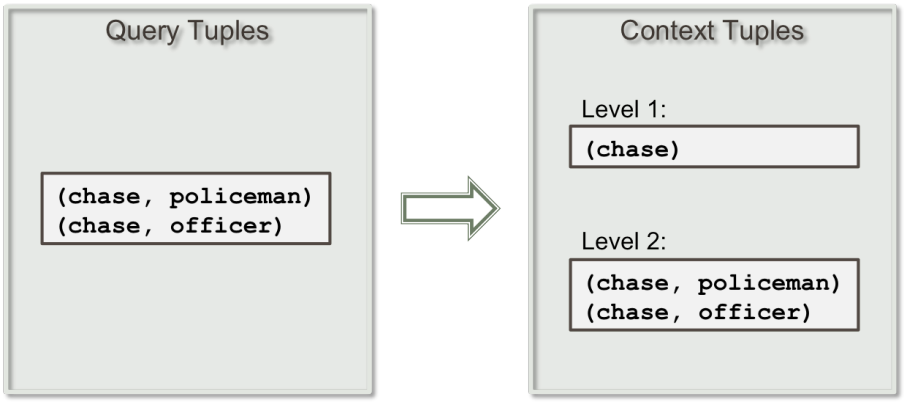


Figure 4.5: Query tuples given to the *vector* model are separated into context tuples of different back-off levels. Each query tuple following the AA pattern generates two context tuples. In this case, the two input query tuples generated the same context tuple at the first level, (*chase*); and each generated a different context tuple at the second level.

From Context Tuples to Mental State Distributions

For each context tuple with multiple terms, such as (*chase*, *policeman*), we find the vector representation for the context by simply combining the vectors representing the search terms. In our experiments, we explored several different ways of combining these vectors. We are specifically interested in combination operators that are both associative and communicative – the goal is for both (*chase*, *policeman*) and (*policeman*, *chase*) to map to the same representative vector. Of the common operators, only addition and multiplication satisfy these desirable properties. In this case, our experiments (see Section 7.2.6) showed that the addition operator works best. This is consistent with findings presented by Mikolov et al. (2013a), where relationships between words were found by manipulating the offsets of vectors (via

addition and subtraction). Thus, the vector representation of a context tuple, also known as the *context vector*, can be found by aggregating the vectors representing the terms from the tuple:

$$\vec{v}_{(chase,\ policeman)} = \vec{v}_{chase} + \vec{v}_{policeman} \ .$$

The context vector for a singleton context tuple is just the vector of the single search term:

$$\vec{v}_{(chase)} = \vec{v}_{chase} \ .$$

After normalizing the context vector, we calculate the distance of each mental state label m to the context tuple by computing the cosine similarity score between $\vec{v}_m$ and the context vector:

$$cos(\Theta_m) = \frac{\vec{v}_m \cdot \vec{v}_{(chase,\ policeman)}}{||\vec{v}_m||\ ||\vec{v}_{(chase,\ policeman)}||} \ .$$

As previously mentioned, the hypothesis here is that mental state labels that are related to the search context will have a RNNLM vector that is closer to the context vector, resulting in a high cosine similarity score. Because the number of latent dimensions is relatively small (when compared to vocabulary size), cosine similarity scores in this latent space tend to be close. To further separate these scores, we raise them to an exponential power:

$$score(m) = e^{cos(\Theta_m)+1} - 1 \ .$$

Since there are 160 different mental state labels in the initial seed set, the processing of each context tuple results in 160 different scores. We build a distribution over mental state labels based on these similarity scores by combining the scores into a single list and normalizing it. This process produces a single distribution for each context tuple.

Back-off Linear Interpolation of Context Levels

The processing of each context tuple yields a distribution over mental state labels. Given that the input query tuples can generate many different context tuples at multiple levels of context, we end up with many distributions over mental state labels for each level. We integrate these distributions into a single distribution for output as follows:

1. At each back-off level, we average the resulting distributions for all context tuples to generate a single representative back-off distribution for that level. In the example of Figure 4.5, the back-off distribution of the first level can be retrieved directly from processing the lone context tuple (*chase*). The back-off distribution for the second level is the average of the distributions retrieved from processing (*chase*, *policeman*) and (*chase, officer*).

2. Once we have back-off distributions at different levels, we combine them via linear interpolation using a similar method to the back-off strategy used by Collins (1997). Once again, we refer to the example in Figure 4.5. Let e_1 and e_2 represent the weights of some mental state label m from the back-off distribution at the first and second level, respectively. Then the interpolated distribution score e for m is:

$$e = \lambda e_1 + (1 - \lambda) e_2 \ .$$

3. Compiling all interpolated scores e from each mental state label m produces a final distribution that incorporates information from each back-off level.

The above algorithm gives us a mean to combine results at multiple back-off context levels into a single output for the *vector* model. Lastly, we prune this final mental state distribution by taking the top ranked mental state labels that make up some γ proportion of the distribution. The pruning parameter γ, as well as the interpolation parameters, are empirically calibrated from a few training examples. We discuss the tuning procedure in more details in Section 7.1.

The final output, i.e., the response distribution, for the *vector* model is the normalized version of this pruned distribution.

4.2.2 Sentence Co-occurrence with Deleted Interpolation

Unlike the previous *vector* model, which operates in a latent conceptual space, the Sentence Co-occurrence (*sentence*) model operates on text and takes a more intuitive approach to the problem: It extracts mental state labels based on the likelihood that they appear in sentences cued by terms from the query tuples. This approach is motivated by the observation that words in the same sentence are more likely to be related.

To help ground the explanation for the rest of this section, we assume that query tuples follow the AA query pattern (though this is not a requirement of the model).

Estimating Sentence Co-occurrence Likelihood

For each query tuple, we need to estimate the conditional probability that we will see a mental state label m in a sentence, where m is from the seed set, given that we already observed the desired activity and actor-type in the same sentence:

$$P(m|\textit{activity}, \textit{actor-type}) \ .$$

We further impose that all terms must appear as the correct part-of-speech (POS). That is, m must appear as an adjective or verb, the activity as a verb, and the actor-type as a noun. (Mental state labels are allowed to appear as verbs because some mental state adjectives are often mis-tagged as verbs; e.g., agitated, determined, welcoming.) Our implementation uses Stanford's CoreNLP[2] toolkit for tokenization and POS tagging.

Note that this probability is similar to a trigram probability in POS tagging, except the triples need not form an ordered sequence but must appear in the same sentence and under the correct POS tagging. In this case, we refer to the sentence

[2] `http://nlp.stanford.edu/software/corenlp.shtml`

length as the *neighborhood* window of the model. While it is possible to compute this trigram probability directly using relative frequency counts from corpus data, it is not robust as the frequency counts of high-ordered n-grams are often unreliable due to data sparsity. Thus, we instead estimate the desired probability as a linear interpolation of unigrams, bigrams, and trigrams.

We define the maximum likelihood probabilities $\hat{P}$, derived from relative frequencies f as follows:

$$\begin{aligned}
\text{Unigram:} \quad & \hat{P}(m) = \frac{f(m)}{N} \\
\text{Bigram:} \quad & \hat{P}(m|\textit{activity}) = \frac{f(m, \textit{activity})}{f(\textit{activity})} \\
\text{Trigram:} \quad & \hat{P}(m|\textit{activity}, \textit{actor-type}) = \frac{f(m, \textit{activity}, \textit{actor-type})}{f(\textit{activity}, \textit{actor-type})}
\end{aligned}$$

for all mental state labels m, activities, and actor types in our queries. N is the total number of tokens in the corpus. We define $\hat{P} = 0$ if the corresponding numerator and denominator are zero.

While computing the relative frequencies f in our corpus, the aforementioned POS requirement is enforced: $f(m)$ is the number of occurrences of m in the entire corpus as an adjective or verb. Generally, the relative frequency count of any unigram is simply the number of times that unigram appears with the correct POS tag in the corpus.

The relative joint frequency for n-grams where $n > 1$, however, is not as straightforward to compute and warrants some discussion. In particular, it is not clear how the count of f should be updated when multiple instances of a search term from the n-gram are present in the same sentence neighborhood. Suppose for example, that our corpus contains the single sentence: "John is happy chasing a happy child." Clearly, all occurrences of the terms "chase" and "happy" occur with the desirable POS. So how should $f(\textit{chase}, \textit{happy})$ be updated? If we decide to update f once for

each occurrence of the word "happy", then $f(chase, happy) = 2$. This is an intuitive counting scheme, as there are two unique pairings of the two terms. However, note that $f(chase) = 1$, which results in the following maximum likelihood estimate:

$$\hat{P}(happy|chase) = \frac{f(happy, chase)}{f(chase)} = \frac{2}{1} > 1 \ .$$

This is clearly wrong. For consistency, we found it is simplest to only increment the joint frequency of these n-grams at most once per neighborhood. Since the neighborhood is defined per sentence, it equates to counting the number of sentences in which all the terms from the n-gram (where $n > 1$) occur at least once in the correct POS. Furthermore, it is worth noting that a single document may contain several sentences that can match the search requirements. Hence under the *sentence* model, a document can contain several neighborhoods. We will contrast this with subsequent models later on.

Deleted Interpolation

As previously mentioned, the maximum likelihood probabilities $\hat{P}$ come in handy because we cannot always compute the trigram probabilities directly from the corpus. There are usually too few instances of each trigram to estimate a probability reliably. As is common in practice, we instead estimate the probability as a linear interpolation of unigrams, bigrams, and trigrams:

$$\begin{aligned} P(m|activity, actor\text{-}type) \ = \ & \lambda_1 \hat{P}(m) \ + \ \lambda_2 \hat{P}(m|activity) \ + \\ & \lambda_3 \hat{P}(m|activity, actor\text{-}type) \ . \end{aligned}$$

Since $\lambda_1 + \lambda_2 + \lambda_3 = 1$, P represents a probability distribution. We use the deleted interpolation algorithm (Brants, 2000) to estimate one set of lambda values for the model, based on trigrams generated from all query tuples from the training

set. This process calibrates a single set of interpolation parameters to be used for all video scenes in the dataset.

Formulating the Response Distribution

For each query tuple, our model generates 160 different trigrams, one for each mental state label m in the seed set, resulting in 160 different conditional probability scores. These scores estimate the conditional probability of seeing each mental state label in the same sentence as the terms from the query tuple (with POS restrictions).

We combine and normalize these scores into a single distribution – the normalized mental state distribution for that query tuple. However, the input into the model usually contains many query tuples; and thus, we need a way to combine several resulting distributions into a final output. We do this by simply taking an average across all resulting distributions.

As with the *vector* model, we prune this final average distribution by taking the top-ranked items that cover a large fraction, γ, of total probability. The pruned distribution over mental state labels is renormalized to yield the response distribution.

4.2.3 Event-centric with Deleted Interpolation

The *sentence* model has two limitations. On one hand, it is too sparse: the single sentence neighborhood window is too small to reliably estimate the frequencies of high-ordered n-grams for the probabilities of mental state labels. On the other hand, it may be too lenient, as it extracts all mental state mentions appearing in the same sentence with the activity, or event, under consideration, whether they apply to this event or not. We address these limitations next with an event-centric model (*event*), where an event simply refers to an activity.

Event-centric Mental State Extraction

Intuitively, the *event* model focuses only on the mental state terms associated with event participants. Formally, these mental state labels are extracted using the fol-

lowing steps (see Figure 4.6 for an overview example):

`Step 1: Identify Event Participants`

The first task is to identify the event and its participants. The identification of events is easy. We do this by searching the corpus for all occurrences of the event and retrieve all sentences containing that event as a verb (e.g., all sentences containing the verb *chase*). Next, we analyze the syntactic dependencies of these sentences to find the subject and object of the target verb. This requires some pattern matching on top of the syntactic relations to extract syntactic phrases referencing the desired participants.

In most cases, the nominal subject and the direct object of the verb are the targets of our extraction. For the verb *chase*, the nominal subject is often the chaser and the direct object is the person being chased. Similarly for *hug*, the nominal subject is typically the person giving the hug, while the direct object is the person being hugged. In addition, we also implemented additional patterns to model passive voice and other exceptions. An example output of this step can be seen in Figure 4.7.

All syntactic patterns used for the identification of participants can be found in Appendix B. We used Stanford's CoreNLP toolkit for syntactic dependency parsing and the downstream coreference resolution.

`Step 2: Identify Relevant Mentions in Other Sentences`

Once the phrases referring to the participants of interest have been pinpointed in the target sentence, we seek all mentions of these participants in the entire document by traversing the coreference chains containing the phrases extracted in the previous step.

It is possible for each participant to be mentioned in multiple coreference chains, and there may be multiple instances of the event in a document (each possibly having several participants). Chains may extend over the entire document and may overlap to share sentences. Therefore, we concatenate all unique sentences traversed in the chains to define the neighborhood area for this model. This defines a single

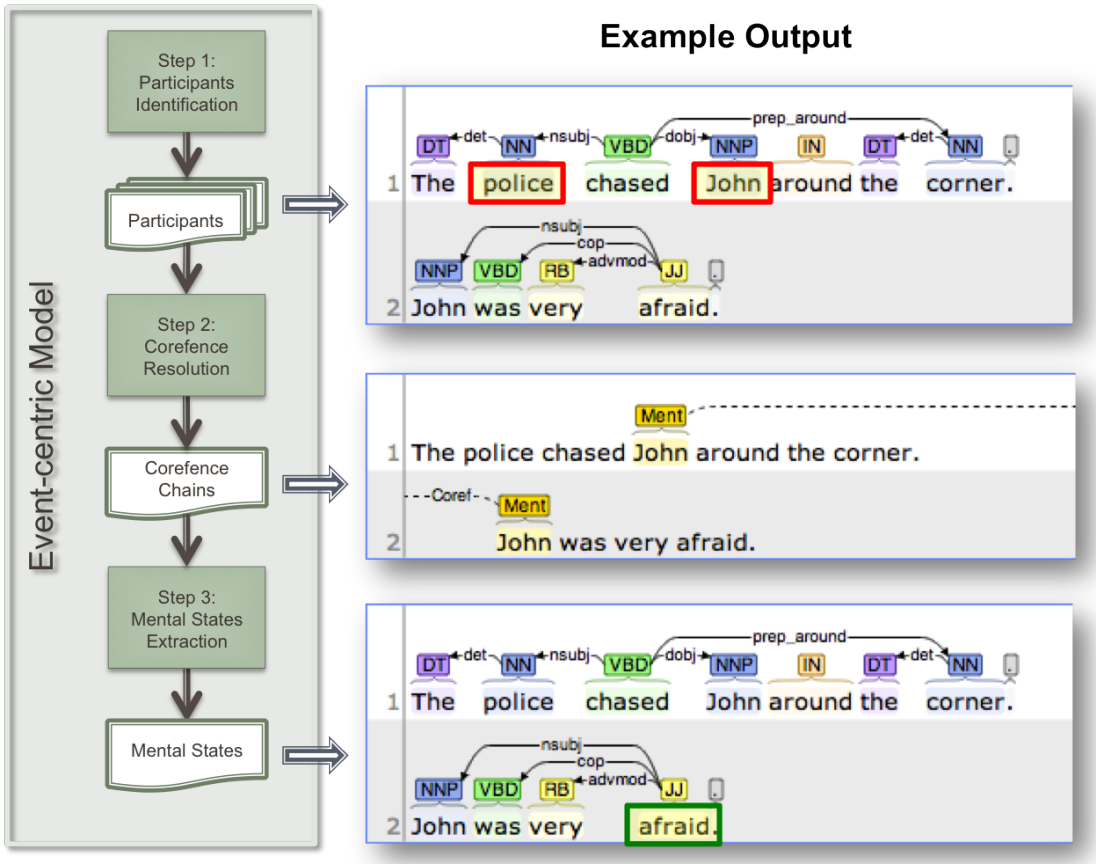


Figure 4.6: High-level design diagram of the *event* model along with illustrated hypothetical output. In the example, the first step in the algorithm begins by identifying that the first sentence contains a relevant event (*chase*) and proceeds to analyze the syntactic dependencies of this sentence to find the subject and object of the verb. Following the *nsubj* and *dobj* relations from the head verb, the algorithm identifies the police and John as the participants of the event (highlighted in the example output by red boxes). The second step traverses the coreference chains containing the police and John to find other sentences referencing these participants. In this case, the coreference chain for John gives us the second sentence. The final step inspects all sentences in the relevant coreference chains to extract mental state labels using a second set of syntactic patterns. In particular, the second sentence fits the common copulative pattern and the adjective *afraid* (green box) is identified as being relevant to John, the object of the event.

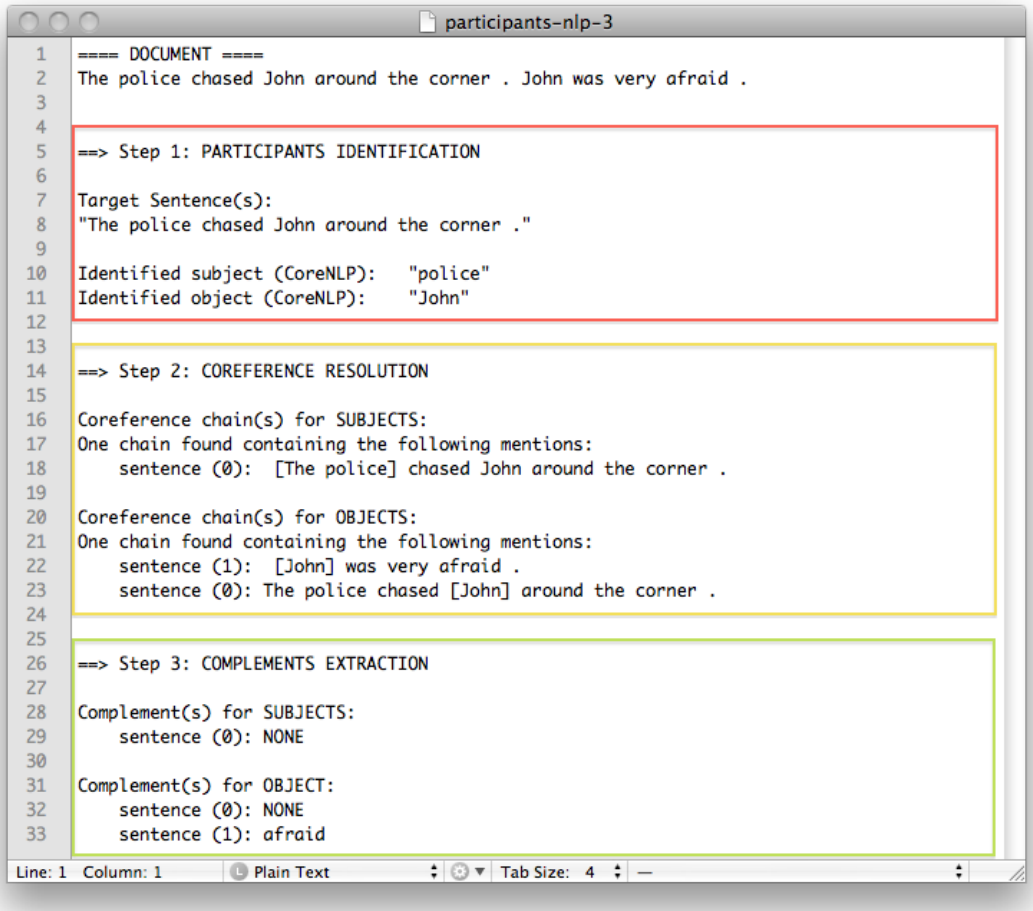


Figure 4.7: Actual output for each step of the *event* model on the example from Figure 4.6. The first step (highlighted by the red box) correctly identified the target sentence and successfully extracted the event participants. The second step (yellow box) found all mentions of the subject and object in other relevant sentences. The final step (green box) extracted the mental state label *afraid* that describes the object participant, John.

neighborhood of relevant sentences for each document. Example output from this step can be found in Figure 4.7.

`Step 3: Identify Mental States of Participants`

Lastly, we identify the mental state terms associated with event participants by analyzing the relevant sentences retrieved from step 2. We do so with a second set of syntactic patterns. First, we inspect several copulative verbs, such as to *be* and *feel*, and extract mental state adjectives from these structures if the corresponding subject is one of the mentions detected above. Second, we search for mental state terms along adjectival modifier relations, where the head is a participant mention. Figure 4.7 shows an example sentence that matches one of our patterns, along with the identified mental state adjective. Details and example illustrations of all syntactic patterns used for the extraction of mental state terms in this model can be found in Appendix C.

For all patterns, we make sure to filter for only mental state labels belonging to the initial seed set. Furthermore, we only consider neighborhoods where all n-gram terms from the associated query tuple are present. For example, we know that the activity verb will be present since it was used as the anchoring event for finding the target sentence. However, we still need to check for the presence of other search terms, such as the actor-type, which is allowed to appear in any sentences within the neighborhood. We note that the same POS restriction as in the *sentence* model also applies.

If all search terms from the query tuple are present and satisfy the POS requirement, then we increment the joint frequency f for the n-gram. As in the *sentence* model, the joint frequency for n-grams, where $n > 1$, is incremented at most once per neighborhood.

Deleted Interpolation and the Response Distribution

As with the *sentence* model, we use the deleted interpolation algorithm to estimate one set of interpolation parameters for the model, based on trigrams generated from

all query tuples from the training set. We similarly combine trigram scores and query tuple distributions to generate a single pruned and normalized distribution of mental state labels as the output response distribution.

Additional Baseline Models

To understand the impact of the various NLP components in the *event* model, we compare it against two additional baseline models. The first baseline investigates the significance of focusing on mental state terms associated with event participants and the second analyzes the importance of coreference resolution to our problem.

`Impact of Participant-specific Extraction`

The first baseline model, called *coref*, implements only the first two steps of the above algorithm, but instead of extracting only mental state terms associated with event participants (the last step), it considers all mental state labels appearing anywhere in the neighborhood output in step 2. In other words, we revert back to the lenient mental state label extraction approach of the *sentence* model. Thus, the *coref* model only addresses the first limitation (data sparsity) of the *sentence* model. As before, the relative joint frequencies of n-grams are then computed by incrementing f once for each neighborhood that contains all terms in the correct POS restrictions.

`Impact of Coreference Resolution`

The second baseline further analyzes the importance of coreference resolution to our problem. This model is similar to *sentence*, with the modification that it increases the size of the neighborhood window to include the immediate neighbors of sentences containing the activity label. We call this the *win-n* model: The window surrounding a target verb contains at most $2n + 1$ sentences. (It could be less than $2n + 1$ if the target sentence does not have enough preceding and following sentences.)

We build the neighborhood area for the model by concatenating all target sentences and their windows together for a given document. This defines a single neighborhood for each document. This contrasts with the *sentence* model, in which the neighborhood is defined for each sentence containing the activity label in the

document, resulting in several possible neighborhoods per document. Figure 4.8 clarifies the different neighborhood areas used by each model.

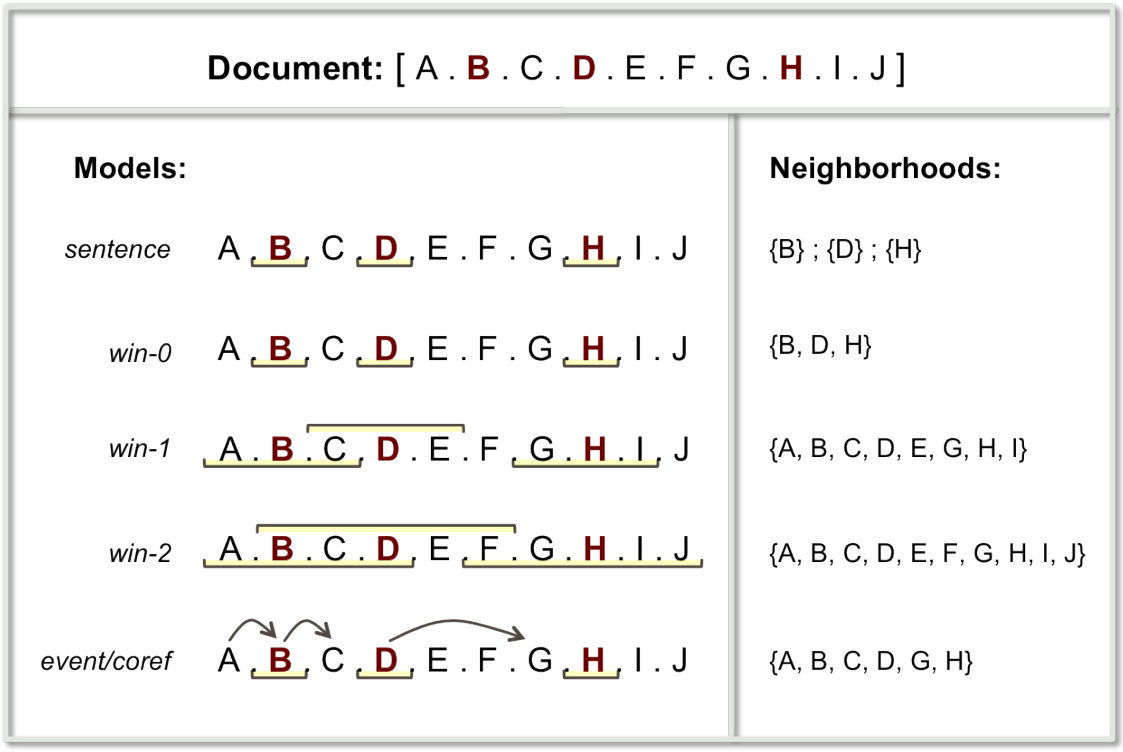


Figure 4.8: In the figure, the letters A through J represent the sentences that form a document. A bolded red letter denotes a target sentence that contains the desired event (e.g., *chase*). The *sentence* model defines a neighborhood for each of the three target sentences, resulting in three distinct neighborhoods for the document. The *win-0* model similarly retrieves only the target sentences, but it pieces them together to form a single neighborhood containing all three sentences. The *win-1* and *win-2* models retrieve additional sentences surrounding each target sentence before concatenating all unique sentences together to form a single neighborhood for the document. The *coref* and *event* models follow the coreference chains (illustrated with arrows) that contain the event participants to form the neighborhood area.

The joint frequency f for each n-gram, where $n > 1$, is computed similarly to the *coref* model: it is incremented once for each neighborhood that contains all the terms from the n-gram in the correct POS. For unigrams, we use frequencies as computed for the *sentence* model.

Semantic Role Labeling

In addition to the baseline models above, which help to analyze the effects of the various NLP components in our model, we are also interested in the effectiveness of semantic role labeling (SRL) in our task. The new model, called *event-srl*, uses semantic role labeling to supplement the search for event participants.

In our implementation, we used SwiRL[3] for this task (Surdeanu et al., 2007). Although SwiRL is capable of identifying several semantic roles for each predicate, or verb, we are only interested in the subject and direct object arguments.

We incorporate the use of SwiRL as follows: First, we run step 1 of the *event* model. For each sentence in which we fail to identify both the subject and direct object of the verb using patterns over shallow syntactic relations, we run SwiRL to extract the desired semantic arguments. If SwiRL succeeds in finding both the subject and direct object arguments, then we use the results from SwiRL; else, we default back to the results obtained from step 1.

In theory, a SRL system like SwiRL should be more successful at identifying the event participants, especially in hard cases where syntactic information alone is not enough. Unfortunately, the use of an external SRL tool also introduces some implementation overheads, namely efficiency and compatibility issues. On average, the SwiRL system takes much longer to process a document, which becomes unfeasible over a large corpus. Hence, we only use it when necessary. Moreover, since the system uses a different underlying parser, the syntactic phrases identified by SwiRL do not always map to a constituent parsed by the CoreNLP parser. This makes combining the results from both methods quite challenging. This incompatibility issue also affects the downstream coreference resolution step. When the syntactic phrases do not align, we cannot find coreference chains containing the arguments output by SwiRL. Our implementation resolves this problem by simply mapping entity mentions together if they share much of the same tokens (i.e., the ratio of their intersection over the union exceeds some pre-defined threshold). Figure 4.9

[3] `http://www.surdeanu.info/mihai/swirl`

shows the output of *event-srl* for an example document from our corpus.

4.3 Ensemble Models

For each deleted interpolation model that operates on text, we combine its output with the *vector* model, which operates in a latent conceptual space, to produce an ensemble model. The ensemble models are named by adding the "*+vector*" suffix to the name of the corresponding deleted interpolation model. The combination is done by averaging the response distributions from both models.

We show in Section 7.2.3 that the ensemble version always outperform each of its individual component alone, suggesting that the information gained from operating on text and operating in a latent conceptual space are highly complementary.

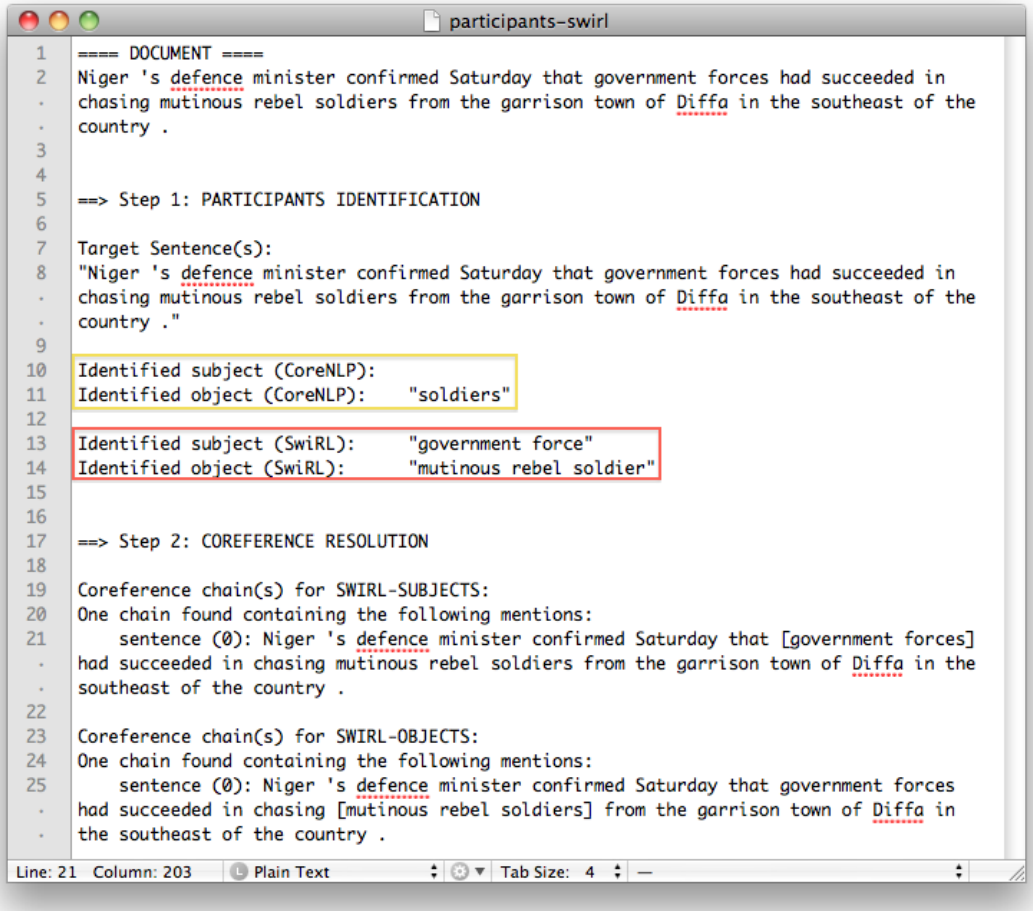


Figure 4.9: Output from the first two steps in the *event-srl* model on a document in our corpus. In the first step, CoreNLP's syntactic dependency failed to give us the subject of the verb (see the yellow highlight box), but SwiRL was able to successfully identify both participants (the red highlight box). Moreover, the syntactic phrase identified by SwiRL for the object participant is more complete. The output from SwiRL is then integrated with CoreNLP's coreference resolution to find the corresponding chains.

CHAPTER 5

PERFORMANCE MEASURES

As previously mentioned, the task proposed here is novel, and hence, does not have any established methods for evaluation. Thus, it is part of our responsibility in defining the task to also construct a good performance measure for the problem.

In Chapter 3, we described a procedure for collecting ground-truth mental state distributions from MTurk workers, and in Chapter 4 we showed several ways to automatically extract these mental state distributions from text. We now explore the best methods for comparing these output.

Let R denote the normalized response distribution of mental state labels produced for a single scene by one of the models described in the previous chapter, and let G denote the gold standard distribution produced for the same video by MTurk workers. If R is similar to G then our model produced similar mental state terms as the workers. Thus, we seek a good performance measure that can properly access the similarity between two mental state distributions. Such a performance measure must account for the following two criteria:

Similarity of distribution shapes. First, a good performance measure for our problem must take into account the similarity between the shapes of the two mental state distributions.

Semantic similarity of distribution elements. Second, the measure must allow semantic comparisons at the level of distribution elements. Suppose R assigns high scores to *angry* and *mad*, only, while G assigns a high score to *happy*, only. Clearly, R is wrong. But if instead G had assigned a high score to *irate*, only, then R would be more right than wrong because, at the level of the individual elements, *angry* and *mad* are similar to *irate* but not similar to *happy*.

We begin our investigation by examining existing methods for comparing distributions (e.g., KL-divergence and chi-square statistics) and discuss their applicability to our problem. We then propose some new measures: a novel score that operates in a latent conceptual space generated by a RNNLM (Mikolov et al., 2013a), and a series of related measures built on top of the familiar F_1 score. We discuss each proposed measure in details and describe their advantages, or disadvantages, with respect to the criteria above.

To further illustrate the effectiveness of each performance measure, we will use the example distributions shown in Table 5.1. In this example, R_5 best matches the shape and meaning of G, because *irate* and *mad* are close synonyms to *angry*, and *scared* is semantically equivalent to *afraid*. R_4 appears to match G semantically, but matches its shape poorly. R_1 misses two of the mental state labels, *afraid* and *guilty*, but contains labels that are semantically close to the weightiest term in G. R_2 and R_3 miss both the shape and the meaning of G; although at first glance, it is hard to estimate which is worse.

Gold G	$(\text{angry}, 0.9), (\text{afraid}, 0.05), (\text{guilty}, 0.05)$
Response R_1	$(\text{angry}, 0.6), (\text{mad}, 0.3), (\text{irate}, 0.1)$
Response R_2	$(\text{guilty}, 0.5), (\text{scared}, 0.5)$
Response R_3	$(\text{sad}, 0.85), (\text{afraid}, 0.05), (\text{angry}, 0.05), (\text{guilty}, 0.05)$
Response R_4	$(\text{guilty}, 0.7), (\text{afraid}, 0.2), (\text{angry}, 0.1)$
Response R_5	$(\text{irate}, 0.45), (\text{mad}, 0.45), (\text{guilty}, 0.05), (\text{scared}, 0.05)$

Table 5.1: An example gold standard distribution G and several candidate response distributions R to be matched against G.

5.1 Known Distribution Similarity Measures

There are many known ways to compare two distributions of elements. Here, we explore a few representative methods: KL divergence, chi-square statistics, and Earth Mover's Distance. We refer interested readers to the work of Rubner et al. (2000) for a more thorough review of other existing measures.

The Kullback-Leibler (KL) divergence is a non-symmetric measure of the difference between two probability distributions. Specifically, the KL divergence of a response distribution R from the gold standard G, denoted as $D_{KL}(G||R)$, is a measure of the information lost when R is used to approximate G. Formally, let $R(w)$ and $G(w)$ denote the probability of w in the response and gold sets, respectively, we have:

$$D_{KL}(G||R) = \sum_i G(w_i) \log \frac{G(w_i)}{R(w_i)} \ .$$

KL-divergence is infamous for its numerical instability. A zero value in the response distribution, caused by a type II (or *false negative*) error, will lead to a division by 0, resulting in an infinite KL value (see example results in Table 5.2). In practice, we can account for these numerical instabilities via some smoothing techniques. Smoothing works well when only a few data points are zero, but can become an issue when the distributions contain a lot of zeros, which is likely for our problem as the set of possible mental state labels to choose from is large. Hence, KL tends to give bad results when distributions are sparse.

Distribution	$\boldsymbol{D_{KL}(G\|\|R)}$
R_1	∞
R_2	∞
R_3	2.60133
R_4	1.77623
R_5	∞

Table 5.2: The KL-divergence of each response distribution R from the gold standard distribution G.

An alternative to KL is the empirically derived Jeffrey divergence, which is a modification of the KL divergence that is numerically stable, symmetric, and robust. It is defined as:

$$D_J(G||R) = \sum_i G(w_i) \log \frac{G(w_i)}{m_i} + R(w_i) \log \frac{R(w_i)}{m_i} \ ,$$

where $m_i = \frac{G(w_i)+R(w_i)}{2}$.

The χ^2 statistics are also popular for comparing distributions. This distance is often used to measure how unlikely it is that one distribution was drawn from the population represented by the other, formulated as:

$$D_{\chi^2}(G, R) = \sum_i \frac{(G(w_i) - m_i)^2}{m_i} \ ,$$

where $m_i = \frac{G(w_i)+R(w_i)}{2}$.

While these measures account well for the dissimilarity between the shapes of the distributions, their major drawback lies in the fact that they do not account for the semantic similarity of the elements. In other words, they only account for the correspondences between *bins*, or elements, of the same index. The Earth Mover's Distance (EMD) presented by Rubner et al. (2000) improves upon these measures by also accounting for the correspondences between different bins. Intuitively, if two distributions are viewed as two collections of dirt piles of varying sizes, then EMD measures the least amount of work needed to rearrange the dirt piles of one collection to match the other. One can associate different costs for moving a unit of dirt between different piles. As such, EMD is very relevant to our work as it satisfies both of our criteria. Upon closer inspections, however, some behaviors of EMD do not seem to naturally fit our problem. For example, EMD allows an element to match (in parts) multiple other elements, i.e., dirt from a single source pile can be moved to multiple target piles. That means the label $(\text{mad}, 0.3)$ in R_1 can potentially match some of its weight to *angry* and some to *afraid* in the gold distribution. This seems counter intuitive to our problem. EMD allows this split because it seeks the minimum cost of transforming one distribution to the other. This search for an optimal solution in EMD requires a super-cubic runtime in the size of the distributions, which is bad news for us as the number of mental state labels in our response distributions can get very large. While EMD is a good case study, we desire a more intuitive and less computationally expensive measure.

5.2 Vector-based Distribution Similarity

The first performance measure we propose, called *vector-based distribution similarity* (VDS), is based on the same approach as our *vector* information extraction model (see Section 4.2.1). The idea is to use the RNNLM of Mikolov et al. (2013a) to project elements (i.e., mental state labels) of a distribution into a latent conceptual space, and then combine the vector representations of these elements, discounted by their weights, to generate a single vector representing the entire distribution, known as the distribution vector. Similarity between two mental state distributions is then trivially computed as the cosine similarity between the two normalized distribution vectors. The hypothesis is that similar distributions will yield similar distribution vectors that will have high cosine similarity scores.

Formally, let $w_1, w_2, ..., w_n$ be the elements of the response distribution R and let $R(w_i)$ denote the probability of w_i in R. We compute the distribution vector for R as follows:

$$\vec{v}_R = \sum_{i=1}^{n} R(w_i)\ \vec{v}_{w_i}\ .$$

The similarity score between a response distribution R and the gold standard distribution G is the cosine similarity between the two distribution vectors:

$$cos(\Theta_R) = \frac{\vec{v}_R \cdot \vec{v}_G}{||\vec{v}_R||\ ||\vec{v}_G||}\ .$$

VDS is a particularly interesting measure as it satisfies both criteria of our problem (it accounts for the distribution weights as well as the semantic similarity of the elements) and appears to work well enough on the examples from Table 5.1 (see Table 5.3). Yet, VDS is not very useful in practice. The primary reason being that semantic in the underlying latent conceptual space is not fully understood. Synonyms are not guaranteed to map with similar vectors in this space; and vice versa, words corresponding to similar vectors are not guaranteed to be similar in meanings. Results from Mikolov et al. (2013b), as well as from our own *vector* model, suggest

that this latent conceptual space does preserve *some* relationships and meanings of words, but it is not guaranteed for all cases. As shown by results in Table 5.3, VDS correctly assigned high scores to R_1 and R_5 and gave lower scores to the remaining three response distributions. However, given that both R_1 and R_5 properly capture the semantic of elements in G, VDS gave a much higher score to R_1 than R_5 even though R_5 better matches the shape of G. This is due to the fact that R_5 contains more synonyms where as R_1 matches the exact terms; and synonyms do not always map to exactly similar vectors in our latent space.

Distribution	**VDS Score**
R_1	0.9372
R_2	0.5037
R_3	0.5204
R_4	0.4467
R_5	0.6744

Table 5.3: The vector-based distribution similarity (VSD) score between the gold standard distribution G and each of the response distribution R. The underlying RNNLM was trained on the English Gigaword (5th Ed) corpus.

Moreover, we have found in our experiments that the cosine similarity scores yielded by VDS tend to be very close. This makes VDS less informative as an evaluation measure since it often returned very similar results for all evaluated models. Due to these setbacks, we opted to not use VDS in our evaluation. Though, we feel that it could prove useful for other research problems, especially when used with a better understood latent space.

5.3 Constrained Weighted Similarity-Aligned F_1

We build the next performance measure by applying a series of improvements to the familiar F_1 score. Each addition to the scoring method is meant to address an evaluation requirement of our problem. The end result is a novel performance measure that fully satisfies the required criteria of our problem, while yielding easy to understand & intuitive results.

5.3.1 F_1 Score

The standard F_1 *score* measures the similarity between two sets of elements, R and G. $F_1 = 1$ when $R = G$ and $F_1 = 0$ when R and G share no elements. Formally, F_1 is defined as the harmonic mean of *precision* and *recall*:

$$precision = \frac{|R \cap G|}{|R|} , \qquad recall = \frac{|R \cap G|}{|G|} , \tag{5.1}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} . \tag{5.2}$$

The F_1 score in its standard form does not satisfy any of our evaluation requirements. It penalizes the responses in Table 5.5 that include semantically similar labels to those in G, and fails to reflect the weights of the labels in G and R.

5.3.2 Similarity-Aligned F_1 Score

Although the standard F_1 does not immediately fit our needs, it is a good starting point. We can incorporate the semantic similarity of distribution elements by generalizing the formulas for precision and recall as follows:

$$\begin{aligned} precision &= \frac{1}{|R|} \sum_{r \in R} \max_{g \in G} \sigma(r, g) , \\ recall &= \frac{1}{|G|} \sum_{g \in G} \max_{r \in R} \sigma(r, g) , \end{aligned} \tag{5.3}$$

where $\sigma \in [0, 1]$ is a function that yields the similarity between two elements. The standard F_1 measure has:

$$\sigma(r, g) = \begin{cases} 1 , & \text{if } r = g \\ 0 , & \text{otherwise} \end{cases} ,$$

but clearly σ can be defined to take values proportional to the similarity of r and g.

We can choose from a wide range of semantic similarity and relatedness measures that are based on WordNet (Pedersen et al., 2004), and the recent RNNLM

of Mikolov opens the door to even more similarity measures based on vector space representations of words (Mikolov et al., 2013a). After experimenting with several measures, we decided on one proposed by Hirst and St-Onge (1998). This method, called HSO, was originally proposed for the problem of finding *lexical chains* in text, where a lexical chain is, in essence, a cohesive chain in which the words in the chain bear some kind of cohesive relationships (not necessarily one specific relationship). HSO represents two lexicalized concepts as semantically close if their WordNet synsets are connected by a path that is not too long and does not have too many "changes of direction" (Hirst and St-Onge, 1998). A change in direction is simply a change in the type of relation connecting the two synsets. We chose this measure because it has a finite range and can accommodate numerous POS pairs. In practice, we found this similarity measure to be quite strict in that it tends to under estimate the similarity of a pair unless the words are close synonyms, as shown in Table 5.4. This is fine for our purpose, fortunately, as a stricter evaluation measure is generally more robust and preferable to a lenient one.

	irate	*mad*	*upset*	*furious*	*rage*
angry	1.0	1.0	0	1.0	0
irate	1.0	0.375	0	0.375	0
outrage	1.0	0	0	0	0.3125
stressed	0	0	0.375	0	0
agitate	0	0	0.3125	0	0
panic	0	0	0.1875	0	0.1875

Table 5.4: The normalized HSO similarity scores for different pairs of mental state lemmas, which are base forms of mental state adjectives, using WordNet 3.0.

Given the generalized precision and recall formulas in Eq 5.3, our *similarity-aligned* (SA) F_1 score can be computed in the usual way, as the harmonic mean of precision and recall (Eq 5.2).

SA-F_1 is inspired by the Constrained Entity-Aligned F-Measure (CEAF) metric proposed by Luo (2005) for coreference resolution. CEAF computes an optimal one-to-one mapping between subsets of reference and system entities before it computes

recall, precision, and F_1. Similarly, SA-F_1 finds optimal mappings between the labels of the two sets based on σ (this is what the max terms in Eq 5.3 do). Table 5.5 shows that SA-F_1 correctly rewards the use of synonyms. The high scores given to R_2 and R_4, however, indicate that it does not measure the dissimilarity between distribution shapes.

5.3.3 Weighted Similarity-Aligned F_1 Score

We address the shortcoming of SA-F_1 by introducing the distribution weights of the elements into the formulas. Let $R(r)$ and $G(g)$ be the probabilities of labels r and g in the R and G distributions, respectively. We can reformulate Eq 5.3 in terms of $R(r)$ and $G(g)$ as follows:

$$\begin{aligned} \mathit{precision} &= \sum_{r \in R} R(r) \cdot \max_{g \in G} \sigma(r, g) \ , \\ \mathit{recall} &= \sum_{g \in G} G(g) \cdot \max_{r \in R} \sigma(r, g) \ , \end{aligned} \tag{5.4}$$

where $R(r) = \frac{1}{|R|}$ for all elements $r \in R$, and $G(g) = \frac{1}{|G|}$ for all elements $g \in G$. In this formulation, we see that the probability terms are already built into the standard precision and recall formulas, except a uniform distribution is assumed for elements in each set.

It is trivial to swap in the actual distribution weights of the elements in Eq 5.4. Moreover, let $\sigma^*_S(\ell)$ denote the best similarity score achievable when comparing elements from set S to label ℓ using the similarity function σ. That is,

$$\sigma^*_S(\ell) = \max_{e \in S} \sigma(\ell, e) \ ,$$

which gives us the following equations for precision and recall:

$$\begin{aligned} \mathit{precision} &= \sum_{r \in R} R(r) \cdot \sigma^*_G(r) \ , \\ \mathit{recall} &= \sum_{g \in G} G(g) \cdot \sigma^*_R(g) \ . \end{aligned} \tag{5.5}$$

The harmonic mean of these new formulas give us the new *weighted similarity-aligned* (WSA) F_1 score. While it may seem that WSA-F_1 satisfies both evaluation criteria of our problem, it is in fact still incomplete, as indicated by the incorrect high scores given to R_4 in Table 5.5.

5.3.4 Constrained Weighted Similarity-Aligned F_1 Score

The main problem with WSA-F_1 is that it does not account for the probability of r in the gold standard distribution G when computing precision; and likewise does not account for the probability of g in the response distribution when computing recall.

An analogy might help here: Suppose we have an unknown "mystery bag" of 100 colored pencils that we will try to match with a "response bag" of pencils. If we fill our response bag with 100 crimson pencils, while the mystery bag contains only 25 crimson pencils, then our precision score should get points only for the first 25 pencils, while the remaining 75 in the response bag should not be rewarded. For recall, the reward given for each color in the mystery bag should be capped by the number of pencils of that color in the response bag. If we apply this concept to the examples from Table 5.1, it means that the precision for R_4 should not be fully awarded the full 0.7 value for matching *guilty* because there is only 0.05 of it available in G. Similarly, while G assigns 0.9 – which is a significant amount – of its weight to *angry*, it cannot recall more than what is available for *angry* in R_4, which is only 0.1.

The analogy is complete when we consider that crimson pencils should perhaps be partially rewarded when matched by cardinal, rose or cerise pencils. In other words, a similarity measure should also account for an accumulated mass of synonyms. Let $M_S(\ell)$ denote the subset of terms from S that have the *best* similarity score to ℓ. That is $M_S(\ell)$ contains the set of synonyms from S that best match some label ℓ:

$$M_S(\ell) = \{e \mid \sigma(\ell, e) = \sigma^*_S(\ell),\ \forall e \in S\}\ .$$

	F_1			SA-F_1			WSA-F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1	p	r	f_1	p	r	f_1
R_1	$0.3\bar{3}$	$0.3\bar{3}$	$0.3\bar{3}$	1	$0.3\bar{3}$	0.5	1	0.9	0.95	1	0.9	0.95
R_2	0.5	$0.3\bar{3}$	0.4	1	$0.6\bar{6}$	0.8	1	0.1	0.18	0.1	0.1	0.1
R_3	0.75	1	0.86	0.75	1	0.86	0.15	1	0.26	0.15	0.15	0.15
R_4	1	1	1	1	1	1	1	1	1	0.2	0.2	0.2
R_5	0.25	$0.3\bar{3}$	0.29	1	1	1	1	1	1	1	1	1

Table 5.5: The precision (p), recall (r), and F_1 (f_1) scores under various evaluation models are presented for the examples from Table 5.1. For simplicity, assume that $\sigma(\textit{angry}, \textit{irate}) = \sigma(\textit{angry}, \textit{mad}) = \sigma(\textit{afraid}, \textit{scared}) = 1$, with σ of any two identical labels being 1, and σ of all other pairs are 0.

We define new forms of precision and recall:

$$\begin{aligned} \textit{precision} &= \sum_{r \in R} \min\left(R(r), \sum_{e \in M_G(r)} G(e)\right) \cdot \sigma^*_G(r) \ , \\ \textit{recall} &= \sum_{g \in G} \min\left(G(g), \sum_{e \in M_R(g)} R(e)\right) \cdot \sigma^*_R(g) \ . \end{aligned} \tag{5.6}$$

The resulting *constrained weighted similarity-aligned* (CWSA) F_1 score is the harmonic mean of these new precision and recall scores. Table 5.5 shows that CWSA-F_1 yields the most intuitive evaluation of the example response distributions, downweighting R_2, R_3, and R_4 in favor of R_5 and R_1.

The computation of CWSA-F_1 is quadratic in the size of the distributions. This is an improvement over the super-cubic runtime of EMD. Although a quadratic runtime still scales quite poorly when comparing large distributions, we don't think it is possible to do any better. At the minimum, each element from one distribution must be compared with every element from the other distribution. This imposes a lower bound of $\Omega(n^2)$ on the problem.

Although the score was specifically design to address the concerns of our problem, we believe that its usefulness extends well beyond our work. In particular, the CWSA-$F1$ score can be used to effectively compare the differences between any two distributions of elements, where the semantic similarity between each element from

the two distributions should also be considered.

Moreover, the presented CWSA-F_1 score fully satisfies both conditions for a good evaluation measure to our problem. It also gives good intuitive results for the tested distributions. In the next chapter, we further investigates the effectiveness of CWSA-F_1 by comparing its performance to those of human annotators in order to test how well CWSA-F_1 scores align with human judgement.

CHAPTER 6

MEASURE EVALUATION

In the previous chapter, we sought to find a good performance measure that can properly access the similarity between two mental state distributions. We defined two primary criteria that a good performance measure must satisfy: (1) it must account for similarity of distribution shapes, and (2) it must allow for semantic comparisons at the level of distribution elements. Our research led to the formulations of several candidate measures. In particular, we showed that the *constrained weighted similarity-aligned* (CWSA) F_1 score satisfies both criteria and yielded the most intuitive evaluation of a few example response distributions.

In this chapter, we present a more thorough investigation into the effectiveness of the CWSA-F_1 score in estimating the similarity between mental state distributions. Using human judgement as the gold-standard, we designed an experiment to compare decisions made based on each performance measure to those of human annotators. We expect a good measure to have a high agreement rate with human selection.

The next few sections detail our experimental procedure and results. We close the chapter with a discussion of our findings.

6.1 Matching Similarity Experiment

To evaluate how well a performance measure corresponds with human judgement, we designed a *matching similarity* experiment, which consists of multiple *choice matching problems*, to test the human-agreement rate of each measure.

6.1.1 Evaluating the Proposed Measures

In a choice matching problem, human annotators are presented with a *target* mental state distribution that represents the mental states of some target person. They are also presented with two additional mental state distributions as choices to choose from. The annotators must select (between the two choices) the option that best resembles the meaning of the target distribution, based on the semantics of the mental state terms and their respective weights (or proportions) in the distributions. The most popular choice among all human annotators is then selected as the *ground-truth choice*.

Similarly, the same choice matching problem is given to a *matching system*. The matching system computes the similarity score between each choice to the target distribution, using some pre-defined similarity measure, and outputs the option with the highest similarity score to the target distribution as the *system choice*.

Thus, given the two choices, the main experimental question to test is: How often does the *system choice* match the human majority *ground-truth choice* for some matching system? Figure 6.1 shows a diagram of the described experimental protocol.

6.1.2 Experiment Data

Our experiment used six different mental state distributions, as shown in Table 6.1. We chose to use mock-up distributions, as opposed to real extraction output, in order to keep the task simple for human annotators. Based on feedback, we found that distributions containing more than five mental state labels become too cluttered, which makes it harder for annotators to understand them. Our goal is to simplify the problem as much as possible to prevent unnecessary confusion.

We divided the six mental state distributions from Table 6.1 into 20 different triplet combinations (e.g., $\{R_0, R_1, R_2\}$, $\{R_0, R_1, R_3\}$, $\{R_1, R_2, R_5\}$, etc.). For each triplet combination, we constructed three different choice matching problems by using each of the element in the triplet as the target distribution once, while leaving

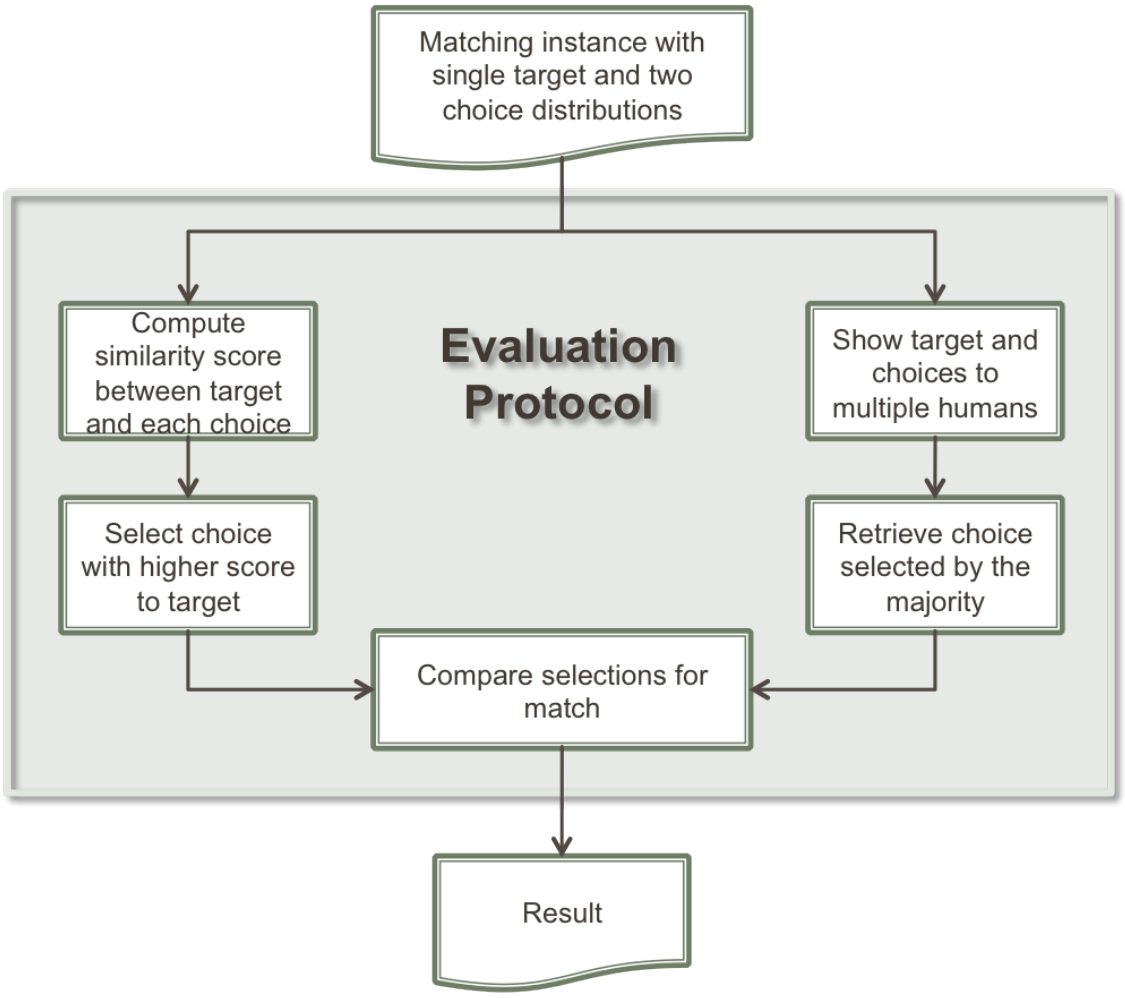


Figure 6.1: Diagram of the evaluation protocol for our matching similarity experiment.

R_0	$(\text{angry}, 0.9), (\text{afraid}, 0.05), (\text{guilty}, 0.05)$
R_1	$(\text{mad}, 0.6), (\text{angry}, 0.3), (\text{irate}, 0.1)$
R_2	$(\text{afraid}, 0.5), (\text{scared}, 0.4), (\text{guilty}, 0.1)$
R_3	$(\text{scared}, 0.6), (\text{angry}, 0.2), (\text{guilty}, 0.2)$
R_4	$(\text{angry}, 0.35), (\text{afraid}, 0.35), (\text{guilty}, 0.3)$
R_5	$(\text{mad}, 0.8), (\text{scared}, 0.1), (\text{guilty}, 0.1)$

Table 6.1: The matching similarity experiment uses six different example mental state distributions. The distributions are purposely kept small and simple to reduce the complexity of the task for human annotators.

the other two distributions as the choices. For example, the triplet $\{R_0, R_1, R_2\}$ would generate the three problem instances in Table 6.2. Thus, we can generate a total of 60 different choice matching problems from the six distributions.

Target	Option A	Option B
R_0	R_1	R_2
R_1	R_2	R_0
R_2	R_0	R_1

Table 6.2: The three possible choice matching problems generated by the $\{R_0, R_1, R_2\}$ distribution triplet. Each triplet can generate three different problem instances by using each of the element as the target distribution once, and leaving the other two distributions as the choices. The order of the choices does not matter. That is, if R_0 is the target, then it does not matter if R_1 is option A and R_2 is option B, or vice versa.

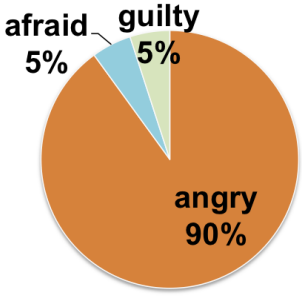
Distribution Representation

Since there are 60 different choice matching problems to be annotated, it is important that the representation of the distribution data is aesthetically pleasing in order to alleviate the repetitive nature of the task. We found that the straightforward list-of-tuples representation, as presented in Table 6.1, is not the best way to portray the distributions. When given a list of values, people have a tough time visualizing and correctly interpreting the different weights associated with the mental state labels.

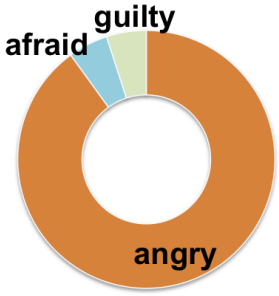
To resolve this issue, we explored several different visual representations for mental state distributions. Figure 6.2 shows the three different representations that

we considered: word cloud, color-coded pie chart, and color-coded doughnut chart. All three representations visually emphasize the different proportions (or weights) associated with each label in the distributions.

(a) Word cloud representation.



(b) Pie chart with values.



(c) Doughnut chart.

Figure 6.2: Different visual representations of the R_0 mental state distribution from Table 6.1.

We presented all three representations to our annotators and collected their feedback. Based on the reactions, we found that doughnut charts appealed better to annotators than pie charts, while word clouds can unintentionally lead people to associate fallacious meanings to the locations of words in clouds (i.e., the idea that two word clouds might be more similar because the same label is located near the same location in both clouds). As the result, we decided to use doughnut charts to represent mental state distributions in our matching similarity experiment.

Figure 6.3 shows the color-coded doughnut chart representation for all six mental state distributions shown in Table 6.1.

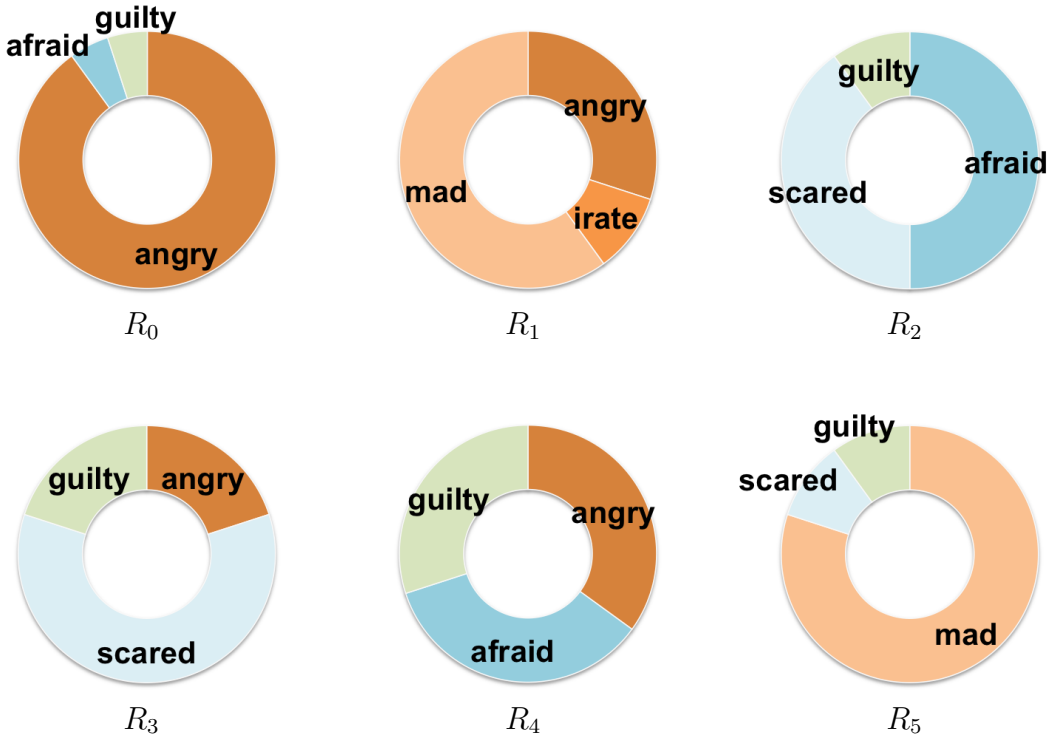


Figure 6.3: Color-coded doughnut chart representations of all mental state distributions from Table 6.1. Synonyms of the label *angry* are colored with different shades of orange, while synonyms of *scared* and *guilty* have different shades of blue and green, respectively.

Human-Majority Annotation

In order to collect ground-truth annotations from humans, we designed and tested our experiment in Amazon Mechanical Turk (MTurk) using the sandbox mode. We created a total of 60 different tasks, each representing a choice matching problem. Figure 6.4 shows a screenshot for one of our tasks.

We collected annotation from three different in-house annotators for each task. We opted to use in-house annotators rather than crowd-sourcing because we only needed a few annotators. Also, the annotation requires a little training that is best

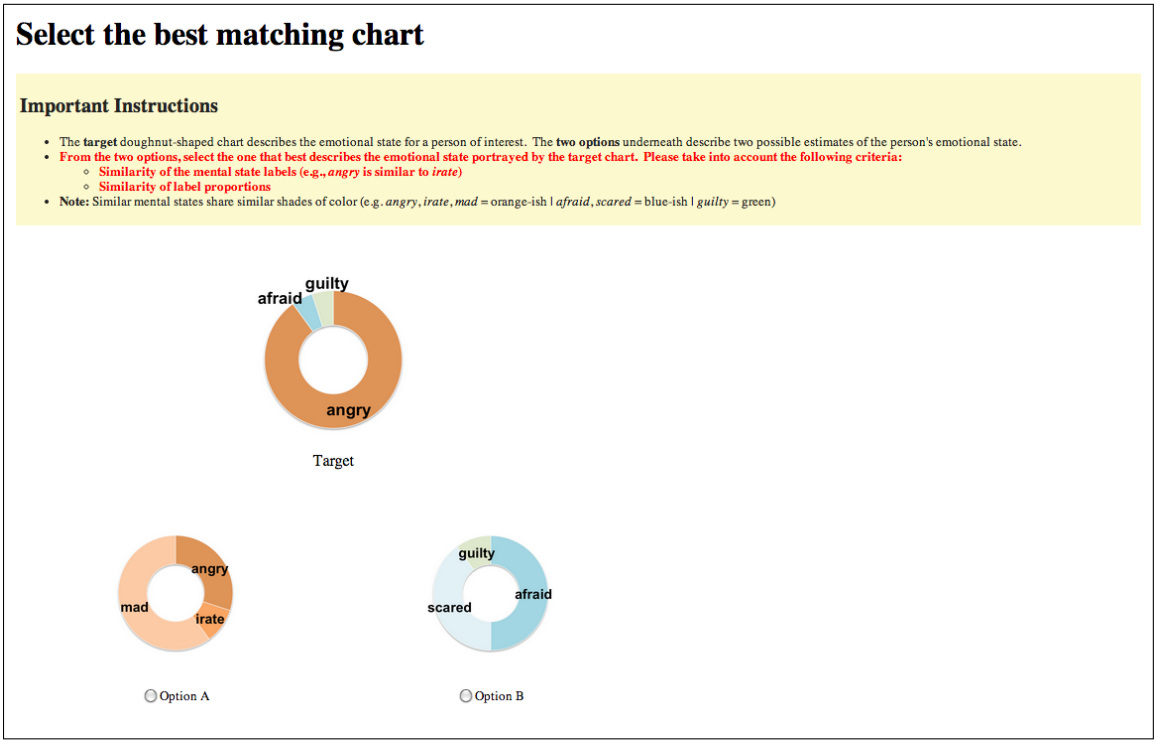


Figure 6.4: Screenshot of a matching similarity experiment's choice matching problem.

done in person. Each annotator was trained using the example shown in Figure 6.5. We explained to each annotator that the target doughnut chart represents the mental state distribution of some target person, while the two options represent guesses about the state of mind of that person. Each annotator was then guided to select option A, as both option A and the target distribution portray a person who is predominantly angry, while option B describe someone who is mostly afraid.

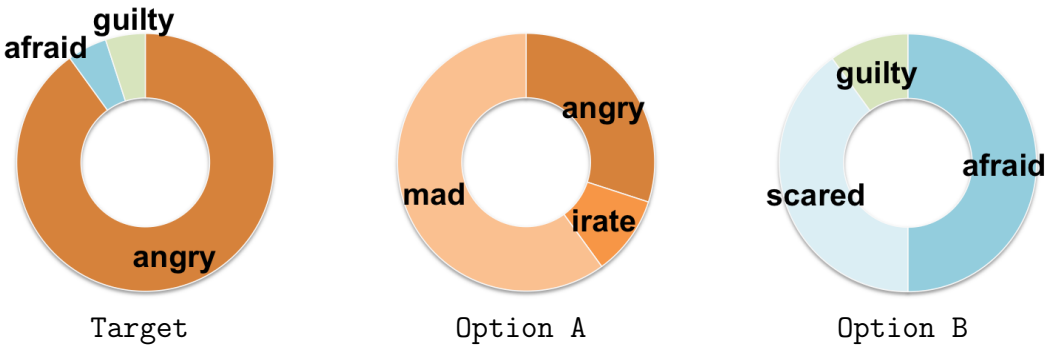


Figure 6.5: The training example given to all human annotators. Each annotator was trained to select option A in this example, since both option A and the target distribution describe a person who is mostly angry, while option B portrays someone who is mostly afraid.

After the training, each annotator was asked to complete the remaining 59 choice matching problems. We tabulated the choices made by all three annotators. The majority choice is then selected to be the ground-truth choice for each problem. Moreover, we found that the three human annotators unanimously agreed 52 times out of the 59 instances. In other words, they arrived at a total agreement 88% of the time.

System Output

We gave the same 60 choice matching problems to our matching systems. For each problem, each system computes a similarity score between each choice to the target distribution using the associated similarity measure. The choice with the highest similarity score is selected as the system choice.

6.2 Matching Experiment Results

We present the matching results for four of our matching systems in Table 6.3. Each matching system relies on one of our proposed performance measures from Chapter 5. The matching accuracy evaluates the human-agreement rate of each system across 59 choice matching problems (one of the problems was held out for annotation training).

System	**Matching Accuracy**
VDS	0.6780
F_1	0.5254
WSA-F_1	0.7627
CWSA-F_1	0.9492

Table 6.3: The matching accuracy of four matching systems, each based on a performance measure from Chapter 5. Choices made based on the CWSA-F_1 score correspond with human judgement almost 95% of the time, while decisions made based on the classical F_1 only did slightly better than chance (50%) at estimating human selections.

The matching system based on the classical F_1 achieved an agreement rate of 52.54%, which is slightly better than chance at 50%. This validates our claim regarding the inapplicability of the classical F_1 score to our problem. The *vector-based distribution similarity* (VDS) score performed better at 67.80% agreement, and the *weighted similarity-aligned* (WSA) F_1 measure was second best at 76.27%. As expected, CWSA-F_1 corresponds best with human judgement. The decision made based on this measure agreed with the human-majority selection almost 95% of the time. In fact, the choices between the two only differed in three out of 59 tested instances (see Figure 6.6).

6.3 Discussion

The results found in this matching similarity experiment strongly supports our claim regarding the effectiveness of the CWSA-F_1 measure in accessing the similarity

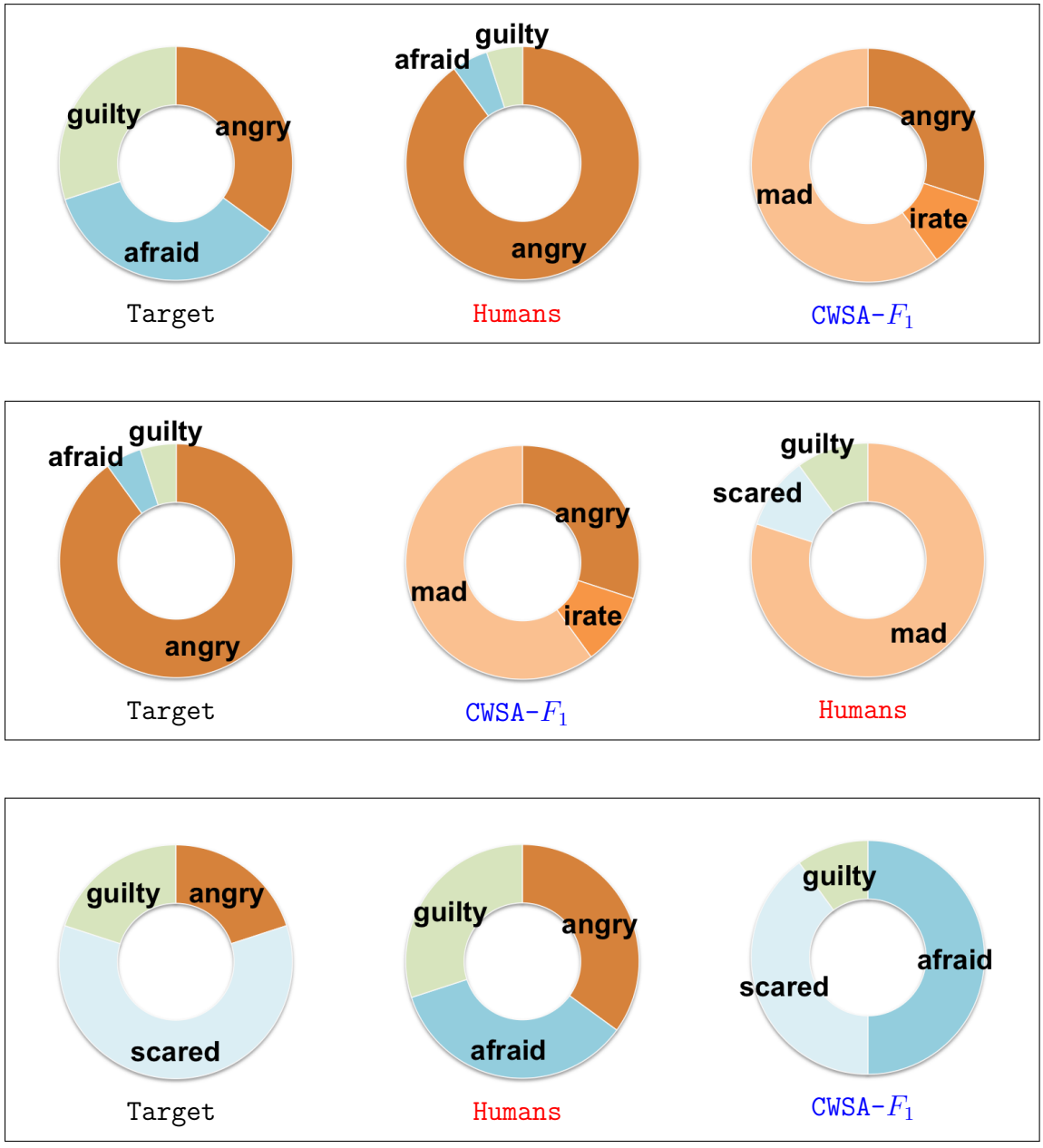


Figure 6.6: Decisions made based on the CWSA-F_1 measure (blue) disagreed with the human-majority selections (red) in just 3 out of 59 test instances.

between mental state distributions. Not only does it satisfy both of our conditions for a good performance measure, by accounting for both similarity of distribution shapes and semantic similarity of distribution elements, CWSA-F_1 is also highly comparable to human judgement. As such, CWSA-F_1 is the primary measure used to evaluate the performance of our extraction models in the next chapter.

CHAPTER 7

EXTRACTION RESULTS

In this chapter, we present empirical results demonstrating the ability of our information extraction models to accurately extract relevant mental state distributions from text. We begin by describing our experimental procedure that evaluates the performance of our models against the ground-truth data generated by MTurk workers. Next, we present and discuss extraction results for videos from the *chase* dataset and explore the impact of various factors, such as the inclusion of NLP components, on the performance of our models. We then explore the generality and robustness of our system by looking at extraction results for videos from the *hug* dataset, as well as examining the performance of our system under noisy detection data. We end the chapter with a brief discussion regarding the scalability of our system.

7.1 Extraction Experimental Procedure

The primary experimental question we are interested in for the extraction experiments is: How well does a normalized mental state distribution generated by MTurk workers, G, match the response distribution output by an automatic extraction model, R, for each video? To answer this question, we devised the experimental protocol shown in Figure 7.1.

As noted in Chapter 3, we generated ground-truth data by showing the videos to MTurk workers and asking them to identify the detections and mental state labels for each video. The mental state labels collected are compiled into one normalized probability distribution over labels for each video, designated as G in our protocol shown in Figure 7.1.

The detections generated by the MTurk workers become input into our system. The idea is to use these workers as a proxy for an automatic detection system that

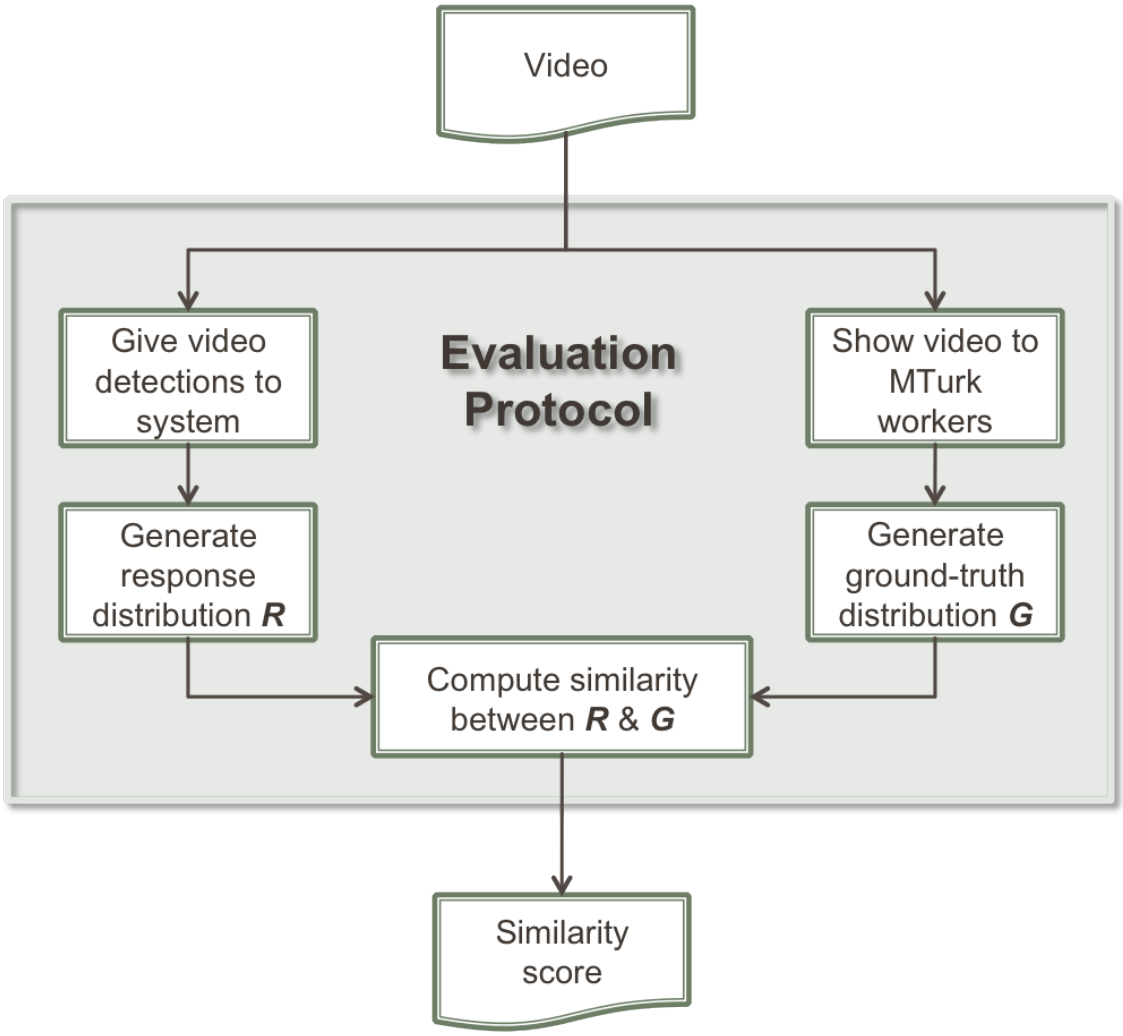


Figure 7.1: Diagram of the evaluation protocol for our extraction experiments.

operates at human-level performance. The detections are formatted by our system into query tuples following predefined patterns, such as the (*activity*, *actor-type*) pattern, which are then given to our neighborhood information extraction models (Section 4.2). The models return a response distribution over mental state labels for each video, designated R.

Thus, the experimental question reduces to, how well does G match R for each video?

7.1.1 Parameters Tuning

Although our neighborhood information extraction models are largely unsupervised, they do require initial calibration of some hyper-parameters, for which we require a little training data. For each dataset, we selected a few videos from the set, four from the *chase* dataset and five from the *hug* dataset, to calibrate the prune parameters γ and the interpolation parameters λ (Section 4.2) for each model. We refer to these as the calibration sets. Of the four chase videos in the *chase* calibration set, one of them contains children, one has police involvement, and two contain adults. The hug calibration set contains one of each of the following relationships between huggers: acquaintances, friends, lovers, siblings, and parent-child.

To generate data for the calibration sets, we asked additional MTurk workers to annotate these videos, yielding an independent set of annotations to be used solely for calibration. Annotations used for calibration were never used for testing, and vice versa, annotations used to generate the ground-truth data were never used for calibration.

We calibrate the parameters by optimizing the CWSA-F_1 score on the calibration sets. Once calibrated, we use the same set of parameters for all videos in the test set. Our models do not require meticulous calibrations for each test video.

7.1.2 AA Query Pattern

We formulated the search context for all models following the (*activity*, *actor-type*) query pattern in all extraction experiments. In other words, our models were given only information regarding the activity and the types of actors involved for each video.

7.2 Mental State Identification in Chase Videos

The primary dataset used to evaluate our models is the *chase* dataset. The *chase* dataset offers several advantages over the *hug* dataset that makes it more suitable for extensive experimentations: (a) chases often invoke clearer mental state inferences (at least in humans) which tend to give us clearer targets to evaluate against, and (b) the English Gigaword corpus contains more instances of the word "chase" than the word "hug", suggesting that it may contain more data about chases than hugs. We focus most of our experiments on the *chase* dataset, but also provide performance benchmarks for the *hug* dataset later on to show the generality of our approach.

7.2.1 Average Performance of Models

We report the average performance of our models, measured in terms of the F_1 score and the CWSA-F_1 score, across all 26 videos from the *chase* dataset in Table 7.1. As discussed in Chapter 5 and verified in Chapter 6, the CWSA-F_1 score most accurately reflects the similarity between two mental state distributions. The F_1 score, although not suited for our problem, is intuitive and easy to understand for most people.

To provide a basis for comparison, we also present scores for two additional baseline methods in Table 7.1. The naïve *uniform* baseline method simply binds R to the initial seed set of 160 mental state labels with uniform probability, while the stronger *frequency* baseline uses the occurrence frequency distribution of the labels from the English Gigaword corpus (note that only occurrences tagged as adjectives or verbs were counted). All average improvements of the ensemble model *event+vector*

over the baseline methods in Table 7.1 are significant ($p < 0.01$). All significance tests were one-tailed and were based on nonparametric bootstrap resampling with $10,000$ iterations.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
uniform	.107	**.750**	.187	.284	.289	.286
frequency	.107	**.750**	.187	.362	.352	.355
sentence	.194	.293	.227	.366	.376	.368
vector	.226	.145	.175	.399	.392	.393
coref	**.264**	.251	.253	.382	.461	.416
event	.231	.303	.256	.446	.488	.463
event+vector	.259	.296	**.274**	**.488**	**.517**	**.500**

Table 7.1: The average evaluation performance across 26 different chase videos are shown against two different baseline methods for our neighborhood information extraction models. Bold font indicates the best score in a given column.

Using the classical F_1 measure, the *coref* model scored highest on precision, while the ensemble method *event+vector* did best on F_1. Not surprisingly, no model can top the baseline methods on recall. This is because both baselines make use of all 160 mental state labels from the seed set for each video, allowing for maximum coverage in terms of recall. Even so, the average recall for the baselines were only 0.750, which means that the initial seed set did not include all labels used by the MTurk annotators.

As we have mentioned, the classical F_1 measure is misleading because it does not credit synonyms. For example, Table 7.2 shows the output from our models for the 10th chase video[1]. In this case, our *event+vector* model correctly identified *upset* and *angry* from G_{10}, but was penalized for the use of *mad* and *furious*. Likewise, the model got *scared* but used *afraid* instead of *fearful*. Uses of these synonyms are classified as *false positive* and *false negative* errors under the classical F_1. Despite these penalties, however, our models actually performed quite well on video 10. The *event+vector* model achieved an F_1 score of 0.431 for this video – a +0.211

[1] `http://youtu.be/CEzHGwtANxM`

improvement from the baselines. Table 7.3 shows the output for the 23rd video[2]. This is an example where our models did not do so well. In fact, according to the performance charts in Figure 7.2, video 23 is one of the most challenging videos for our models. The *event+vector* model also did best in this video, achieving an F_1 score of 0.154 , which improves slightly over the baselines by +0.052.

Under the CWSA-F_1 performance measure, which correctly accounts for both synonyms and label probabilities, the ensemble model *event+vector* performed best in Table 7.1. The average CWSA-F_1 score of this ensemble model improves upon the simple uniform baseline by almost 75%, and over the stronger frequency baseline by over 40%. It also outperforms each of its individual component in all measured F_1 scores. These improvements were also found to be significant. This strongly suggests that the *vector* and *event* models are complementary, not entirely redundant. Furthermore, the steady improvements going from the *sentence* model to the *coref* model, and subsequently from *coref* to the *event* model show the benefits of focusing on event-centric mental state terms using various NLP techniques. We will further explore the impacts on performance of ensemble models, NLP components, and other factors in the following subsections.

Moreover, it is worth noting that results in Table 7.1 also shows that the frequency baseline outperforms the simple uniform-probability baseline by 24%. This performance gain strongly suggests that the inherent occurrence frequency of mental state terms in the underlying data source (the English Gigaword corpus) is informative for identifying latent mental state information in chase videos.

Figure 7.2 plots the CWSA-F_1 scores for each model across 26 different chase videos. As expected, the ensemble method *event+vector* consistently yielded the highest score, losing out to another model on only four occasions. The *event* model is next best. The plot also shows that our models behave quite consistently with each other. We did not see any cases where one model performed significantly better while a better model (based on average performance) performed significantly worse. There are some videos that seem to be more challenging for all of our models,

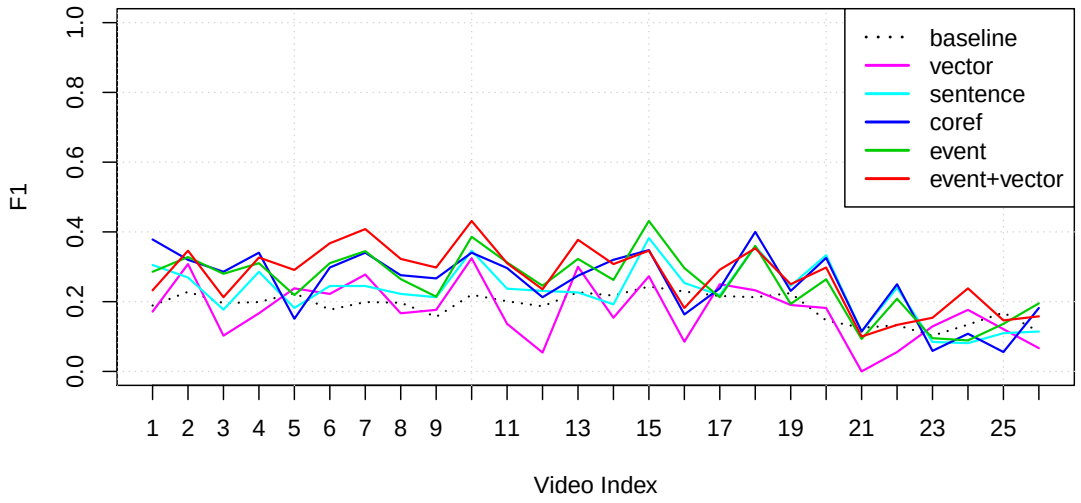
[2] `http://youtu.be/MhBvvq6eHLc`

Gold G_{10}	(angry, 0.19)(scared, 0.14)(excited, 0.08)(frantic, 0.06)(fearful, 0.06), (energize, 0.03)(guilty, 0.03)(hopeless, 0.03)(panic, 0.03)(protective, 0.03) (vigilant, 0.03)(eccentric, 0.03)(frightened, 0.03)(concerned, 0.03) (determine, 0.03)(surprised, 0.03)(upset, 0.03)(crazy, 0.03) (aggressive, 0.03)(startled, 0.03)(desperate, 0.03)(focus, 0.03)
vector	(**frantic**, 0.07)(**crazy**, 0.07)(**aggressive**, 0.07)(hurry, 0.07) (**surprised**, 0.07)(bored, 0.07)(fun, 0.07)(worthless, 0.07) (**desperate**, 0.07)(furious, 0.07)(weird, 0.07)(jealous, 0.07)(nervous, 0.07) (giddy, 0.06)(**scared**, 0.06)
sentence	(**angry**, 0.11)(relax, 0.09)(calm, 0.09)(**desperate**, 0.08)(serious, 0.06) (**focus**, 0.06)(happy, 0.05)(sad, 0.04)(miserable, 0.04)(pleased, 0.04) (afraid, 0.04)(weary, 0.03)(motivated, 0.03)(energetic, 0.03)(eager, 0.02) (**concerned**, 0.02)(**determine**, 0.02)(**upset**, 0.02)(violent, 0.02) (romantic, 0.02)(**crazy**, 0.01)(**guilty**, 0.01)(reluctant, 0.01) (**aggressive**, 0.01)(cautious, 0.01)(unhappy, 0.01)(amuse, 0.01) (interested, 0.01)(worried, 0.01)(welcome, 0.01)
coref	(calm, 0.23)(happy, 0.06)(serious, 0.06)(**determine**, 0.05)(**focus**, 0.05) (**guilty**, 0.05)(**aggressive**, 0.04)(**angry**, 0.04)(afraid, 0.03)(violent, 0.03) (**upset**, 0.03)(unhappy, 0.03)(exhaust, 0.03)(**crazy**, 0.03)(interested, 0.03) (tired, 0.03)(sad, 0.02)(alert, 0.02)(furious, 0.02)(welcome, 0.02) (disappointed, 0.02)(competitive, 0.02)(mad, 0.02)(**desperate**, 0.02) (romantic, 0.02)
event+vector	(happy, 0.08)(**crazy**, 0.06)(**upset**, 0.04)(afraid, 0.04)(**determine**, 0.04) (tired, 0.04)(**frantic**, 0.04)(**aggressive**, 0.03)(hurry, 0.03) (**surprised**, 0.03)(bored, 0.03)(fun, 0.03)(worthless, 0.03) (**desperate**, 0.03)(furious, 0.03)(weird, 0.03)(jealous, 0.03)(nervous, 0.03) (giddy, 0.03)(**scared**, 0.03)(**angry**, 0.03)(**focus**, 0.03)(impressed, 0.03) (violent, 0.03)(**guilty**, 0.03)(disappointed, 0.03)(relax, 0.02) (interested, 0.02)(mad, 0.02)

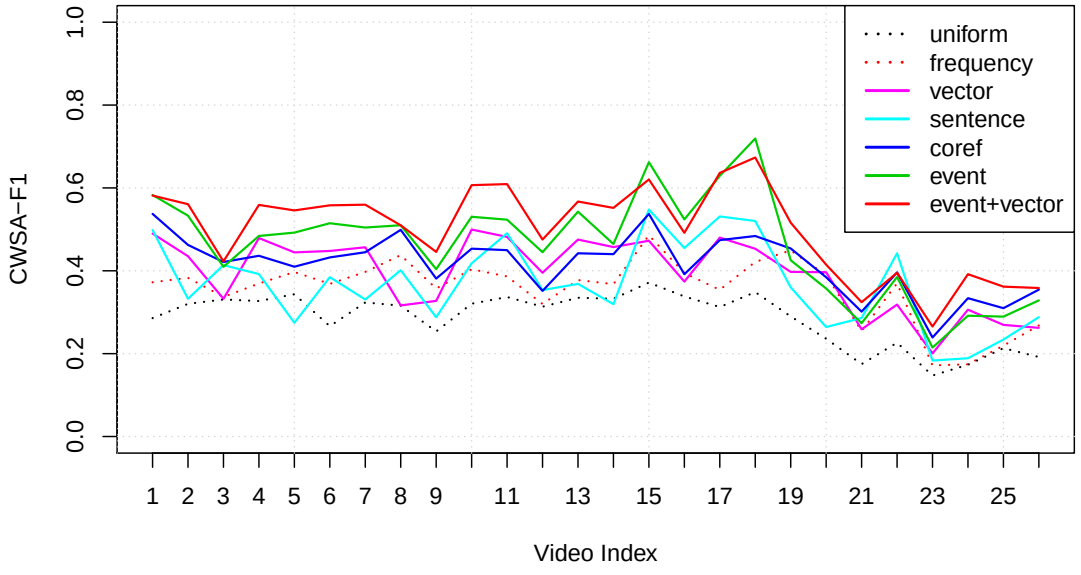
Table 7.2: Example output for the *chase10* video (i.e., the video at index 10 from the *chase* dataset). Each distribution is represented as a list of elements and their associated weights. The gold standard distribution G_{10} is generated by aggregating mental state annotations from 10 different MTurk workers. The response distributions are given by various different neighborhood information extraction models. Bold font in a response distribution indicates an exact match with an element from G_{10}.

Gold G_{23}	(happy, 0.31)(playful, 0.27)(excited, 0.06)(joyful, 0.06)(adrenalin, 0.04) (fun, 0.04)(laughable, 0.04)(entice, 0.02)(wary, 0.02)(enthusiastic, 0.02) (lead, 0.02)(ecstatic, 0.02)(enjoyable, 0.02)(intrigued, 0.02) (delighted, 0.02)(aggressive, 0.02)
vector	(frantic, 0.07)(crazy, 0.07)(**aggressive**, 0.07)(surprised, 0.07)(hurry, 0.07) (bored, 0.07)(**fun**, 0.07)(desperate, 0.07)(furious, 0.07)(giddy, 0.07) (worthless, 0.07)(nervous, 0.07)(weird, 0.07)(scared, 0.06)(jealous, 0.06)
sentence	(determine, 0.07)(angry, 0.06)(focus, 0.05)(**aggressive**, 0.04) (serious, 0.04)(worried, 0.04)(**happy**, 0.03)(reluctant, 0.03) (desperate, 0.03)(violent, 0.03)(interested, 0.03)(distract, 0.03) (nervous, 0.03)(alert, 0.02)(eager, 0.02)(upset, 0.02)(welcome, 0.02) (friendly, 0.02)(romantic, 0.02)(frustrated, 0.02)(exhaust, 0.02) (frightened, 0.02)(crazy, 0.02)(terrify, 0.02)(afraid, 0.01) (competitive, 0.01)(curious, 0.01)(cautious, 0.01)(tired, 0.01)(frantic, 0.01) (bitter, 0.01)(furious, 0.01)(content, 0.01)(tense, 0.01)(irate, 0.01) (peaceful, 0.01)(weary, 0.01)(threatening, 0.01)(disappointed, 0.01) (guilty, 0.01)(relax, 0.01)(calm, 0.01)(**wary**, 0.01)(drunken, 0.01) (enraged, 0.01)(optimistic, 0.01)(hopeful, 0.01)(mad, 0.01)(hostile, 0.01) (rage, 0.01)(surprised, 0.01)(anxious, 0.01)(ambitious, 0.01)(lively, 0.01) (concerned, 0.01)
coref	(optimistic, 0.15)(protective, 0.13)(**happy**, 0.10)(guilty, 0.08) (determine, 0.07)(mad, 0.07)(focus, 0.05)(serious, 0.05)(curious, 0.04) (angry, 0.03)(scared, 0.03)(rage, 0.03)(afraid, 0.03)(welcome, 0.03) (crazy, 0.03)(nervous, 0.03)(discourage, 0.02)(violent, 0.02)
event+vector	(optimistic, 0.10)(scared, 0.09)(**happy**, 0.07)(nervous, 0.07) (guilty, 0.06)(determine, 0.05)(afraid, 0.04)(frantic, 0.04)(crazy, 0.03) (**aggressive**, 0.03)(surprised, 0.03)(hurry, 0.03)(bored, 0.03)(**fun**, 0.03) (desperate, 0.03)(furious, 0.03)(focus, 0.03)(giddy, 0.03)(worthless, 0.03) (weird, 0.03)(jealous, 0.03)(rage, 0.03)(mad, 0.02)

Table 7.3: Example output for the *chase23* video (i.e., the video at index 23 from the *chase* dataset). Each distribution is represented as a list of elements and their associated weights. The gold standard distribution G_{23} is generated by aggregating mental state annotations from 10 different MTurk workers. The response distributions are given by various different neighborhood information extraction models. Bold font in a response distribution indicates an exact match with an element from G_{23}.



(a) F_1 scores for *chase*.



(b) CWSA-F_1 scores for *chase*.

Figure 7.2: The performance scores for each neighborhood information extraction model across all 26 videos from the *chase* dataset.

including the baselines, than others. For example, there is a noticeable drop in performance across all models starting with the video at index 19. Indeed this drop corresponds with the start of sports and children related videos, as shown in Table 7.4. Videos at indices $19, 20, 21$, and 22 all involve sport-related scenes, while videos from index 21 to 26 all involve children (videos 21 and 22 involve children playing sports). Thus, it would seem that the category of the video may have some correlation with how challenging it is to our models. We will further explore this hypothesis in Section 7.2.7.

Category	Video Indices
children	$3, 21, 22, 23, 24, 25, 26$
police	$1, 15, 16, 17, 18$
sports	$19, 20, 21, 22$
others	$2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14$

Table 7.4: Different categories of chase videos. The *children* and *police* categories refer to any chase videos involving a child or a policeman participant, respectively. The *sports* category describes all sport-related videos. Videos not belonging to one of the first three categories are labelled as *others*, such as videos involving normal civilian adults. A video may fall into multiple categories.

7.2.2 Related Work Comparisons

Since the proposed task and the corresponding performance measure are novel, we cannot directly compare our results to previous research. Fortunately, we can use our *sentence* model as a strong baseline to indirectly gauge the performance of our other models, because *sentence* employs the same collocation-search methodologies commonly found in other state-of-the-art information extraction technologies. For instance, Marneffe et al. (2010) learned the meanings of scalar adjectives by extracting scalars, such as children's ages, that were commonly collocated with landmark phrases, such as "little kids", in web documents. In similar fashions, our *sentence* model seeks mental state terms that are commonly found near keywords cued by the video context, such as "chase" and "policeman", in the English Gigaword (5th Ed) corpus.

Thus, our *sentence* model can be viewed as a strong baseline that is representative of common state-of-the-art technologies. Table 7.1 shows that our best ensemble model outperforms this baseline (the *sentence* model) by over 35%, which reassures that we are, in fact, getting good results.

7.2.3 Ensemble Models

As shown in Table 7.1, the ensemble method *event+vector* significantly outperformed each of its individual model. In general, we found that any ensemble model formed by combining a deleted interpolation model that operates on text, with the *vector* model, which operates in a latent conceptual space, will outperform each of its individual component by a significant amount, as shown in Table 7.5.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
vector	.226	.145	.175	.399	.392	.393
sentence	.194	.293	.227	.366	.376	.368
sentence+vector	.192	.377	.250	.434	.444	.438
coref	.264	.251	.253	.382	.461	.416
coref+vector	.231	.337	.271	.448	.481	.462
event	.231	.303	.256	.446	.488	.463
event+vector	.259	.296	.274	.488	.517	.500

Table 7.5: The average evaluation performance across 26 chase videos for different ensemble combinations.

These empirical results show that the information gained from operating on text and operating in a latent conceptual space are highly complementary, and a combination of both is needed to attain the best performance. This finding suggests that we can independently improve our text driven models and our latent space model, and any improvements made within each model will also improve the corresponding ensemble model.

7.2.4 Coreference Resolution

Table 7.1 also shows that the *event* model performs considerably better than *coref*. This performance gain emphasizes the importance of focusing on mental state terms associated with event participants rather than considering all mental state labels collocated in the same neighborhood as an actor or action verb. We also found that the *coref* model improves over the *sentence* model, which validates the needs for expanding the neighborhood area beyond a single sentence. In this section, we explore the effectiveness of coreference resolution in expanding the neighborhood area. In particular, we would like to know if a naïve windowing approach can do just as well, or better.

Model	CWSA-F_1	Versus *coref*	p-value
win-0	0.388682	−0.027512	0.0067
win-1	0.415328	−0.000866	0.4629
win-2	0.399777	−0.016417	0.0311
win-3	0.392832	−0.023362	0.0029

Table 7.6: The average CWSA-F_1 scores for the windowing model with different window parameters. The *coref* model outperformed all tested cases, though the difference is not significant for *win-1*. The p-values, based on the average differences, were obtained using one-tailed nonparametric bootstrap resampling with $10,000$ iterations.

Table 7.6 compares the performance of *coref* versus the windowing method under several different window sizes. The *coref* model outperformed the simple windowing method under every tested configuration. However, the improvement over windowing with $n = 1$ is not significant. This is largely explained by the fact that if a text is cohesive and coherent, then successive sentences are likely to refer to similar concepts and topics. Thus, the sentences immediately surrounding a target sentence are very likely to be relevant, while sentences further away are not. Also, since newswire articles tend to be short and contain few sentences, the neighborhood defined by *win-1* is very likely to be similar to that of *coref*. We see evidence of this similarity between *coref* and *win-1* in Table 7.7. Both models extracted roughly the same number of sentences related to chase events from the entire corpus. As the

window size grows, however, the total number of sentences extracted increased due to the inclusion of irrelevant sentences and the naïve windowing method became less dependable.

Model	Total Sentences
win-0	90,399
win-1	260,423
coref	281,666
win-2	418,827
win-3	567,706

Table 7.7: The total number of sentences from the corpus that were deemed to be related to chase events under different models.

In general, *coref* does not do worse than a simple windowing method, while also having the bonus advantage of providing references to the participants of interest for downstream processes. This functionality alone warrants the added complexity of coreference resolution. It is, however, reassuring to know that we cannot do any better with a simpler method.

7.2.5 Semantic Role Labeling

Similar to coreference resolution, we would also like to know whether or not our model can benefit from semantic role labeling (SRL). As explained in Section 4.2.3, the use of an external SRL tool (SwiRL) comes with extra overheads, one of which is CPU time. Thus it is worth experimenting on a small sample of documents from the Gigaword corpus first. For that, we took a sample of 82 documents from the corpus, each containing at least one occurrence of the word "chase". The primary question to answer is whether or not it is worth running a full experiment on all documents with SwiRL.

Table 7.8 shows some summary statistics from our initial experiment. Out of 82 total documents, our model identified 59 unique instances of the verb "chase" for which we can expect to extract a subject and object participant. Using patterns on top of the shallow syntactic dependencies of CoreNLP, we were able to extract the

subject-verb-object (SVO) triplet for 29 out of 59 chases, yielding a 49% find rate. Of the remaining 30 instances where we failed to find all participants using CoreNLP dependencies, SwiRL was able to identify the complete SVO triplet in 20 of those cases – a 67% identification rate in cases where CoreNLP failed. Moreover, SwiRL on its own identified the SVO triplet for 48 out of the 59 instances (81% success rate overall). We note that a successful identification is defined as successfully binding each SVO component to a syntactic clause. These numbers do not reflect the accuracy of the actual identifications. However, through visual inspection, it seems that both methods yielded roughly similar accuracy rates.

Description	Count
Number of documents parsed with CoreNLP	82
Number of documents parsed with SwiRL	82
Number of relevant *chase* instances	59
Number of times CoreNLP found both participants	29
Number of times SwiRL found both participants	48
Number of times CoreNLP failed to find at least one participant and SwiRL found both	20

Table 7.8: Summary statistics regarding the number of times both participants of a chase were identified by either CoreNLP or SwiRL. The corpus is compiled of 82 documents containing the word "chase" from the Gigaword corpus.

From the initial experiment, it appears that SwiRL does offer some additional advantages to our model, which would warrant a full experiment on the entire corpus. However, due to limited resources, we could only process about 41K of the 141K documents that contain the word "chase" from the Gigaword corpus. For the remaining 100K documents, we relied solely on the shallow dependency parser of CoreNLP. Table 7.9 shows the summary statistics of the full experiment. Even with only 41K parsed documents, SwiRL contributed to 10K out of 51K different instances where the use of CoreNLP alone failed to identify all participants.

Table 7.10 shows the performance of our model with SwiRL integration. Surprisingly, the addition of SRL via SwiRL actually decreased the CWSA-F_1 score of our *event* model by a statistically significant amount. Similarly, the ensemble method

Description	Count
Number of documents parsed with CoreNLP	141,875
Number of documents parsed with SwiRL	41,725
Number of relevant *chase* instances	91,167
Number of times CoreNLP found both participants	40,145
Number of times SwiRL found both participants	22,001
Number of times CoreNLP failed to find at least one participant and SwiRL found both	10,056

Table 7.9: Summary statistics regarding the number of times both participants of a chase were identified by either CoreNLP or SwiRL. The corpus is compiled of all documents from the Gigaword corpus that contain the word "chase".

event-srl+vector only achieved an average CWSA-F_1 score of 0.496 – a statistically significant drop from the performance of *event+vector* in Table 7.1. As before, all significance tests were one-tailed and were based on nonparametric bootstrap resampling with 10,000 iterations.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
event	.231	.303	.256	.446	.488	.463
event-srl	.232	.306	.259	.437	.487	.458

Table 7.10: The average evaluation performance across 26 different chase videos for the *event* and *event-srl* models.

These empirical results would suggest that SRL is not very useful to our problem, which is quite puzzling as we expected a SRL system like SwiRL to be more helpful at identifying the event participants, especially in hard cases where syntactic information alone is not enough. There are many possible explanations as to why this happened. The first and most obvious explanation is that we did not process enough of the documents. Though, we feel that processing close to 30% of the total number of documents should be more than enough to expose any differences. In fact, statistics from Table 7.9 do show that SwiRL did contribute to at least 10K out of 91K total chase instances. A better reason could be because of the way we integrated SwiRL into our model. We only used the output of SwiRL if the shallow dependency parser of CoreNLP fails to detect both the subject and object of an

event. Thus, we effectively only rely on SRL in "hard" cases. It is conceivable that for complex sentences where CoreNLP fails, even a SRL system, like SwiRL, would have a hard time accurately identifying the participants. Therefore by only using SRL for hard cases, we overlook the common "normal" cases where the output of SwiRL might be more accurate and useful than CoreNLP. A trivial solution then would be to use the output of SwiRL for all cases. If we did this, then the 41K parsed documents from SwiRL would have contributed to identifying both participants in 22K of the 91K chase instances (Table 7.9), which is promising. Unfortunately, this is not viable as it is not straight forward to integrate the output of SwiRL into our model, which could be another reason why the inclusion of SRL did not help. Since SwiRL and CoreNLP depend on different parse trees, a syntactic clause identified by SwiRL does not always map directly to one from CoreNLP. This mapping is necessary as the downstream coreference resolution step uses the CoreNLP parse tree. Currently, we map syntactic clauses if they share at least some predefined amount of terms. It is possible that this simple mapping mechanism is introducing errors into the model that could negatively affect performance.

7.2.6 Vector-Space Combination Operators

In this section, we explore two different ways for combining vector representations of search terms from context tuples in order to generate context vectors. In particular, we are specifically interested in combination operators that are both associative and communicative – the goal is for both (*chase*, *policeman*) and (*policeman*, *chase*) to map to the same representative vector.

Results from Table 7.11 clearly show that the addition operator works best, which is consistent with the findings presented by Mikolov et al. (2013a). In their work, relationships between words were found by manipulating the offsets of vectors (via addition and subtraction).

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
vector	.226	.145	.175	.399	.392	.393
vector-mult	.162	.103	.125	.333	.343	.335

Table 7.11: The average evaluation performance across 26 different chase videos for the *vector* and *vector-mult* models. The *vector-mult* model is the *vector* model with the modification that the vector representations of search terms (from the context tuples) are *multiplied* together, as opposed to being added, to generate the context vector.

7.2.7 Underlying Data Biases

In Table 7.12, we show the performance results of our models based on different types of chase scenarios happening in the videos. These scenarios are equivalent to the categories presented in Table 7.4. The average uniform baseline scores for chase videos involving children and sporting events are lower than those for chases involving the police and others. This is also evidence in the plot shown in Figure 7.2. This trend strongly suggests that our seed set of 160 mental state labels (see Appendix A) is biased towards the latter types of events, and is not as fit to describe chases involving children.

Category	Unif. Baseline	*event+vector*	Change
children	0.2082	0.3599	+0.1517
police	**0.3313**	**0.6006**	**+0.2693**
sports	0.2318	0.4126	+0.1808
others	0.3157	0.5457	+0.2300

Table 7.12: The average CWSA-F_1 scores for the ensemble model *event+vector* are shown in comparison to the uniform baseline performance, categorized by video scenarios.

Moreover, videos involving police officers show the biggest improvement in the CWSA-F_1 scores, on average, over the uniform baseline method (+0.2693), whereas videos involving children received the lowest gain (+0.1517). We believe this is the bias effect of the underlying data source. The English Gigaword text corpus is a comprehensive archive of newswire text, and thus is heavily biased towards

high-speed and violent chases involving the police. The Gigaword corpus is not the place to find descriptions of playful chase scenes involving children. Sport-related chases, which are also news-worthy, also have a higher gain than children's videos on average.

Thus, biases introduced from different input sources (e.g., the underlying corpus and the seed set of mental states labels) can have a large effect on the performance of our system. In this dataset, biases from both sources favor violent police chases.

7.2.8 Actor-Specific Mental State Identification

All of our previous experiments have focused on the task of *activity-wide* mental state identification. That is, we have been generating a single distribution over mental state labels for each video to describe the entire activity in general. We have not distinguished between the mental states of the chaser versus the person being chased (a.k.a., the chasee), while in reality these participants may be in very different states of mind. Since our *event* model is fully capable of making this distinction, it allows us to next examine the task of *actor-specific* mental state identification. In general, this is a more challenging problem as it demands more specific identification.

We follow the same experimental protocol of Figure 7.1. However, we modify the generation of G by only collecting mental state annotations regarding a specific participant. Similarly for the system, we only pass in detections pertaining to one of the actors, and follows only the corresponding coreference chain during extraction. The resulting distributions are compared as normal.

Tables 7.13 and 7.14 show the average performance of our models for the subject-specific and the object-specific identification tasks, respectively. Once again, the ensemble model *event+vector* gives the best F_1 and CWSA-F_1 scores. In both tasks, our best model improves over the uniform baseline CWSA-F_1 scores by over 100%. It also improves the standard F_1 scores by a considerable margin. As with the activity-wide experiments, all improvements of the ensemble model *event+vector* over the baselines are significant.

As suspected, the task of identifying actor-specific mental states is more chal-

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
uniform	.067	**.796**	.124	.196	.195	.195
frequency	.067	**.796**	.124	.290	.282	.285
vector	.118	.129	.122	.351	.338	.342
event	.164	.362	.220	.353	.340	.340
event+vector	**.171**	.338	**.224**	**.395**	**.400**	**.396**

Table 7.13: The average evaluation performance for identifying the mental state distributions of the **subject** across 26 different chase videos. Bold font indicates the best score in a given column.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
uniform	.064	**.760**	.117	.191	.181	.185
frequency	.064	**.760**	.117	.246	.229	.235
vector	.136	.221	.167	.358	.374	.363
event	.135	.208	.162	.383	.407	.391
event+vector	**.148**	.325	**.202**	**.389**	**.415**	**.399**

Table 7.14: The average evaluation performance for identifying the mental state distributions of the **object** across 26 different chase videos. Bold font indicates the best score in a given column.

lenging than the activity-wide equivalent. The baseline scores for both the subject-specific and object-specific tasks are much lower than the baseline scores for the activity-wide task in Table 7.1. The added difficulty is also reflected in the overall lower scores across all models. Nonetheless, the results shown in Tables 7.13 and 7.14 are very encouraging as they prove that our models can be successful at making actor-specific mental state identifications.

7.3 Mental State Identification in Hug Videos

To show the generality of our models, we also evaluated our system on the *hug* dataset. Hug videos differ from chase videos in several ways. The differences in mental state distributions between a hug involving two friends versus a hug involving two siblings are much more subtle. Moreover, hugs are often mutually given. In

most cases, it is hard to discern between the hugger versus the person being hugged. For this reason, we opted not to experiment with the actor-specific mental state identification task for the *hug* dataset.

We report the average performance of our models, measured in terms of the F_1 score and the CWSA-F_1 score, across all 45 videos from the *hug* dataset in Table 7.15.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
uniform	.073	**.487**	.126	.226	.210	.217
frequency	.073	**.487**	.126	.254	.225	.238
sentence	.192	.253	.213	.388	.378	.382
vector	**.251**	.218	.230	.347	.334	.339
event	.156	.230	.183	.406	.384	.394
event+vector	.231	.255	**.239**	**.443**	**.437**	**.439**

Table 7.15: The average evaluation performance across 45 different hug videos are shown against two different baseline methods for our neighborhood information extraction models. Bold font indicates the best score in a given column.

Under the classical F_1 measure, the *vector* model scored highest on precision, while the ensemble method *event+vector* did best on F_1. As before, the baseline methods gave the best recall, but it is at a significant drop from the 0.750 value seen in the *chase* dataset. This means our seed set of mental state labels is more unfit to describe hug videos than chase videos. Moreover, we also see less of an improvement between the frequency baseline and the uniform-probability baseline (9% improvement versus 24% from the *chase* dataset). This suggests that the inherent occurrence frequency of mental state terms in the English Gigaword corpus is also less compatible with hug videos.

As in the *chase* dataset, the ensemble model *event+vector* performed best in terms of CWSA-F_1. The average CWSA-F_1 score of this ensemble model improves upon the uniform baseline score by over 100%, outperforms the stronger frequency baseline by over 80%, and raises performance over each of its individual component in all measured F_1 scores. Overall, performance in the *hug* dataset is very consistent with the performance achieved in the *chase* dataset. That is, ensemble models

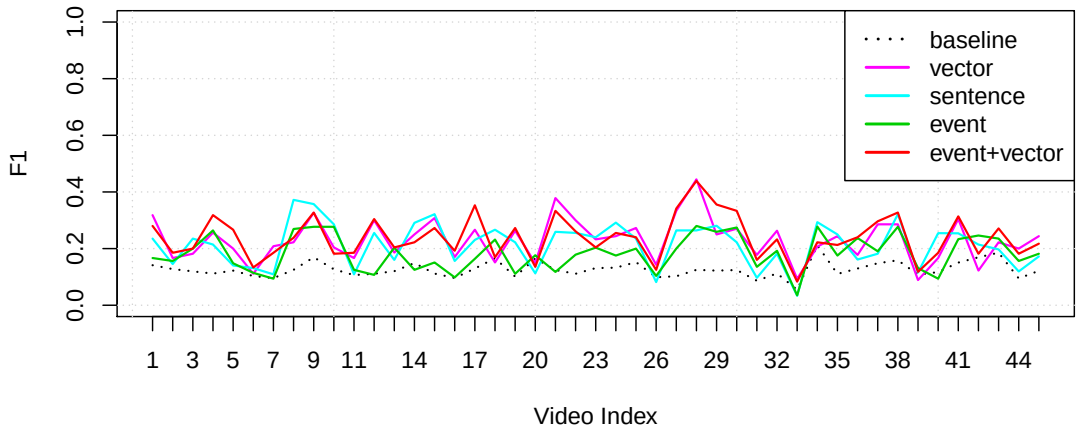
still outperform their respective components, the inclusion of NLP techniques to extract event-centric mental state labels remains helpful, and difficult videos are still consistently challenging for all models (see Figure 7.3).

In general, the scores we get on hug videos tend to be lower, on average, than our performance on the chase videos. This is due to several contributing factors. We already discussed the poor fit of our initial seed set to the *hug* dataset. Another reason has to do with the biases of the underlying text corpus. The Gigaword corpus generally contains less information about hugs than it does for chases. There are 141K different documents containing at least one occurrence of the word "chase" in the Gigaword corpus and only 41K unique documents containing the word "hug". This means we have much less data to work with, which makes it harder to get good frequency estimates of n-grams in our models. Moreover, happy and affectionate hug scenarios are also not very common in newswire articles. Despite all these challenges, however, our system still performed quite well, especially when compared against the baselines. These empirical results help to support our claim of a general solution that works well across different activity types.

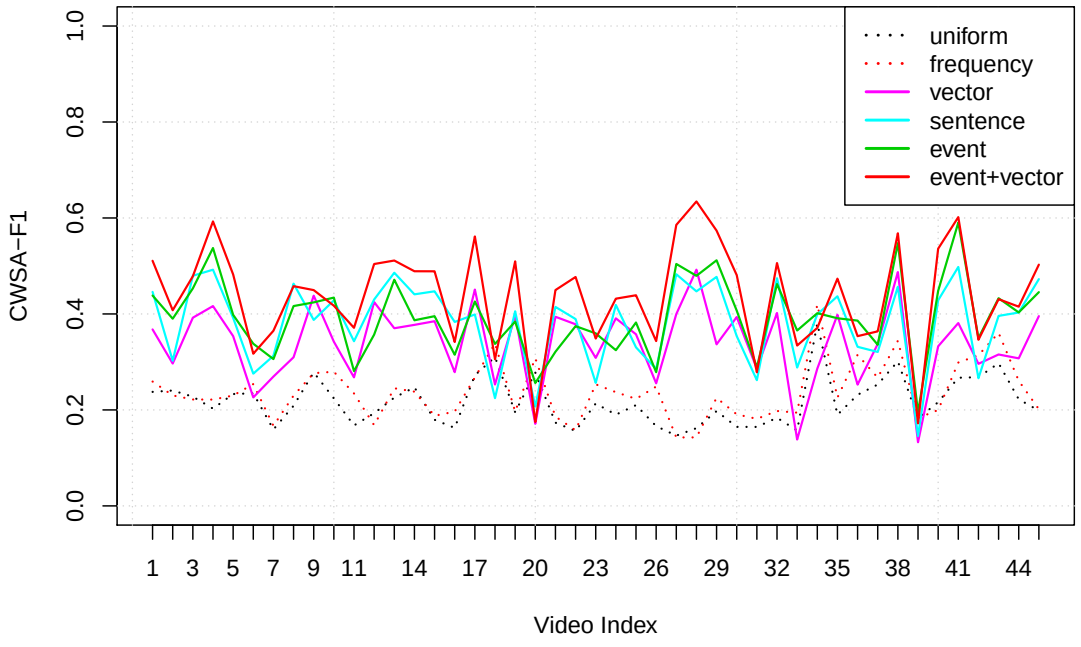
7.4 Artificially Induced Noise Experiment

In previous experiments, we have relied on ground-truth detections that came from highly accurate human annotations. In reality, however, vision-based detection algorithms are much more susceptible to noise. Thus, it is important that our information extraction models can tolerate noisy detection data.

Ideally, we would like to integrate our models with actual detection algorithms in a complete end-to-end system in order to test their robustness. Unfortunately, the integration of a fully functional computer vision detection system is a daunting task – one that we currently do not have the resources to complete. So as a first approximation, we instead simulate the noisy nature of detectors by introducing artificial noise into our annotated ground-truth detections. We then give the noisy detections to our extraction models to test their performance under noisy conditions.



(a) F_1 scores for *hug*.



(b) CWSA-F_1 scores for *hug*.

Figure 7.3: The performance scores for each neighborhood information extraction model across all 45 videos from the *hug* dataset.

We refer to this as the *artificial noise* experiment.

In this experiment, we degrade the quality of ground-truth detections for videos in the *chase* dataset by artificially adding *false positive* and *false negative* errors into the ground-truth. The rates at which we introduce the errors are based on published results in computer vision literature. This allows us to realistically simulate the performance of current state-of-the-art vision-based detectors.

Given a set of ground-truth detections for a particular scene, we add a false positive error to the scene by inserting a new erroneous detection into the set. To introduce a false negative, we remove an existing ground-truth detection from the set. We take care to never remove the actual activity detection (e.g. *chase* or *hug*). In our opinions, the activity detection is crucial to the scene because it anchors the search. It is not possible to retrieve latent information relevant to an activity if we do not know what the activity is. Thus, in this particular experiment, the performance robustness of our system is only guaranteed if the activity is recognized.

This means we are always guaranteed to have at least one detection in the set of context detections (i.e., the activity detection). It is possible, however, that all other detections are removed due to false negative errors, which would leave the extraction models with very little context to work with. This is especially worrisome if we rely only on the activity and actor-type detections, as is the case in all of our previous experiments. Since there are only a few actor-type detectors, the chance of all actor-type detections being removed is quite large. In a realistic setting, the system would have many more detector types to fall back on in cases where all actor-type detectors fail to trigger. As such, we decided to include location detections in this experiment in addition to actor-type.

7.4.1 Adding Location Detections

As mentioned in Section 3.2, our annotation experiment also collected annotation for 24 different categories of location. In the 26 chase videos, however, only 19 different location types were identified by MTurk workers (see Table 7.16). These location detections were derived from known categories used in natural scenes and settings

detection research (Lazebnik et al., 2006; Quattoni and Torralba, 2009; Xiao et al., 2010).

Location Types			
alley	arena	backyard	beach
(open) country	field	forest	highway
home	indoor	industrial	office
outdoor	park	parking-lot	playground
store	street	suburb	

Table 7.16: The 19 different location types used in the annotation of chase videos.

As with actor-type detections, we further expand the pool of each location tag by retrieving the synonym set (synset) of its most frequent sense (i.e., sense #1) from WordNet. Once again, we ignore multi-words synonyms. The addition of location detections allows us to formulate query tuples following new query patterns, such as the (*activity*, *location*) (AL) and the (*activity*, *actor-type*, *location*) (AAL) pattern. We report the average performance of our ensemble model *event+vector*, which has consistently proven to be our best model, for different types of query patterns across all 26 videos from the *chase* dataset in Table 7.17.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
AA	.259	.296	.274	.488	.517	.500
AL	.202	.390	.261	.414	.421	.416
AAL	.229	.336	.270	.469	.514	.489

Table 7.17: The average evaluation performance across 26 different chase videos for the ensemble *event+vector* model using different query patterns. The (*activity*, *actor-type*) (AA) query pattern can be viewed as a baseline to access the usefulness of location detections to chase videos.

Using the performance of the ensemble model based on the (*activity*, *actor-type*) (AA) query pattern as a baseline, we can access the usefulness of location detections to the task of mental state identification in chase videos. We see a significant drop in performance, in all measured scores, between the AA baseline and the AL pattern. This strongly suggests that when it comes to mental state

identification, information regarding the types of actors involved in the chase is much informative than information about the location of the chase. This supports our intuitive understanding of the problem. For example, in a police chase scene, the involvement of a police officer is more predictive of an angry suspect than the fact that the chase took place in the street, or the park.

Furthermore, we also expected the combination of all detection types in the AAL pattern to outperform the AA baseline, but this is not the case. Adding location knowledge on top of activity and actor-type information actually decreased extraction performance. This is quite counter-intuitive. However, we do not think this is conclusive of the fact that the addition of location information is not useful. It is likely that our current extraction models are not making effective use of the new location information. For instance, several location tags do not have corresponding synsets in WordNet (e.g., *backyard*, *beach*, *indoor*, *street*); and for those that do, the most frequent sense sometimes portrays the incorrect meaning (see Table 7.18). Moreover, it is possible that location search terms are more susceptible to noise, and more sophisticated extraction patterns are needed to properly accommodate for these terms in our extraction process.

Location	Most Frequent Sense Synset
arena	sphere, domain, area, orbit, field, arena
(open) country	state, nation, country, land, commonwealth

Table 7.18: Example location tags with the incorrect most frequent sense synset from WordNet.

Although the inclusion of location detections did not improve performance in mental state identification as expected, it did well enough to support our artificial noise experiment, which is the main focus. As such, we defer the investigation regarding the effective use of location information to future work (see Chapter 9).

7.4.2 Introducing Artificial Noise to Detections

Given a set of ground-truth detections for a scene, we artificially introduce noise into the detections by adding false positive and false negative errors based on published performance rates of state-of-the-art detectors. We base the performances of actor-type detections on performance rates taken from the Visual Object Classes (VOC) 2012 detection challenge (Everingham et al., 2014). For location detections, we use the performance rates from Lazebnik et al. (2006), Quattoni and Torralba (2009), and Xiao et al. (2010).

In computer vision literature, detector performance is often given only by the average precision rate. Thus, given a list of detectors, each with an associated precision rate, we introduce errors into the ground-truth detection set $S_{\text{ground-truth}}$ for some video V as follows:

For each detector d with precision rate p from the list of all available detectors:

- If $d \in S_{\text{ground-truth}}$, then with probability $(1 - p)$ remove d from $S_{\text{ground-truth}}$.
- If $d \notin S_{\text{ground-truth}}$, then with probability $(1 - p)$ add d to $S_{\text{ground-truth}}$.

As one might notice, in the above noise-induction algorithm, we treat our simulated detection system as a collection of binary detectors, as opposed to a single large multi-class detector. In our experience, this is typical of most large-scale practical vision-based detection systems. Binary detectors are useful when the classes to be recognized are not mutually exclusive, which is often the case in real-world scenarios. For instance, in the Mind's Eye[3] challenge presented by DARPA, teams were expected to recognize 48 different verbs from videos. The solution submitted by almost every team, including our own from the University of Arizona, made use of many binary detectors, or many small multi-class detectors. Likewise, Everingham

[3] http://www.darpa.mil/Our_Work/I2O/Programs/Minds_Eye.aspx

et al. (2014) took the best submissions from the VOC2012 challenge and constructed an ensemble method by building a binary classifier for each VOC class. This *super* method outperformed all the state-of-the-art submissions at the time.

Once the set of detections have been artificially degraded with errors, we give the error-prone context to the extraction models and carry out the same mental state identification task as before. Due to the random nature of the noise-induction, a video may experience no added errors, or it may be infused with numerous errors. Hence, the results for a video from any single run of the system may differ. For example, given some arbitrary video with ground-truth detections containing both actor-type and location information, the algorithm may not introduce any errors, which would result in an AAL query pattern, or it may remove all detections of a particular type, forcing the extraction models to run the AA or AL query pattern. On rare occasions, we may only have the (*activity*) (A) query pattern. Thus, it is necessary to run several trials and average the performance across all trials to get a more informative evaluation.

7.4.3 Noise Experiment Results

Table 7.19 shows the average performance of our ensemble model *event+vector* across 20 different noise experiment trials, where each trial was evaluated over 26 different chase videos with artificially induced noise.

	F_1			CWSA-F_1		
	p	r	f_1	p	r	f_1
event+vector	.231	.255	.239	.443	.437	.439

Table 7.19: The average evaluation performance across 20 different noise experiment trials for the ensemble *event+vector* model. Each trial was evaluated over 26 different chase videos with artificially induced noise.

Even under noisy conditions, our ensemble model still did quite well. It still significantly outperformed the baselines from Table 7.1. As expected, however, the CWSA-F_1 score of .439 is a drop off from the .500 mark achieved using error-free ground-truth activity and actor-type detections.

In Table 7.20, we present some additional summary statistics regarding the amount of errors introduced per movie in this experiment.

Stat Type	No. Occurrences per Movie
True Positives	3.17
False Negatives	2.83
True Negatives	9.75
False Positives	7.25

Table 7.20: Summary statistics regarding the number of errors introduced per movie, averaged across 520 movies (i.e., 20 trials times 26 movies per trial).

On average, we removed 2.83 correct detection labels and added 7.25 incorrect detections to each movie. As is the case with most vision-based detection systems, false positive errors are much more prominent than false negative errors. It is also interesting to note that on average, our extraction model has to work with more incorrect context, 7.25 false positive errors, than correct context, 3.17 true positive detections. Despite this fact, our model still performed quite well. These results prove that our proposed information extraction model is very robust and can effectively tolerate noisy detection data.

7.5 System Scalability

While running our experiments, we also benchmarked the runtime of our models to track their scalability. We found that by caching the necessary data, we can significantly reduce the processing time of all models.

For the deleted interpolation models, we precomputed and cached the relative joint frequencies of all relevant n-grams. In the *chase* dataset, the AA query pattern generated $3,541$ different combinations of n-grams. The *hug* dataset required $3,058$ different combinations. On average, it takes about $3-8$ hours to precompute these frequencies on a standard quad core laptop, depending on the added complexity of the extraction process. The number of possible n-gram combinations can explode with larger query tuples. However, this is not a big concern as we can always

precompute these values offline. Once computed, it is fast to look up these relative frequencies at runtime.

For the *vector* model, we cached the output distribution for each context tuple. We can then quickly retrieve these distributions for back-off linear interpolation. It takes no longer than an hour to pre-compute all relevant distributions, as there aren't as many combinations of context tuples. This allows us to quickly process any sets of detections and yield the corresponding response distributions quickly.

On a standard quad core laptop, each of our models can produce a response distribution for a given set of detections within 2 seconds. This should allow us to efficiently scale to videos of any length, given that we have the resources to precompute the necessary data offline.

CHAPTER 8

CONCLUSIONS

In this work, we introduced the novel task of identifying latent attributes in video scenes given some contextual information about the scenes. We showed that these attributes can be identified with a simple approach: Use the explicit visual features of the videos as contextual information to query text corpora, and from the resulting texts extract attributes that are latent in the video.

We grounded our research to the problem of identifying the mental state labels of participant actors in video scenes, and presented several largely unsupervised methods for accomplishing this task. Our latent attribute extraction models take a list of video detections along with an initial set of mental state labels as input, and output a distribution over mental state terms for each video. The first model we proposed is a lexical semantic model that extracts attributes that are highly similar to the context of the scene in a latent, conceptual space generated by a recurrent neural network language model. The next is an information retrieval model that identifies attributes that commonly appear in sentences related to the explicit scene context. Our final model is an event-centric model that focuses on the attributes associated with the participants of relevant event (identified using syntactic patterns and coreference resolution).

To evaluate the performance of our models, we defined a new similarity measure for comparing distributions of mental state labels. The *constrained weighted similarity-aligned* (CWSA) F_1 score can effectively account for both semantic relatedness of the labels and their probabilities in the response and gold standard distributions. Furthermore, our investigation also showed that the CWSA-F_1 score corresponds highly with human judgement in determining the similarity between mental state distributions. Thus, it is the primary performance measure used in our extraction evaluation.

8.1 Findings

Our extraction experiments showed that very little information from videos are needed to produce good results that can significantly outperform simple baseline methods. The models in our experiments only needed information about the activity and actor-type for each video to perform well. Our ensemble model *event+vector* consistently produced the best performance in all experiments. In general, we found that any ensemble model formed by combining a deleted interpolation model that operates on text, with our lexical semantic *vector* model, which operates in a latent conceptual space, will outperform each of its individual component by a significant amount. This is strong evidence showing that the information gained from the two representations are highly complementary.

The results of our experiments also revealed that the use of syntactic patterns and coreference resolution to identify event participants and their relevant attributes can significantly improve the performance of our models. This is an important finding, as it helps validate the effectiveness of NLP techniques in extraction models. The inclusion of semantic role labeling (SRL), on the other hand, appeared to hurt our model more than it helped. Though, we don't think the results are conclusive and more experiments are needed to fully evaluate the effectiveness of SRL to our problem. We delay discussion of this future work to the next chapter (Sec. 9.1).

Using the event-centric model, we showed that our system is capable of identifying mental state descriptions at the actor level. That is, we can yield a different mental state distribution for the subject of the event versus the object. Our experiments showed that the ensemble model *event+vector* also performed well in this harder task. In addition, we found that our models performed consistently across both tested datasets, *chase* and *hug*, which shows the generality of our approach.

Furthermore, results from our artificial noise experiment showed that our ensemble model *event+vector* performed well even with noisy detection data. This is highly encouraging as it strongly suggests that our information extraction model is robust enough to work well with real vision-based detection systems.

Lastly, we showed that the resource intensive components of our models can be done in an offline stage. Once all necessary data have been pre-computed, our end-to-end system can yield an output for a video clip within seconds by simply looking up the cached frequencies and context distributions in order to build a new distribution over mental state labels for the video. Thus, scalability is not an issue at runtime.

8.2 Final Remarks

We believe that the approach presented in this dissertation is not restricted to the proposed problem of mental state label identification. Our work provides a methodology and framework that can be used to identify any latent attributes in a video scene, given some contextual knowledge about the scene. For instance, given that we know some information about the current and past activities of the actors, we could use the same approach to extract from texts latent information regarding their goals, motives, intentions, and beliefs. In fact, the same novel idea can apply across different domains. One could conceivably apply the same method to the problem of medical diagnosis, which might go as follows: Given the symptoms as context, automatically query specialized medical texts to find possible causes that relate to the context symptoms.

As a closing remark, we would like to note that the idea of going to text to find latent information is powerful. We, as humans, do it on a daily basis. Whenever we need to perform an online search, we begin by providing some query terms (or contextual information) to the search engine, and then from the returned web results parse the texts for relevant answers. Our work simply extended this idea to the problem of mental state identification in videos. It is the opinion of this author that this general information-seeking strategy can be applied to many existing problems across all different areas within the field of Artificial Intelligence, and possibly beyond.

CHAPTER 9

FUTURE WORK

As a guide for future efforts, we now outline some important problems and challenges uncovered during the course of our research.

9.1 Model Improvements

We begin by addressing current problems and limitations of our models, and suggest possible improvements for future work. As discussed in Chapter 7, all of our models are affected by the information bias in the underlying data source. Since the English Gigaword corpus is compiled from newswire articles, it heavily favors news-worthy events, such as police chases, and does not contain as much information on daily-life events, such as people hugging or children playing. It would be interesting to see if performance changes significantly when we replace the Gigaword corpus with the ClueWeb09 corpus. The ClueWeb09 corpus contains information from 1 billion different web pages, which would require a large amount of resources to index. The data contained in web pages should hopefully cover more of the daily-life events that the Gigaword corpus lacks. However, perhaps a combination of the two datasets is the best option as the ClueWeb09 corpus may also have its own biases.

In addition to the corpus, our models also use an initial seed set of 160 mental state labels compiled from different mental and emotional state dictionaries. At the moment, this set is treated as a closed set of labels, which means labels not in this set are never considered by the models. This imposes great restrictions on the extraction coverage of our models. A possible alternative is to simultaneously extract new mental state labels as we generate new mental state extraction patterns via a mutual bootstrapping method, similar to that of (Riloff and Jones, 1999). This method would offer two advantages over our current models: (1) The ability to

obtain new mental state labels, and (2) the ability to learn new extraction patterns.

We have shown that our models require limited context from the videos to perform well, namely just the activity and actor-type. We also tried adding location information to chase videos, which did not seem to help performance. However, it is likely that we did not adapt the models accordingly to make effective use of the added location information. This modification should yield immediate beneficial results. It would also be interesting to find out what kind of information, and how much of it, can be added to increase performance. A thorough investigation could yield interesting findings regarding the relationship between the size of the input query tuples versus the model's performance. From a machine learning perspective, the problem of automatically learning which types of contextual information are most informative for a given activity/verb is quite interesting.

Lastly, we had high hopes that the inclusion of semantic role labeling (SRL) in our model would improve the identification rate of event participants, and ultimately the overall performance of our model. We were quite surprised and puzzled to find that the integration of SRL actually decreased performance and would like to further investigate this issue. The first step in this line of research will be to process the SRL tool SwiRL for all documents in the corpus, instead of just a portion of the corpus. Secondly, a more sophisticated mapping heuristic might be needed in order to map the syntactic clauses between the output of SwiRL and the downstream coreference resolution process of CoreNLP. If we can figure out why the addition of SRL generated poor results, we can potentially fix it and improve the performance of our model. We also plan to run the SRL-infused model on the *hug* dataset to validate our initial findings.

9.2 Computer Vision Integration

The current formulation of our problem predicates on the detections of explicit visual cues from videos. Computer vision technologies offer us many ways to retrieve these detections from video images. We have shown that our information extraction

model can effectively tolerate noisy detection data, which means it should perform well on output from actual automatic detection algorithms. Our goal is to integrate these algorithms into our system. If successful, this integration would give us a complete end-to-end system that takes videos as input and outputs latent attribute descriptions of the videos. This would be an incredible achievement that promises many useful applications.

9.3 Other Applications

One interesting bi-product of our research is the construction of the CWSA-F_1 score for comparing two distributions of elements. We believe this performance measure can be used in many other applications. For example, one could use CWSA-F_1 to measure the distance between images for the task of image retrieval. Images are often represented as histograms features, such as gray-scale intensities, RGB values, and others, which can be naturally represented as distributions of values. It would be interesting to see if the CWSA-F_1 score can be a good indicator of perceptual similarity in images, like the Earth Mover's Distance (Rubner et al., 2000).

Lastly, we would like to extend our work to cover more scenarios and applications. The immediate step is to try our models on more activities. We've already examined *chase* and *hug*, and would like to know whether or not the performance can hold for harder, more ambiguous events, such as *run* and *walk*. The activities *chase* and *hug* often illicit clearer mental state inferences, where as *run* and *walk* do not. We wonder if our system can agree with human annotators in these cases, assuming that the annotators can more or less agree among themselves. Beyond video latent attribute identification, we suspect (and would like to verify) that this approach can achieve success in problems of other domains, such as medical diagnosis.

APPENDIX A

MENTAL STATE LABELS

We compiled an initial seed set of 160 mental state labels for our models from popular mental and emotional state dictionaries, including the Profile of Mood States (McNair et al., 1971) and Plutchik's wheel of emotions. These labels also include frequently used terms gathered from synsets found in WordNet.

admiring	cranky	fearful	instinctive	pleased	sympathetic
afraid	crazy	focused	interested	protective	tense
aggressive	curious	forgetful	irate	raging	terrified
agitated	cynical	frantic	irritated	rebellious	terrifying
alarmed	demented	friendly	jealous	refreshed	thankful
alert	depressed	frightened	joyful	relaxed	threatening
ambitious	desperate	frustrated	joyous	relieved	tired
amazed	determined	fun	lively	reluctant	trustful
amused	devious	furious	loathsome	remorseful	trusting
angry	disappointed	giddy	lonely	resentful	uncomfortable
annoyed	discontent	glamorous	loved	restless	uneasy
anxious	discouraged	gleeful	mad	revengeful	unhappy
apprehensive	disgusted	grateful	mellow	romantic	unworthy
ashamed	distracted	grumpy	merciless	sad	upset
assertive	drunken	guilty	mischievous	satisfied	urgent
bitter	eager	happy	miserable	scared	vengeful
bored	ecstatic	helpful	motivated	selfish	vigilant
calm	encouraged	helpless	naughty	selfless	vigorous
carefree	energetic	homicidal	nervous	serious	violent
cautious	energized	hopeful	numb	shaky	wary
cheerful	enraged	hopeless	optimistic	shocked	weary
competitive	enthusiastic	hostile	panicked	sickened	weird
complacent	envious	hurried	panicky	spiteful	welcoming
concerned	excited	impressed	peaceful	stressed	worried
confused	exhausted	indifferent	peeved	submissive	worthless
considerate	exhilarating	inebriated	pessimistic	surprised	
content	fatigued	infuriated	playful	suspenseful	

APPENDIX B

PARTICIPANTS EXTRACTION PATTERNS

This appendix lists all syntactic patterns used by our models to identify the participants of an event, or activity. Given a sentence containing the target activity verb, e.g., *chase*, we look for the following patterns of syntactic dependencies to identify the nominal subject and direct object of the verb. Our system uses the syntactic dependency parser offered by the CoreNLP toolkit. Please refer to Marneffe and Manning (2013) for additional details on each relation. All except one of the patterns below are general patterns that apply to all types of events. The last pattern is specialized for the verb *chase*.

***nsubj* (nominal subject) + *dobj* (direct object)**

In most cases, the nominal subject (*nsubj*) and the direct object (*dobj*) of the verb are the dependents that we seek. For the verb *chase*, the nominal subject is often the chaser and the direct object is the person being chased. Given a sufficiently complex sentence, however, we often cannot retrieve the nominal subject of the verb. Thus, our system extracts these syntactic phrases independently from each other (i.e., it is okay to retrieve just the direct object of the verb even if the *nsubj* relation is not present). Figure B.1 illustrates some example sentences where both participants can be identified and Figure B.2 shows cases where only the direct object can be retrieved. In the case of *chase*, dependents of the *nsubj* relation are extracted as the chasers and dependents of the *dobj* relations are the people being chased.

***nsubjpass* (passive nominal subject) + *agent* (agent)**

According to Marneffe and Manning (2013), an agent is a complement of a passive verb which is introduced by the preposition "by" and does the action. Thus, in the presence of the preposition "by", the roles of the participants are switched. That is,

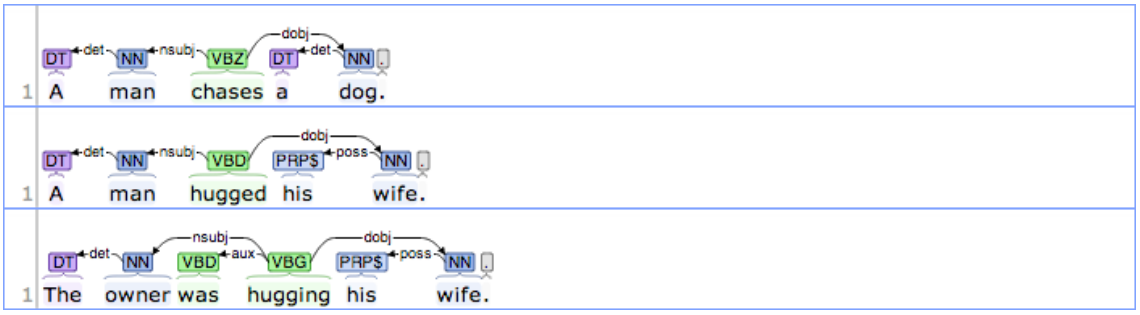


Figure B.1: Example sentences where both the nominal subject *nsubj* and the direct object *dobj* relations are present. In these cases, both participants of the event can be identified.

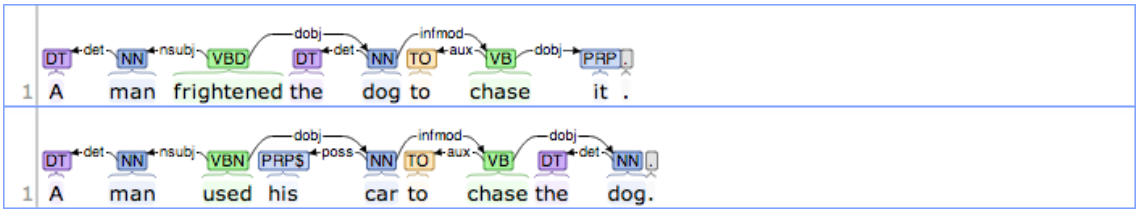


Figure B.2: Example sentences where only the direct object *dobj* relation for the target verb is present; and hence, only the object of the verb can be identified.

the syntactic subject of the sentence, captured by the passive relation *nsubjpass*, is now the object of the verb while the subject of the verb (a.k.a., the participant that does the action) is given by the syntactic relation *agent*. Figure B.3 shows some examples of this pattern.

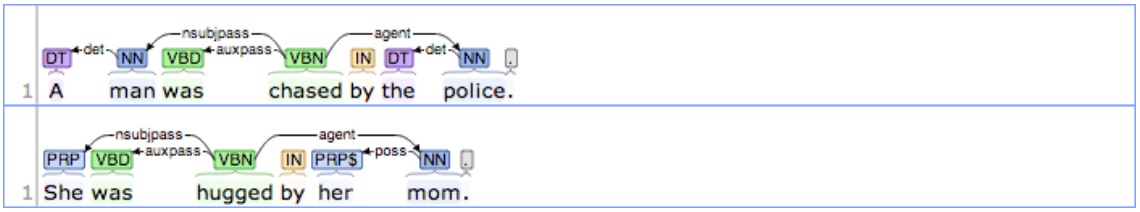


Figure B.3: When the preposition "by" is present in the sentence, the agent of the corresponding passive verb is extracted as the subject of the verb (i.e., the one that does the action), while the syntactic subject is the object the verb (a.k.a, the one being acted upon).

***prep_after* (object of the preposition "after")**

Although we try to design general patterns that work across all types of events/activities, some events have specific patterns that appear so frequently in text that they warrant their own special recognition, such as the "chase after" pattern. In sentences following the "chase after" pattern, the object of the verb *chase* is usually parsed as the object of the preposition "after," captured in CoreNLP's collapsed dependencies by the *prep_after* syntactic relation. Figure B.4 shows some example sentences following this pattern.

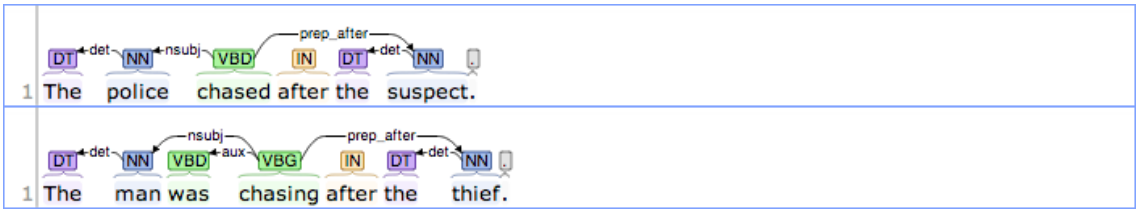


Figure B.4: The "chase after" pattern is a specialized pattern that applies only to the verb *chase*. In this pattern, the object of the preposition "after" is the object of the verb. Stanford's CoreNLP collapses the preposition "after" and its object into the *prep_after* relation.

APPENDIX C

MENTAL STATE EXTRACTION PATTERNS

This appendix lists all syntactic patterns used by our models to extract mental state labels referencing specific event participants. We search for these labels by pattern matching the syntactic dependencies of relevant sentences to look for common patterns that involve the participants of interest and their adjective complements. Our system uses the syntactic dependency parser offered by the CoreNLP toolkit. Please refer to Marneffe and Manning (2013) for additional details on each relation.

amod (adjectival modifier)

Perhaps the most common and relevant syntactic relation of interest is the adjectival modifier relation, known as *amod*. This dependency describes any adjectival phrase that serves to modify the meaning of a noun phrase. Since we already know the noun phrases of interest (from the coreference entities), we can search the list of syntactic dependencies for *amod* relations that directly connect these noun phrases to their modifying adjectival phrases. Figure C.1 illustrates some example sentences fitting this pattern. The adjectival phrases are extracted as potential complements, though further downstream filtering is required to get mental state labels.

nsubj (nominal subject) / *nsubjpass* (passive nominal subject)

Noun phrases that serve as the syntactic subjects (or nominal subjects) for a clause are often captured as dependents of the syntactic relations *nsubj* and *nsubjpass*. Note that we are only interested in nominal subject relations for which an entity of interest from the coreference chains is the dependent. We refer to these as valid *nsubj* relations to distinguish them from all other *nsubj* relations that do not reference an event participant. In many cases, the root (or head) clause of these syntactic relations can oftentimes be interpreted as complements.

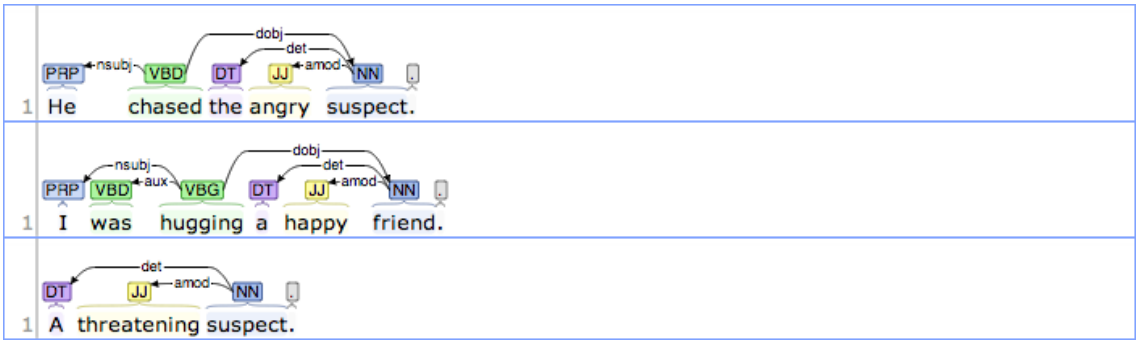


Figure C.1: Example sentences containing the common adjectival modifier (*amod*) syntactic relation. The graphical representation shows the dependency relationship as an arrow pointing from the head of the relation, which is the noun phrase, to the modifying adjectival phrase.

The most common of these cases occur when the sentence also contains a copula relation (*cop*), which captures the relationship between the complement of a copular verb and the copular verb. The Stanford representation chooses to take a copula as a dependent of its complement. Hence, when the *cop* relation shares the same head clause as a valid *nsubj* clause, we can extract this common head clause as a complement. Figure C.2 shows some example sentences containing the copular verb and mental state complements.

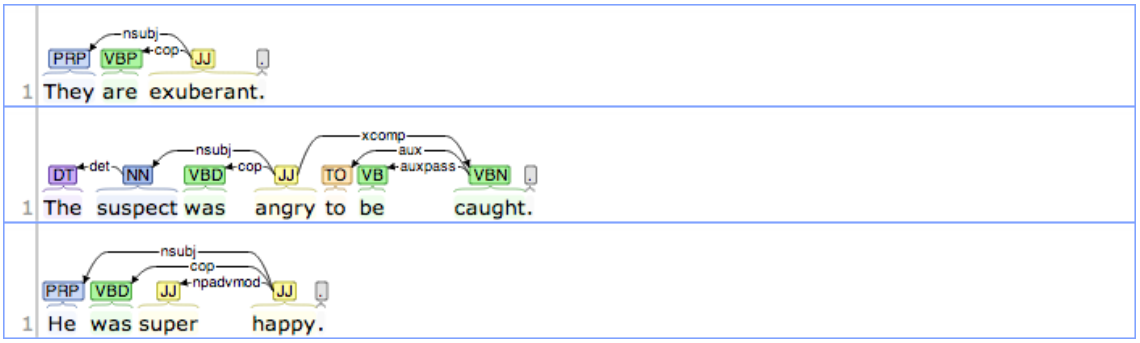


Figure C.2: Example sentences showing the indirect association between an entity of interest and a complement via a copular verb. The complement to be extracted is the head clause for both the *nsubj* and the *cop* relations.

Oftentimes, however, the adjective label is mistakenly tagged as a verb by the part-of-speech tagger. In these cases, we do not get the *cop* relation as above.

Instead, we often see the auxiliary (*aux*) or passive auxiliary (*auxpass*) relation in its place. These relations describe the non-main verb of the clause, e.g., a modal auxiliary, or a form of "be", "do" or "have" in a periphrastic tense. This makes sense; as when the adjective labels are incorrectly tagged as verbs, we end up with multiple verbs in each of these sentences, inducing the auxiliary verb relations. As with the *cop* relation, when the *aux* relation shares the same head clause as a valid *nsubj* relation (and similarly for *auxpass* and *nsubjpass*), we can extract this common head clause as a complement. See Figure C.3 for some example sentences demonstrating this pattern.

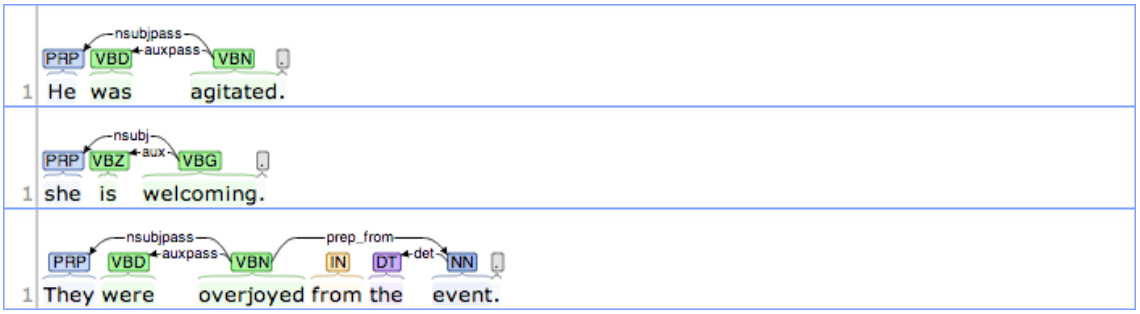


Figure C.3: Example sentences showing the indirect association between an entity of interest and an incorrectly tagged complement via the main copular verb. The complement to be extracted is the head clause for both the *nsubj* and the *aux*, or *nsubjpass* and *auxpass* relations.

In practice, we have noticed that it is generally acceptable to always treat the head clause of a valid *nsubj* or *nsubjpass* relation as a possible complement for extraction, without needing to check for a matching *cop*, *aux*, or *auxpass* relation. This is because the part-of-speech tagger is error-prone and often mis-tag adjective complements. Hence we find that it is simpler to be more lenient during the extraction process in order to catch more incorrectly tagged complements, rather than constructing a different pattern for each possible error case. Though, we still enforce that the clause being extracted must be related to an entity of interest, via the *nsubj* or *nsubjpass* relation.

***nsubj* of "feel"**

Asides from the copular to *be* verb, the verb *feel* is also highly indicative of the presence of a mental state label. However, *feel* differs from the copular verb pattern above in that the verb itself is often the root of two syntactic relations, one connecting the verb to the nominal subject and one connecting the verb to the complement. The former is the familiar *nsubj* relation, where the head of the relation is the verb *feel* and the dependent is the nominal subject. The second relation, which connects the verb to a complement, is usually the adjectival complement (*acomp*) relation. Like the name suggests, an adjectival complement of a verb is an adjectival phrase which functions as the complement, such as the object of the verb. Thus, if we find an *acomp* relation that shares the same head verb *feel* as a valid *nsubj* relation, then we can extract the dependent of the *acomp* relation as a complement. Figure C.4 shows some examples of this pattern.

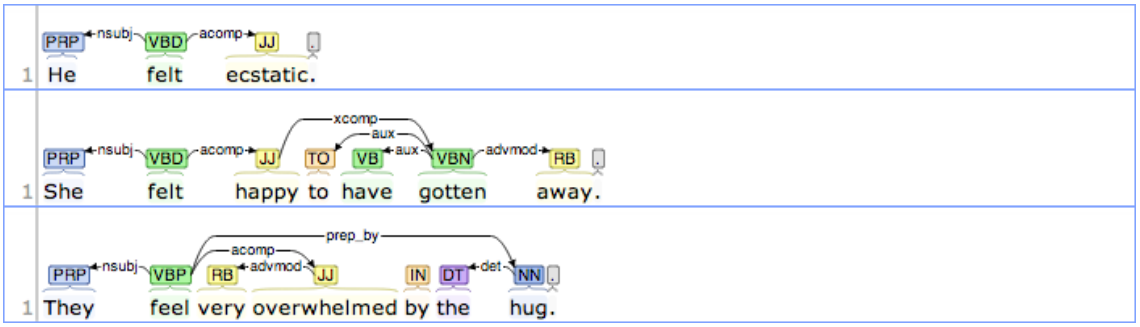


Figure C.4: Example sentences showing the indirect association between an entity of interest and a complement via the verb *feel*. The complement to be extracted and the subject both share the same head verb. The complement is connected to the verb via the relation *acomp* while the nominal subject is connected to the verb via the relation *nsubj*.

Also as with previous patterns, when the adjective complement is mistakenly tagged as a verb by the part-of-speech tagger, we cannot count on the syntactic dependency processor to correctly give us the *acomp* relation. Instead, we often find the *dep* relation in its place, which is used when the system is unable to determine a more precise dependency relation between two words (see Figure C.5).

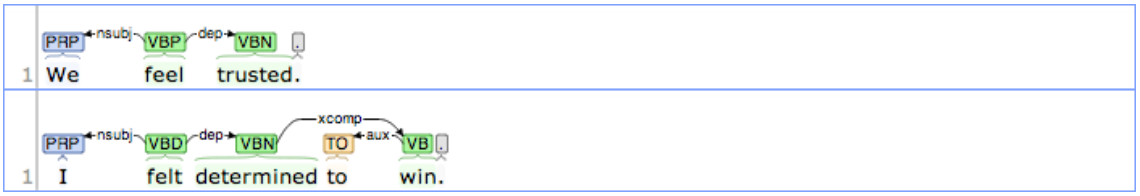


Figure C.5: Example sentences showing the indirect association between an entity of interest and an incorrectly tagged complement via the verb *feel*. The complement to be extracted and the subject both share the same head verb. The complement is connected to the verb via the relation *dep* while the nominal subject is connected to the verb via the relation *nsubj*.

REFERENCES

Abbasi, A. R., M. N. Dailey, N. V. Afzulpurkar, and T. Uno (2009). Student mental state inference from unintentional body gestures using dynamic Bayesian networks. *Journal on Multimodal User Interfaces*, **3**(1-2), pp. 21–31. ISSN 1783-7677. doi:10.1007/s12193-009-0023-7.

Baltrusaitis, T., D. McDuff, N. Banda, M. Mahmoud, R. el Kaliouby, P. Robinson, and R. Picard (2011). Real-time inference of mental states from facial expressions and upper body gestures. In *Face and Gesture 2011*, pp. 909–914. IEEE. ISBN 978-1-4244-9140-7. doi:10.1109/FG.2011.5771372.

Brants, T. (2000). TnT: A statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing*, pp. 224–231. Association for Computational Linguistics, Morristown, NJ, USA. doi:10.3115/974147.974178.

Brau, E., D. Dunatunga, K. Barnard, T. Tsukamoto, R. Palanivelu, and P. Lee (2011). A generative statistical model for tracking multiple smooth trajectories. In *CVPR 2011*, pp. 1137–1144. IEEE. ISBN 978-1-4577-0394-2. doi:10.1109/CVPR.2011.5995736.

Brau, E., J. Guan, K. Simek, L. D. Pero, C. R. Dawson, and K. Barnard (2013). Bayesian 3D Tracking from monocular video. In *The IEEE International Conference on Computer Vision (ICCV)*.

Cai, J. and M. Strube (2010). Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 28–36.

Chambers, N. and D. Jurafsky (2009). Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP Volume 2 ACLIJCNLP 09*, pp. 602–610.

Chambers, N. and D. Jurafsky (2011). Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pp. 976–986.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th annual meeting on Association for Computational Linguistics* -, pp. 16–23. Association for Computational Linguistics, Morristown, NJ, USA. doi:10.3115/976909.979620.

El Kaliouby, R. and P. Robinson (2004). Real-Time Inference of Complex Mental States from Facial Expressions and Head Gestures. In *2004 Conference on Computer Vision and Pattern Recognition Workshop*, pp. 154–154. IEEE. doi: 10.1109/CVPR.2004.427.

Everingham, M., S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2014). The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*. ISSN 0920-5691. doi: 10.1007/s11263-014-0733-5.

Fei-Fei, L. and P. Perona (2005). A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pp. 524–531. IEEE. ISBN 0-7695-2372-2. doi:10.1109/CVPR.2005.16.

Felzenszwalb, P., D. McAllester, and D. Ramanan (2008). A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE. ISBN 978-1-4244-2242-5. doi:10.1109/CVPR.2008.4587597.

Gabbard, R., M. Freedman, and R. Weischedel (2011). Coreference for learning to extract relations: yes, Virginia, coreference matters. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, pp. 288–293.

Gendron, M., D. Roberson, J. M. van der Vyver, and L. F. Barrett (2014). Cultural relativity in perceiving emotion from vocalizations. *Psychological science*, **25**(4), pp. 911–20. ISSN 1467-9280. doi:10.1177/0956797613517239.

Giebel, J., D. Gavrila, and C. Schnörr (2004). A bayesian framework for multicue 3d object tracking. In *Computer Vision-ECCV 2004*, pp. 241–252. doi: 10.1007/978-3-540-24673-2_20.

Han, M. and T. Kanade (2001). Multiple motion scene reconstruction from uncalibrated views. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pp. 163–170. IEEE Comput. Soc. ISBN 0-7695-1143-0. doi:10.1109/ICCV.2001.937513.

Herman, M. and T. Kanade (1986). Incremental reconstruction of 3D scenes from multiple, complex images. *Artificial Intelligence*, **30**(3), pp. 289–341. ISSN 00043702. doi:10.1016/0004-3702(86)90002-0.

Hirst, G. and D. St-Onge (1998). Lexical chains as representations of context for the detection and correction of malapropisms. In Fellbaum, C. (ed.) *WordNet: An*

Electronic Lexical Database (Language, Speech, and Communication), pp. 305–332. The MIT Press.

Lazebnik, S., C. Schmid, and J. Ponce (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)*, volume 2, pp. 2169–2178. IEEE. ISBN 0-7695-2597-0. doi: 10.1109/CVPR.2006.68.

Lee, H., A. Chang, Y. Peirsman, N. Chambers, M. Surdeanu, and D. Jurafsky (2013). Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, **39**(4), pp. 885–916. ISSN 0891-2017. doi:10.1162/COLI_a_00152.

Li, L., H. Su, L. Fei-Fei, and E. Xing (2010). Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Advances in Neural Information Processing Systems*.

Liu, Z. and S. Wang (2011). Emotion recognition using hidden Markov models from facial temperature sequence. In *ACII'11 Proceedings of the 4th international conference on Affective computing and intelligent interaction - Volume Part II*, pp. 240–247.

Luo, X. (2005). On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing - HLT '05*, pp. 25–32. Association for Computational Linguistics, Morristown, NJ, USA. doi:10.3115/1220575.1220579.

Marneffe, M. D. and C. Manning (2013). Stanford typed dependencies manual.

Marneffe, M. D., C. Manning, and C. Potts (2010). "Was it good? It was provocative." Learning the meaning of scalar adjectives. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 167–176.

McKenna, S. J., S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler (2000). Tracking Groups of People. *Computer Vision and Image Understanding*, **80**(1), pp. 42–56. ISSN 10773142. doi:10.1006/cviu.2000.0870.

McNair, D. M., M. Lorr, and L. F. Droppleman (1971). Profile of Mood States (POMS).

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, pp. 1–12.

Mikolov, T., W.-t. Yih, and G. Zweig (2013b). Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, **38**(11), pp. 39–41. ISSN 00010782. doi:10.1145/219717.219748.

Moeslund, T. B. and E. Granum (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, **81**(3), pp. 231–268. ISSN 10773142. doi:10.1006/cviu.2000.0897.

Moeslund, T. B., A. Hilton, and V. Krüger (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, **104**(2-3), pp. 90–126. ISSN 10773142. doi:10.1016/j.cviu.2006.08.002.

Mohtarami, M., H. Amiri, M. Lan, and C. L. Tan (2011). Predicting the uncertainty of sentiment adjectives in indirect answers. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, p. 2485. ACM Press, New York, New York, USA. ISBN 9781450307178. doi:10.1145/2063576.2063998.

Ng, C., Y. Tay, and B. Goi (2012). Recognizing human gender in computer vision: a survey. *PRICAI 2012: Trends in Artificial Intelligence*, **7458**, pp. 335–346. doi:10.1007/978-3-642-32695-0_31.

O'Hara, S. and B. A. Draper (2012). Scalable action recognition with a subspace forest. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1210–1217. IEEE. ISBN 978-1-4673-1228-8. doi:10.1109/CVPR.2012.6247803.

Pedersen, T., S. Patwardhan, and J. Michelizzi (2004). WordNet::Similarity: measuring the relatedness of concepts. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pp. 1024–1025. San Jose, CA.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image and Vision Computing*, **28**(6), pp. 976–990. ISSN 02628856. doi:10.1016/j.imavis.2009.11.014.

Quattoni, A. and A. Torralba (2009). Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 413–420. IEEE. ISBN 978-1-4244-3992-8. doi:10.1109/CVPR.2009.5206537.

Ramanan, D. (2012). Face detection, pose estimation, and landmark localization in the wild. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2879–2886. IEEE. ISBN 978-1-4673-1228-8. doi:10.1109/CVPR.2012.6248014.

Ramanan, D., D. a. Forsyth, and A. Zisserman (2007). Tracking people by learning their appearance. *IEEE transactions on pattern analysis and machine intelligence*, **29**(1), pp. 65–81. ISSN 0162-8828. doi:10.1109/TPAMI.2007.22.

Reschke, K., A. Vogel, and D. Jurafsky (2011). Generating recommendation dialogs by extracting information from user reviews. In *Proceedings of the 51st annual meeting of the Association for Computational Linguistics*.

Riloff, E. and R. Jones (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the sixteenth national conference on Artificial intelligence (AAAI-1999)*, pp. 474–479.

Rubner, Y., C. Tomasi, and L. Guibas (2000). The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, **40**(2), pp. 99–121. doi:10.1023/A:1026543900054.

Sadanand, S. and J. J. Corso (2012). Action bank: A high-level representation of activity in video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1234–1241. IEEE. ISBN 978-1-4673-1228-8. doi:10.1109/CVPR.2012.6247806.

Schuldt, C., I. Laptev, and B. Caputo (2004). Recognizing human actions: a local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, pp. 32–36 Vol.3. IEEE. ISBN 0-7695-2128-2. doi:10.1109/ICPR.2004.1334462.

Slabaugh, G. and B. Culbertson (2001). A survey of methods for volumetric scene reconstruction from photographs. In *Proceedings of the 2001 Eurographics conference on Volume Graphics*, pp. 81–101. doi:10.1007/978-3-7091-6756-4_6.

Sokolova, M. and G. Lapalme (2011). Learning opinions in user-generated web content. *Natural Language Engineering*, **17**(04), pp. 541–567. ISSN 1351-3249. doi:10.1017/S135132491100012X.

Surdeanu, M., L. Màrquez, X. Carreras, and P. Comas (2007). Combination Strategies for Semantic Role Labeling. *Journal Of Artificial Intelligence Research (JAIR)*, **29**, pp. 105–151.

Weinland, D., R. Ronfard, and E. Boyer (2011). A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, **115**(2), pp. 224–241. ISSN 10773142. doi:10.1016/j.cviu.2010.10.002.

Xiao, J., J. Hays, K. a. Ehinger, A. Oliva, and A. Torralba (2010). SUN database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492. IEEE. ISBN 978-1-4244-6984-0. doi:10.1109/CVPR.2010.5539970.

Yang, Y. and D. Ramanan (2011). Articulated pose estimation with flexible mixtures-of-parts. In *CVPR 2011*, pp. 1385–1392. IEEE. ISBN 978-1-4577-0394-2. doi:10.1109/CVPR.2011.5995741.

Zhou, G. and J. Su (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, July, p. 473. Association for Computational Linguistics, Morristown, NJ, USA. doi:10.3115/1073083.1073163.