Spring 1-1-2016

Semantic Likelihood Models for Bayesian Inference in Human-Robot Interaction

Nicholas Sweet
University of Colorado at Boulder, nisw6751@colorado.edu

Recommended Citation

Sweet, Nicholas, "Semantic Likelihood Models for Bayesian Inference in Human-Robot Interaction" (2016). *Aerospace Engineering Sciences Graduate Theses & Dissertations*. 151.
https://scholar.colorado.edu/asen_gradetds/151

Semantic Likelihood Models for Bayesian Inference in Human-Robot Interaction

by

Nicholas Sweet

B. Eng. in Computer Engineering, Concordia University, 2014

A thesis submitted to the
Faculty of the Graduate School of the
University of Colorado in partial fulfillment
of the requirements for the degree of
Master of Science
Department of Aerospace Engineering Sciences
2016

This thesis entitled:
Semantic Likelihood Models for Bayesian Inference in Human-Robot Interaction
written by Nicholas Sweet
has been approved for the Department of Aerospace Engineering Sciences

Prof. Nisar Ahmed

Prof. Eric Frew

Date ___________________

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Sweet, Nicholas (Aerospace Engineering)

Semantic Likelihood Models for Bayesian Inference in Human-Robot Interaction

Thesis directed by Prof. Nisar Ahmed

Autonomous systems, particularly unmanned aerial systems (UAS), remain limited in autonomous capabilities largely due to a poor understanding of their environment. Current sensors simply do not match human perceptive capabilities, impeding progress towards full autonomy. Recent work has shown the value of humans as sources of information within a human-robot team; in target applications, communicating human-generated 'soft data' to autonomous systems enables higher levels of autonomy through large, efficient information gains. This requires development of a 'human sensor model' that allows soft data fusion through Bayesian inference to update the probabilistic belief representations maintained by autonomous systems. Current human sensor models that capture linguistic inputs as semantic information are limited in their ability to generalize likelihood functions for semantic statements: they may be learned from dense data; they do not exploit the contextual information embedded within groundings; and they often limit human input to restrictive and simplistic interfaces. This work provides mechanisms to synthesize human sensor models from constraints based on easily attainable a priori knowledge, develops compression techniques to capture information-dense semantics, and investigates the problem of capturing and fusing semantic information contained within unstructured natural language. A robotic experimental testbed is also developed to validate the above contributions.

Dedication

À mes parents, en reconnaissance du temps qu'ils ont perdu pour leur fils

Acknowledgements

I have received immeasurable help along the course of this work. Of particular note is the support provided by Nisar Ahmed, my advisor, who constantly went above and beyond the task of advising: he was always available to help guide my thinking, to help me as I wrapped my mind around difficult concepts, and to support my intellectual development both within and outside engineering. Other members of my committee, Eric Frew and Mike Mozer, have also helped mold me through our discussions and interactions inside and outside the classroom.

I have shared lab spaces and great discussions with wonderful peers. Thanks, in no particular order, to: Will Silva, Anthony Carfang, Neeti Wagle, Jeremy Muesing, Sierra Williams, Steve McGuire, Matt Aitken, Ian Loefgren, Brett Isrealsen, Luke Burks, Roger Lawrence, Tevis Nichols, Drew Ellison, Craig Turansky and Jason Durrie.

Outside the lab, thanks to my friends near and far: Robert Gouldson, Chuck Wilson, Sandra Witzen, John Gemperline, Leah Isaman, Laura Grace Beckerman, Mehdi Sabzalian, Alex Potapov, Alex Teodor, Tiago Leao, Lambert Le, Riccardo Biciola, Chris Bridgeman, Graham Lau, Amanda Williams, Alisa Soukhodolskaya, Maude Gupta, and many others who I simply don't have the space to thank.

Most importantly, thanks to my family: Doug Sweet, Alec Sweet, Lauren Narcross and Marjolaine Boutin-Sweet, as well as the full Boutin and Sweet families, for providing unwavering encouragement and inspiration.

Contents

Chapter

Tables

Table

Figures

Figure

Abbreviations

BSM binary softmax model

CRF conditional random field

DCG distributed correspondence graph

DMTSP dynamic multi-target search problem

G^3 generalized grounding graph

GM Gaussian mixture

GMM Gaussian mixture model

HDCG hierarchical distributed correspondence graph

HMM hidden markov model

LWIS likelihood-weighted importance sampling

MAP maximum a posteriori

MMS multimodal softmax

NLP natural language processing

PCCG probabilistic combinatory categorical grammar

pdf probability density function

PM particle model

ROS robot operating system

SGNS skip-gram negative sampling

UAS unmanned aerial system

VBIS variational bayes with importance sampling

VOI value of information

WiSAR wilderness search and rescue

Nomenclature

D_k Random variable of softmax class selected at time k

k Discrete time index

L Set of selection indices for D_k

l Single selection index for D_k

m Number softmax classes within a softmax model

n Dimensionality of the state space

N_o Number of softmax models

O_k Unstructured natural language input

$\mathbf{T}$ Set of possible tokenizations of an input statement O_k

X_k n-dimensional vector of state variables at time k

ζ_k Hard sensor data k

Chapter 1

Introduction

The late 20th century has seen a proliferation of autonomous systems in commercial, military and home environments. While a universally accepted definition of autonomy proves elusive, we can generally describe autonomy as any self-governing system. A prominent form of these systems is the robot: a physical embodiment of autonomy, with the ability to sense, plan and act within an environment. A major driving factor to build and integrate these autonomous systems is delegation; we give robots tasks that we would rather not do, primarily because these tasks are dull, dirty or dangerous. This is underscored by the fact that the word *robot* itself comes from the Czech word *robota* meaning *forced labour* – our motivation is to build robots that we can command to perform unpleasant tasks in our place.

The focus on delegation, however, is countered by the importance of interaction. Full autonomy is often seen as the pinnacle of autonomous systems development – we would like to delegate high-level tasks to an autonomous system, and have it carry the task out with minimal guidance. The main point in delegating tasks is to not have to specify a complete breakdown of the task structure, watch over the autonomous system as it performs the task, and to correct it at every failing; we want to minimize our effort in the process of delegation.

However, full autonomy is both complex and, in many cases, insufficient. Parasuraman's landmark paper on levels of automation [1] argues for autonomous systems to be designed based on expected levels human interaction – we must consider not only delegation mechanisms, but interaction mechanisms as well. Interaction supports delegation through clarity of task assignment and provides an avenue through which autonomy can better understand the world. While interaction may add additional overhead to the process of delegation, it may also enable a robot to perform tasks it would otherwise be unable to do, while giving the interacting human a better understanding of the robot.

This is particularly important in human-robot teaming. Humans excel at certain things robots find difficult – perception, interpretation and forethought. Conversely, robots tend to excel at high-speed calculation, optimal decision-making, and performing repetitive tasks. In general, both humans and robots posses complementary skillsets that support one-another when working in concert. When designing for human-robot teaming, one should attempt to maximize the mutual benefit of interaction while minimizing its overhead.

For example, much of the work done by Goodrich et al. [2–6] focuses on minimizing operator workload in teams of humans and unmanned aerial systems (UASs). In wilderness search and rescue (WiSAR), each individual UAS may be supported by a UAS payload operator, a UAS flight operator, a mission manager, and ground operations personnel. This presents the issue of scale: the overhead of interaction required by the UAS requires multiple humans to continuously delegate lower-level tasks, as the UAS is not developed enough to provide higher levels of autonomous reasoning. Aside from minimizing the workload of each operator with respect to the UAS, it is also desirable to invert this human-to-autonomy ratio, such that a single operator is able to delegate to multiple robots simultaneously. Both these goals can be achieved through greater efficiency in delegation through higher levels of autonomy.

Many autonomous systems are limited in their ability to sense their surrounding environment. Visual perception in particular is a difficult problem which robots have yet to truly solve; yet, humans are adept at interpreting visual sensory information. It follows that if a human were able to provide sensory information to a robot, as so-called *soft data*[7], the robot would be able to fuse this information with its model of the world and information it collects from other sensors, so-called *hard data*. While providing soft data to a robot necessarily adds overhead to the human-robot interaction, we posit that this is of net benefit to the human-robot team until autonomous systems are equipped with perceptive abilities equal to a human's.

The goal, then, is to define mechanisms for providing soft data to autonomous systems so that the autonomy is able to make better decisions based on a better understanding of the environment. This notion runs in parallel to delegation: we provide a robot with some task, then assist the robot with its task when appropriate. If the goals of the human and robot are aligned (which is to be expected, as the latter are designed), then we expect greater achievement of the human-robot team through interaction. From the perspective of the robot, the human is considered to be both an issuer of tasks as well as one of its sensors. As with any other sensor, a *human sensor model* must be developed and calibrated in order to provide soft data to the autonomous system.

The primary mechanism for communication considered in this work is *semantic language*: a set of signifiers (words, phrases, symbols) that each represent a unit of information. The complexity of sensory perception is well mirrored by the richness and flexibility of semantic language; for example, the statement, "The target was under an oak tree three minutes ago and is now limping north along the river's east bank," contains multiple units of information that may be simple for a human to perceive, but hard to communicate to a robotic counterpart. The challenge lies in mapping semantic information to a form that can be well-represented by an autonomous system. We take the Bayesian approach to this problem, in which the observation likelihood function maps from categorical semantic labels to probabilistic state information; with properly modeled likelihood functions, we can translate from semantic language to probabilistic information representations.

This work contributes two techniques to enable the development of human sensor models: one technique allows for the synthesis of likelihood functions through imposition of constraints rather than calibration using dense training data; the other technique provides compression algorithms for information-rich semantic statements to allow efficient online data fusion. The remainder of the work provides progress toward the problem of mapping from unstructured natural language to well-known human sensor models. This work is considered within the context of an experimental setup where a human and remote robot share information to accomplish the task of finding other robots located near the remote robot.

Chapter 2

Background

If our goal is to help humans and autonomous systems communicate, we need to speak in a common language. Just as humans have verbal and written languages to communicate ideas, so have we developed mathematical languages to communicate information. This work makes use the *softmax* function as a 'translator' between categorical semantic labels and probabilities associated with dynamic state information. This chapter contains the formal background required to understand both the key properties of the softmax function and how to learn functions from human training data. We also discuss related work in sharing information between humans and robots, whether in developing human sensor models or providing natural language commands to robotic partners. The guiding thought behind this chapter is that we can relate elements of human language to observation likelihood functions that can be used to update a belief over dynamic states.

2.1 Uncertainty in Semantic Communication

To guide discussion, we nominally consider examples from a simple target search problem, in which a human and robot jointly search for one or more targets of interest. There are several communication pathways a human can use to specify state information, from language (both written and spoken) to graphical interfaces [8] to gestures [9]. Each communication pathway trades between precision and communication speed; for instance, it would be faster to use qualitative language (e.g. "Target is near the chair.") rather than quantitative language (e.g. "Target is exactly 1.45 meters

North and 0.31 meters East of the chair.") albeit less precise. While previous work has explored quantitative language [10], we focus on qualitative language only.

In communicating a message from a source to a destination, multiple sources of uncertainty can be considered. Take the phrase, "I think the person is near that house." The sources of uncertainty include:

(1) *Knowledge uncertainty*, where the source provides an uncertain observation (i.e. "I *think* the person is near that house."). Capturing this uncertainty requires a model of the source's accuracy; in other words, calibrating the source's sensor noise[1] .

(2) *Transmission uncertainty*, where some noise corrupts the transmitter-receiver pathway (i.e. I said, "I think the person is near that house," but you heard, "I know the person is far from that house."). This is most common in speech- and gesture-based communication, but chat and graphical interfaces greatly reduce or eliminate transmission uncertainty.

(3) *Semantic uncertainty*, where the meaning behind chosen words is uncertain (i.e. what does *near* mean?). Quantitative language eliminates semantic uncertainty, often at the expense of knowledge uncertainty due to inherent human biases in quantitative value estimation which must be learned [10].

(4) *Lexical uncertainty*, where multiple words can be used to describe similar concepts (i.e. how well *close to*, *nearby* and *next to* describe *near*?). Using a language in which all words have a one-to-one mapping to distinct concepts eliminates lexical uncertainty, but this requires either a limited language or modeling the entire language.

Knowledge uncertainty is closely tied to the data-association problem, which asks, "Given multiple possible targets, which measurements are associated with which targets?" The observation data may also be associated with no target (commonly understood as a false observation that should be simply ignored). This is a non-trivial problem, though radar tracking literature in particular

[1] In the case of a human source, this describes the source's true uncertainty rather than perceived uncertainty, although perceived uncertainty is often an indicator of true uncertainty.

provides typical approaches for capturing this knowledge uncertainty; simpler techniques such as the interacting multiple model multi-hypothesis tracker (IMM-MHT) [11] provide a weighted sum over hypotheses provided by a bank of Kalman filters. While not a solved problem, capturing knowledge uncertainty is well-understood in the literature and not a focus of this work.

Transmission uncertainty is similarly well-understood by the signal processing community. Within the space of human-robot interaction, work typically revolves around defining measures for a robot's perception of the corruption of the source signal, i.e. Sattar and Dudek's work in estimation of uncertainty in gesture recognition [9]. The transmission uncertainty, coupled with the cost of misinterpreting the gesture, are then evaluated to determine whether the robot should ask for clarification before performing a task. This type of uncertainty is heavily dependent on the transmission medium, and is not a focus of this work.

Semantic uncertainty is potentially the least well-understood of the four types of uncertainties defined above within the context of human-robot interaction. Qualitative language is necessarily vague and its semantic meaning is typically captured either as a fuzzy set [12] or probabilistically [13]. As discussed in [14], the vagueness of language is reflected by its inherent epistemologically uncertain semantic meaning – that is, this uncertainty should not necessarily be minimized, but rather maintained as useful information. Much of the efforts in capturing semantic uncertainty for human-robot interaction are related to robot control, in which a human utterances must be grounded to semantically meaningful concepts, typically through a probabilistic relation (e.g. in [15–23]). Capturing semantic uncertainty in the context of human sensing typically requires a probabilistic representation of semantic meaning (e.g. normalized exponential likelihood [24], random finite sets [25], variable-sized grids [26]). This uncertainty measure is the primary focus of this work, covered in Chapter 3.

Lexical uncertainty has long been a focus of the natural language processing (NLP) community. Recent efforts in word sense matching have provided robust similarity measures between words, particularly from Mikolov's skip-gram negative sampling (SGNS) approach that allows high-dimensional word vector embeddings to be easily and quickly trained from large corpora [27]. This is an optimization of the shallow, two-layer neural net approach dubbed `word2vec`. Example extensions to this approach provide greater word sense matching accuracy (e.g. capturing differences between homographs like 'bear,' which could be either an animal or an action) through: training based on syntactic dependency contexts rather than bag-of-words contexts [28]; embedding part-of-speech tags alongside word vectors [29]; and infinite-dimensional word embeddings [30] for efficient multimodal sense matching. Capturing lexical uncertainty is a secondary focus of this work, covered in Chapter 4.

In developing some translation mechanism for phrases like, "I think the person is near that house," we must be able to account for all these types of uncertainty embedded within the phrase. These are not necessarily the only types of uncertainty found in linguistic human-robot communication, but it is argued that these are the predominant ones.

A major source of prior work in capturing semantic uncertainty is the work performed by Ahmed in both modeling semantic statements and fusion of these modeled likelihoods with multimodal prior beliefs [8, 24, 31–35]. This work will be discussed in the following subsections.

2.1.1 Bayesian Inference with Semantic Observations

We now consider the formal problem of data fusion with semantic observations using Bayesian inference. One benefit of semantics is the richness of possible state variables to consider: a human may want to describe a target's position, speed, heading, movement patterns, size, weight, expected position, movement path, etc. This represents a problem of tracking a state with both continuous random variables and discrete random variables, as well as mapping the semantic observations to these states. We consider only the case of continuous states within this work, but hybrid continuous and discrete distributions are well-studied in the literature [36]. What follows is a brief overview of a method that develops and uses this mapping between categorical semantics and n continuous states, predominantly developed by Ahmed [24].

The state vector $X_k \in \mathbb{R}^n$ is composed of n dynamic, continuous state variables, where $k \in \mathbb{Z}^{0+}$ is the time index. Hard sensor data ζ_k may be provided in the form of continuous measurements (e.g.

from ultrasonic range sensors) or discrete measurements (e.g. from mechanical bumper sensors). Soft sensor data D_k may be provided as one of $m \in \mathbb{Z}^+$ categorical semantic labels for human observations. The hard data joint conditional observation likelihood $p(\zeta_k \mid X_k)$ is assumed known. The soft data conditional observation likelihood is $P(D_k = i \mid X_k)$, where $i = 1, 2 \ldots, m$ indexes one of the mutually exclusive and exhaustive semantic labels such that $\sum_{i=1}^{m} P(D_k = i \mid X_k) = 1$. The sequences of all observations accumulated until time k are denoted $\zeta_{1:k} \equiv \{\zeta_1, \zeta_2, \ldots, \zeta_k\}$ and $D_{1:k} \equiv \{D_1, D_2, \ldots, D_k\}$.

With known initial prior probability density function (pdf) $p(X_0)$ and known state transition pdf $p(X_k \mid X_{k-1})$, state information at time k conditioned on all previous information $D_{1:k-1}$ and $\zeta_{1:k-1}$ can be found through the Chapman-Komolgorov [37] equation:

$$p(X_k \mid \zeta_{1:k-1}, D_{1:k-1}) = \int p(X_k \mid X_{k-1}) p(X_{k-1} \mid \zeta_{1:k-1}, D_{1:k-1}) dX_k \tag{2.1}$$

The hard data measurement update step fuses information contained in ζ_k with the latest available information $p(X_k \mid \zeta_{1:k-1}, D_{1:k-1})$ via Bayes' rule:

$$p(X_k \mid \zeta_{1:k}, D_{1:k-1}) = \frac{p(\zeta_k \mid X_k) p(X_k \mid \zeta_{1:k-1}, D_{1:k-1})}{\int p(\zeta_k \mid X_k) p(X_k \mid \zeta_{1:k-1}, D_{1:k-1}) dX_k} \tag{2.2}$$

Similarly, the soft data measurement update step fuses information contained in D_k with the latest available information $p(X_k \mid \zeta_{1:k}, D_{1:k-1})$ via Bayes' rule:

$$p(X_k \mid \zeta_{1:k}, D_{1:k}) = \frac{P(D_k \mid X_k) p(X_k \mid \zeta_{1:k}, D_{1:k-1})}{\int P(D_k \mid X_k) p(X_k \mid \zeta_{1:k}, D_{1:k-1}) dX_k} \tag{2.3}$$

In general, recursive Bayes' filters such as the Kalman filter (or a non-linear variant) [37], particle filter or Gauss-sum filter [38] can be used to find the pdfs for (2.1) and (2.2). As well, Ahmed [24] demonstrates a variational bayes with importance sampling (VBIS) method to perform the soft data fusion step (2.3) using a Gaussian mixture (GM) prior and *softmax* likelihood to generate a GM posterior. This accumulates all available information into an estimate of the state X_k at time k, but requires a well-defined soft observation likelihood $P(D_k \mid X_k)$. For the remainder of this work, we focus on the soft data measurement update step, denoting $p(X_k) \equiv p(X_k \mid \zeta_{1:k-1}, D_{1:k-1})$ and $p(X_k \mid D_k) \equiv p(X_k \mid \zeta_{1:k-1}, D_{1:k}$ as the Bayesian prior and posterior (respectively) in (2.3), leading to the following reformulation of (2.3):

$$p(X_k \mid D_k) = \frac{P(D_k \mid X_k)p(X_k)}{\int P(D_k \mid X_k)p(X_k)dX_k} \tag{2.4}$$

The GM prior is a weighted sum of N_{mix} n-dimensional normal distributions:

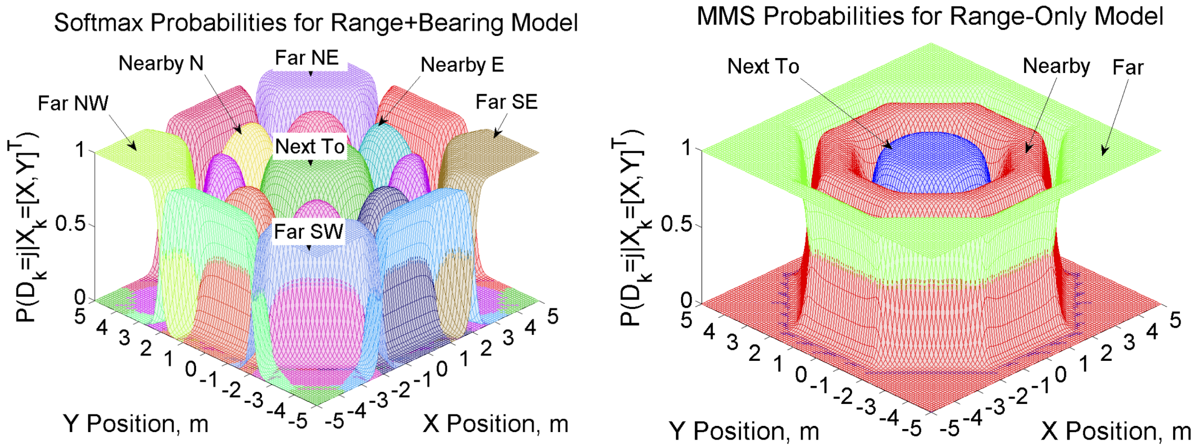
$$p(X_k) = \sum_{l=1}^{N_{mix}} w_l \mathcal{N}(X_k; \mu_l, \Sigma_l) \tag{2.5}$$

Where mixand l is weighted by a factor $0 \leq w_l \leq 1$ such that $\sum_{l=1}^{N_{mix}} w_l = 1$, and parametrized by mean $\mu_l \in \mathbb{R}^n$ and covariance $\Sigma_l \in \mathbb{R}^{n \times n}$. This weighted sum of Gaussians is able to represent arbitrary smooth distributions compactly, providing several advantages over similar multimodal probabilistic state representations, such as grid-based or particle-based methods. Primarily, a fixed upper bound for N_{mix} limits the computation expense of (2.4) while GM merging minimizes the information loss of reducing N_{mix}[39], allowing selection of N_{mix} to trade between accuracy and computability. While grids may trade between accuracy and computability through variable cell size [26], they suffer from the curse of dimensionality for high-dimensional states, they do not interact easily with hard sensor fusion techniques (e.g. Gaussian posteriors generated from extended and unscented Kalman filters) and their representations are not compact. Similarly, particle representations trade accuracy and computability through the number of particles used, but will not be compact for highly multimodal states, suffer from sample degeneracy (e.g. when observations are given for states with relatively few nearby particles), and also scale poorly in high-dimensional state spaces.

The softmax likelihood function is a normalized exponential:

$$P(D_k = i \mid X_k) = \frac{e^{w_i^T X_k + b_i}}{\sum_{j=1}^{m} e^{w_j^T X_k + b_j}} \tag{2.6}$$

Where parameters $\Theta_i = \{w_i, b_i\}$ determine the shape of each *softmax class* $P(D_k = i \mid$



(a) Softmax model of joint range and bearing labeled classes.

(b) Multimodal softmax model of range superclasses.

Figure 2.1: Observation likelihoods for categorical labels represented as (a) softmax classes and (b) softmax superclasses. Reprinted from [24], ©2013 IEEE.

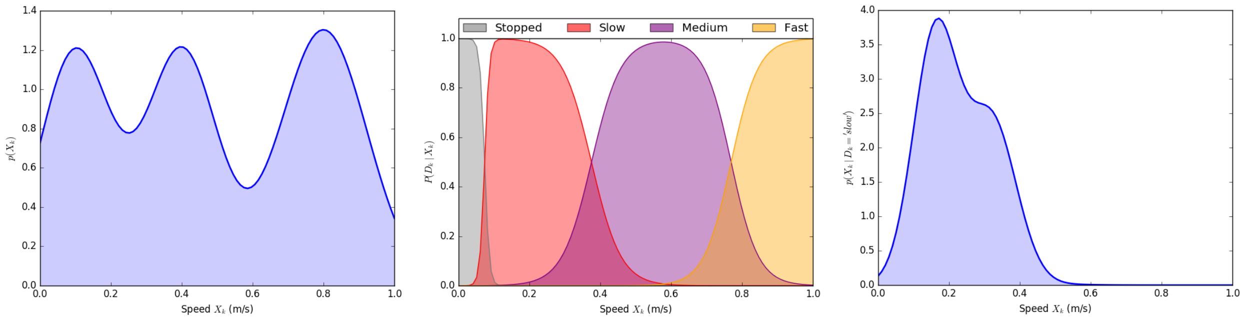
X_k) specified by *class label* i relative to all m class labels within the complete *softmax model*. Specification of these parameters is crucial and discussed further in both this and later sections. Each softmax class is convex in X_k, thus, unimodal. To represent multimodal likelihoods, we can sum over multiple unimodal classes to generate a multimodal softmax (MMS) model:

$$P(D_k = i \mid X_k) = \frac{\sum_{r \in \sigma(i)} e^{w_r^T X_k + b_r}}{\sum_{j=1}^{S} e^{w_j^T X_k + b_j}} \tag{2.7}$$

where the numerator now takes the sum over each subclass r in the set of subclasses $\sigma(i)$ for each class i, and $S = \sum_{j=1}^{m} |\sigma(j)|$ represents the total number of subclasses.

An example of a two-dimensional softmax model with joint range and bearing semantic class labels is shown in Fig. 2.1a. A MMS representation is shown in Fig. 2.1b, where the range-only classes (i.e. *near*) are obtained by summing over all joint range-bearing subclasses containing the parent class (i.e. *near east*, *near north-east*, etc.). For simplicity, we will consider both cases of softmax as well as MMS likelihoods when referring to softmax likelihoods, unless otherwise specified.

Figure 2.2 illustrates a simple one-dimensional case of VBIS data fusion approximation to (2.4). The softmax class label *slow* (corresponding to the red curve in Fig. 2.2b) is observed.



(a) Gaussian mixture prior.

(b) Softmax model likelihoods.

(c) Gaussian mixture posterior.

Figure 2.2: Example of VBIS fusion for a 1D state space with a semantic observation of ‘slow’, shown in red.

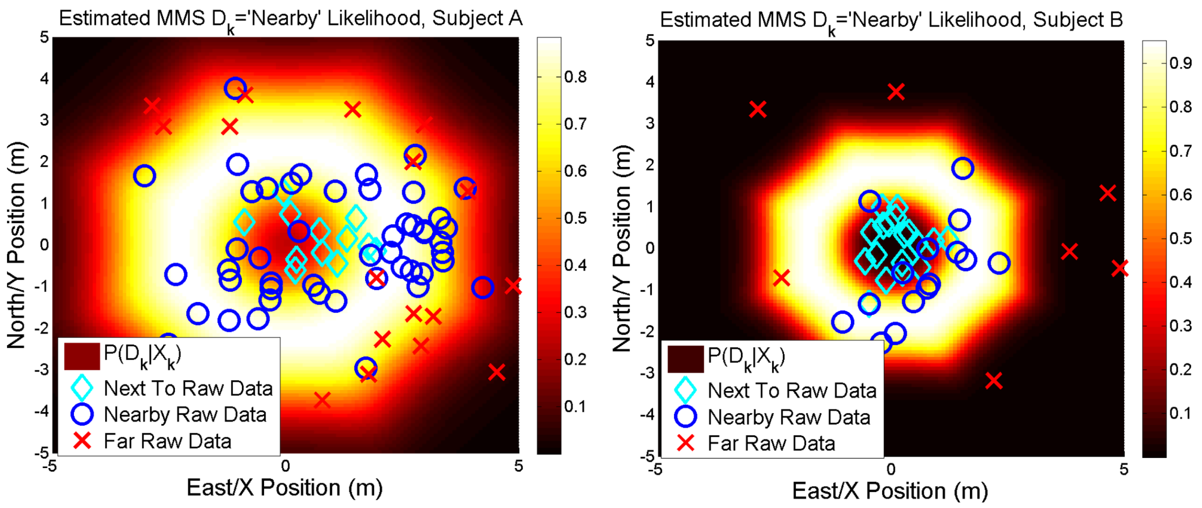
With multiple hypotheses represented in both the prior and posterior GMs, we see a significant information gain from the data fusion of with the selected semantic observation – in this example, we are far more confident that the target’s speed is close to 0.2 after the observation data has been fused.

While the softmax function is shown to be an ideal method of mapping semantics to likelihood functions, what remains to be seen is how to specifically construct these softmax likelihoods – how do we select parameters to shape softmax classes that best represent semantic uncertainty?

2.1.2 Softmax Parameter Learning

One approach to defining softmax parameters is to use training data from human subjects to ‘calibrate’ the mapping of semantics to state space. Maximum likelihood learning can be performed to find softmax parameters Θ_i that best fit the labeled training data. This is the inverse of classification, which asks: given a state, which semantic labels best describe that state? We assume the human can perceive the state (perhaps only a portion of the complete state), and provides a classification label that probabilistically relates to each state variable.

For example, Fig. 2.3 shows learned probabilities for the semantic label *nearby* for two test subjects from [24]. Each subject was tasked with labeling the spatial relation between multiple objects. X_k, the 2-dimensional distance vector between objects, was known exactly and the human



(a) (b)

Figure 2.3: Learned probability of *near* for two different test subjects. Reprinted from [24], ©2013 IEEE..

was restricted to range-only descriptors *next to*, *nearby* and *far*. Gradient descent in parameter space was used to optimize randomly initialized parameters. The end result is a subject-specific softmax model that provides data-calibrated likelihood functions for three range-only descriptors.

However, parameter learning techniques have several drawbacks, largely related to their inability to generalize models in both size and shape. Scaling softmax models based on relative reference object size is critical for capturing contextual information in semantic statements (i.e. can *near* a square desk be mapped to *near* a square building?). Similarly, the experiment results of Fig. 2.3 shows softmax parameters learned for the comparison of two points; what if each object had some larger geometry? The shape of “beside a river” would clearly differ from the shape of “beside a tree”, and it would be useful to transfer the notion of *beside* between the two rather than retraining on every new reference object. As well, dense datasets are required to generate for well-formed softmax likelihoods, as each softmax model contains $|\Theta| = m(n+1)$ parameters (this becomes $|\Theta| = S(n+1)$ for MMS models).

Related to this, we have so far assumed mutual exclusion in semantic class labels – this is an approximation: *near* is not mutually exclusive with *far*, rather, it is mutually exclusive with *not* *near*. A binary softmax model (BSM), composed of one label and its negation, would be a more accurate representation of semantics, but would require exponentially more data to learn as both a semantic label and its negation would require training data (i.e. data for *far* no longer informs parameters for *near*). Additionally, we have only discussed single-observation softmax likelihoods. Semantic language provides rich information that may contain multiple observations in one set of semantic signifiers, but this is not well-handled by current softmax models. Chapter 3 will provide insight into how we can use properties of the softmax function to resolve these issues and create generalizable softmax functions that capture rich semantic statements with reference to contextual information.

As well, we have only considered the softmax likelihood $P(D_k = i \mid X_k)$ based on a known semantic class label indexed by i. This assumes we are able to uniquely identify i from some semantic language input; while natural language has associated structure, this is not a trivial problem. Ahmed [24] bypasses this issue by allowing humans to select archetypal semantic statements from a fixed dictionary, but this approach is rigid and unscalable. Chapter 4 provides some developments towards incorporating chat-based unstructured natural language sensory inputs in human-robot teaming.

2.2 Related Work

Literature related to the problem of providing natural language inputs as soft data for autonomous systems typically falls into one of two categories: either developing human sensor models, or providing natural language commands robot teammates.

2.2.1 Human Sensor Models

Kaupp et al. [10] provided one of the early works in defining a model of a human sensor focused on mapping between human-specified quantitative measurements to state-space observation likelihoods. They encode the soft data observation likelihood as a unimodal conditional Gaussian, which can be learned as described in section 2.1.2. Decentralized Bayesian data fusion using this quantitative human sensor model was experimentally validated in later work [40]. While this conditional Gaussian likelihood function relates human-provided quantitative/continuous observations to a continuous state space, it is neither able to relate qualitative/categorical labels to a continuous state space nor able to consider multimodal observations (i.e. requiring both range and bearing simultaneously, rather than range-only measurements). The ideal interface would allow for both quantitative and qualitative multimodal inputs from human sensors.

Frost et al. [26] took steps towards developing a human sensor model based on qualitative spatial language observations. Their method defines model likelihoods on an occupancy grid map with variable-sized cells, defining functions that modify the likelihood that each cell is described by a spatial relation. However, these functions are defined on a case-by-case basis: the likelihood for *between* two reference objects is exactly 1 for each grid cell within the convex hull, and informally defined outside of the convex hull as a function of distance to the center of the convex hull. Problems associated with grids notwithstanding, limiting qualitative semantic information to spatial relations is limiting, and more rigor is required when defining likelihood functions.

Similar to the tradeoff between accuracy and computation provided by Gaussian mixtures, it becomes clear that data-driven approaches provide representational fidelity at the cost of dense learning requirements, whereas functional approaches provide simple models that may not be well-mapped to an individual human's semantic meaning. The argument presented in section 3.2 is that functional constraints may be imposed, alleviating data requirements for model parameter learning.

2.2.2 Natural Language Commanding

Delegation mechanisms have received more attention than information-providing mechanisms in human-robot teaming literature, although progress in one domain is often applicable to both. The primary difference between natural language commanding and natural language sensing is in the Bayesian inference problem: commanding requires that a single action be taken, which means taking the most likely hypothesis from Bayesian inference; conversely, sensing need not be constrained to considering a single hypothesis, so many natural language commanding techniques throw away information that may be valuable to a natural language sensing approach. Nevertheless, lessons learned from task delegation through natural language commands can be applied to providing soft data through natural language sensing. The core difficulty in both commanding and sensing is in *grounding* natural language to meaning – how do the words in the phrase, "Go to the third door on the left," map to actions and objects in the physical world?

Matuszek et al. [15] focuses on a semantic parsing of natural language commands to generate sequences of desired actions to be taken in an uncertain environment. This semantic parser uses experimental training data to define a distribution over a fixed set of robot control actions given some natural language input. Language grounding is resolved through a probabilistic combinatory categorical grammar (PCCG) which can be trained on natural language and semantic command sentence pairs. Features trained include lexical features related to the natural language input and semantic features related to a robot control language, but these are both based on binary co-occurrence statistics between natural language and commands, requiring either significant amounts of training data or a limited lexicon.

Similar work by Tellex et al. [17] approaches the grounding problem through the development of a generalized grounding graph (G^3): a bipartitie factor graph relating pre-categorized natural language input to available figures, spatial relations and landmarks. Training data is collected to learn weights for feature functions relating groundings, natural language inputs and known binary correspondence variables between the two. This produces a massive joint inference problem over all groundings and action sequences available; the approximate solution is found through a beam search over natural language-grounding pairs. Following this work, distributed correspondence graph (DCG) model is presented in [19] to infer constraints on the action space rather than finding the optimal action sequence, significantly minimizing the computational cost of inference without accuracy loss. More recently, a hierarchical distributed correspondence graph (HDCG) extension [41] imposes rules to limit the search space of correspondences between natural language and groundings, providing inference computation speeds permissible for online computation.

These models for grounding natural language commands provide a framework for development of natural language sensing. First steps towards a full natural language human sensor model are discussed in chapter 4.

Chapter 3

Likelihood Function Generation

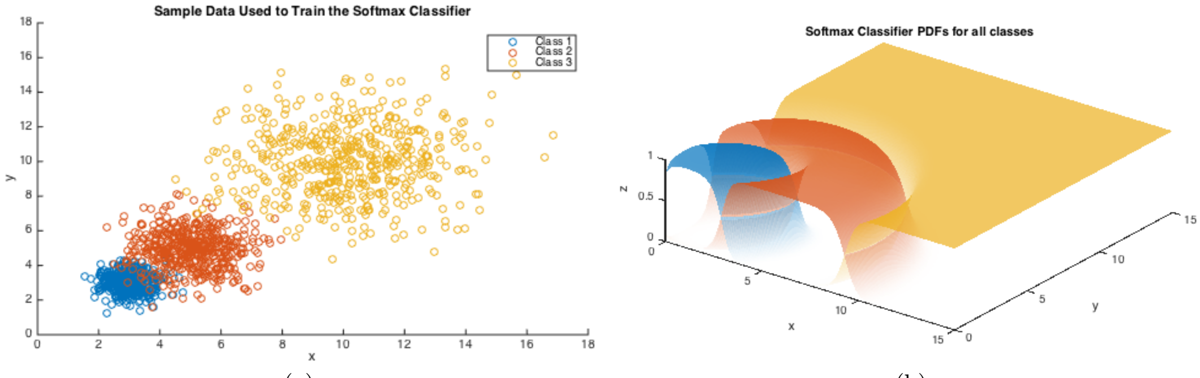
The softmax function, also known as the normalized exponential or the logistic function, provides a smooth decomposition of the state space. We argue this function is key in the representation of semantics for Bayesian data fusion. This chapter begins with a discussion some key properties of the softmax function, particularly in the derivation of equiprobable log-odds hyperplanes that uncover structure in the softmax model. In the second section, this hidden structure is exploited to establish a technique capable of synthesizing softmax models using no or minimal training data. Model synthesis techniques can then be used to compress information-dense multi-observation likelihoods for online fusion, as discussed in the final section.

3.1 Properties of the Softmax Function

Section 2.1.2 identified the key issue with generating softmax functions to represent semantic information: parameter selection. We investigate the softmax function, (2.6), in greater detail to determine properties that may be helpful in either constraining or fully specifying parameters for a given softmax model.

3.1.1 Nonlinear Normalized Exponentials

We have so far been discussing the linear formulation of the softmax model, where we assume a linear state representation. In general, it is possible to construct arbitrary normalized exponentials using nonlinear transformations on state variables:



(a) (b)

Figure 3.1: Example normalized exponential function with quadratic state variables.

$$P(D_k = i \mid X_k) = \frac{e^{w_i^T \phi(X_k) + b_i}}{\sum_{j=1}^{m} e^{w_j^T \phi(X_k) + b_j}} \tag{3.1}$$

Where the *basis function* $\phi(X_k)$ performs a transformation on the state. For example, a quadratic form, in which $\phi(X_k) = \phi\left(\begin{bmatrix} x & y \end{bmatrix}\right) = \begin{bmatrix} x & y & xy & x^2 & y^2 \end{bmatrix}$, is shown in Fig. 3.1. This is useful when considering highly nonlinear state decompositions, as well as synthesizing softmax models around nonlinear constraints. However, in many cases, it also leads to a greater number of parameters to specify, since $w_i, \phi(X_k) \in \mathbb{R}^d$. For simplicity, we will continue to use linear exponential functions, $\phi(X_k) = X_k$, for our softmax models, with the understanding that the linearity is not a limiting factor in model generation, as a greater number of subclasses may be introduced to approximate nonlinear features for MMS models.

Regardless of the dimensionality of the state, the softmax parameters Θ_i are comprised of weights w_i and biases b_i for each class i. The bias term is a state-independent offset, whereas the weights scale the influence of state variables. Alternative notations combine the bias term with the other weight parameters, then augment the state variable such that $X_k' = \begin{bmatrix} X_k^T & 1 \end{bmatrix}^T$. Our notation maintains the separation of weights and bias to underscore the parallels between weights/biases and the slope/offset representation for linear hyperplanes, as will be discussed in section 3.1.4.

3.1.2 Parameter Relativity

We can also demonstrate the *relativity* of these softmax parameters - any class associated with one set of parameters is dependent on the parameters of all other classes within the model. This allows for modification of all model parameters while maintaining the same likelihood function:

$$\begin{aligned}
P(D_k = i \mid X_k; \Theta) &= \left(\frac{e^{w_i^T X_k + b_i}}{\sum_{j=1}^{m} e^{w_j^T X_k + b_j}} \right) \left(\frac{e^{v^T X_k + a}}{e^{v^T X_k + a}} \right) \\
&= \frac{e^{(w_i + v)^T X_k + (b_i + a)}}{\sum_{j=1}^{m} e^{(w_j + v)^T X_k + (b_j + a)}} \\
&= \frac{e^{(w_i')^T X_k + b_i'}}{\sum_{j=1}^{m} e^{(w_j')^T X_k + b_j'}} \\
&= P(D_k = i \mid X_k; \Theta')
\end{aligned} \tag{3.2}$$

where $\Theta' = \{\{w_1', b_1'\}, \{w_2', b_2'\}, \ldots, \{w_m', b_m'\}\}$, $w_i' = w_i + v$ and $b_i' = b_i + a$. Thus, when specifying parameters for a model with m classes, we only need to specify $m - 1$ parameter sets. This is often useful as we can simply set one class' parameters to zero and offset the remainder.

3.1.3 Parameter Transformations

The relativity of parameters allows us to perform transformations rotational and translational on the softmax model as a whole. For instance, if we translate the state space by some β, we can arrive at a translated state space $X_k' = X_k + \beta$, where $\beta \in \mathbb{R}^n$. But, can we represent this shift simply by adjusting our parameters, rather than redefining our state vector? If we examine $P(D_k = i \mid X_k'; \Theta)$ in the translated state space, we see the following:

$$\begin{aligned}
P(D_k = i \mid X_k'; \Theta) &= \frac{e^{w_i^T (X_k + \beta) + b_i}}{\sum_{j=1}^{m} e^{w_j^T (X_k + \beta) + b_j}} \\
&= \frac{e^{w_i^T X_k + w_i^T \beta + b_i}}{\sum_{j=1}^{m} e^{w_j^T X_k + w_j^T \beta + b_j}} \\
&= \frac{e^{w_i^T X_k + b_i'}}{\sum_{j=1}^{m} e^{w_j^T X_k + b_j'}}
\end{aligned} \tag{3.3}$$

Where $b_j' = b_j + w_j^T \beta$ for $j = 1, 2, \ldots, m$. This retains our original state and modifies only our biases. Similarly, we can apply a rotation to the translated state space, to get a fully transformed softmax model:

$$X_k'' = R(\theta)X_k' = R(\theta)(X_k + \beta) \tag{3.4}$$

Where $R(\theta)$ is some n-dimensional rotation matrix. Again, we can apply this transformation to the parameters rather than the state:

$$\begin{aligned} P(D_k = i \mid X_k''; \Theta) &= \frac{e^{w_i^T(R(\theta)X_k + R(\theta)\beta) + b_i}}{\sum_{j=1}^m e^{w_j^T(R(\theta)X_k + R(\theta)\beta) + b_j}} \\ &= \frac{e^{w_i^T R(\theta)X_k + w_i^T R(\theta)\beta + b_i}}{\sum_{j=1}^m e^{w_j^T(R(\theta)X_k + R(\theta)\beta) + b_j}} \\ &= \frac{e^{(w_i'')^T X_k + b_i''}}{\sum_{j=1}^m e^{(w_j'')^T X_k + b_j''}} \end{aligned} \tag{3.5}$$

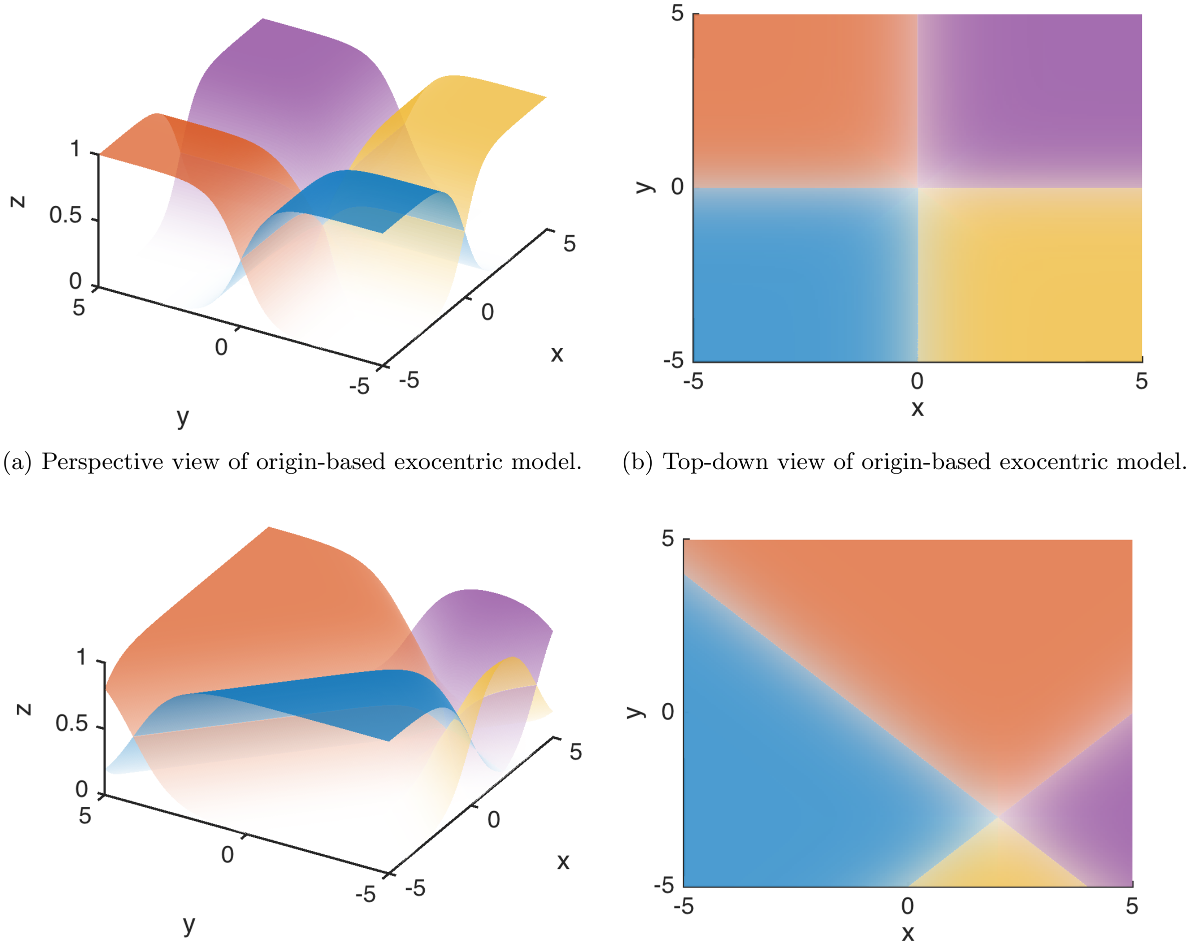
Where,

$$\begin{aligned} w_j'' &= R(\theta)^T w_j \\ b_j'' &= w_j^T R(\theta)\beta + b_j \end{aligned} \tag{3.6}$$

We can manipulate any softmax model by translation and rotation as necessary. This is important when grounding semantics to physical objects with known positions. Many semantic relations are either egocentric or exocentric – dependent on the observer's viewpoint or the inertial viewpoint. For instance, the egocentric *to my right* would require some base model of *right* and *left* to be transformed to the current position and orientation of the observer, while the exocentric *north of the kitchen* could be defined once and fixed.

3.1.3.1 Example

If we want to specify the location of a target, one option is to use the four intercardinal directions (*northeast*, *southeast*, *southwest* and *northwest*). These map to a 2-dimensional state space, $X_k = \begin{bmatrix} x & y \end{bmatrix}^T$, and can be considered wither exocentrically or egocentrically. In this simple problem, weights correspond to each quadrant of the Cartesian plane, thus:



(a) Perspective view of origin-based exocentric model.

(b) Top-down view of origin-based exocentric model.

(c) Perspective view of reference-based egocentric model.

(d) Top-down view of reference-based egocentric model.

Figure 3.2: Softmax model of intercardinal directions undergroing transformation from global frame to ego frame.

$$w_{SW} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad b_{SW} = 0$$

$$w_{NW} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad b_{NW} = 0$$

$$w_{SE} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad b_{SE} = 0$$

$$w_{NE} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad b_{NE} = 0$$

These weights correspond to the model shown in figs. 3.2a and 3.2b. All semantics are currently with reference to the origin; however, we can specify them with relation to some arbitrary translated and rotated point to provide egocentric measures.

Given a reference pose of $p = (2, -3, \frac{\pi}{4})$ representing (x, y, θ_r) in (m, m, clockwise rad from $+y$), we can find the egocentric parametrization by using $\beta = \begin{bmatrix} -x & -y \end{bmatrix}$ and $\theta = \theta_r$ with (3.6):

$$w''_{SW} = \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix}^T \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -\sqrt{2} \\ 0 \end{bmatrix}$$

$$w''_{NW} = \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix}^T \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix}$$

$$w''_{SE} = \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ -\sqrt{2} \end{bmatrix}$$

$$w''_{NE} = \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix}^T \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$$

$$b''_{SW} = \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix} \begin{bmatrix} -2 \\ 3 \end{bmatrix} = 2\sqrt{2}$$

$$b''_{NW} = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix} \begin{bmatrix} -2 \\ 3 \end{bmatrix} = 3\sqrt{2}$$

$$b''_{SE} = \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix} \begin{bmatrix} -2 \\ 3 \end{bmatrix} = -3\sqrt{2}$$

$$b''_{NE} = \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & -\sin\left(\frac{\pi}{4}\right) \\ \sin\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) \end{bmatrix} \begin{bmatrix} -2 \\ 3 \end{bmatrix} = -2\sqrt{2}$$

Using these parameters, we achieve the result in figs. 3.2c and 3.2d. – a translated and rotated softmax model.

3.1.4 Log-odds Hyperplanes

For any two classes, we can take the ratio of their probabilities to determine the odds of one class with respect to the other:

$$L(i,j;X_k) = \frac{P(D=i|X_k)}{P(D=j|X_k)} = \frac{\frac{e^{w_i^T X_k + b_i}}{\sum_{c=1}^{m} e^{w_c^T X_k + b_c}}}{\frac{e^{w_j^T X_k + b_j}}{\sum_{c=1}^{m} e^{w_c^T X_k + b_c}}} = \frac{e^{w_i^T X_k + b_i}}{e^{w_j^T X_k + b_j}} \tag{3.7}$$

When $L(i,j;X_k) = 1$, the two classes have equal probability, so any X_k for which $L(i,j;X_k) = 1$ is part of the *equiprobable level set* of classes i and j. We can take the logarithm to get the log-odds ratio:

$$log\left(L(i,j;X_k)\right) = (w_i^T X_k + b_i) - (w_j^T X_k + b_j) = (w_i - w_j)^T X_k + (b_i - b_j) \tag{3.8}$$

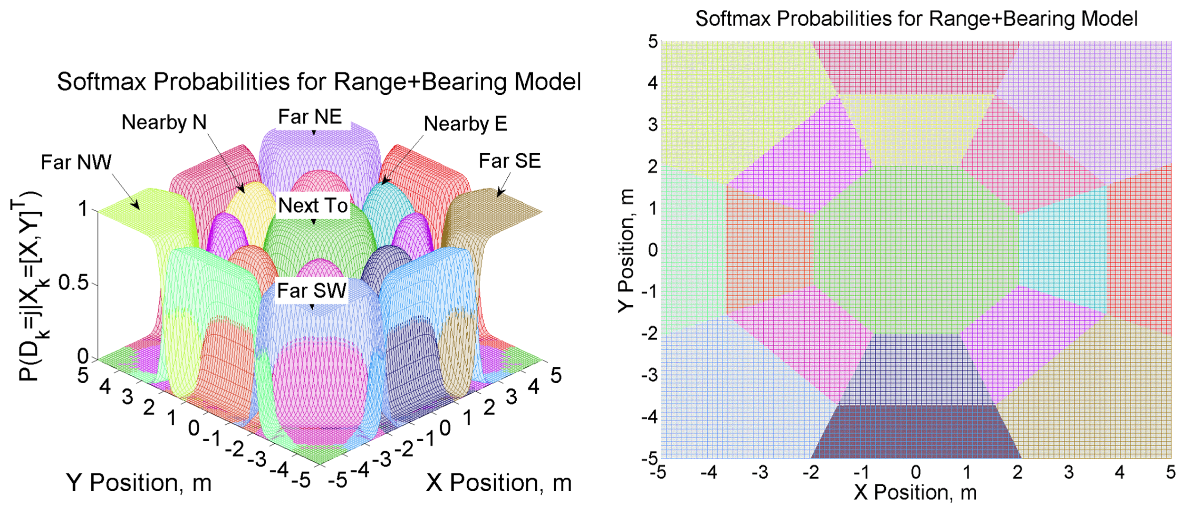
When $log\left(L(i,j;X_k)\right) = 0$, the equiprobable level set takes the form of a linear hyperplane. These log-odds hyperplanes are defined for all combinations of classes i, j if $i \neq j$, and are critical in the synthesis of softmax models covered in section 3.2.

3.1.5 Softmax Model Regions

Given that any non-MMS softmax class is convex, finding the most likely labels over state space partitions the space into convex regions, in which each region maps to a class that has greater probability than any other class. This is shown in Fig. 3.3, where each region is represented by a different color. Furthermore, any border between two dominant regions is necessarily a linear hyperplane, as defined by (3.8). Note, however, that the log-odds hyperplanes between *any* two classes can be defined (e.g. between *far northeast* and *far southwest*) – these hyperplanes do not only bound dominant regions.

3.1.6 Binary Softmax Models

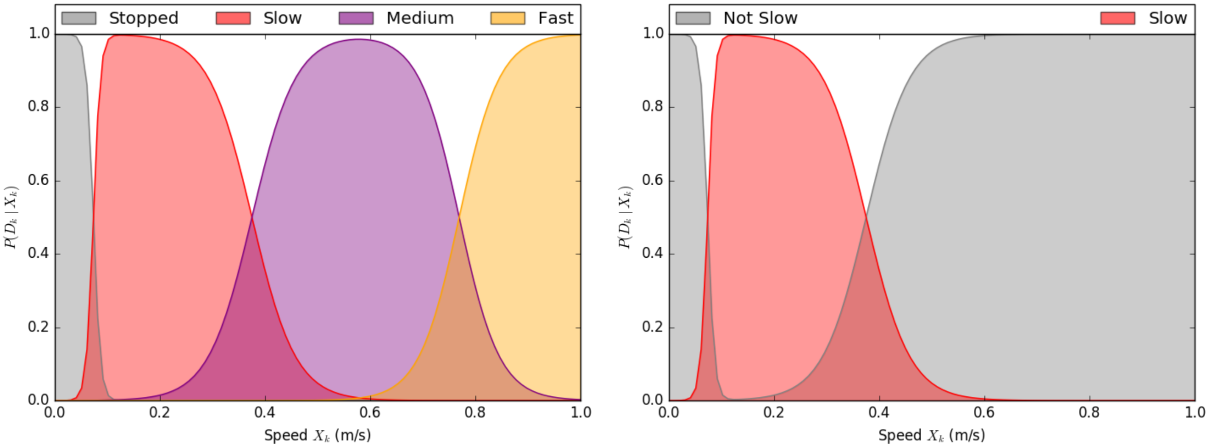
Observations of semantic language mapping to class labels in a softmax model are deemed mutually exclusive events – this is the traditional categorical softmax model. This model definition



(a) Softmax model of joint range and bearing labeled classes.

(b) Dominant softmax class regions.

Figure 3.3: Projection of softmax likelihoods from (a) onto state space, shown in (b). Reprinted from [24], ©2013 IEEE.



(a) Categorical speed model.

(b) Binary softmax model for *slow*.

Figure 3.4: Conversion of a categorical speed model into a BSM.

is problematic when representing non-mutually exclusive semantics; for instance, *near*, *next to*, *beside* and *around* are clearly not mutually exclusive spatial relations. The only semantic statement which is affirmatively mutually exclusive with *near* would be *not near*. In general, this means that semantics should be considered as binary mutually exclusive pairs, modeled through a BSMs, with only the semantic label and its negation as the two composing classes.

However, this increases the training data requirement for parameter learning (since we would need sufficient training data for each class as well as its negation). As an approximation, it is possible to simply take a categorical model and construct a binary MMS model containing that class and a summation over all other classes as the other class. This maintains the assumption of mutual exclusivity between all semantic class labels, but allows for an approximate definition of negative terms for each possible semantic label for D_k. An example using 1-dimensional speed state space is shown in Fig. 3.4.

Given the mutual exclusion assumption between labels, it becomes key that the categorical softmax models be defined with a set of archetypal terms (such as *next to*, *near*, and *far* for range descriptions). In other words, we define categorical softmax models without synonymous labels. This greatly limits the potential semantic labels that can be used, but also limits the amount of training required, as models need not be trained on the entire set of possible semantically meaningful labels (nominally, prepositional phrases in English). In learning categorical softmax models, as well as their negations through the BSM representation, we naturally limit semantic language to a predefined dictionary of learned archetypal labels; section 4.1 provides an approach that allows for arbitrary semantic labels to be used.

3.2 Constraint-based Softmax Synthesis

With these properties of the softmax function defined, we can now begin to consider the problem of specifying softmax parameters with little or no training data. We note that, since human sensor statements focus on describing grounded language, we may consider other sources of knowledge as a method of generating softmax models; for instance, if we know the shape of a desk, we may be able to use its geometry to generate a softmax model without requiring any training. Building on the log-odds hyperplane and dominant region properties discussed in section 3.1, we consider the converse of the result proved in [42]: can a Θ always be found to embed any set of probabilistic polytopes within a softmax model such that X_k is completely partitioned? What follows is a description of this ‘constraint-based softmax synthesis problem’, based on previous work [43].

We define the constraint-based synthesis problem as the following: given a set of m semantic (sub)class labels with 1-to-1 correspondence to m convex probabilistic polytopes in a complete mutually exclusive convex decomposition of X_k, find the parameters Θ that produce log-odds boundaries for the desired polytopes. Unlike purely learning-based methods, such as shown in section 2.1.2, the synthesis problem here relies only on a given polytope geometry specification to identify Θ (which may not be unique). This a priori geometric polytope information is often available in the context of semantic spatial sensing [44], and can be used to define human sensor models. Suppose we are given $\mathcal{B} = \left\{n_{ji}^T, c_{ji}\right\}$ where

$$n_{ji}^T X_k + c_{ji} = 0 \tag{3.9}$$

and where $n_{ji} \in \mathbb{R}^n$ and $c_{ji} \in \mathbb{R}$ for $i, j \in \{1, ..., m\}$ where $i \neq j$. Let $\mathcal{B}$ define a set of surface normal parameters describing the log-odds hyperplanes for the faces of m convex polytopes, which together form a complete mutually exclusive convex decomposition of X_k. Since the state space may be unbounded, assume the polytopes may be unbounded as well (e.g. the non-negative orthant). Assume these convex polytopes correspond to desired dominant regions for class (or subclass) labels $i, j \in \{1, ..., m\}$, such that each face of the class i polytope corresponds to the equiprobability level set with its neighbouring class j (i.e. the faces correspond to the log-odds hyperplanes defined by (3.8)). $\mathcal{B}$ here defines only the desired boundaries for i and j whose dominant region polytopes are neighbors in X_k. We denote the set of all neighboring class label pairs (i, j) that have boundaries specified by $\mathcal{B}$ as $S_{\mathcal{B}}$. The constraint-based softmax synthesis problem is then to find softmax parameters $\Theta = \{w_c, b_c\}_{c=1}^m$ such that the log-odds boundaries for $i, j \in \{1, ..., m\}$ correspond exactly to the specified boundaries in $\mathcal{B}$. Equating the left-hand side of (3.9) and (3.8) when $log\left(L(i, j; X_k)\right) = 0$,

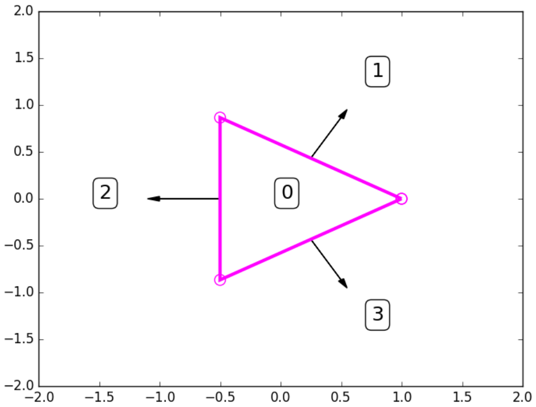


Figure 3.5: Example geometry specification of a regular triangle.

$$\begin{aligned} \Delta w_{ji}^T X_k + \Delta b_{ji} = n_{ji}^T X_k + c_{ji} = 0 \\ \Rightarrow w_j - w_i = n_{ji}, \quad b_j - b_i = c_{ji} \quad \forall (i,j) \in S_{\mathcal{B}}, \end{aligned} \tag{3.10}$$

Where $\Delta w_{ji} = w_j - w_i$ and $\Delta b_{ji} = b_j - b_i$. If $S_{\mathcal{B}}$ defines N_S boundary pairs, then (3.10) leads to a system of $N_S(n+1)$ linear equations that must be satisfied by Θ.

We define the stacked parameter vector as,

$$\vec{\theta} = [w_1^T, b_1, w_2^T, b_2, ..., w_m^T, b_m]^T$$

and the stacked vector of desired polytope boundary surface normal parameters as,

$$\vec{\beta} = [n_{v(1)}^T, c_{v(1)}, n_{v(2)}^T, c_{v(2)}, ..., n_{v(N_S)}^T, c_{v(N_S)}]^T$$

where $v(t)$ denotes the t^{th} pair of (i,j) class labels in $S_{\mathcal{B}}$ for $t \in \{1, ..., N_S\}$. Then, the required equations can be written as $M\vec{\theta} = \vec{\beta}$, where each row of $M \in \mathbb{R}^{N_S(n+1) \times m(n+1)}$ performs the appropriate differencing operations on $\vec{\theta} \in \mathbb{R}^{m(n+1)}$ to obtain $\vec{\beta} \in \mathbb{R}^{N_S(n+1)}$.

To better understand the relationship between geometry and parameters, take the triangle polytope specification $\mathcal{B}$ shown in Fig. 3.5. The triangle decomposes the state space into two regions (*inside* and *outside* the triangle), which can be decomposed into four convex regions: region 0 at the interior of the polytope; and regions 1 through 3 defined on one side by each polytope face and unbounded elsewhere. The magnitude of the normal of each face can be used to determine the boundaries between unconstrained regions (e.g. regions 2 and 3). Our equation $M\vec{\theta} = \vec{\beta}$ mapping the known polytope normals to unknown weights becomes:

$$\begin{bmatrix} c_{1,0} \\ n_{1,0}^T \\ c_{2,0} \\ n_{2,0}^T \\ c_{3,0} \\ n_{3,0}^T \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ w_0^T \\ b_1 \\ w_1^T \\ b_2 \\ w_2^T \\ b_3 \\ w_3^T \end{bmatrix} \tag{3.11}$$

We have 9 equations and 12 unknowns, given that $n, w \in \mathbb{R}^2$. However, recall the relativity of softmax parameters as discussed in section 3.1.2: we can simply set $b_0 = 0$ and $w_0 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, then define a set of weights relative to the arbitrarily determined interior class. This allows us to easily specify weights and biases as $w_i = n_{i,0}$ and $b_i = c_{i,0}$ for $i = 1, 2, \ldots, m-1$.

In general, given $(\mathcal{B}, S_{\mathcal{B}}) \rightarrow (\beta, M)$, the existence and uniqueness of solutions to constraint-based softmax synthesis follows from the Rouché-Capelli theorem. If $A = [M, \vec{\beta}]$, then the following solutions $\vec{\theta}$ are possible:

(1) $\text{rank}(A) = \text{rank}(M) = m(n+1) \Leftrightarrow$ unique solution

(2) $\text{rank}(A) = \text{rank}(M) < m(n+1) \Leftrightarrow$ infinitely many solutions

(3) $\text{rank}(A) \neq \text{rank}(M) \;\; \Leftrightarrow$ no exact solutions

Hence, at least one solution $\vec{\theta}$ exists for the softmax synthesis problem iff $\vec{\beta}$ lies in the span of the columns of M, whose only entries are 1,0, and -1 (cases 1 and 2). Otherwise, For inconsistent specifications (case 3), $\vec{\theta}$ does not have enough degrees of freedom to produce the desired polytope boundaries for m classes, meaning that a greater number of classes *must* be created for a consistent model. Certain log-odds boundary specifications between m convex polytopes are thus not realizable using only m softmax parameters. In fact, for solutions to exist, the following 'loop summation constraint' must hold (analagous to Kirchoff's votage law),

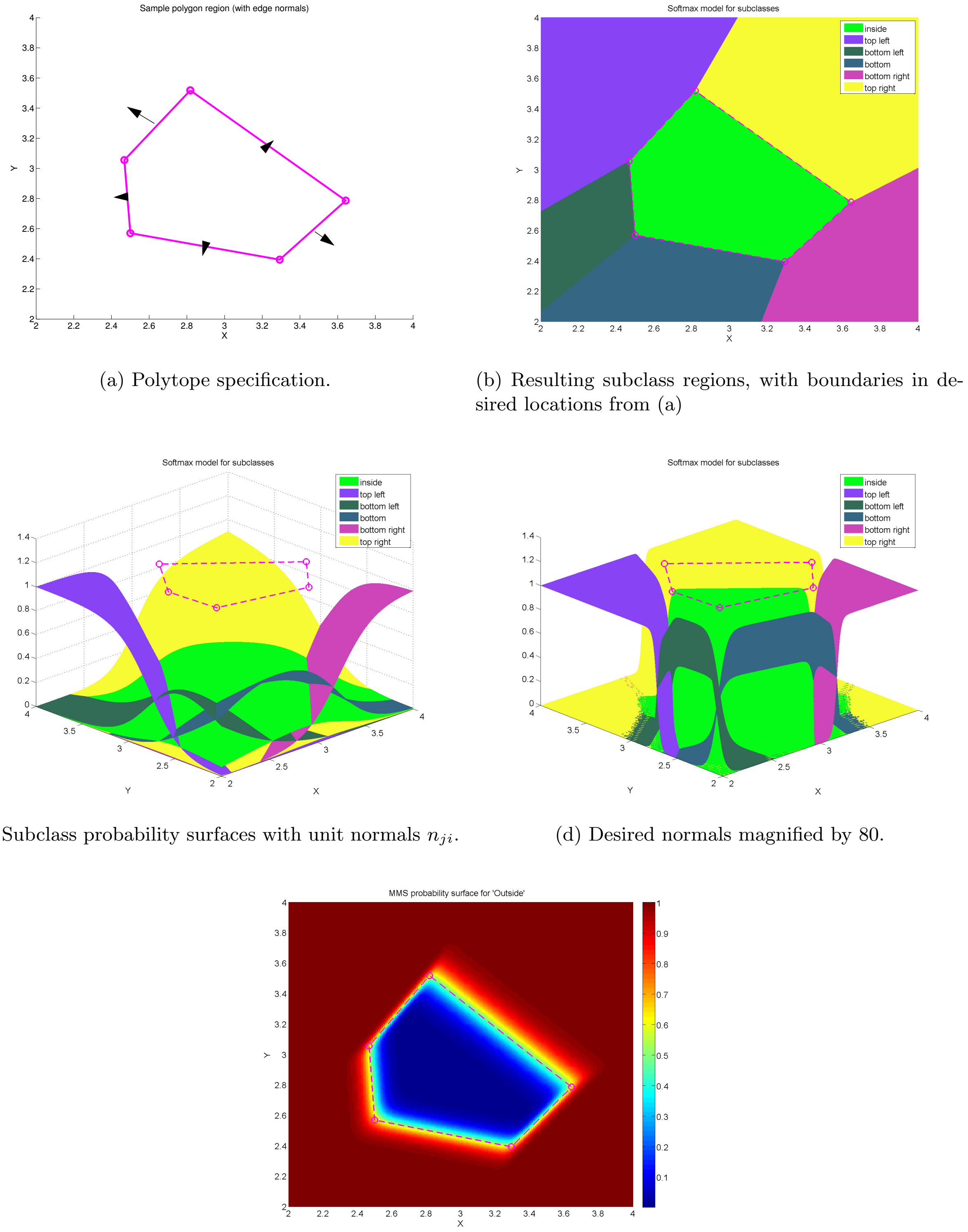
$$\begin{aligned} &\Delta w_{jy_1} + \Delta w_{y_1y_2} + \Delta w_{y_2y_3} + ... + \Delta w_{y_h,i} + \Delta w_{ij} = 0 \\ \Rightarrow\ & n_{jy_1} + n_{y_1y_2} + n_{y_2y_3} + ... + n_{y_h,i} + n_{ij} = 0 \end{aligned} \tag{3.12}$$

where (3.12) follows from (3.10), and the indices y_r for $r = 1, ..., h$ and $2 < h \leq m$ denote any subset of class indices other than i and j (similarly for Δb_{ji} and c_{ji}). Note that the Δw constraints are automatically satisfied for any given set of softmax model parameters. However, if the desired polytope boundaries in $\mathcal{B}$ do not satisfy (3.12) for any subset of class labels $\{i, j, y_1, ..., y_h\}$, then no $\vec{\theta}$ exists such that $M\vec{\theta} = \vec{\beta}$ using only m softmax classes.

3.2.1 Example

Fig. 3.6 shows an irregular convex polygon for defining *inside* and *outside* spatial likelihood model regions. Since the *outside* region is non-convex, this requires an MMS model featuring at least 5 softmax subclasses to make up the *outside* class. Each *outside* subclass corresponds to one polygon face: *top left*, *top right*, *bottom left*, *bottom right*, and *bottom*. A single subclass makes up the *inside* class. Fig. 2(a) shows the desired normals n_{ji} for each face between the *outside* and *inside* classes polytopes (black arrows), indicating that the likelihood model should force the *inside* (sub)class to become less probable at the perimeter, but dominate the interior.

This specification of $N_s = 5$ polytope faces for $m = 6$ subclasses with dimensionality $n = 2$ leads to a system of $N_S(n+1) = 15$ linear equations in $m(n+1) = 18$ parameters. As with the previous example, to find a particular realization satisfying this specification, we can arbitrarily assume $w_i = 0$ and $b_i = 0$ for $i =$*inside*, leaving 15 parameters in 15 difference equations. Thus, $M = [\tilde{M}; 0]$, where $\tilde{M} = I$ (identity), and so $\vec{\theta} = \vec{\beta}$.



(a) Polytope specification.

(b) Resulting subclass regions, with boundaries in desired locations from (a)

(c) Subclass probability surfaces with unit normals n_{ji}.

(d) Desired normals magnified by 80.

(e) Non-convex MMS likelihood for *outside*.

Figure 3.6: Construction of MMS likelihood function for *outside* a given polytope. Note that internal boundaries between *outside* subclasses vanish. Reprinted from [43], ©2016 IEEE.

3.3 Multi-observation Likelihood Compression

The GM fusion approximation assumes that $P(D_k = i|X_k)$ captures all information contained within a semantic human observation D_k. However, D_k could contain mixed information about different states within X_k (e.g. target position and heading). As it is difficult to capture all possible semantic data combinations within a single MMS model, $P(D_k = i|X_k)$ could be decomposed into atomic likelihoods that model relevant semantic information after D_k is parsed by a natural language processing front-end (as will be discussed in chapter 4). For instance, if $D_k^a = i$ and $D_k^b = j$ correspond to "target near building" and "target moving quickly away from building", then the likelihood for the joint observation $D_k = ([D_k^a = i] \wedge [D_k^b = j])$ is the product of the corresponding likelihoods (assuming both are conditionally independent given X_k), $P(D_k|X_k) = P(D_k^a|X_k)P(D_k^b|X_k)$. D_k^a and D_k^b could also represent observations from different human sensors at time k.

With N_o such observations D_k^o, $o \in \{1, ..., N_o\}$, each D_k^o could be processed sequentially via repeated application of the GM approximation (2.5). However, this becomes expensive for a single time step k, since GM updating and compression must be executed for each D_k^o update. The accuracy of these steps also depends on the order in which each D_k^o is processed. We thus seek to extend the GM fusion method of [24] to handle the general case of 'batch' semantic measurement updates,

$$p(X_k|D_k^{1:N_o}) \propto p(X_k) \prod_{i=1}^{N_o} P(D_k^o = i_o|X_k). \tag{3.13}$$

Ideally, if a single softmax/MMS model $L(D_k^{1:N_o}|X_k)$ captured all information from the RHS product, then GM fusion and merging methods only need to be applied once at step k. This section describes several ways to accomplish this, using the likelihood synthesis approach described previously.

Information-dense semantic may be combined into a single batch likelihood $L(D_k^{1:N_o}|X_k)$ as long as no dynamics prediction step is required between any two component measurements (as would be the case in the example statement, “The target left the house, turned right, and walked down the street.”). We first describe compression strategies for the case where each $P(D_k^o = i_o|X_k)$ is described by a softmax model with m_o class labels. We then generalize these compression strategies to MMS component likelihoods. In both cases, the key idea is to extract the relevant boundaries for probabilistic polytopes that correspond to the intersection of dominant regions for all observed class labels. Once these boundaries are identified, they can be used to synthesize more sparsely parameterized semantic likelihoods, which lead to more computationally efficient Bayes fusion updates.

3.3.1 Compression Methods for Multiple Observations

We first examine how to exactly produce one softmax model from the product of multiple softmax likelihoods. If we assume N_o conditionally independent measurements, the combined likelihood $P(D_k^{1:N_o}|X_k)$ is,

$$\prod_{i=1}^{N_o} P(D_k^o = i_o|X_k) = \frac{e^{w_{\mathcal{I}}^T X_k + b_{\mathcal{I}}}}{\prod\limits_{l=1}^{N_o} \sum\limits_{c_l=1}^{m_l} e^{w_{c_l}^T X_k + b_{c_l}}} \tag{3.14}$$

where $\mathcal{I} = \{i_1, i_2, \ldots, i_{N_o}\}$ is the set of N_o class observations taken from the N_o softmax models, where $i_o \in \{1, ..., m_o\}$ and m_o is the number of classes for the oth softmax model. Assume w.l.o.g. that the class labels are ordered within each softmax model. The product model parameters are defined as $w_{\mathcal{I}} = \sum_{o=1}^{N_o} w_{i_o}$ and $b_{\mathcal{I}} = \sum_{o=1}^{N_o} b_{i_o}$. In general, each measurement $i_o \in \mathcal{I}$ comes from a different softmax model.

Now, (3.14) can be exactly expressed as a softmax function,

$$\frac{e^{w_{\mathcal{I}}^T X_k + b_{\mathcal{I}}}}{\prod\limits_{l=1}^{N_o} \sum\limits_{c_l=1}^{m_l} e^{w_{c_l}^T X_k + b_{c_l}}} = \frac{e^{w_{\mathcal{I}}^T x + b_{\mathcal{I}}}}{\sum\limits_{t=1}^{\bar{m}} e^{w_t^T X_k + b_t}} = L(D_k^{1:N_o}|X_k) \tag{3.15}$$

Thus, the product of N_o conditionally independent softmax likelihoods can be exactly described as a single softmax likelihood over $m_1 \times m_2 \times \cdots \times m_n = \bar{m}$ ‘product classes’, which appear in the denominator. The w_t terms cover all $\bar{m}$ combinations for the sum of the weights from the product of N_o softmax models (likewise for the b_t terms). Equation (3.15) is referred to as the *product expansion* softmax model for N_o semantic observations. Figure 3.7c shows an example of a product model for two joint semantic observations (each a shifted/scaled version of the softmax model template shown in Fig. 3.7a). When fused via VBIS with the GM prior in Fig. 3.7b, the posterior in Fig. 3.7g is obtained.

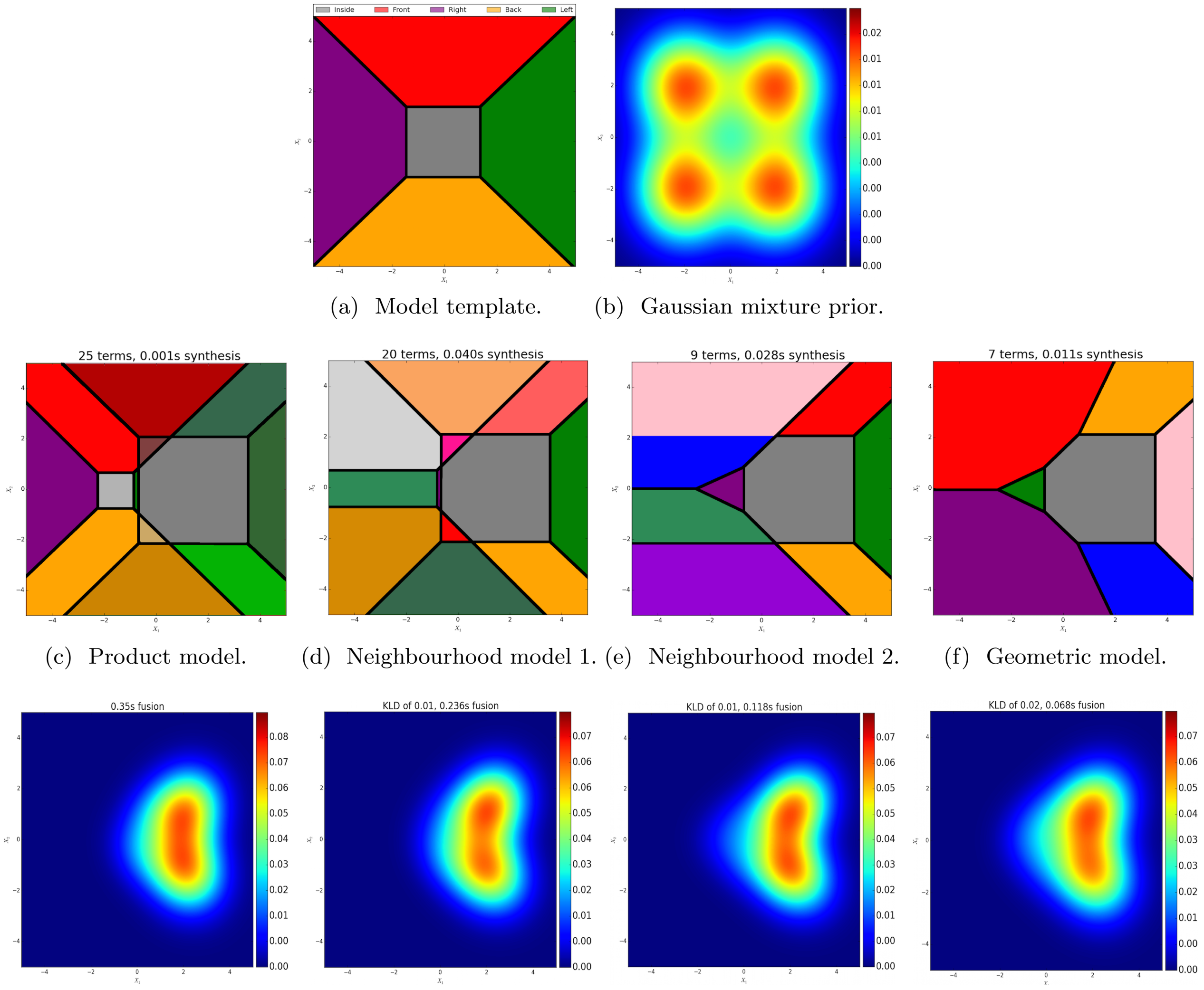
Note that $\bar{m}$ increases exponentially with N_o, making (3.15) computationally expensive in general. However, as with GM compression algorithms, this provides motivation to investigate methods for approximating (3.15) via parameter compression techniques, to balance tradeoffs between fusion accuracy and online computational efficiency. That is, we seek

$$L(D_k^{1:N_o}|X_k) \approx P(\tilde{D}_k = i^*|X_k) = \frac{e^{\tilde{w}_{i*}^T X_k + \tilde{b}_{i*}}}{\sum_{c=1}^{m_*} e^{\tilde{w}_c^T X_k + \tilde{b}_c}} \tag{3.16}$$

where $m_* \ll \bar{m}$, and $\tilde{w}_c, \tilde{b}_c$ are based on some approximation technique. The following geometric and neighbourhood compression approximations are based on the constraint-based softmax synthesis approach presented in section 3.2, and trade speed for accuracy to enable online GM fusion calculations with batch semantic measurements.

3.3.1.1 Geometric Compression

Geometric compression attempts to extract the relevant information in $L(D_k^{1:N_o}|X_k)$ according to minimal set of log-odds boundaries needed to specify the dominant region polytope of the product class that appears in the numerator of (3.15). This is equivalent to finding the intersection of the dominant region polytopes for the N_o semantic observations in $\mathcal{I}$ (where each dominant region can be unbounded). This implicitly assumes that the set of measurements in $\mathcal{I}$ is consistent, i.e. that no one $i_o \in \mathcal{I}$ contradicts any of the others; probabilistic data association techniques can be used to handle the possible 'false alarm' observations, e.g. see [24]. If any of the polytopes are non-intersecting (i.e. measurements are inconsistent, such as "Target is in front of the house,"



(a) Model template. (b) Gaussian mixture prior.

(c) Product model. (d) Neighbourhood model 1. (e) Neighbourhood model 2. (f) Geometric model.

(g) Product posterior. (h) Neighbourhood 1 posterior. (i) Neighbourhood 2 posterior. (j) Geometric posterior.

Figure 3.7: Comparison of softmax likelihood compression methods for two measurements: *right* of a small box and *inside* a large box, based on models in (a). The joint measurement class is shown in dark grey for (c)-(f). Posteriors (g)-(j) arise from fusion of the joint measurement class with the GM prior (b) via VBIS. Reprinted from [43], ©2016 IEEE.

and, simultaneously, "Target is behind the house") geometric synthesis fails to product a softmax model. We thus identify from (3.15) the relevant set of denominator parameters $\tilde{w}_c$ and $\tilde{b}_c$ defining the log-odds boundaries of the dominant region polytope for the product class i^* in (3.16).

Consider first how the dominant region for any class $i_o \in \mathcal{I}$ can be determined from the set of **all** possible log-odds boundaries that could be formed with respect to all other classes $j_o \in \{1, ..., m_o\}$ within a single softmax model. By changing the rightmost equality in (3.8) to an inequality, we define the dominant region of $i_o = i$ as

$$\mathcal{S}_i = \{X_k \in \mathbb{R}^n | G_{i,j} X_k \leq h_{i,j}, \forall j \in m_o, j \neq i\} \tag{3.17}$$

where $G_{i,j} = w_j - w_i$ and $h_{i,j} = b_i - b_j, \forall j \in m_o, j \neq i$. We may more compactly express this as $G_i X_k \preceq h_i$. The number of constraints in (3.17) is generally greater than the number needed to define $\mathcal{S}_i$; that is, some constraints generated by eq. 3.8 may not restrict $\mathcal{S}_i$. Our goal is to find the *irredundant* constraints $G'_i X_k \preceq h'_i$ to define $\mathcal{S}'_i$, the same feasible region with only the necessary constraints, and use the irredundant constraints to create an approximate softmax likelihood to the original via direct synthesis. Several techniques from the optimization literature may be used to prune redundant linear constraints; here, we use the linear programming (LP) method of Caron et al. [45] to prune log-odds-based inequality constraints.

Let $\mathcal{S}_i^j$ denote the feasible set $\mathcal{S}_i$ with the jth constraint removed. If $\mathcal{S}_i = \mathcal{S}_i^j$, then $G_{i,j} x \leq h_{i,j}$ is a redundant constraint. By iterating through all constraints, the redundant ones are identified: for each constraint, construct the LP,

$$\max z_{i,j} = G_{i,j} x, \quad \text{s.t.} \quad G_i^{(j)} x \preceq h_i^{(j)} \tag{3.18}$$

where $G_i^{(j)}$ and $h_i^{(j)}$ are the G_i and h_i matrices with row j removed, and $z_{i,j}$ is the optimal value of the maximization problem. If $z_{i,j} \leq h_{i,j}$, the jth constraint is redundant; otherwise the jth constraint is irredundant. Once we have found the irredundant constraints $G'_i x \preceq h'_i$ to define $\mathcal{S}'_i$, we can synthesize our approximate softmax model using the structured algebraic synthesis approach outlined in section 3.2, where we transform the halfspace inequality constraints $\mathcal{S}_i'$ into specifications $\mathcal{B}$ for the dominant region polytope face of the observed class i. For an l-sided convex polytope (bounded or unbounded), a minimum of $l+1$ classes are required to specify a softmax model with all critical class boundaries equivalent to the l polytope bounds. In general, we can set $w_i = \mathbf{0}$ and $b_i = 0$ to greatly simplify specification of the reduced softmax likelihood (cf. example from section 3.2). Note that this approach allows us to retain only the information necessary to reconstruct the dominant region polytope for the **observed** class i, but no other classes. Thus, the weights and biases for our newly 'compressed' softmax model are

$$\tilde{W} = [0^T, \tilde{w}_2^T, ..., \tilde{w}_l^T + 1]^T = [0^T, G'^T]^T, \tag{3.19}$$

$$\tilde{B} = [0, \tilde{b}_2, ..., b_{l+1}]^T = [0, -h^T]. \tag{3.20}$$

To prune redundant constraints from N_o observations, form $G_{\mathcal{I}}$ as a stacked matrix of G_{i_o} and $h_{\mathcal{I}}$ as a stacked vector of h_{i_o} for $o \in \{1, ..., N_o\}$ to obtain the $G_{\mathcal{I}} X_k \preceq h_{\mathcal{I}}$. Using the same reduction technique as before, define $G_{\mathcal{I}}' X_k \preceq h_{\mathcal{I}}'$. This reduced set of constraints allows us to synthesize the compressed softmax model

$$P(\tilde{D} = i^* | X_k) = \frac{1}{\sum\limits_{c \in \mathcal{M}} e^{w_c^T X_k + b_c}} \tag{3.21}$$

where $\mathcal{M}$ contains all the indices for the parameters specified by $\tilde{W}$ and $\tilde{B}$. Note that the numerator is simply 1 since the parameters of the new compressed product observation class i^* are set to 0. As shown in Fig. 3.7f, the likelihood for i^* in (3.21) will have the same dominant region boundaries as the corresponding joint observation class obtained via the product synthesis method. The likelihood surface for (3.21) will differ from the product likelihood (3.15). In this case, since we have removed additional information provided by all other classes in the product model, the geometric compression likelihood will be more conservative and the probability surfaces will be less steep while meeting log-odds equiprobability constraints. However, as Fig. 3.7j shows, such errors are often acceptable, given the decrease in softmax compression time, T_c, and GM fusion time, T_f; in the example case, $T_c + T_f = 0.085s$ using the geometric model vs. $T_c + T_f = 0.428s$ using the exact product model. This results in an (often negligible) loss in GM fusion accuracy as measured by the Kullback-Leibler divergence (KLD) of $D_{KL} = 0.027$ nats between the two results. Timing and accuracy results for all compression techniques for softmax and MMS joint observation likelihoods are summarized in Table 3.1.

3.3.1.2 Neighbourhood Compression

Neighbourhood compression extends geometric compression by retaining additional information about the dominant region polytopes for the neighbouring classes of each observed class $i_o \in \mathcal{I}$. The main idea is that parameters for unobserved classes whose dominant regions are far from those of the observed classes $i_o \in \mathcal{I}$ will not contribute significantly to (3.15), and thus can be ignored without major information loss. The neighbours of a given class in any softmax model can be determined offline via constraint reduction. For class i, we identify the irredundant constraints $G_i' x \preceq h_i'$ as above. Any class j whose weights were kept to form G' is considered to be a neighbour of class i. The neighbours of class i, $\mathcal{Y}_i$, are recorded offline, before any observations are received. When fusing multiple observations online, we perform product expansion, as in (3.15), on the measurements $\mathcal{I}$ but use only the weight/bias parameters of the measurements and their neighbours in the denominator, excluding all other parameters.

$$P(\tilde{D} = i^*|X_k) = \prod_{o=1}^{N_o} \tilde{P}_{\mathcal{Y}_o}(D_k^o = i_o|X_k) = \frac{e^{w_{\mathcal{I}}^T X_k + b_{\mathcal{I}}}}{\prod_{o=1}^{N_o} \sum_{c\in\mathcal{Y}_o} e^{w_c^T X_k + b_c}}$$

where $\mathcal{Y}_o$ contains the indices only of the associated neighbours of $i_o \in \mathcal{I}$ (including i_o itself). This new compressed model is still exponential in N_o, although with a reduced number of classes per component model. To further reduce the complexity, neighbourhood compression can be performed again, this time online, producing the a second-iteration neighbourhood model which contains a minimal representation of the product likelihood for the jointly observed classes. An example of the results for one and two iterations of neighbourhood compression is shown in Fig. 3.7d and 3.7e, with posteriors in Figs. 3.7h and 3.7i; a more conservative posterior is obtained, but computation time decreases from the product method and accuracy is slightly greater than for the geometric method.

3.3.2 MMS Product Compression

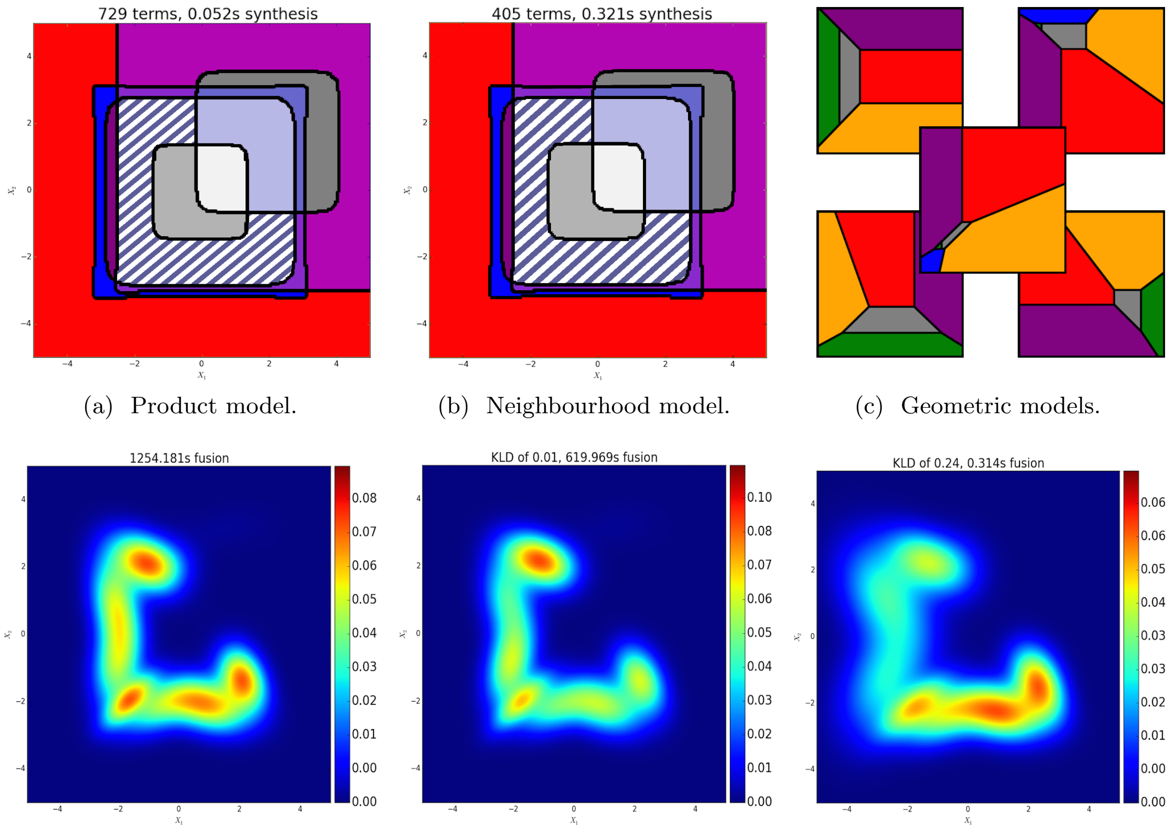
The aforementioned compression techniques directly extended to products of MMS likelihood functions, with some modifications. As in (3.15), the exact product expansion is

$$P(D_k^{1:N_o}|X_k) = \frac{\sum\limits_{r\in\sigma(\mathcal{I})} e^{w_r^T X_k + b_r}}{\sum\limits_{c=1}^{\bar{s}} e^{w_c^T X_k + b_c}} = L(D_k^{1:N_o}|X_k) \tag{3.22}$$

where $s_1 \times s_2 \times \cdots \times s_{N_o} = \bar{s}$ is the total number of product subclasses from all MMS models, s_o is the number of subclasses for $o \in \{1, ..., N_o\}$ and $\sigma(\mathcal{I})$ is the set of all $\prod_{o=1}^{N_o} q_{i_o} = \bar{q}$ product subclasses formed from the product of the N_o observed classes in $\mathcal{I}$. The compressed softmax approximation from (3.16) could be extended to MMS in one of two ways. First, we could find one set of parameters to represent the dominant regions for the product subclass polytopes within a single latent softmax model corresponding to an MMS model closely approximating (3.22). However, this 'holistic' compression approach leads to a possible complications for geometric or neighbourhood compression, since these may generate convex decompositions of X_k that cannot be embedded with the desired number of subclass parameters (i.e. case 3 from section 3.2). Alternatively, (3.22) could be approximated by a sum of $\bar{q}$ different softmax models, each representing the dominant region polytope for a different combination of the $\bar{q}$ subclass product terms,

$$L(D_k^{1:N_o}|X_k) \approx \sum_{z=1}^{\bar{q}} \frac{e^{\tilde{w}_{i^*,z}^T X_k + \tilde{b}_{i^*,z}}}{\sum_{c_z=1}^{m_z} \tilde{w}_{c_z}^T X_k + \tilde{b}_{c_z}}, \tag{3.23}$$

where each softmax summand z models a different portion (i.e. product subclass) of the (non-convex) probabilistic polytope for the joint product class label i^* of all N_o class observations.



(a) Product model.

(b) Neighbourhood model.

(c) Geometric models.

(d) Product posterior.

(e) Neighbourhood posterior.

(f) Geometric posterior.

Figure 3.8: MMS likelihood compressions for three measurements: *near* a small box centered at the origin, *near* a medium box offset from the origin, and *inside* a large box centered at the origin, shown by the striped area in (a) and (b), the latter containing an equal number of dominant classes but significantly fewer hidden subclasses. All five component models generated from geometric compression are shown in (c). Posteriors (d) through (f) are generated w.r.t the prior from Fig. 3.7b. Reprinted from [43], ©2016 IEEE.

Table 3.1: Accuracy and computation time of softmax and MMS compression method examples (30 trial avg.)

	Product	**Nbhd. 1**	**Nbhd. 2**	**Geometric**
Softmax Example				
Compression time T_c (s)	0.001	0.045	0.029	0.012
Fusion time T_f (s)	0.427	0.267	0.132	0.073
Total time $T_c + T_f$ (s)	0.428	0.312	0.161	0.085
m_*	25	20	9	7
KLD w.r.t Product (nats)	-	0.006	0.021	0.027
MMS Example				
Compression time T_c (s)	0.0520	0.321	-	0.112
Fusion time T_f (s)	1250	619	-	0.314
Total time $T_c + T_f$ (s)	1250	620	-	0.426
m_*	729	405	-	27[1]
KLD w.r.t Product (nats)	-	0.01	-	0.24

Although less accurate, this latter ‘summation’ approach is adopted here since it is more computationally efficient (parallelizable) and generally leads to relatively small errors for GM fusion in practice (see section 3.3.3). With this in mind, the geometric and neighborhood compression techniques can be straightforwardly adapted with extra bookkeeping to produce $\tilde{w}_z, \tilde{b}_z$ for all $\bar{q}$ summands. Future work will explore techniques for automatically generating subclasses to efficiently implement the former holistic approach.

Figure 3.8 shows compression results for a product of three MMS models that are shifted/scaled from a base MMS model describing *inside*, *near* and *outside* of the gray rectangular region in Fig. 3.7a. The imprecise approximation from (3.23) leads to noticeably more conservative fusion results for the geometric approximation. However, there is a great speedup for geometric compression ($T_c + T_f = 0.426s$) over the exact product and neighbourhood methods (20 minutes and 10 minutes for GM fusion, respectively), for a fairly small and acceptable sacrifice in fusion accuracy relative to the product fusion result ($D_{KL} = 0.24$ nats for VBIS GM fusion via geometric compression vs. $D_{KL} = 0.01$ nats with neighborhood).

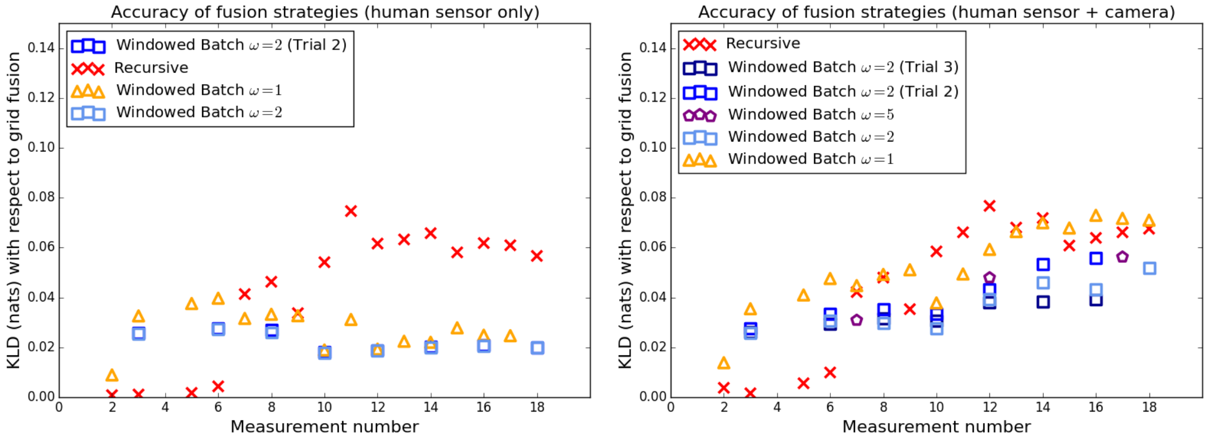
3.3.3 Fusion Strategies

The goal of likelihood compression is to maintain accuracy while minimizing computation for multiple semantic data observations. We perform multiple data aggregation methods on our dataset to compare these two objectives: *sequential recursive fusion* uses one uncompressed measurement per update step; *windowed batch fusion* uses $\omega + 1$ compressed measurements per update step. Varying ω shows differences in computational load and accuracy; since one human operator provides non-composite measurements, our approach is to compress and fuse $\omega + 1$ measurements in batch. We use an initial GM prior with $m_p = 6$ and limit to 20 posterior mixands using Runnalls' method [39].

Data was collected from a full static target search experiment described in section 5.1, in which the human observer had prior knowledge of the environment but no knowledge of the target locations. This resulted in approximately 20 human sensor measurements, as well as a full history of the robot teammate's trajectory. At each timestep, the robot teammate collects hard sensor data ζ_k as visual target detection information; this data is fused at each timestep via an efficient VBIS GM fusion approach.

Figure 3.9a demonstrates the accuracy results (in terms of KLD with respect to a grid-based posterior) of geometric MMS with VBIS GM fusion using different data aggregation methods (ignoring hard sensor data to assess the effects of only using the human sensor updates). In contrast to results shown in figure 3.9b (in which the hard sensor data fused at each of the 1300 simulation timesteps), the human-only data requires significantly fewer fusion steps, and thus fewer posterior approximations via GM merging. This leads to larger KLDs from the true grid-based posterior when the camera is activated. Furthermore, while all windowed fusion methods provided a greater accuracy than sequential recursive GM fusion after 8 measurements (likely also due to fewer posterior approximations), the computation time increases. Note that the GM merging algorithm causes sequential recursive fusion to start discarding more information than the batch methods. On average, recursive fusion required less than one second per update, as did $\omega = 1$,

[1] Spread across 5 complete softmax models.



(a)

(b)

Figure 3.9: Using geometric MMS compression only, we show fusion KLDs with respect to a grid-based truth model at the time of each human sensor measurement.

whereas $\omega = 2$ required 2-5 seconds per update, and $\omega = 5$ requires minutes per update.

Slow fusion times for larger values of ω point to the need for exploitation of the parallelizability of both VBIS fusion steps as well the geometric compression method. Additionally, the multiple fusion steps required by the summation geometric MMS fusion is inefficient in both computation time and accuracy; a technique must be developed to generate a single non-convex geometry for synthesis of an MMS model from geometric compression. This problem is non-trivial, but closely related to convex shape decomposition techniques currently being researched in computer vision and computer graphics domains [46, 47].

Chapter 4

Communicating Semantic Data

We have focused so far on modeling semantically uncertain statements to be fused within a Bayesian framework, with the primary purpose of providing an autonomous system with a better understanding of its environment. In other words, we treat the human as an information source, akin to one of the robot's on-board sensors, that can be used to update a robot's belief. These interactions are often decomposed [48] into the following four communication patterns:

- *human-push*, where a human voluntarily offers information to the robot;
- *human-pull*, where a human queries the robot for information;
- *robot-push*, where a robot voluntarily offers information to the human;
- *robot-pull*, where a robot queries the human for information.

The human-directed communication patterns (human-pull and robot-push) are important aspects to be considered for human-robot teaming, and new areas of research are quickly forming to develop human-directed information passing strategies. For instance, developing a human's trust in an autonomous system through its expression of self-confidence [49]. However, since our goal is to improve the robot's understanding of the world, human-directed communication patterns are less relevant to the discussion.

The robot-pull pattern is characterized by a request for information by the robot. Previous work by Kaupp [48] established the value of information (VOI) metric as a method of computing

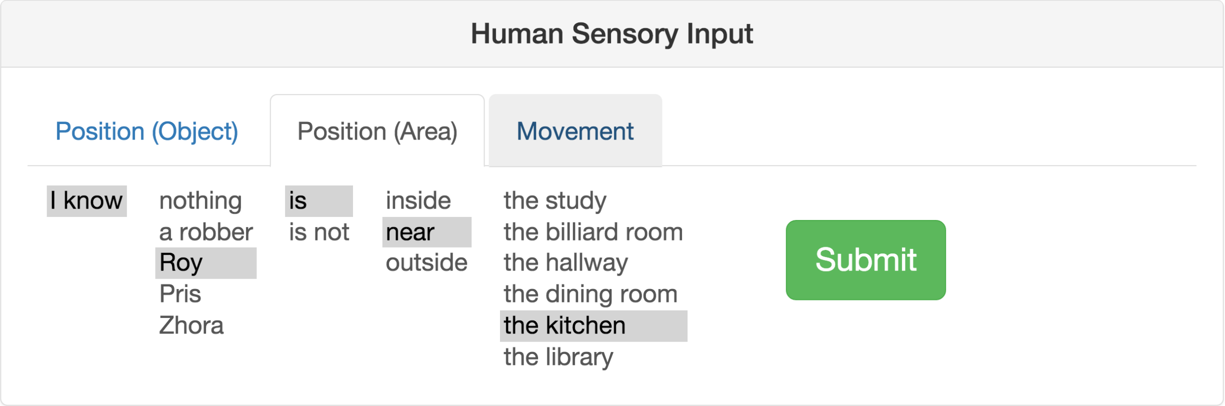


Figure 4.1: Fixed dictionary communication method.

the expected information value of a given query, which could be compared a cost metric (e.g. the expected increase in the human's cognitive workload [5]) to decide whether the query is of sufficient importance to be asked. Lore et al. [50] examine deep learning techniques for *non-myopic* VOI querying, which determines the VOI for query sequences over extended time spans rather than a *myopic* approach of assessing the VOI for only the current time step. Though non-myopic VOI is of particular interest for queries regarding dynamical systems, an in-depth consideration of (non-myopic) VOI is beyond the scope of this work, as we focus on the human-push communication paradigm.

In human-push, a human volunteers information; we focus on volunteering semantic state information embedded within natural language. Translation of language to semantic information requires that the language be *grounded*; that is, language needs be imparted meaning to relate to physical or abstract elements in the environment [51]. In the phrase, "The red robot is left of the green fern," we see three signifiers that must be assigned meaning: "The red robot" is a target object that must be uniquely identified among all physical objects; "is left of" is a spatial relation between two physical objects; and "the green fern" is a reference object in relation to which the location of the target object can be found. The semantic uncertainty in the phrase is captured by the spatial relation, while knowledge uncertainty and lexical uncertainty may be found in all three signifiers.

As a first step to communicating semantic information, we can develop a template-based communication method that eliminates significant amounts of uncertainty while limiting richness of semantic statements (as used by Ahmed [24]). The discussion of BSMs introduced the concept of archetypal semantic labels used to define dominant regions in softmax models; restricting spatial relations to these archetypes eliminates all lexical uncertainty, and results in a *fixed dictionary* of possible semantic labels. Similarly, if all targets and reference objects are limited to a set of unique identifiers for all physical objects, we reduce data-association based knowledge uncertainty and eliminate lexical uncertainty. An example of this *direct selection* approach is shown in Fig. 4.1, where:

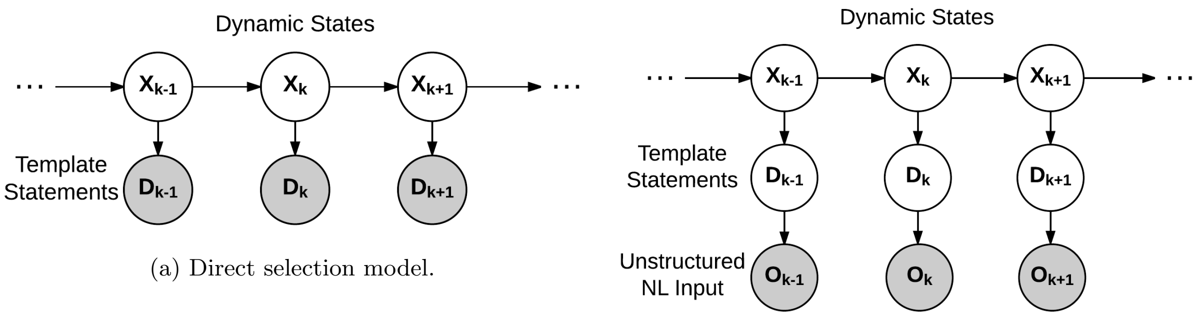
- the *targets* in column 2 selects one or more belief maps for Bayesian inference;
- the *positivities* in column 3 selects one class in a BSM;
- the *spatial relations* in column 4 selects one complete BSM;
- the *reference areas* in column 5 selects geometries which can be used for on-the-fly synthesis of BSM models.

Note that these four fixed dictionary categories represent only a subset of semantics that can be modeled. In a dynamic search, for instance, we can describe *movement type*, such as *moving quickly* or *stopped* which map to velocity states rather than the position states informed by spatial relation statements.

Direct selection provides a rigid, structured framework for communicating semantic information with minimal uncertainty. But, selection of items from a list may require a significant time expenditure (delaying time-sensitive information), is unscalable (with possible groundings easily reaching into the hundreds for complex environments), and constrains human language to only archetypal labels.

In the following sections, we discuss one approach to solve these problems[1] : a human-

[1] See [50] for greater detail of a robot-pull, question-based solution in which the robot teammate generates a series of questions based on a fixed semantic dictionary, ranked by VOI.



(a) Direct selection model.

(b) Natural language model.

Figure 4.2: Graphical models representing the relationships between key variables in the two problem forms. A greyed circle denotes an observed variable, whereas a white circle denotes an unobserved variable.

push, chat-based interface that takes unstructured natural language and relates each utterance to one or more pre-defined human sensor statements. The first section of this chapter introduces the problem formally and defines the terms required for Bayesian inference. The second section provides a scenario-specific natural language pipeline approach to compute the probabilities required for Bayesian inference. The final section provides preliminary results for the scenario-specific pipeline within the context of a target-search problem.

4.1 State Estimation with Natural Language Inputs

The direct selection approach is a bottom-up strategy for capturing semantic information: we generate semantically meaningful likelihood functions, then constrain the shared human-robot language set to a set of labels that uniquely specify these models. Instead, consider a top-down strategy, the natural language approach: allow humans to provide semantic information in the form of unstructured natural language *utterances*, and attempt to relate the input utterance to a pre-defined set of semantically meaningful likelihood functions. While this introduces lexical and knowledge uncertainties, it also provides a more convenient, scalable and rich communication interface.

When selecting archetypal semantic labels, the human exactly specifies the BSM likelihood function $p(D_k \mid X_k)$ in the general Bayesian inference problem (2.4) – that is, D_k is an observed variable. In contrast, D_k becomes a latent unobserved variable in the natural language approach, but some utterance O_k which relates to a template sensor statement D_k is now observed[2] . Fig 4.2 gives a comparison of these two problems as graphical models used for Bayesian inference.

The key consideration is the mapping between O_k and D_k. The general, any utterance O_k may relate to multiple semantic observations, $D_k^{N_o}$. For example, the unstructured natural language utterance, "The blue robot is in the kitchen, heading to the library," could be represented by the statements, "I know Roy is in the kitchen" and "I know Roy is moving towards the library," each of which are mapped to softmax classes. As discussed previously in section 3.3, it is possible compress multi-observation likelihoods into a single fusion update step. What has not been discussed is how to identify multiple template statements from one unstructured utterance. Section 4.2.3 provides a solution to this problem.

Formally, the Bayesian inference problem from (2.4) is now:

$$\begin{aligned}
p(X_k \mid O_k) &= \frac{\sum_{D_K^{1:\mathcal{M}}} p(D_k^{1:\mathcal{M}}, O_k, X_k)}{p(O_k)} \\
&= \frac{p(X_k)}{p(O_k)} \sum_{i=1}^{\mathcal{M}} \sum_{j=1}^{m_i} P(O_k \mid D_k^i = j, X_k) P(D_k^i = j \mid X_k) \\
&= \frac{p(X_k)}{p(O_k)} \sum_{i=1}^{\mathcal{M}} \sum_{j=1}^{m_i} P(O_k \mid D_k^i = j) P(D_k^i = j \mid X_k) \\
&= p(X_k) \sum_{i=1}^{\mathcal{M}} \sum_{j=1}^{m_i} \frac{P(D_k^i = j \mid O_k) P(D_k^i = j \mid X_k)}{P(D_k)}
\end{aligned} \tag{4.1}$$

Where $\mathcal{M}$ is the number of observation likelihood functions defined in the direct selection approach (i.e. all possible template statements to which an utterance may be mapped). Line 3 of (4.1) is arrived at through assertion of the conditional independence of O_k and X_k given D_k^i – that is, knowledge of the state gives no information about the utterance, if we already know the template semantic statement that describes the state. We also note the separation of semantic uncertainty, as

[2] This is a simplification: $p(D_k = i \mid X_k; \Theta)$ may simply represent a single BSM, where i selects either binary option, from a bank of BSMs, each with a different parameterization Θ. To simplify discussion, we assume that i selects both the softmax model as well as the softmax class.

captured by $P(D_k^i = j \mid X_k)$, and lexical uncertainty, as captured by $P(D_k^i = j \mid O_k)$, the template statement observation likelihood. Given previous discussion in capturing semantic uncertainty from chapters 2 and 3, the only term that is not easily attainable in (4.1) is $P(D_k^i = j \mid O_k)$. All other terms in (4.1) have been accounted for, whether through likelihood synthesis, VBIS fusion or multi-observation likelihood compression. Our focus then shifts from a sensor modeling problem to a linguistic grounding problem.

This problem bears strong resemblance to the grounding problem for natural language commanding [15,17,19,41]. Our objective is to relate elements of natural language input to pre-defined archetypal semantic labels which are grounded to pre-determined meanings (in a sense, grounding the unstructured language to the structured language in order to complete the link between unstructured language and meaningful semantic information).

In parallel with these approaches to natural language commanding problems, we will focus on a parameter-learning technique to capture $P(D_k^i = j \mid O_k)$, requiring generation of scenario-specific training data. While many off-the-shelf NLP tools provide mechanisms to tokenize, label and compare signifiers, the probabilities required at each step are not necessarily well-defined for our purposes. Thus, while we employ some off-the-shelf tools (e.g. `word2vec`), general-purpose tools are often lacking for our specific problem formulations, and we require scenario-specific tools to ensure both the proper probabilistic outputs, but also to restrict the problem in scope. This will be discussed further in section 4.2 for each component of the comparison process.

In contrast to the natural language commanding problem, our formulation allow us to be fully Bayesian about the method in which we fuse the sensory information given by the human: some probability exists that each utterance maps to each template sensor statement, and we consider *all* possible template statements, weighted by the likelihood that those statements relate to the utterance. This is a fundamentally different approach from command information given by humans to a robot; one expects that a command interface would select the most likely command rather than a weighted sum of commands (a robot performing a mixture of “go right” and “go left” would often be inadvisable). In the same sense of capturing and maintaining semantic uncertainty, rather than simply fusing information concerning the most likely observed state, we can capture and maintain lexical uncertainty as a useful property of the utterance; for example, if our archetypal labels for range measurements limited to *near* and *far*, it becomes possible to take a mixture of the two labels to produce a unique softmax function for *in between*[3] .

The definition of $P(D_k^i = j \mid O_k)$ is a non-trivial problem, as we condition on the entirety of unstructured language. One approach may limit O_k to merely a subset of language (i.e. only defining this probability for utterance components that have been viewed in the scenario-specific training corpus, as in [15]); however, this prevents the use of unmodeled language and is philosophically similar to simply generating more template sensor statements. A tradeoff between corpus size and scenario specificity is evident: using a larger and more diverse corpus allows a greater range of natural language inputs, but makes language less scenario-specific. The goal of this work is to explore solutions that allow sufficiently general language that could reasonably be expected to be used in a given scenario.

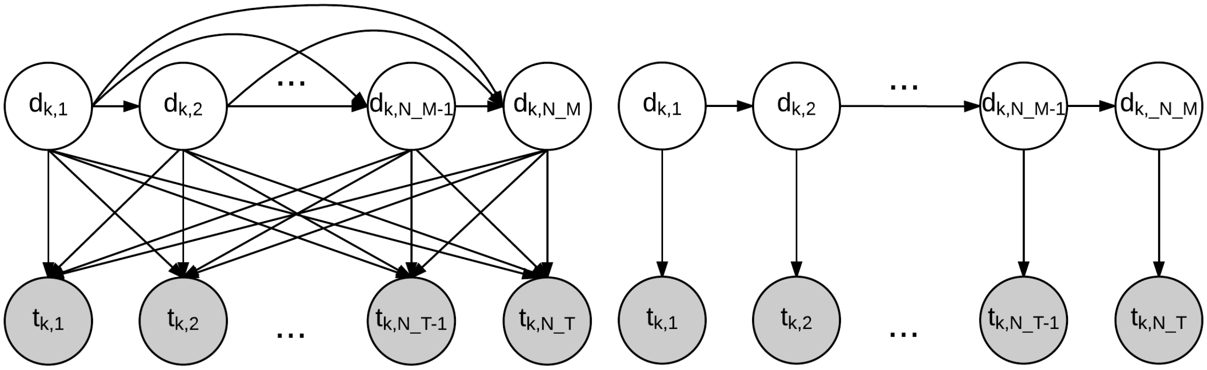
The following provides some deeper intuition into the obstacles to natural language human sensing, as well as one possible solution for how to arrive at the template statement observation likelihood $P(D_k^i = j \mid O_k)$.

4.2 Scenario-Specific Natural Language Pipeline

At the root of the problem in defining $P(D_k^i = j \mid O_k)$ is the notion of *word sense matching*: how similar is the sense[4] of one word, compared to the sense of another word? State-of-the-art methods in comparing words rely on embedding some representation of the word in a metric space, then defining a distance metric between words in that space. For instance, the `word2vec` framework can be trained in massive corpora to learn word embeddings for the majority of words in a language.

[3] This combination of labels requires a search over the power set of all labels; while this is an intriguing avenue of research, it is outside the scope of our current work due to computational requirements. As with categorical labels discussed in section 3.1.6, we assume mutual exclusion between all possible statements.

[4] Note that we must consider word senses rather than words themselves, as individual words may have many meanings (e.g. *plant* in the phrase "near the plant" may refer to a building or vegetation.). In general, we can discuss comparing words, though it is important to remember that their senses must be compared rather than the words themselves.



(a) Full token-matching problem.

(b) Token-matching problem reformulated as a Hidden Markov Model.

Figure 4.3: Probabilistic graphical models of the token-matching problem between structured statement tokens and observed, processed unstructured statement tokens.

We will discuss the embedded similarity measures used to compare words in section 4.2.4, but we must first identify the points of comparison.

In our natural language framework, O_k is an unstructured utterance – this means we do not know which elements of O_k map to structured elements that select D_k^i. In order to compare elements, we must first divide divide the unstructured language into *tokens*: meaningful units that may be comprised of one or more words. For example, Fig. 4.1 includes one possible area reference as, *the billiard room*; this multi-word token retains an indivisible unit of meaning that changes based on component modification ("a billiard room" implies one of many, rather than a specific room; "the ball room" characterizes an entirely different room type; "the billiard ball" may reference an object in the environment, rather than a room).

Once we achieve this tokenization of the observed utterance $O_k = \{t_1, t_2, \ldots, t_{N_T}\}$, we can similarly decompose our our unobserved semantic statement into tokens $D_k = \{d_1, d_2, \ldots, d_{N_M}\}$, using the categorical archetypes shown in Fig. 4.1. These components form the basis for comparison, with a graphical representation of this comparison shown in Fig. 4.6a. To explore this comparison in a tractable manner, we assert that the Markov property holds between unobserved tokens, arriving at the graphical model shown in Fig. 4.6b. Thus, we define the probabilistic relationship between O_k and D_k as follows:

$$\begin{aligned} P(D_k \mid O_k) &\propto P(d_{k,1}, d_{k,2}, \ldots, d_{k,N_T}, t_{k,1}, t_{k,2}, \ldots, t_{k,N_T}) \\ &= P(d_{k,1})P(t_{k,1} \mid d_{k,1})P(d_{k,2} \mid d_{k,1})P(t_{k,2} \mid d_{k,2}) \times \ldots \\ &\quad \times P(d_{k,N_T} \mid d_{k,N_T-1})P(t_{k,N_T} \mid d_{k,N_T}) \end{aligned} \tag{4.2}$$

Now, we can probabilistically compare O_k and D_k by defining transition probabilities between template components $P(d_{k,i+1} \mid d_{k,i})$ and observation probabilities of utterance components given template components $P(t_{k,i} \mid d_{k,i})$. However, this formulation imposes a one-to-one mapping between observed utterance tokens and unobserved statement tokens, imposing a constraint of $N_T = N_M$. As well, ordering must be defined to provide proper transition and observation probabilities.

In order to ensure proper tokenization and ordering, we define a pipeline to process the natural language input (shown graphically in Fig. 4.4), the components of which will be discussed in sequential order throughout the following subsections. This pipeline pre-processes natural language input, decomposes the input into one or more sets of ordered tokens, and performs the token-to-token comparison to arrive at $P(D_k \mid O_k)$.

Some components of this pipeline are scenario-specific, as they require training on scenario-specific experimental data, such as full utterances, hand-generated tokenizations and hand-generated categorical labels of multi-word tokens. A general pipeline using off-the-shelf components (such as a part-of-speech tagger) and without any training data is outside the scope of this work due to the complexity required in integrating several general-purpose tools into a problem-specific environment. The training data was collected for three purposes: to better define the natural language used by humans; to determine how well the pre-defined template sentences cover the space of given utterances; and to be used as training data for pipeline components. The specifics of data collection are discussed in more detail in concert with the experimental setup in section 5.2.

4.2.1 Multi-word Tokenization

The *tokenizer* portion of the natural language pipeline decomposes an input utterance O_k into a distinct set of meaningful single- or multi-word units, $\{t_1, t_2, \ldots, t_{N_T}\}$. This is achieved

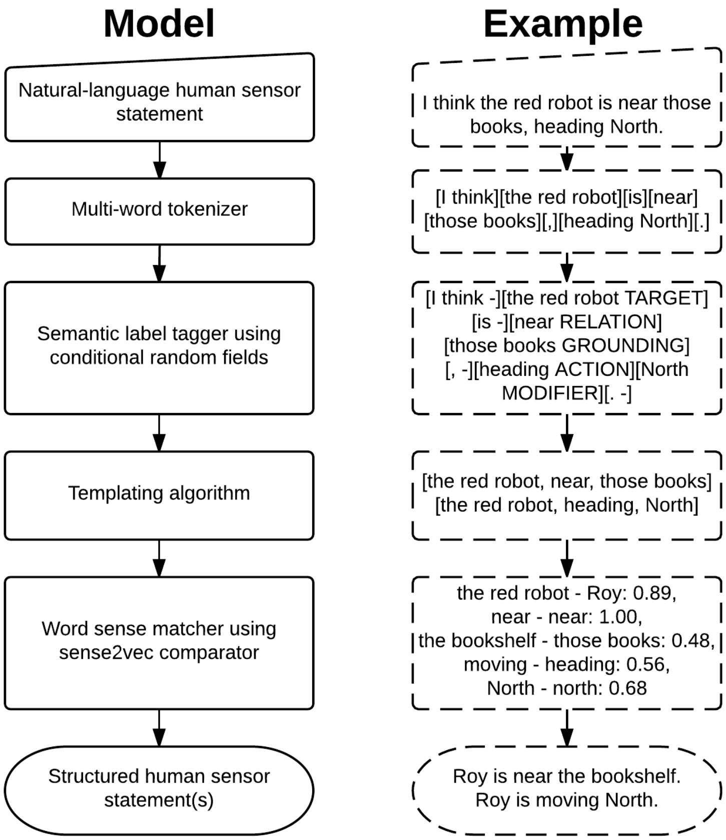

Model
Example
Natural-language human sensor statement
Multi-word tokenizer
Semantic label tagger using conditional random fields
Templating algorithm
Word sense matcher using sense2vec comparator
Structured human sensor statement(s)
I think the red robot is near those books, heading North.
[I think][the red robot][is][near] [those books][,][heading North][.]
[I think -][the red robot TARGET] [is -][near RELATION] [those books GROUNDING] [, -][heading ACTION][North MODIFIER][. -]
[the red robot, near, those books] [the red robot, heading, North]
the red robot - Roy: 0.89, near - near: 1.00, the bookshelf - those books: 0.48, moving - heading: 0.56, North - north: 0.68
Roy is near the bookshelf. Roy is moving North.

Figure 4.4: Proposed NLP pipeline (left) with example (right).

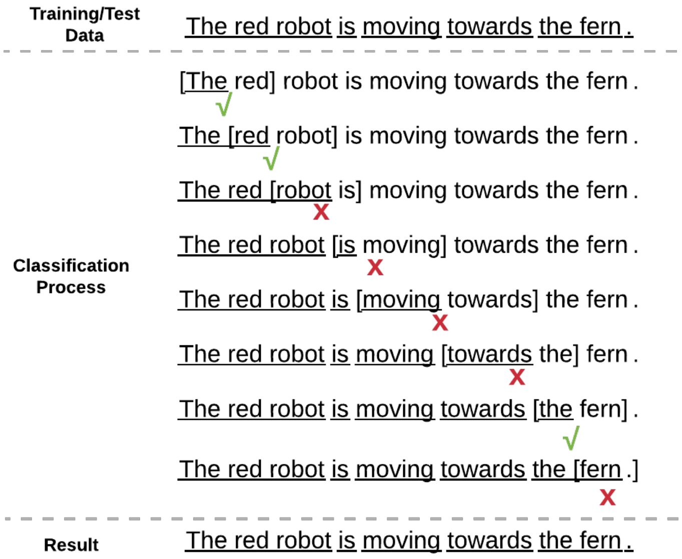
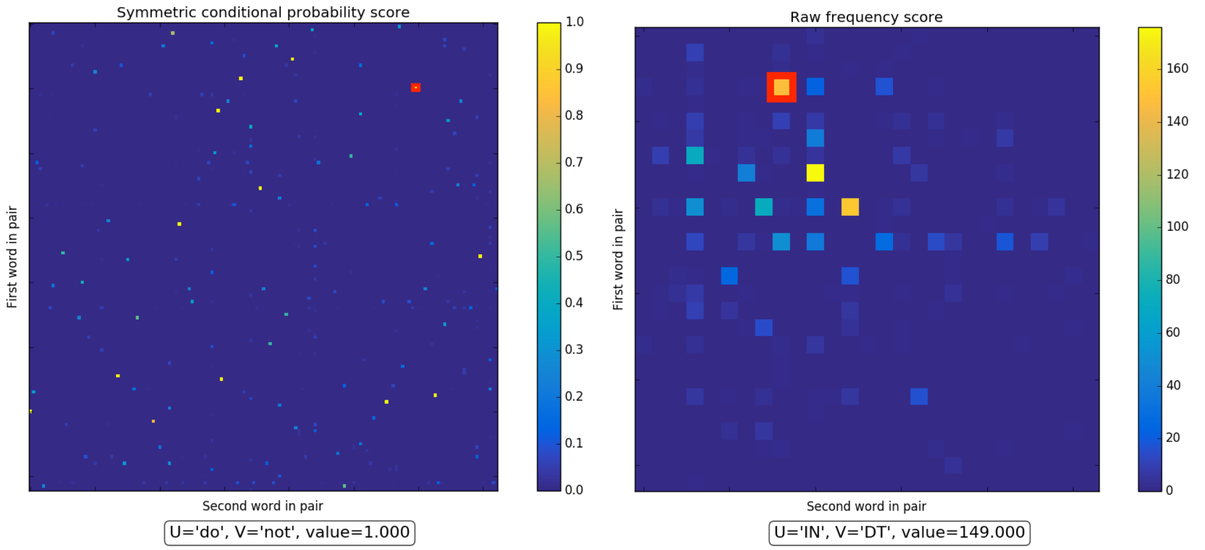

Training/Test Data
The red robot is moving towards the fern .
Classification Process
[The red] robot is moving towards the fern .
√
The [red robot] is moving towards the fern .
√
The red [robot is] moving towards the fern .
X
The red robot [is moving] towards the fern .
X
The red robot is [moving towards] the fern .
X
The red robot is moving [towards the] fern .
X
The red robot is moving towards [the fern] .
√
The red robot is moving towards the [fern .]
X
Result
The red robot is moving towards the fern .

Figure 4.5: Step-by-step multi-word tokenization process. Brackets encapsulate each considered word-pair, underlines denote individual tokenizations, a checkmark denotes that a word-pair has been classified as grouped and an x mark denotes that a word-pair has been classified as not grouped.



(a) Symmetric conditional probability score for lemmatized word pairs.

(b) Raw frequency score for part-of-speech tagged word pairs.

Figure 4.6: Examples of two association measures used to score transformations on word-pairs. The (u, v) word pair selected is highlighted in red, with the value of its association measure shown in below the graph.

through classifying adjacent words based on whether or not they should be grouped within the same token. Each ordered word pair, (u, v) is assigned a probability of being grouped, $g = 1$, or ungrouped, $g = 0$, based on a pre-trained softmax classifier, $P(g \mid u, v)$. The process is illustrated in Fig. 4.5.

In order to train a word-pair grouping classifier, we first identify some features which we can use for supervised learning with our training data. Michelbacher [52] identifies eight different *association measures* between word-pairs; these are a measure of statistical joint predictiveness for the event of one word occurring within the same word-pair as another word. The association measures vary from simple (the *frequency* measure, which scores based on the number of occurrences of a word pair in a corpus) to complex (the *log-likelihood hypothesis* measure, which finds the likelihood ratios of co-occurrence hypotheses). For a full description of these association measures, see section 3.3 of [52].

To better improve classification accuracy, we consider association measures on transformations on the input words rather than on the input words themselves. We score *lemmata*, the canonical form of a words (e.g. “run” instead of “running”, “ran” or “runs”), by transforming each word into its lemma and finding association measures between all lemmata. We also score *part-of-speech tags*, the categories of words that conform to similar grammatical rules (e.g. nouns, adverbs, prepositions, etc.), by finding association measures between each part-of-speech pair. Examples of these are shown in Fig. 4.6. Note the sparsity of data - some word pairs, such as (‘do’, ‘not’) have strong association scores do to their frequent pairwise occurrence within the training corpus, yet many others, such as (‘the’, ‘forward’) have weak or nil association scores.

Given 8 association scores and 2 word transformations, we now have a 16-dimensional feature vector with which to train our classifier. We can train a softmax classifier, as described in section 2.1.2, with labeled training data to learn the 16-dimensional weight vector corresponding to the features. Note that some lemmata may be previously unobserved; their association scores are excluded. However, since part-of-speech tags are far more limited than lemmata, an 8-dimensional feature vector can easily be recovered. In short, lemmata-based association measures provide specificity, whereas part-of-speech-based association measures provide generality.

4.2.2 Semantic Label Tagging

Once we have multi-word tokens, our goal is to assign labels to each token so that we can match each token to a place within the pre-defined template categories (*target*, *positivity*, *spatial relation*, etc.). Linear-chain conditional random field (CRF)s [53] are a well-known technique for assigning labels to sequences of inputs, similar to hidden markov models (HMMs) but undirected.

HMMs are generative models, in that they model joint probabilities $P(S,T)$ between a random variable over observed values T and a random variable over hidden variables S. This requires the enumeration of all possible sequences of observed variables, which is often intractable. In contrast, CRFs are discriminative models, defining only the conditional probability $P\left(S \mid T\right)$, which is significantly less complex as only the observed values of T are considered (i.e., we need not model the prior $P(T)$).

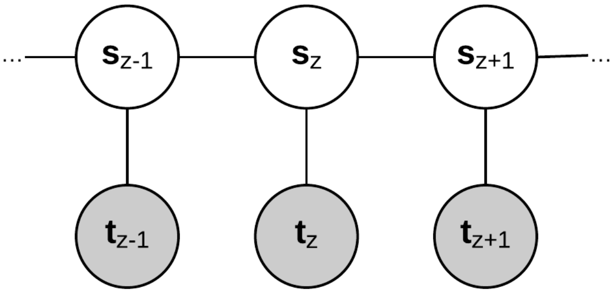


Figure 4.7: Linear-chain CRF model, where shaded circles are observed and open circles are hidden variables.

Within our NLP context, we assign semantic labels $s_z \in S$ which range over some finite set of possible labels $\mathcal{S}$ to each observed token $t_z \in T$ for token index $z = 1, 2, \ldots, Z$. We assume this is a linear-chain CRF, in that the graph is composed of two undirected sequences of random variables $S = (s_1, s_2, \ldots, s_Z)$ and $T = (t_1, t_2, \ldots, t_Z)$ as shown in Fig. 4.7. This graphical assumption allows us to take the context of nearby labels and tokens when assigning the specific label s_z to the specific token t_z.

We can evaluate the joint distribution over the label sequence S given the observed tokens T:

$$P_\theta(S|T) \propto \exp\left(\sum_{e\in E,k} \lambda_k f_k(e, S|_e, T) + \sum_{v\in V,k} \mu_k g_k(v, S|_v, T)\right) \tag{4.3}$$

$f_k(\cdot)$ and $g_k(\cdot)$ are *transition feature functions* and *state feature functions*, respectively, that are defined a priori. λ_k, μ_k are parameters to be learned from training data. We pre-define the set of feature functions f_k and g_k based on the CRF++ framework [54], with feature function templates shown in Listing 4.1 for both unigram and bigram models that each generate a set of indicator feature functions.

Once θ has been learned, we can take

$$\operatorname*{argmax}_{s} P_\theta(S|T) \tag{4.4}$$

to find the most likely labels associated with individual tokens within context of both neighouring labels and neighbouring tokens. At this state of the pipeline, we have a set of label-token pairs $\{s_z, t_z\}_{z=1}^{Z}$ that can be used to match structured templates.

Listing 4.1: CRF++ feature function templates

```
# Unigram
U00:%x[-2,0]
U01:%x[-1,0]
U02:%x[0,0]
U03:%x[1,0]
U04:%x[2,0]
U05:%x[-1,0]/%x[0,0]
U06:%x[0,0]/%x[1,0]

# Bigram
B
```

4.2.3 Sensor Statement Templates

Given our label-token pairs, we can use pre-defined template structures to ensure both correct ordering between utterance tokens and statement tokens, as well as define the statement token transition probabilities $P(d_{k+1} \mid d_k)$.

Known dictionary-based templates can be transformed into a tree structure, e.g. the problem-specific structure shown in Fig. 4.8. Each node represents a category containing one or more possible tokens all associated with the same category label; for instance, the *Spatial Relation: Object* node may contain, *front*, *left*, *back*, *right* and *near*, whereas the *Spatial Relation: Area* node may contain *inside*, *near* and *outside*. If each categorical node is expanded into a set of token nodes, then each path through the tree from the root to a leaf node generates one statement. Thus, transition probabilities between d_{k_i} and $d_{k_{i+1}}$ can be derived by parent-child relationships:

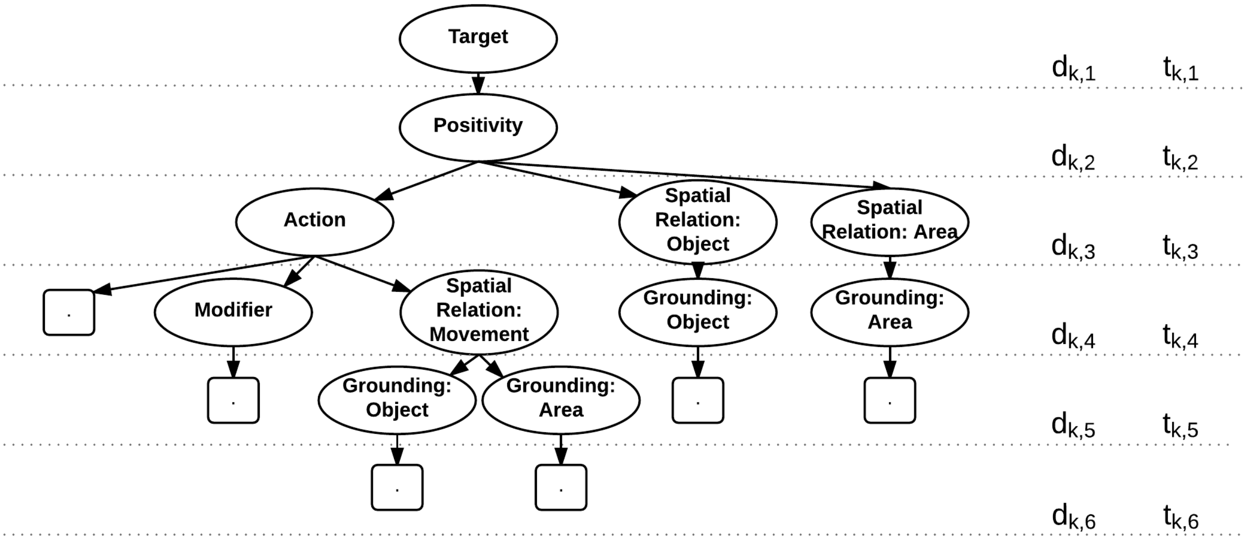


Figure 4.8: Tree-based human sensor statement template. Ellipses denote category nodes, each containing one or more possible archetypal labels.

$$P(d_{k,i+1} \mid d_{k,i}) = \begin{cases} \frac{1}{\nu} & \text{if PARENT}_{d_{k,i+1}} = d_{k,i} \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

where ν is the number of tokens captured by the template node that contained $d_{k,i+1}$. This assumes all possible paths are equally likely within the constrained paths specified by the template.

As well, this structure can be used to identify dense utterances that map to multiple statement templates. Once each utterance token has been labeled through the CRF, these labels can be compared to paths through the statement template tree to enumerate all contained template statements. For example, the phrase, "The blue robot is in the kitchen, heading to the library," contains one target, two spatial relations and two groundings, mapping to two different paths through the statement template tree. A greedy algorithm can then be used to identify common elements and distinct elements, providing one or more structured orderings for the labeled unstructured utterance tokens.

4.2.4 Word Sense-Matching

Calculation of the observation probabilities asks the question, 'What is the probability of observing the token $t_{k,i}$, given that I have selected the template token $d_{k,i}$?'. This question relates to the core issue of lexical uncertainty in our problem: the probability of observing, for instance, *near* given that the template word is *next to* is expected to be non-unity; furthermore, since $\sum_{t_{k,i}} P(t_{k,i} \mid d_{k,i}) = 1$ and $t_{k,i}$ spans the English language, we notice little discrimination between two words from the conditional probability.

To compare tokens, we use word similarity measures provided by the `word2vec` framework. The efficient SGNS method developed by Mikolov et al. uses a softmax function (c.f. [27] eq. 2) to relate the probability of one word, w_i, represented by its input and output vectors v_i and v_i', given another word, w_j, represented by its input and output vectors v_j and v_j':

$$p(w_i \mid w_j) = \frac{exp\left(v_i'^{\,T} v_j\right)}{\sum_{w=1}^{W} exp\left(v_w'^{\,T} v_j\right)} \tag{4.6}$$

Since W is the number of words in the vocabulary (likely in the millions for unstructured language), $p(w_i \mid w_j)$ will be both small and nearly flat over the vocabulary. A common measure of word similarity, in place of conditional probability, is to instead take the cosine similarity measure between output word vectors. This similarity measure need not sum to 1 for all words and is exactly 1 for identical words. A first step approach would be to replace the token observation probability $P(t_{k,i} \mid d_{k,i})$ with a token observation similarity score $s(t_{k,i}, d_{k_i})$, weight all possible statements by their joint scores:

$$\begin{aligned} s(D_k, O_k) = P(d_1)s(t_1, d_1)P(d_2 \mid d_1)s(t_2, d_2) \times \ldots \\ \times s(d_{N_T}, d_{N_T - 1})P(t_{N_T} \mid d_{N_T}) \end{aligned} \tag{4.7}$$

While this bypasses the Bayesian probabilistic framework developed in section 4.1, it provides a useful first step towards relating unstructured utterances to template statements.

The following section provides proof-of-concept results in generating $s(D_k, O_k)$ for some test

Table 4.1: Observation likelihood values $P(D_k \mid O_k)$ for select unstructured natural language inputs.

Unstructured input phrase							
I know a robot is near the bookcase.		**I know a robot is next to the bookcase.**		**I know a robot is not next to the bookcase.**		**I am pretty sure one of those guys is somewhere close to that pile of books**	
Template statement	$s(D_k, O_k)$	**Template statement**	$s(D_k, O_k)$	**Template statement**	$s(D_k, O_k)$	**Template statement**	$s(D_k, O_k)$
I know a robot is near the bookcase.	0.0111	I know a robot is in front of the bookcase.	0.0073	I know a robot is not in front of the bookcase.	0.0084	I think a robot is right of the bookcase.	0.0037
I know a robot is near the fridge.	0.0103	I know a robot is in front of the fridge.	0.0067	I know a robot is not in front of the fridge.	0.0081	I know a robot is right of the bookcase.	0.0037
I know a robot is near the hallway.	0.0099	I know a robot is right of the bookcase.	0.0067	I know a robot is not in front of the desk.	0.0078	I think a robot is in front of the bookcase.	0.0035
I know a robot is near the billiard room.	0.0095	I know a robot is right of the fridge.	0.0063	I know a robot is not right of the bookcase.	0.0078	I think a robot is right of the fridge.	0.0035
I know a robot is near the kitchen.	0.0095	I know a robot is in front of the desk.	0.0061	I know a robot is not right of the fridge.	0.0075	I know a robot is in front of the bookcase.	0.0035

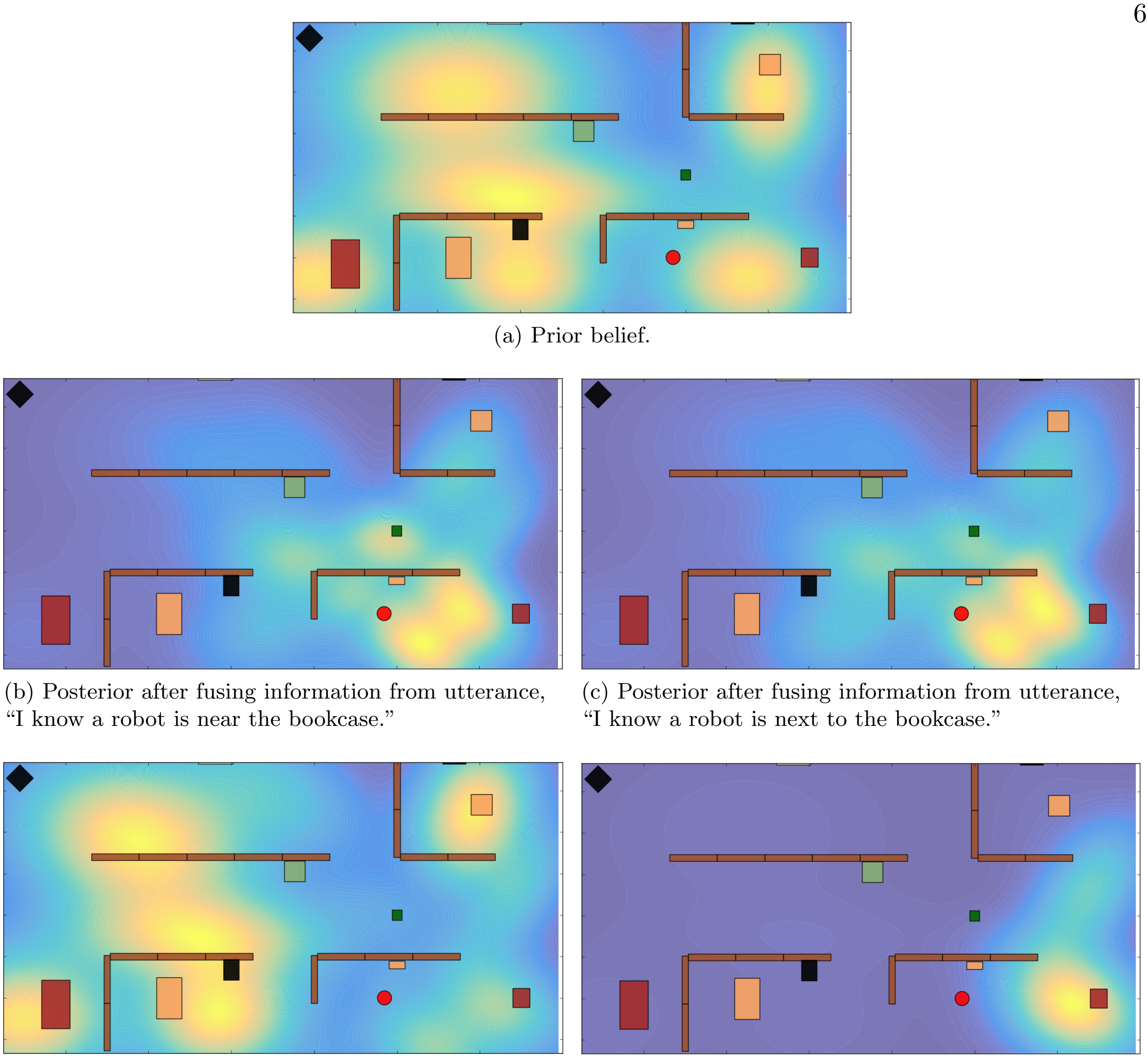
phrases and using the $\text{argmax}_{D_k}\, s(D_k, O_k)$ to select template sensor statements that are most similar to the input utterance for Bayesian fusion.

4.3 Preliminary Natural Language Sensing Results

To gain intuition for the word sensing matching component of the NLP pipeline, four utterances with varying degrees if lexical distance from pre-defined template statements are considered, where each utterance is used for data fusion. These utterances are compared to 2682 possible template statements. The four input phrases are shown and assumed to have been tokenized and labeled correctly by previous pipeline elements in order to demonstrate the sense matching portion of the pipeline independently.

These four utterances, as well as the 5 highest scoring template statements based on $s(D_k, O_k)$, are shown in Table 4.1. With a known semantic map and a GM prior, the Bayesian inference updates using the most likely sensor template as selected by the natural language pipeline are shown in Fig. 4.9.

From left-to-right, the four columns in Table 4.1 demonstrate the effects of increasing dissimilarity with template statements: the first input sentence is exactly a sensor statement template; the second input sentences replaces a spatial relation template token, ‘near’, with a non-template token, ‘next to’; the third input sentence replaces the grounding and changes the positivity; and the fourth is an imprecise reformulation of the first sentence.



(a) Prior belief.

(b) Posterior after fusing information from utterance, "I know a robot is near the bookcase."

(c) Posterior after fusing information from utterance, "I know a robot is next to the bookcase."

(d) Posterior after fusing information from utterance, "I know a robot is not next to the bookcase."

(e) Posterior after fusing information from utterance, "I am pretty sure one of those guys is somewhere close to that pile of books."

Figure 4.9: Example results for Bayesian inference with natural language human sensor statements. The target is marked as a red circle, and is positioned near a reference object labeled as a *bookcase* marked as a small tan rectangle. Both are positioned in the lower-right quadrant of the map.

We see that the first utterance, "I know a robot is near the bookcase", is best characterized by its exact matching template statement; the value of $s(D_k, O_k)$ for the most likely template statement is not particularly distinct from the other template statements. The statement token transition probability $P(d_{k,i+1} \mid d_{k,i})$ is expected to be uniform over all statements whose components are derived from the same categorical labels, and the token similarity measure $s(d_{k,i}, t_{k,i})$ is expected to help distinguish between template statements given some known score. However, since this similarity measure takes the cosine distance between word vector representations within the context of the entire corpus, the similarity between contextually dissimilar terms (e.g. *fridge* and *bookcase*) is not apparent. Clearly, a similarity measure that takes in the appropriate context is required for greater disambiguation between phrases.

Similar results are seen for the other three test utterances: the most similar statements are a good approximation of the input utterance, but the difference between template statements is minimal.

Chapter 5

Experimental Validation Testbed

An experimental testbed was developed alongside the synthesis, compression and natural language sensing techniques covered in previous chapters. This allowed practical validation of the developed algorithms and methodologies, as well as data collection for a greater understanding of what informative natural language statements human would provide.

This chapter contains two sections: the first describes the indoor search problem conceived for experimental validation of human sensor models; the second describes a pilot study in which human subjects provided natural language information to assist their robotic partner to be used as and training data for the pipeline described in section 4.2.

5.1 Cops and Robots Testbed

We apply our techniques to a target search scenario we call ‘Cops and Robots’, in which one ‘cop’ robot searches for multiple ‘robber’ robots, with the assistance of a human ‘deputy’. The objective of the human-robot team (the cop and deputy) is to ‘capture’ the robber robots, achieved by the cop robot orienting itself toward the robber robot within a 1 meter radius from the robber. The cop robot maintains a state estimate X_k^r for each robber r (nominally, this scenario is performed with three visually distinct robbers, $r \in \{\mathrm{Roy}, \mathrm{Pris}, \mathrm{Zhora}\}$), explores the area by taking a planned path to the maximum a posteriori (MAP) point of the GM fusion posterior for one robot r (chosen from a preset target capture order). At each time step, the cop fuses negative information corresponding to its target detection viewcone generated by hard camera data ζ_k. Human sensor data D_k is fused as it arrives. The scenario terminates once all robbers are captured.

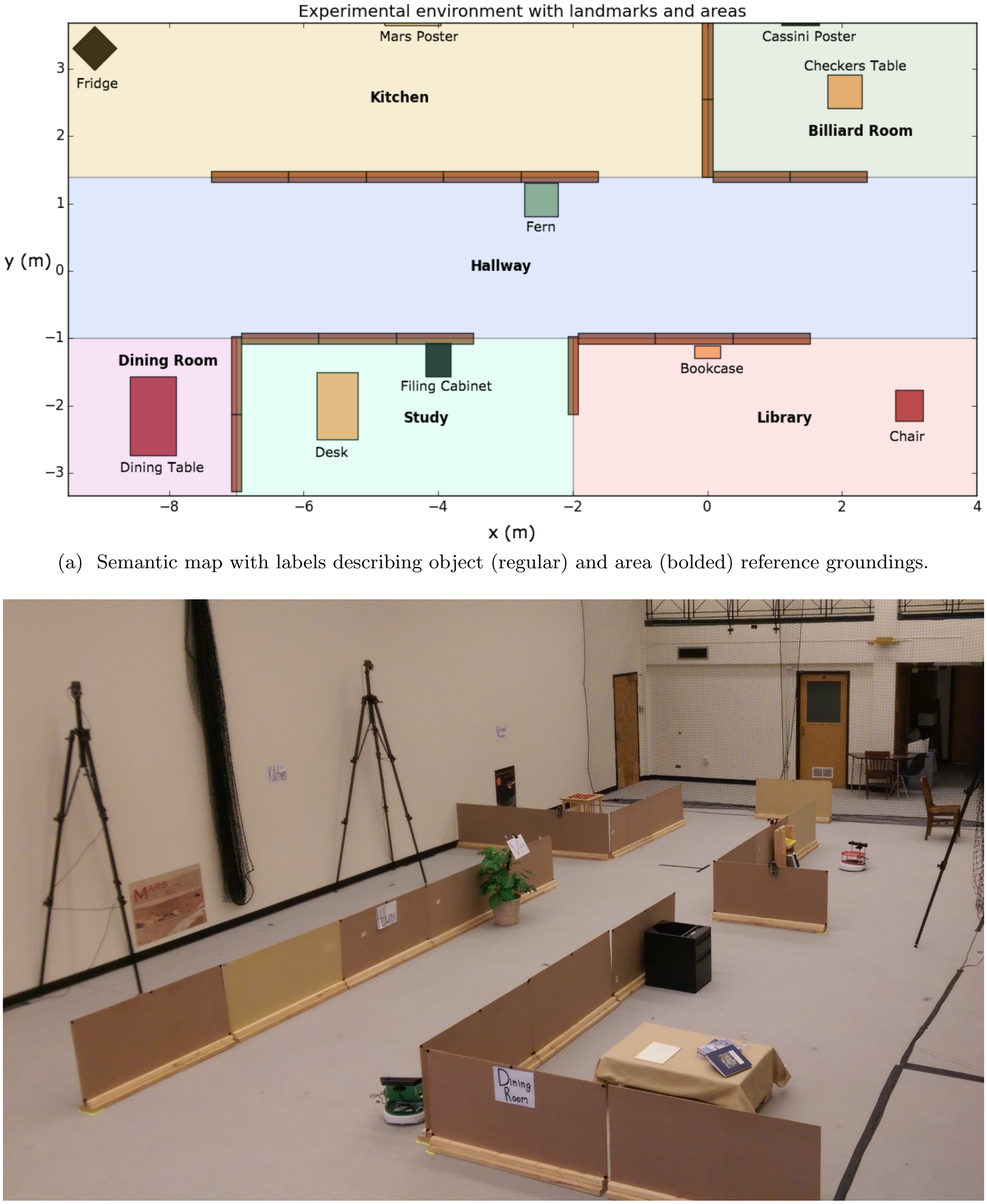
Each robot is modeled after the open-source Turtlebot design provided by Willow Garage, with: a Microsoft Kinect for RGB visuals and depth data to avoid obstacles; an Odroid U3 carrying 2GB of RAM and an 1.7 GHz quad-core ARM processor for on-board computation; and an iRobot Create base for mobility. Traditional wheel encoders and accelerometers are not used.

The indoor search environment consists of semantically distinct landmarks and areas, as shown as a map in Fig. 5.1a and as a physical environment in Fig. 5.1b. Fourteen Vicon Bonita-series motion-capture cameras surround the field, providing localization of all map elements (primarily the robots). This also allows objects to be tracked in a simulation environment (Gazebo) in real time.

The human deputy is located remotely and uses an web-based interface shown in Fig. 5.2 to communicate with the cop. The human can either view the cop's camera feed, or select a view from one of three security cameras in the environment. The human can also select from a list of observations to send a sensor statement to the cop. These statements are structured by five selection parameters: *certainty, target, spatial relation, grounding* (lower-left of Fig. 5.2). Alternatively, questions generated by the cop using VOI [50], seen lower-right, can be answered by the human to the same effect.

A remote backend simulation written in Python provides the mission and goal planning infrastructure, the pre-generated human sensor models generated through synthesis of human sensor models with respect to all reference groundings (e.g. Fig. 5.3), a prior belief based on the size and shape of each area (seen in Fig. 5.4), as well as the VBIS fusion updates. This interacts with a robot operating system (ROS) environment that provides a communication infrastructure, as well as navigation software for global A* planning and a dynamic window approach for local, reactive planning around unforeseen obstacles [55].

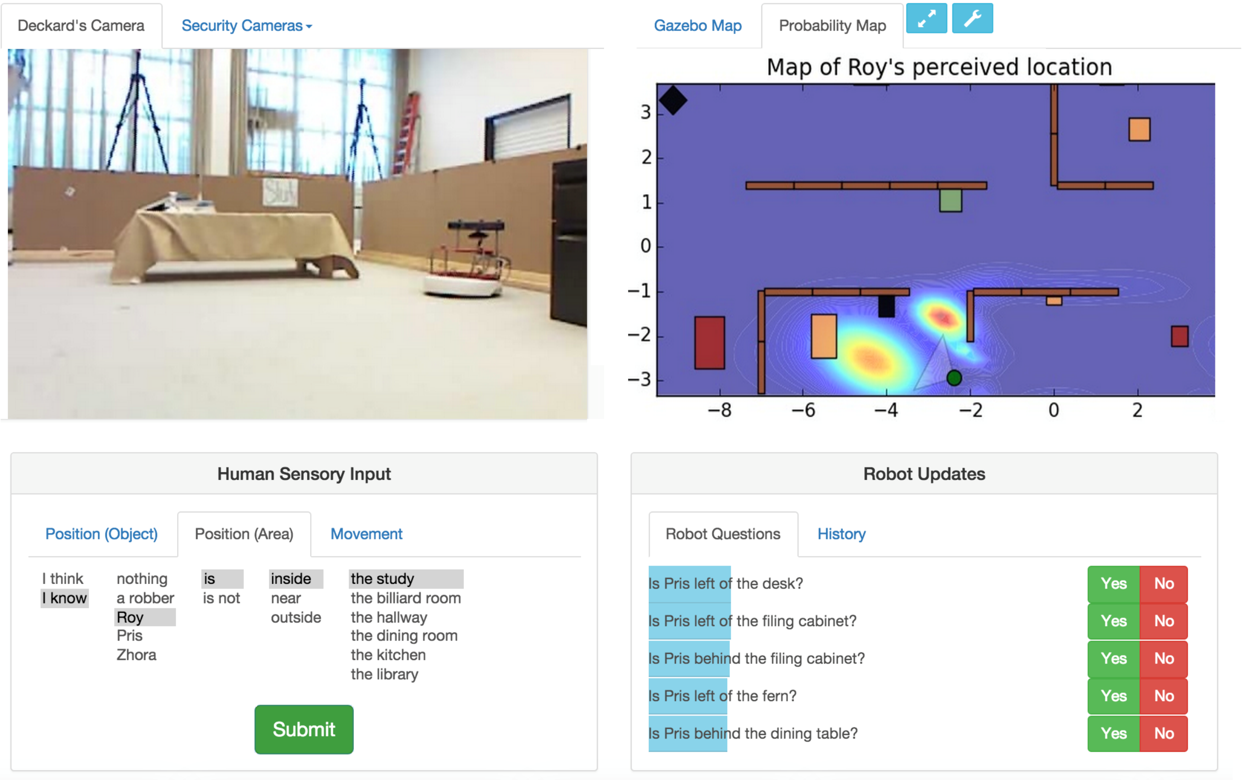
The robbers can either be static targets, remaining in place for the duration of a scenario, or dynamic targets, moving to randomly selected goal poses within the map.



(a) Semantic map with labels describing object (regular) and area (bolded) reference groundings.

(b) Physical indoor search environment.

Figure 5.1: One configuration of the experimental indoor search space with semantically distinct reference objects/areas and active robotic agents.



I think	nothing	is	inside	the study
I know	a robber	is not	near	the billiard room
	Roy		outside	the hallway
	Pris			the dining room
	Zhora			the kitchen
				the library

Is Pris left of the desk? Yes No

Is Pris left of the filing cabinet? Yes No

Is Pris behind the filing cabinet? Yes No

Is Pris left of the fern? Yes No

Is Pris behind the dining table? Yes No

Figure 5.2: Web-based human-robot interface. Clockwise from top-left are the camera views, the target location probability map, a questioner module showing questions posed by the cop robot, and a codebook for human sensory input (yes/no answers to queries or free-form selectable codebook inputs possible).

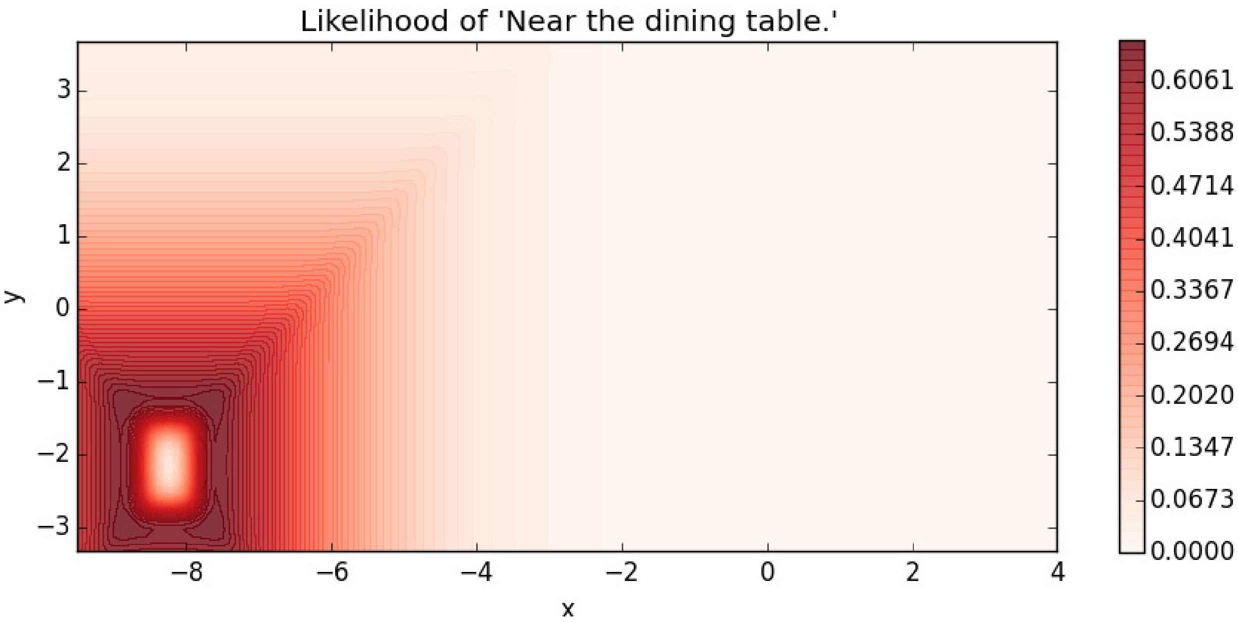


Figure 5.3: Synthesized likelihood for near a reference grounding.

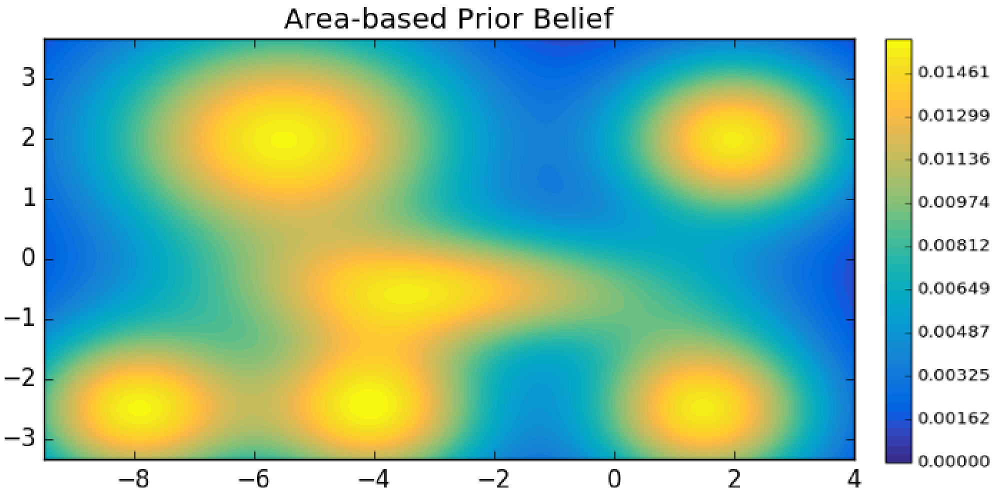
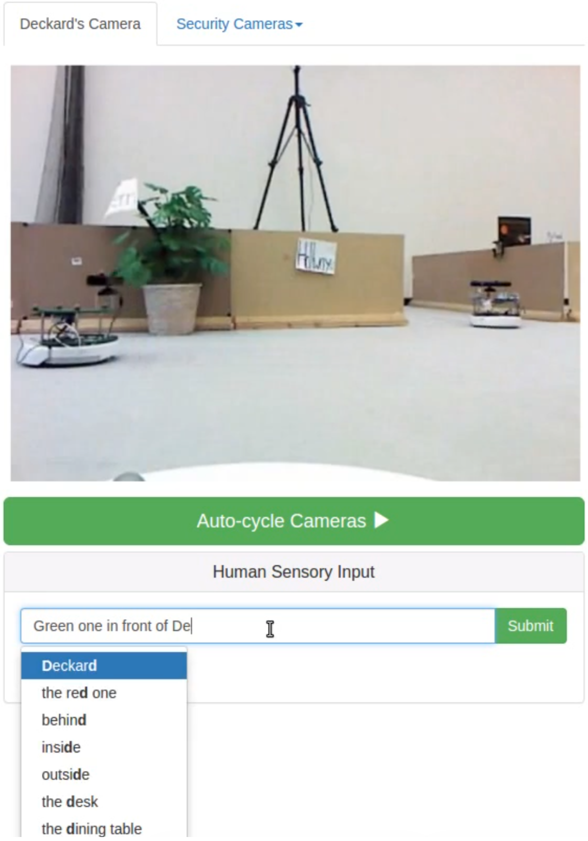


Figure 5.4: Prior target location belief based on known map areas.

5.2 Integrated Human-Robot Search Pilot Study

A pilot study was performed to better understand unstructured natural language inputs humans may provide. Human subjects were given five minutes of pre-recorded video feed from a previously executed dynamic search scenario and asked to inform their robot partner about the robber robots. They provided unstructured natural language input through a chat interface, Fig. 5.5, which shows one frame of the video feed provided to the human, as well as the chat interface with suggested terms based on the template dictionaries. The green button toggles between manual camera selection and a 5-second cycle between all available camera feeds, in an effort to encourage the human to provide information (either negative or positive information) from several viewpoints rather than remaining locked to one.

The 12 subjects, all current or prospective students at the time the data was taken, provided 200 total utterances. Of these, 36 were identified as uninformative (e.g. "Get him!!!", "Get your act together."). Each of the 164 remaining utterances were matched by hand to reasonably close template sensor statements. For example, the unstructured utterance, "Pris might be in the hallway..." was associated by hand to the template statement, "I think Pris is inside the hallway." In a minority of cases, individual utterances were associated with multiple template statements (e.g. the utterance, "i can see movement in the hallway," associates with the joint observations



Green one in front of De

Figure 5.5: Data collection interface for unstructured natural language sensor statements.

of, “I know a robot is moving,” and “I know a robot is inside the hallway.”). With 2682 possible template statements related to the known semantic map, only 6 were deemed unable to associate with a template statement. These cases required information that was unavailable or ambiguous, such as “A robot is in front of the other robot.”

While this collected data was necessary for the development of the NLP pipeline described in section 4.2, useful information was also gleaned from subject feedback. A primary point of concern included the disorientation and lack of situational awareness due to the auto-cycling of cameras – awareness is an critical factor in human-robot teaming [56]. This particularly frustrated subjects while they were typing utterances, as the camera view would often switch during text entry. The number of utterances provided in 5 minutes varied significantly between test subjects, from eight utterances to 24 utterances. It was noted that subjects who were both more familiar with the experimental setup as well as nominally faster typists would provide more utterances, as expected. In several cases, human subjects provided ambiguous references such as, “near the black box” in an environment with two reference objects that could conceivably both be described as black boxes (a fridge and a filing cabinet).

A rigorous human subject study would be instrumental in providing generalizable insight into unstructured natural language human sensor statements, but these preliminary results from the pilot study are nevertheless useful in guiding development of the natural language interface.

Chapter 6

Conclusion

This chapter recaps the motivating problems of the work, summarizes the main contributions outlined in this thesis, and provides suggestions for future avenues of research.

In human-robot teaming, robots are often limited in autonomous reasoning capabilities due to a lack of information provided by sensors. The perceptive capabilities of humans, which continue to outpace those of robots until robotic perception is solved, make humans a worthwhile information source for their robotic partners. Semantic language is an efficient communication mechanism for human-generated soft data, but must be translated into a likelihood function in order for the robot to perform Bayesian inference and incorporate the data into its probabilistic state belief.

6.1 Contributions

6.1.1 Softmax Likelihood Synthesis

One of the most promising techniques for Bayesian inference with categorical semantic labels requires dense data to define softmax likelihood functions. In investigating properties of the softmax function, it was noted that the log-odds hyperplanes that function as equiprobable class boundaries can be used as constraints on model parameters. A priori knowledge, such as known geometry of reference objects, can be used to impose these constraints on model parameters, allowing for either uniquely defined parameter set or an underdetermined parameter set that may use sparse data for parameter tuning. This allows softmax models to be constructed with minimal data, while accounting for the size and shape of reference objects.

6.1.2 Multi-observation Likelihood Compression

Softmax likelihood synthesis is leveraged to enable efficient multi-observation likelihood compression. Some input semantic statements such as, "The target is in front of the house, inside the yard," contain observations relating to multiple softmax classes simultaneously. However, when combining these likelihoods through direct multiplication, the softmax denominator experiences combinatorial blowup due to the exponential set of joint observations in each model, causing the data fusion step to become intractable. This paper shows that two methods, geometric compression and nearest neighbour compression, can achieve great speedup at minimal accuracy loss. We showed that geometric fusion for multi-observation likelihoods can be computed online for simple MMS models.

6.1.3 Natural Language Human Sensor Models

Current research for natural language communication in the context of human-robot teaming focuses primarily on providing natural language commands to robot counterparts. We have investigated the preliminaries of natural language human sensing, in which an unstructured natural language utterance is mapped to semantically meaningful template statements and then used for Bayesian inference. A formal problem definition was presented and promising example results were shown from passing natural language inputs through a scenario-specific natural language pipeline.

6.2 Future Research

6.2.1 Softmax Likelihood Synthesis

A priori knowledge is currently used to defined softmax constraints; however, on-the-fly semantic map generation in uncertain environments [21] could be used to provide both labels and geometries in place of a priori knowledge.

As well, synthesis of models around non-convex polytopes remains an open issue, which impacts the speed and accuracy of the geometric MMS compression technique.

6.2.2 Multi-observation Likelihood Compression

Current MMS geometric compression techniques generate multiple softmax models with convex joint observation likelihoods with disjoint dominance regions. An ideal solution would be to construct a single non-convex joint observation likelihood from these disjoint dominance regions. This requires careful consideration of which normals for the component polytopes should be maintained at the MMS level, as well as a non-convex synthesis technique.

Furthermore, results for multi-observation likelihood compression require generalization: sensitivity analysis on elements such as the total number of classes contained in the compressed model would be helpful in selecting the ideal tradeoff between compression and accuracy.

6.2.3 Natural Language Human Sensor Models

Lastly, a great deal of work remains before natural language human sensing is a solved problem. Two main avenues of research are identified: first, a context-specific similarity measure would help disambiguate terms which are quite similar within the context of a massive training corpus; second, context-independent natural language components, such as part-of-speech taggers, could be leveraged to help generalize the translation mechanism between unstructured utterances and structured template statements.

Bibliography

[1] Raja Parasuraman, Thomas B. Sheridan, and Christopher D. Wickens. A model for types and levels of human interaction with automation. IEEE transactions on systems, man, and cybernetics. Part A, Systems and humans : a publication of the IEEE Systems, Man, and Cybernetics Society, 30(3):286–297, 2000.

[2] Michael A Goodrich, Bryan S Morse, Damon Gerhardt, Joseph L Cooper, Morgan Quigley, Julie A Adams, and Curtis Humphrey. Supporting wilderness search and rescue using a camera-equipped mini uav. Journal of Field Robotics, 25(1-2):89–110, 2008.

[3] Bryan S Morse, Cameron H Engh, and Michael A Goodrich. Uav video coverage quality maps and prioritized indexing for wilderness search and rescue. In Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction, pages 227–234. IEEE Press, 2010.

[4] Lanny Lin and Michael A Goodrich. Sliding autonomy for uav path-planning: Adding new dimensions to autonomy management. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, pages 1615–1624. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[5] JJ Moore, R Ivie, TJ Gledhill, Eric Mercer, and Michael A Goodrich. Modeling human workload in unmanned aerial systems. In AAAI Spring Symposium Series: Formal Verification and Modeling in Human-Machine Systems, pages 44–49. Citeseer, 2014.

[6] T.J. Gledhill, Eric Mercer, and Michael a. Goodrich. Modeling UASs for Role Fusion and Human Machine Interface Optimization. 2013 IEEE International Conference on Systems, Man, and Cybernetics, pages 1929–1937, oct 2013.

[7] David L. Hall, Michael McNeese, James Llinas, and Tracy Mullen. A framework for dynamic hard/soft fusion. Proceedings of the 11th International Conference on Information Fusion, FUSION 2008, 2008.

[8] Nisar R. Ahmed, Rina Tse, and Mark E. Campbell. Enabling Robust Human-Robot Cooperation through Flexible Fully Bayesian Shared Sensing. AAAI Spring Symposium: The Intersection of Robust Intelligence and Trust in Autonomous Systems, pages 2–10, 2014.

[9] Junaed Sattar and Gregory Dudek. Reducing Uncertainty in Human-Robot Interaction: A Cost Analysis Approach. In Oussama Khatib, Vijay Kumar, and Gaurav Sukhatme, editors, The 12th Annual Symposium on Experimental Robotics, volume 79 of Springer Tracts in Advanced Robotics, pages 81–95, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[10] Tobias Kaupp, Alexei Makarenko, Fabio Ramos, Ben Upcroft, Stefan Williams, and Hugh Durrant-Whyte. Adaptive human sensor model in sensor networks. 2005 7th International Conference on Information Fusion, FUSION, 1:748–755, 2005.

[11] Donald Reid. An algorithm for tracking multiple targets. IEEE transactions on Automatic Control, 24(6):843–854, 1979.

[12] Lotfi A Zadeh. The concept of a linguistic variable and its application to approximate reasoningi. Information sciences, 8(3):199–249, 1975.

[13] David M Mark and Ferenc Csillag. The nature of boundaries on area-class maps. Cartographica: The International Journal for Geographic Information and Geovisualization, 26(1):65–78, 1989.

[14] Daniel R Montello, Michael F Goodchild, Jonathon Gottsegen, and Peter Fohl. Where's downtown?: Behavioral methods for determining referents of vague spatial queries. Spatial Cognition & Computation, 3(2-3):185–204, 2003.

[15] Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In Experimental Robotics, pages 403–415. Springer, 2013.

[16] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 259–266. IEEE, 2010.

[17] Stefanie Tellex, Thomas Kollar, and Steven Dickerson. Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation. In AAAI Conference on Artificial Intelligence, pages 1507–1514, 2011.

[18] Luís Seabra Lopes and António Teixeira. Human-robot interaction through spoken language dialogue. In Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, volume 1, pages 528–534. IEEE, 2000.

[19] Thomas M. Howard, Stefanie Tellex, and Nicholas Roy. A natural language planner interface for mobile manipulators. IEEE International Conference on Robotics and Automation (ICRA), pages 6652–6659, 2014.

[20] Jacob Arkin and Thomas M. Howard. Towards Learning Efficient Models for Natural Language Understanding of Quantifiable Spatial Relationships. In Robotics: Science and Systems 2nd Workshop on Model Learning for Human-Robot Communication, 2016.

[21] Sachithra Hemachandra, Felix Duvallet, Thomas M Howard, Nicholas Roy, Anthony Stentz, and Matthew R Walter. Learning models for following natural language directions in unknown environments. In 2015 IEEE International Conference on Robotics and Automation (ICRA), pages 5608–5615. IEEE, 2015.

[22] Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Clarifying commands with information-theoretic human-robot dialog. Journal of Human-Robot Interaction, 2(2):58–79, 2013.

[23] Cynthia Matuszek, Dieter Fox, and Karl Koscher. Following Directions Using Statistical Machine Translation. In IEEE International Conference on Human-Robot Interaction, 2010.

[24] Nisar R. Ahmed, Eric Sample, and Mark Campbell. Bayesian Multicategorical Soft Data Fusion for Human–Robot Collaboration. IEEE Transactions on Robotics, 29(1):189–206, 2013.

[25] Bahador Khaleghi, Alaa Khamis, and Fakhreddin Karray. Random finite set theoretic based soft/hard data fusion with application for target tracking. 2010 IEEE Conference on Multisensor Fusion and Integration, pages 50–55, sep 2010.

[26] Jamie Frost, Alastair Harrison, Stephen Pulman, and Paull Newman. Mapping Spatial Language to Sensor Models. IEEE International Conference on Robotics and Automation, 2010.

[27] T Mikolov and J Dean. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 2013.

[28] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 302–308, 2014.

[29] Andrew Trask, Phil Michalak, and John Liu. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. arXiv preprint arXiv:1511.06388, 2015.

[30] Eric Nalisnick and Sachin Ravi. Infinite dimensional word embeddings. arXiv preprint arXiv:1511.05392, 2016.

[31] Nisar Ahmed and Mark Campbell. Variational bayesian learning of probabilistic discriminative models with latent softmax variables. IEEE Transactions on Signal Processing, 59(7):3143–3154, 2011.

[32] Nisar Ahmed, Eric Sample, Tsung-Lin Yang, Daniel Lee, Lucas de la Garza, Ahmed Elsamadisi, Arturo Sullivan, Kai Wang, Xinxiang Lao, Rina Tse, et al. Towards cooperative bayesian human-robot perception: Theory, experiments, opportunities. In Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence, Washington, 2013.

[33] Nisar R. Ahmed, J.R. Schoenberg, and Mark Campbell. Fast Weighted Exponential Product Rules for Robust General Multi-Robot Data Fusion. Robotics: Science and Systems, 2012.

[34] Nisar Ahmed, Eric Sample, Ken Ho, Tauhira Hoossainy, and Mark Campbell. Categorical soft data fusion via variational bayesian importance sampling with applications to cooperative search. In Proceedings of the 2011 American Control Conference, pages 1268–1273. IEEE, 2011.

[35] Nisar Ahmed and Mark Campbell. Variational bayesian data fusion of multi-class discrete observations with applications to cooperative human-robot estimation. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 186–191. IEEE, 2010.

[36] Frank Havlak and Mark Campbell. Discrete and continuous, probabilistic anticipation for autonomous robots in Urban environments. IEEE Transactions on Robotics, 30(2):461–474, 2014.

[37] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. Estimation with applications to tracking and navigation: theory algorithms and software. John Wiley & Sons, 2004.

[38] Rong Chen and Jun S Liu. Mixture kalman filters. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 62(3):493–508, 2000.

[39] Andrew R. Runnalls. Kullback-Leibler approach to Gaussian mixture reduction. IEEE Transactions on Aerospace and Electronic Systems, 43(3):989–999, 2007.

[40] Tobias Kaupp, Bertrand Douillard, Fabio Ramos, Alexei Makarenko, and Ben Upcroft. Shared environment representation for a human-robot team performing information fusion. Journal of Field Robotics, 24(11-12):911–942, 2007.

[41] Thomas M. Howard, Istvan Chung, Oron Propp, Matthew R. Walter, and Nicholas Roy. Efficient Natural Language Interfaces for Assistive Robots. In IROS Workshop on Rehabilitation and Assistive Robotics, 2014.

[42] Shun Taguchi, Tatsuya Suzuki, Soichiro Hayakawa, and Shinkichi Inagaki. Identification of Probability weighted multiple ARX models and its application to behavior analysis. Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference, (5):3952–3957, dec 2009.

[43] Nicholas Sweet and Nisar Ahmed. Structured Synthesis and Compression of Semantic Human Sensor Models for Bayesian Estimation. In American Controls Conference, 2016.

[44] Peter Gärdenfors. The geometry of meaning: Semantics based on conceptual spaces. MIT Press, 2014.

[45] R.J. Caron, J.F. McDonald, and C.M. Ponic. Classification of linear constraints as redundant or necessary. Journal of Optimization Theory and Applications, 62(2):225–237, 1989.

[46] Hairong Liu, Wenyu Liu, and Longin Jan Latecki. Convex shape decomposition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 97–104, 2010.

[47] Jyh-Ming Lien and Nancy M Amato. Approximate convex decomposition of polygons. In Proceedings of the twentieth annual symposium on Computational geometry, pages 17–26. ACM, 2004.

[48] Tobias Kaupp, Alexei Makarenko, and Hugh Durrant-Whyte. Human–robot communication for collaborative decision makinga probabilistic approach. Robotics and Autonomous Systems, 58(5):444–456, 2010.

[49] Nicholas Sweet, Nisar R Ahmed, Ugur Kuter, and Christopher Miller. Towards self-confidence in autonomous systems. In AIAA Infotech@ Aerospace, page 1651, 2016.

[50] Kin Gwn Lore, Nicholas Sweet, Kundan Kumar, Nisar Ahmed, and Soumik Sarkar. Deep value of information estimators for collaborative human-machine information gathering. In 2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS). IEEE, 2016.

[51] Rohan Paul, Jacob Arkin, Nicholas Roy, and Thomas M. Howard. Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. In Proceedings of Robotics: Science and Systems, AnnArbor, Michigan, June 2016.

[52] Lukas Michelbacher. Multi-Word Tokenization for Natural Language Processing. PhD thesis, Universitt Stuttgart, 2013.

[53] John Lafferty, Andrew McCallum, and Fernando C N Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning, 8(June):282–289, 2001.

[54] Taku Kudo. CRF++: Yet Another CRF toolkit, 2003.

[55] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine, 4(1):23–33, 1997.

[56] Jill L Drury, Jean Scholtz, and Holly A. Yanco. Awareness in Human-Robot Interactions. In IEEE International Conference on Systems, Man and Cybernetics, 2003.