UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Perceptual Algorithms for Social Robots in Early Childhood Education

Permalink

https://escholarship.org/uc/item/2c24321w

Author

Malmir, Mohsen

Publication Date

2017

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Perceptual Algorithms for Social Robots in Early Childhood Education

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy

in

Computer Science

by

Mohsen Malmir

Committee in charge:

Professor Garrison W. Cottrell, Chair
Professor Sanjoy Dasgupta
Professor Lawrence K. Saul
Professor Zhuowen Tu
Professor Nuno M. Vasconcelos

2017

Copyright

Mohsen Malmir, 2017

All rights reserved.

The Dissertation of Mohsen Malmir is approved and is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2017

DEDICATION

To my wife,
and to my parents in Iran.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 2, in full, is a reprint of the material as it appears in International Conference on Computer Vision Workshops (ICCVW) 2013. Malmir, Mohsen; Forster, Deborah; Youngstrom, Kendall; Morrison, Lydia; Movellan, J avierR., copyright IEEE, 2013. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in International Conference on Human Robot Interaction Workshops (HRIW) 2014. Movellan, J avierR.; Malmir, Mohsen; Forster, Deborah. copyright IEEE, 2014. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in Proceedings of the British Machine Vision Conference (BMVC) 2015. Malmir, Mohsen; Sikka, Karan; Forster, Deborah; Movellan, Javier R.; Cottrell, Garrison W. copyright BMVA Press, 2015. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in full, is a reprint of the material as it appears in Computer Vision and Image Understanding (CVIU) 2017. Malmir, Mohsen; Sikka, Karan; Forster, Deborah; Fasel, Ian; Movellan, Javier R.; Cottrell, Garrison W. copyright Elsevier, 2017. The dissertation author was the primary investigator and author of this paper.

Chapter 6, in full, is a reprint of the material as it submitted to IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2017. Malmir, Mohsen; Cottrell, Garrison W.; The dissertation author was the primary investigator and author of this paper.

VITA

2006	Bachelor of Engineering, Shahid Chamran University of Ahvaz, Iran
2009	Masters of Science, AmirKabir University of Technology, Iran
2017	Doctor of Philosophy, University of California, San Diego

PUBLICATIONS

M. Malmir, K. Sikka, D. Forster, I. Fasel, J. R. Movellan, G. W. Cottrell, Deep active object recognition by joint label and action prediction, Computer Vision and Image Understanding, Volume 156, Pages 128-137, ISSN 1077-3142, 2017.

M. Malmir, K. Sikka, D. Forster, J. Movellan and G. W. Cottrell. Deep Q-learning for Active Recognition of GERMS: Baseline performance on a standardized dataset for active learning. In Xianghua Xie, Mark W. Jones, and Gary K. L. Tam, editors, Proceedings of the British Machine Vision Conference (BMVC), pages 161.1-161.11. BMVA Press, September 2015.

J. R. Movellan, M. Malmir and D. Forester, HRI as a tool to monitor socio- emotional development in early childhood education., In IEEE International Conference on Human Robot Interaction Workshop (HRIW), Bielefeld, Germany, 2014.

M. Malmir, D. Forster, K. Youngstrom, L. Morrison, J. R. Movellan, Home Alone: Social Robots for Digital Ethnography of Toddler Behavior, Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on , vol., no., pp.762,768, 2-8 Dec. 2013.

M. Malmir, S. Shiry, A Model of the Primate Visual Cortex based on Category Specific Redundancies in Natural Images, Connection Science, vol.22, issue.4, pp. 313-329, Taylor and Francis, 2010.

M. Malmir, S. Shiry, Object recognition with statistically independent features: a model Inspired by the primate visual cortex, RoboCup 2009, LNAI 5949, pp. 204–214. Springer, Heidelberg (2010).(Best Paper Award from 13th international Robocup symposium).

M. Malmir, S. Shiry, Class specific redundancies in natural images: a theory of extrastriate visual processing, International Joint Conference on Neural Networks (IJCNN), Atlanta, GA, USA, 2009.

M. Malmir, S. Shiry, A Model of Angle Selectivity in Area V2 with Local Divisive Normalization, IEEE Symposium on Computational Intelligence for Multimedia Signal and Vision Processing (CIMSVP), Nashville, TN, USA, 2009.

FIELDS OF STUDY

Major Field: Computer Science

ABSTRACT OF THE DISSERTATION

Perceptual Algorithms for Social Robots in Early Childhood Education

by

Mohsen Malmir

Doctor of Philosophy in Computer Science

University of California, San Diego, 2017

Professor Garrison W. Cottrell, Chair

Social robots are becoming a part of everyday life, including household companions, education and healthcare assistants, museum guides and hotel delivery bots. These robots can benefit from machine perception algorithms that automatically recognize people, their facial expressions and the surrounding environment. In this thesis, we develop perceptual algorithms for a social robot called RUBI, who is used to interact with toddlers in the classroom and enrich the early childhood education environment. We develop a classic computer vision pipeline for RUBI to recognize and analyze toddlers' faces in the classroom. In the second and third chapters of this thesis, we show how

RUBI can use this capability to extract the social structure of a group of toddlers, as well as the children's preferences for different activities and their playmates.

In the remainder of this thesis, we develop Active Object Recognition (AOR) models for RUBI. While interacting with children, RUBI can teach object names to children, to extend their vocabulary and monitor their learning skills. In chapter 4, we introduce GERMS, a dataset designed to accelerate progress on AOR in the context of human robot interaction. This dataset is designed for RUBI to recognize toys in her grippers so that she can name them for the children. We use deep Q-learning to train RUBI to optimize her actions (rotating the toy in her gripper) for toy recognition. In chapter 5, we improve her performance by training a deep neural network end-to-end, for joint object label and action prediction. A generative model of object similarities based on Dirichlet distribution is proposed and embedded in the network for encoding the state of the system. In chapter 6, we propose a method for supervised learning of AOR policy. We formulate AOR as a Partially Observable Markov Decision Process (POMDP) and extract rollouts of object inspection using Belief Tree search. We then train a Long Short Term Memory network (LSTM) on these rollouts to predict the best next action. We show improvement in recognition performance by iteratively optimizing the observation function, and then retraining the supervised LSTM policy network.

Chapter 1

Introduction

1.1 Perceptual Primitives for Social Robots

Social robots are becoming prevalent in all aspects of everyday human life. There are numerous companies and startups that are developing robots as family companions, health care companions, for delivering groceries, as museum guides, and for educational and therapeutic purposes. These robots come in a variety of forms and sizes, with some, such as Nao®, adapting a humanoid design, while others, such as Jibo®, are designed to express social behavior without an explicit human form. Despite the wide range of appearances and applications of social robots, they all need *perceptual capabilities* that enable them to perceive and understand the surrounding environment, people, and activities, and respond in an appropriate way.

In this thesis, we develop perceptual algorithms for a social robot called RUBI, which is used in a daycare classroom for teaching and monitoring the learning performance of children. The application of robots as a teaching agent to deliver educational content and monitor students' learning is becoming a common research trend. For example, robots have been used to teach vocabulary [51, 21], science [35, 42], geometric shapes [33], language [31, 70, 22], dance and music [23] and story telling [26]. See figure 1.1 for example settings of child-robot interaction.

Figure 1.1. Top-left: Nao teaching children to sort objects by shape, top-right: DragonBot teaching words, bottom-left: IROBI teaching English, bottom-right: iCat teaching language.

In these works, researchers are trying to answer questions such as, is a physical robot more effective than on-screen interactive technology [22] or an animated virtual robot [42]? When taught by a physical robot, will the children form a friendship with the robot [31], and would they treat the robot as a peer [26]? Most of these studies were carried out in a wizard-of-Oz (WOZ) setting in which a person remotely controlled the robot and its behavior. In contrast to the WOZ setting, an autonomous robot is one that can automatically recognize people and their behavior, understand social cues and generate appropriate behavior in response to these external factors. A robot equipped with such capabilities reduces the manual labor required to analyze the results of experiments in using the robot to educate children, by keeping track of the amount of interaction with each child and that child's performance on the tasks. A robot with these capabilities can also serve as an automated monitoring tool for teachers on the child's developing skills. Finally, such a robot can extract the social structure of the environment by collecting statistics on interactions between the children.

The primary subject of this thesis is to develop *perceptual primitives* for RUBI, and to showcase examples of how these capabilities can be used effectively in the classroom. We develop face recognition for RUBI, and show that when RUBI is equipped with face recognition, she can extract useful information about the social structure of the classroom. We develop active object recognition methods for RUBI, in order to provide her with the capability of recognizing toys handed to her by the children.

In the next section, we provide a brief overview of the RUBI project. Then we describe the contributions and outline of this thesis.

1.2 RUBI Project

The RUBI project started in 2004, with the goal of enriching the classroom environment in early childhood education [55]. Over 10 years, six different prototypes of RUBI were developed and tested in the Early Childhood Education Center (ECEC) at UCSD (see Figure 1.2). Earlier versions of RUBI were Wizard of Oz experiments, while later versions were fully autonomous in generating behavior and interacting with the children. Over the course of 10 years, various perceptual primitives were gradually added to RUBI, enabling the robot to generate more complex behavior while interacting with the children.

One of the earliest perceptual primitives added to RUBI was the ability to detect faces and recognize facial expressions [7]. This enabled early versions of RUBI to perform real-time social interaction by extracting dynamics of facial expressions from images and automatically evaluating RUBI's interactions with the children [19, 39, 40, 74]. In chapter 2 and 3, we will use facial expression recognition on the data collected by RUBI-5 in the classroom to predict the children's preference over different activities while playing with the robot.

Along with these perceptual capabilities, the ability to *learn to respond* to the

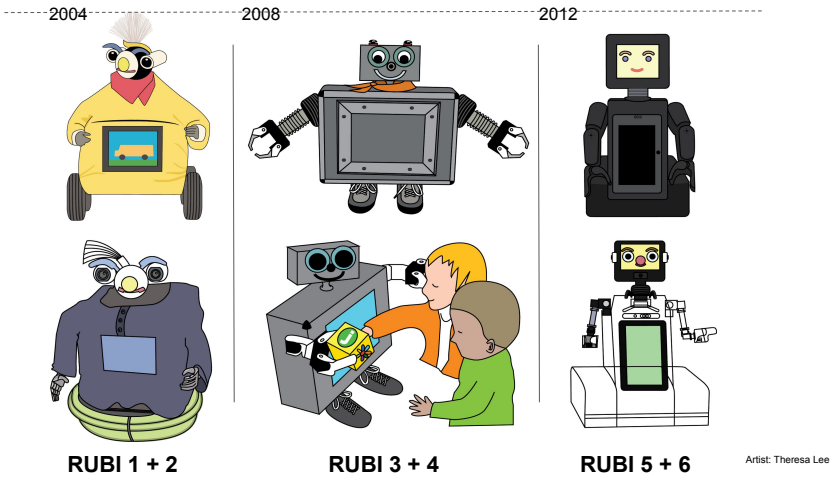


Figure 1.2. RUBI 1-6. The robots exhibit different levels of autonomy and complexity.

environment in an appropriate way was implemented in the form an Infomax controller, allowing RUBI to model the social contingencies of the environment and respond to sounds in a meaningful way [52]. The ability to perceive touch allowed RUBI to categorize the behavior of children towards itself and to conclude that caring behavior and object-related touches were sustained over longer periods of time, compared to curiosity-driven touch behavior [80]. The Infomax controller and touch sensing capability were later used in a study to demonstrate the children's sustained interest in RUBI over long periods of time [79]. Visual saliency detection allowed RUBI to focus its visual sensors more often on the people in the environment [13]. Automatic cry [68] and mood detection [67] allowed RUBI to act in a manner consistent with the current activities in the classroom.

1.3 Contribution of this Thesis

This thesis can be divided to two parts based on the technical contributions. In the first part, we develop a face recognition model for RUBI and show how RUBI can use that to enrich its analysis of the emotional dynamics and social structure of the classroom.

In the second part, we focus on active object recognition, where we develop models for in-hand object recognition for RUBI. Active object recognition can be used by RUBI to teach object naming to toddlers during "give-and-take" activity.

1.3.1 Face Recognition and Social Structure of the Classroom

In the first two chapters, we develop a face recognition module for RUBI by collecting a large number of face images, manually labeling the images and training a supervised classifier on the extracted visual features. We show that RUBI, equipped with person recognition, extracts interesting social dynamics from the classroom. We also showcase the usefulness of facial expression recognition in RUBI, by analyzing the emotional development of the classroom during song-and-dance activity.

In a field study that lasted more than 6 months, RUBI collected images of children in the classroom during different activities. We extract faces from these images, and compiled that into a face recognition dataset, called ECEC faces. The dataset contains more than 7K images of 16 toddlers and 12 adults. The face images vary greatly in depth rotation and facial expressions. It is specially challenging since it captures a period of 6 months during which faces of toddlers changes dramatically. Figure 1.3 shows images of one toddler from the ECEC faces dataset.

Figure 1.3. Example face images from ECEC face dataset. We show the temporal evolution of a face of one toddler over 5 months. Each image is labeled with its capture date.

We develop a face recognition module for RUBI by building a preprocessing,

feature extraction and classification pipeline. The preprocessing is minimal. We crop faces detected by the OpenCV face detector. For feature extraction, we compare HOG, LBP and Daisy features and show that LBP is more efficient and leads to higher classification accuracy. We test the system with multinomial regression and SVM classifiers, and conclude that the resulting accuracy is very similar for both of these classifiers. This pipeline achieves 84% accuracy on the test set portion of the ECEC faces dataset.

We show that using face recognition, RUBI can analyze the social structure of the classroom. We analyze the preferences of children to play with each other, or to avoid each other while in front of RUBI. The result is a diagram, called a RUBI-gram, that depicts the children as nodes of a graph and their preference as the edges. Teachers at ECEC have expressed their desire for such automated social interaction information as can be gained from the RUBI-gram.

We show how RUBI can use the ability to recognize facial expressions to extract emotional dynamics of the classroom and toddlers' preferences for different activities. We used the FACET SDK that extracts six channels depicting different emotions from face images. We showed that RUBI can predict the preference of toddlers for different activities, using the *joy* channel. Her predictions are as good as the teachers' ratings of the children's preferences. As an example analysis, we extracted the dynamics of joy channel, averaged over multiple trials, during "the bus song". Interesting information is revealed in this analysis, which shows *peaks* of joy during certain verses of the song.

1.3.2 Active Object Recognition for Object Naming

Object naming is an important development stage in toddlers, and is the gateway to developing language. We would like RUBI to teach object names to toddlers using one of RUBI's most beloved activities called "give-and-take", in which toddlers bring an object and hand it to RUBI. The robot then examines the object and says its name, after

which returns the object to the toddler. In the context of give-and-take, RUBI has the opportunity to teach object names to children, assess their knowledge of object names, and compare the efficacy of 3D in-hand object learning to 2D on-screen object learning. In the second part of this thesis, we focus on developing an active object recognition model for RUBI to enable this activity.

We collected a dataset of 136 "giant microbes" objects to teach object naming to children. We select giant microbes to minimize the effect of children's prior knowledge of objects on the results of experiments. For each object in this dataset, we put the object in RUBI's gripper and let RUBI collect images while rotating the object. For each object, 10 different in-hand views are collected, 6 of which are used for training and the rest for test set. The images are accompanied by the gripper location that rotates the object, allowing simulation of different actions while examining objects. We call this dataset *GERMS*, and make it publicly available to the active object recognition community. Comparing to other datasets, GERMS is more challenging, includes larger number of objects and views, and has complex background that simulates in-class situation.

We developed a model for active object recognition of GERMS by training a neural network using reinforcement learning to predict the action-values for object exploration actions. The learning rule for network parameters was derived using an iterative Q-learning approach. In this work, the appearance of objects is fixed before learning object inspection policy. We show the performance of the proposed Deep Q-network on active recognition of GERMS, and compare it to a random strategy. These results serve as a baseline for the GERMS dataset, to which other researchers can compare the performance of their methods.

We extended the previous model to learn the object appearance along with object exploration policy. We trained a neural network end-to-end using classification and action-value prediction errors. We experimentally show that the network can be trained

using errors from prediction of label and action modalities. The network predicts actions and label in two different layers, and images are selected based on the proposed action to train the network. We found that a plain Deep Q-learning network fails to perform any better than random in this case, due to overfitting to a subset of images. To alleviate this problem, we introduce a Dirichlet layer that encodes the belief of objects based on the selected action. The Dirichlet layer is embedded in the network and trained using gradient descend along with other parameters of the network. We show that adding Dirichet layer significantly improves the performance of the layer, and partly cures the overfitting problem. We show that the proposed network achieves state of the art results on the GERMS dataset, surpassing the plain Deep Q-learning network.

In previously mentioned approaches, and other recent methods on active object recognition, this problem has been approached as an semi-supervised learning paradigm in which optimal trajectories for object inspection are not known and to be discovered by reducing label uncertainty or training with reinforcement learning. These methods have no guarantee of their optimality, even on the training set. In the last contribution of this thesis, we formulate Active Object Recognition (AOR) as a Partially Observable Markov Decision Process (POMDP) and find near-optimal values and corresponding action-values of training data using Belief Tree Search (BTS) on the AOR belief Markov Decision Process (MDP). We train a Long Short Term Memory (LSTM) network on these values to predict the best next action on the training set rollouts and experimentally show that our method generalizes well to explore novel objects and novel views of familiar objects with high accuracy. We compare this supervised scheme against guided policy search, and show that the LSTM network reaches higher recognition accuracy compared to the guided policy search and guided Neurally Fitted Q-iteration. We further look into optimizing the observation function to increase the total collected reward during active recognition. We also look into optimizing the observation function by deriving a

gradient-based update to increase the total expected reward. We show that by optimizing the observation function and retraining the supervised LSTM network, active object recognition performance on the GERMS improves significantly.

1.4 Thesis Outline

The perceptual capabilities we mentioned above allowed RUBI to better understand the surrounding social dynamics in the classroom. However the ability to recognize people and the environment could largely benefit the robot, researchers, and also the teachers and students. In this thesis, we add person and object recognition abilities to RUBI. Person recognition will reduce the analysis time of the experiments significantly, while enabling RUBI to extract subject-specific statistics from the classroom. Object recognition allows RUBI to improve its interaction with children and teach object names to children. Here, we briefly mention the outline of this thesis.

In chapter 1, we go into the details of preparing RUBI-5 hardware and software for its first field study. We delve into the details of preparing RUBI, and refining its facial expression generation. Then we describe the person recognition algorithm implemented in RUBI-5. This robot operated autonomously in classroom 1B in Early Childhood Education Center (ECEC) at UCSD, collecting images of children and teachers over a period of 6 months. To train the person recognition system, we extract and manually label faces of teachers and toddlers from images that RUBI-5 collected during this time. Using the extracted facial expressions and identities, we predict the children's preference over different activities. We show that by using joy as the indicator, RUBI-5 can predict the children's preference over several games *as good as* teachers.

In chapter 2, we demonstrate the utility of person recognition in the extraction of social dynamics of the classroom. In this work, we analyze the data collected during a one month field study, in which RUBI collected facial expressions and identity of toddlers

in the classroom. First we analyze the dynamics of the classroom affect during the sing-and-dance activity. Then we present the analysis of social structure of the classroom, in which the preference of children to avoid each other or play together in front of RUBI is shown in the *RUBI-gram* graphical form.

Object naming is an essential skill for developing children, and can reveal a great deal about their current assessments of the referents of the words they know, as well as provide scaffolding for relational thinking and analogical reasoning. In chapter 3, we develop an object recognition algorithm for RUBI, that allows the robot to teach object names to toddlers. The main contribution of chapter 3 is to introduce GERMS, a dataset for active object recognition, that we use in the classroom to teach object names and assess children's ability to learn new objects. We collect a dataset of 136 different objects by using RUBI-6 to hold the objects in its gripper and capture images while rotating the object. We use GERMS to train an *active object recognition* model using deep Q-learning, and show that the proposed method for active recognition of GERMS is superior to random selection of actions while examining objects for recognition.

In chapter 4, we delve deep into development of a deep learning model for active object recognition. Deep learning has become the most active branch in computer vision, and large progress has been made on object recognition. We develop a deep learning model that is trained to perform active object recognition by predicting the best next action given the current view of the object. The network is trained using an stochastic update rule based on deep Q-learning. The proposed network is trained end-to-end to simultaneously predict object label and action. In order to reduce overfitting of the model to specific objects, a layer of Dirichlet belief encoding is added to the network that is intended to provide a generative model of object-action beliefs. We derive a gradient-based update rule for the Dirichlet layer, and show that its performance is improved significantly compared to the regular deep Q-learning network.

In chapter 5, we solve the active object recognition by reducing it to a supervised learning problem. We formulate the active object recognition as a Partially Observable Markov Decision Process (POMDP), and adapt a belief tree search algorithm to calculate the values of object beliefs that are arbitrarily close to the optimal value. The extracted values are then used to train a long short-term memory (LSTM) network that predicts the best next action given the current view of the object. The proposed method has advantages over previous approaches, such as fast training and convergence to the desired policy, no requirement for a generative model of objects and generalization to novel objects. We show that this method achieves state of the art performance on the GERMS dataset. We also show that the active recognition of objects can be used to update the image classification model, using a weighted version of training data, that leads to increased accuracy of static and active object recognition.

Chapter 2

Home Alone: Social Robots for Digital Ethnography of Toddler Behavior

2.1 Abstract

We describe the results of a field study in which the social robot RUBI-5, was left alone for a 28 day period to interact autonomously with 16 toddlers at an Early Childhood Education Center. The study is part of the RUBI project, which started in 2004 with the goal of exploring the potential of social robotics for research and enrichment of early childhood education environments. As part of the 28 day field study RUBI-5 collected data about the facial expressions, activities, and spatio-temporal proximity of the toddlers. We found that RUBI-5 could use the facial expression data to accurately predict the children's preference for different activities: on average robot agreed with human judges as much (Pearson Correlation =0.67) as human judges agreed with each other (Pearson Correlation = 0.68). In addition RUBI discovered some useful aspects of the social structure of the toddler's group. The study is an important milestone in social robotics, both for the length of time the robot could interact autonomously with children, and for the richness of the data that it provided. The results indicate that social robots have the potential to act as low cost, autonomous "digital ethnographers" in a manner that may revolutionize the science and technology of early childhood education.

2.2 Introduction

An unprecedented number of children in the US start public school with major deficits in basic academic skills [77]. Scientific evidence shows that children who have early failure experiences in school are those who are most likely to become inattentive, disruptive, or withdrawn later on. Empirical research using longitudinal randomized control studies is now showing that early childhood education programs can effectively prevent academic deficits [77]. However, due to their high costs, such programs may not find widespread use. Thus it is critical to find innovative ways to gather and analyze data on early childhood education so as to better understand toddler's behavior and to conduct rapid, big-data experiments. As part of this overall vision, the RUBI project started back in 2004 with the goal of studying the potential of social robot technologies in early childhood education. Since then 5 different robot prototypes have been developed and immersed on an early childhood education center for sustained periods of time (see Figure 2.1). The early prototypes (RUBI-1, 2) were remotely operated by humans. RUBI-3 was a transitional design. RUBI-4 was the first prototype to operate autonomously for a period of 15 days. RUBI-4 provided useful data about toddler behavior. In particular it was shown that a 2-week period of interaction with RUBI-4 resulted in improvement of vocabulary skills in 18-24 month olds [51]. However many of the results found with RUBI-4 required for human ethnographers to analyze hundreds of hours of video, a process that was both slow and costly. The latest prototype (RUBI-5) (Figure 2.1, bottom row) was designed to operate as an autonomous "digital ethnographer" that would embed itself on the daily routine of the toddlers life and enrich their environment while gathering and analyzing the observed behaviors.

After a construction period that lasted several years, including hundreds of hours of short-term field studies, we felt that RUBI-5 was ready for a long-term study. One of

Figure 2.1. Top row: RUBI progression: from left to right RUBI-1, RUBI-2, RUBI-3 & RUBI-4. RUBI-1 and 2 were remotely operated. RUBI-3 was the first fully autonomous design. Bottom row: RUBI-5 playing with kids at ECEC classroom 2A.

our goals was to break the previous record of 15 days of autonomous operation, achieved by RUBI-4. More importantly we wanted to explore whether RUBI-5 could autonomously data-mine the toddler's behavior and provide insights about what activities the children like most. To this end RUBI-5 was equipped with off-the-shelf computer vision tools to recognize the children she interacted with and to analyze their facial behavior. Here we show that these tools, while still not perfect, provided very useful information, including the preference of children for different activities, as well as sociograms indicating which children tend to interact with RUBI and one another on a daily basis.

The rest of the paper is organized as follows. In Section 2, we describe the design and modification process for RUBI-5 along with the software architecture we used to make this study feasible. In Section 3, we describe the specification of the study in terms of experiments setup, number of toddlers involved and the data collection and processing procedures. Section 4 describes the results for the experiment and is followed by a conclusions section.

2.3 Robot Design

For more than 8 years the RUBI robot series have been built and improved using the experiences achieved from thousands of hours of field studies [19, 55, 54, 68, 69, 79, 30]. During these studies, robot prototypes were immersed in an uncontrolled environment to interact with toddlers at the Early Childhood Education Center (ECEC) at the University of California, San Diego. User-friendly appearance, ability to operate autonomously, low cost design and implementation, and safety of interaction with toddlers were among the early design criteria that emerged from these field studies. As a better understanding was gained of the sort of interactions that emerge between toddlers and robots, the emphasis switched from Wizard-of-Oz studies in which RUBI was tele-operated by a human, to sustained, full autonomy. RUBI-4 was the first robot of the series that operated autonomously for 15 consecutive days. To achieve this end a relatively simple design was used (e.g., single degree of freedom head, no facial expressions, 2 degree of freedom arms). While RUBI-4 provided very useful data, confirming that children that interact with the robot show significant increases in vocabulary skills, the analysis of the obtained data was very time consuming and expensive. The main bottleneck was the human coding of 14 days times 8 hours per day times 3 cameras. This coding ended up taking more than a year of time. Thus the design of RUBI-5, the robot prototype presented in this document, focused on two aspects: (1) increase the complexity of the robot so as to support more complex forms of interactions than RUBI-4 could do, and (2) use machine perception software to automatically analyze the sensory data. The goal was for RUBI-5 to become an autonomous "digital ethnographer" that could be left alone with a group of toddlers to analyze their behavior while enriching their educational environment. In the following two subsections we describe RUBI-5 and go into the details of hardware components and software architecture.

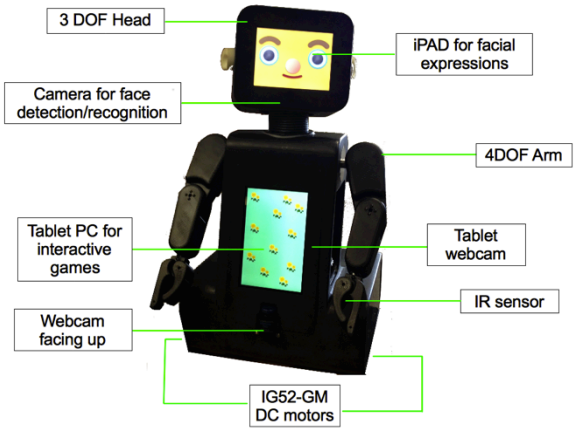


Figure 2.2. RUBI-5 appearance and hardware list.

2.3.1 Hardware

Figure 2.2 shows RUBI-5, the latest prototype of the RUBI series. This was the first prototype developed using modern digital fabrication methods. The robot is 24x20x37 inches. Her drive train consists of two Shayang Ye IG-52GM motors with magnetic encoders, controlled by Roboteq's MDC2250 Motor Controller board. Each of Rubi's arms has 4 DOF, independently controlled using Robotis' Dynamixel servos EX106+, RX64 and RX28. Each hand has an IR sensor inside the gripper that can detect an object. The head has 3 DOF, a webcam for image capture and an iPad2 for the face. RUBI's "belly" has a touchscreen tablet PC, which is used to display educational games and popular songs. A MacMini server with 2 GHz Intel Core i7 processor and 8 GB of RAM runs the Robot Operating System (ROS), the machine perception engines (face detection, person recognition, expression recognition), activity scheduling, and motor control algorithms.

Because the toddlers can stand in different distances to RUBI-5 while playing the games or dancing with the songs, we installed 3 cameras, each capturing part of the robot's surrounding area: One in the head, another in the torso and the last one under the tablet. Figure 2.3 shows shots of the same scene from 3 different cameras.

2.3.2 Software

ROS: software architecture is based on the Robot Operating System (ROS). The entire system is distributed and works by passing ROS messages between ROS nodes that provide a variety of services. A node called RUBIScheduler is a finite state machine that schedules the activity to play with the children. This is based on the previous activities, the amount of time since the children touched RUBI's belly and the constraints of the ECEC classroom's daily schedule.

Games: There were two types of material presented on the touch-screen: (1) RUBI sings popular songs ("Wheels on the Bus", and "Monkeys on the Bed") while she physically dances and shows an animation in her belly's touchscreen. (2) Educational Games. These are Flash-based educational games targeting vocabulary development. For example, in one game 4 images are presented on the screen and RUBI asked to touch one of them (e.g., where is the orange?). These games combine sounds and visuals presented on RUBI's belly, as well as physical actions, like clapping, looking towards the screen when the child touches it, and smiling. In addition RUBI can play "Give and Take" games (see Figure 2.1 bottom row). In this game children give objects to RUBI. She takes the object, looks at it and gives it back to the child saying "Thank You".

If RUBI's belly is not touched for a period of 10 seconds the RUBIScheduler puts her in "Idle Mode", in which she makes randomly scheduled idle movements, and displays simple visuals on her belly. When children touch RUBI's belly the scheduler chooses a new activity, provided it is consistent with the classroom's daily schedule.

Facial Expression Production: RUBI's facial expressions are controlled by a ROS node, called RUBIFace, running on an iPad. RUBIFace animates facial expressions using 23 different parameters. The set of facial expressions used at ECEC field studies were selected using a survey conducted on 30 different adults. These people we asked

Figure 2.3. Scene shots from RUBI-5's three different webcams. The left image is the picture from camera under the tablet. The middle picture is the tablet webcam's shot and the right one is captured by the webcam inside the head.

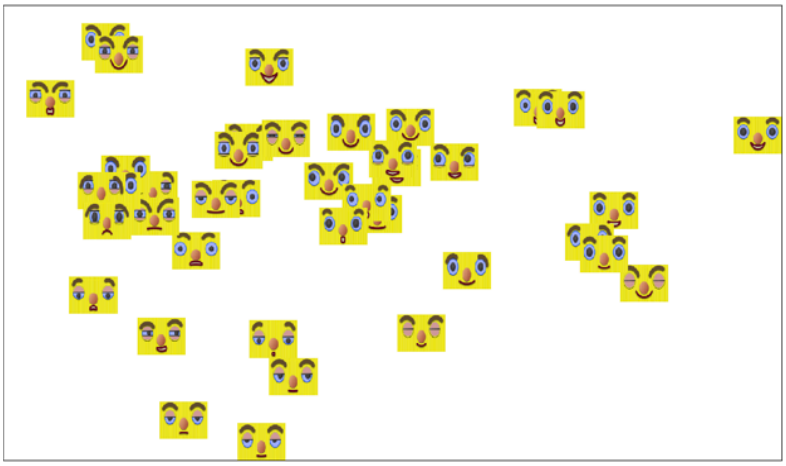


Figure 2.4. Visualization of the results of RUBI-5 online facial survey. People were asked to give their opinions about each face by grading them in different aspect such as positive-negative, active-passive and like-dislike.

to give scores to a wide range of facial expressions based on multiple questions. We performed MDS on the survey's outputs to represent the different expressions using a 2 dimensional space, which can be roughly interpreted as "valence" and "energy". Figure 2.4 shows the results for 40 different faces. We picked the top 10 images with the highest positive valence.

Person Recognition: RUBI-5 captures the ongoing scene using three cameras, located in the head, right and center of the belly's tablet and bellow the belly. Currently these cameras are used to identify who is playing with RUBI and what facial expressions they are making. The face recognition system was developed by us, using the following

Figure 2.5. Example faces from our dataset. Each row represents different views of the same child. The images were collected over a course of 8 months at ECEC.

pipeline. After an image is captured, we use the OpenCV face detector to find the faces in the image. Each detected face is normalized to a specific size, it is converted to a 4 layer Gaussian image pyramid with a between layer downscale of 1.2. Daisy features [81] are extracted from overlapping image patches from each layer of this pyramid. PCA is then used to reduce the dimensionality of the features. A Multinomial Logistic Regression classifier is used to recognize the different participants. The classifier was trained using 7000 images of 28 subject collected by RUBI. Of these 28 subjects, 16 were toddlers and 12 were adults, including classroom teachers and researchers accompanying RUBI. Because adults were also present in the field while RUBI-5 was interacting with kids, we added them to our dataset to reduce the number of false positives. Figure 2.5 shows some examples of this dataset. We divided the entire dataset into two non-overlapping sets: train and test. The test set size was 35% of the entire dataset. After training using the training set we tested the system on the test set using the following procedure: For each image on the test set the system had to choose amongst 28 possible alternatives (16 toddlers, plus 12 adults). The system guessed the correct alternative with 93% accuracy.

Facial Expression Recognition: Facial expression recognition was performed

using the FACET R1.1 SDK from Emotient.com. FACET is the commercial version of CERT [41], one of the most popular and accurate facial expression recognition systems. FACET R1.1 recognizes 6 primary expressions of emotion: anger, disgust, fear, joy, sadness and surprise. Since FACET R1.1 was trained to recognize adult facial expressions from faces that deviate no more than 15 degrees from frontal, it was not clear whether the system would prove useful to recognize toddler facial expressions.

2.4 Study Design

Participants: 16 toddlers (ages 11 to 23 months) from room 2A of UCSD's Early Childhood Education Center (ECEC) enrolled during the period of Jan 24 to September 11, 2013. The total number of children at any given time ranged from 9-12. Two teachers informally observed the interaction between the children and RUBI. A research assistant under the supervision of an ethnographer took notes to characterize the observed interactions between children and robot using standard ethnographic methods.

Procedure: RUBI was left alone in Room 2A of ECEC, starting on Jan 24, for increasingly longer periods of field testing. On Aug 12 we brought RUBI to ECEC with the intention of continuing the study until she stopped operating. This happened on September 11, 2013. During this period RUBI was relatively stationary, making only small rotational movements with the drivetrain, thus allowing to obtain power using a standard electrical outlet. RUBI-5 ran on two types of schedules: continuously while research staff were on location; and an automated schedule designed to coincide with activity periods chosen by the educational staff as curriculum appropriate. During every session, RUBI-5 was on idle state until someone touched her belly. At this time, she chose either a game or a song. The songs always end after a specific pre-determined time, while the games continue until no one touches the belly for 10 seconds. After game or song has finished, RUBI goes back to the idle state, showing the idle game on the belly

and looking around while moving slightly her arms and head. This cycle continues until the session finishes.

Data: During each session, RUBI kept the log of the games and songs that were played. She also recorded images from the three RUBI cameras. The head and belly mounted cameras captured an image when they detect a face and a game is being played. The tablet camera captured images every 2 seconds during the game episodes. These pictures were then processed to extract the identity using the face recognition described in section 2.2. Facial expressions were also extracted using the FACET SDK.

2.5 Results

Expression Recognition: One of our goals was to test whether RUBI could automatically detect which kind of activities the children liked most. To this end we asked 2 teachers and the research assistant to rank from last (1) to first (10) how much the children liked the 10 activities they did with RUBI: "Give and Take", "Bus Song", "Monkey Song", "Animals Game", "Balloons Game", "Fruits Game", "Transportation Game", "Photos Game", "Triangles Game", "Objects Game". The average Pearson correlation between the three human judges was 0.68. We then computed the correlation between the output of the different emotion recognition channels and the activity rankings averaged across the 3 human judges. The independent variable was the total number of images greater than 0.95 on the corresponding emotion channel. This means that FACET was at least 95% sure that the face exhibited the expression of the target emotion. There was a statistically significant correlation (r=0.73, $p < 0.05$, 2-tails) between the output of the Joy channel and the average human rankings of the different games (see Table 2.1). The average agreement between the Joy channel and each of the human judges was 0.67, which was very close to the average agreement between the 3 judges (0.68). We were surprised that some of the channels associated with negative emotions (e.g.,

Table 2.1. Correlation between expert observer ranking of preferred game and FACET output channels.)

Channel	Pearson Correlation	Significance: 2-tailed t-test (9 df)
Anger	0.179	$p > 0.05\%$
Disgust	0.534	$p > 0.05\%$
Sadness	0.011	$p > 0.05\%$
Surprise	0.463	$p > 0.05\%$
Fear	0.534	$p > 0.05\%$
Joy	0.733*	$p < 0.05\%$

Fear, Disgust) showed relatively large, though not significant, correlation with game preferences. One explanation provided by the teachers is that some children get upset when some of their favorite games are about to end.

Figure 2.6 shows rows of 8 faces corresponding to the highest values of joy for 10 toddlers and faces corresponding to the lowest values of joy for the same toddlers. The numbers written on each side of the figure are the averages of joy value calculated from the faces in the corresponding row. The Figure shows that while, not perfect, on average the images that FACET chosen as being more joyful, do indeed look more joyful than those chosen as being less joyful.

RUBIGrams: Next we map the frequency of dyadic activities between the different children. The goal was for RUBI to automatically generate a RUBI-centric sociogram [82, 20], which here we call a RUBIGram. To this end, RUBI partitioned the time she was interacting with children into intervals defined by the start and end time of a game or a song. For each interval, RUBI identified the toddlers interacting with her during that activity. For each pair of children RUBI computed the number of activities in which the two children were seen together. A "RUBIGram" was then constructed as follows: Children that are seen together often were connected with wide lines, and children that are seldom seen together were connected with thin lines. Figure 7 shows the resulting RUBIGram. The graph shows clear patterns in the playing styles of the

Figure 2.6. Faces corresponding to largest and smallest values of the joy channel. Faces in the left side have the top 8 values of joy for the corresponding toddler. Faces in the right side have the lowest value of joy for the same subjects. The average of joy values for the faces in each row are written on the sides.

different children. Some children play with RUBI while other children are present. Other children tend to play with RUBI alone.

2.6 Discussion

We presented results of a field study in which a social robot (RUBI-5) was left alone at UCSD's Early Childhood Education Center for a period of 28 days in a classroom of 16 toddlers. During this time RUBI interacted with the children in a fully autonomous manner playing "Give and Take" games, singing, dancing, and playing educational games that have been previously shown to improve children's vocabulary skills [51]. More importantly during this time RUBI collected data from 3 cameras, two touchscreens and, proximity sensors. Off-the-shelf computer vision tools allowed RUBI to recognize who she was playing with, what games they were playing, and what facial expressions they

were making. Using this information RUBI could detect which activities the children enjoyed most, with as much accuracy as experienced teachers and ethnographic observers could do. In addition RUBI discovered some aspects of the social structure of the group of toddlers, including cliques of children that tend to play together with RUBI.

While the results are preliminary they show that low cost social robots like RUBI could be used as autonomous digital ethnographers, to run experiments, and datamine the children's social, affective, and cognitive behaviors. Based on the experience with the previous robot prototype, RUBI-4, it would have taken thousands of hours to manually code and analyze the data from the 3 video cameras. Instead RUBI-5 could automatically analyze the data in real time, at little or no cost. The results of the data analyses, e.g., sociograms, preferred games, typical facial expressions, scores on the different games, could be provided to teachers on a periodic base, to keep track of the children's social, affective and cognitive state. More importantly a network of such robots may open opportunities to run high volume, low cost experiments, to better understand, monitor, and improve the learning and development in our early childhood education centers.

2.7 Acknowledgements

The research presented here was funded by NSF IIS 0968573 SoCS, IIS INT2-Large 0808767, and NSF SBE-0542013.

Chapter 2, in full, is a reprint of the material as it appears in International Conference on Computer Vision Workshops (ICCVW) 2013. Malmir, Mohsen; Forster, Deborah; Youngstrom, Kendall; Morrison, Lydia; Movellan, J avierR., copyright IEEE, 2013. The dissertation author was the primary investigator and author of this paper.

Chapter 3

RUBI as a Tool to Monitor Socio Emotional Development in Early Childhood Education

3.1 Abstract

Sociable robots are benefiting from machine perception systems that automatically recognize social behavior (e.g., detect and recognize people, recognize their facial expressions and gestures). These systems can be used to support sophisticated forms of human-robot interaction. In addition the data provided by the perceptual systems can be data-mined to discover the socio-emotional structure of the environments where the robot operates. In this paper we analyze the data collected by a social robot, named RUBI-5, during a field study at an Early Childhood Education Center in which the robot autonomously interacted with 16 toddlers for a period of 28 days. RUBI-5 was equipped with face detection, person identification and automatic recognition of facial expressions of emotion. The data automatically collected by RUBI during the 28-day period revealed the children's preferences for different activities as well as each toddler's preferences to play with or to avoid playing with other specific children. The study illustrates that social robots may become a useful tool in early childhood education to discover socio-emotional patterns over time and to monitor their development. The data provided by the robots

could be used by educators and clinicians to discover problems and provide appropriate interventions.

3.2 Introduction

Previous research shows that relatively simple sociable robots can generate rich forms of socio-emotional interaction with toddlers that are sustained for months [79]. In addition randomized pretest/posttest studies have shown that interaction with these robots can result in measurable gain in vocabulary skills [51]. Recent advances in machine perception are making possible the automatic recognition of emotion-relevant behavior in real time (e.g., detect and recognize faces and facial expressions of emotion). These new systems can be used to support sophisticated forms of HRI. In addition the sensory data used by the robot can be stored and data-mined. In this paper we analyze the data collected by a social robot, named RUBI-5, during a 28 day long field study at the UCSD Early Childhood Education Center. RUBI-5 was equipped with 3 cameras connected to computer vision systems to detect people, recognize them and analyze their facial expressions. The results of the analysis show that the data collected by social robots can indeed be very useful to discover socio-emotional patterns and to monitor their development over time.

The study is part of the RUBI project, which started in 2004 with the goal of studying the potential value of social robot technologies in early childhood education environments [19, 55, 54]. Figure 3.1 shows the different robot prototypes used in the project, starting with QRIO and ending with RUBI-5. The diagram organizes the different prototypes by level of mechanical complexity, degree of robot autonomy, and quality of the observed human-robot interactions. The latest prototype so far is RUBI-5, the prototype we used in the field study described here. In the study presented here RUBI-5 functioned autonomously for 28 consecutive days with 16 toddlers in real life conditions.

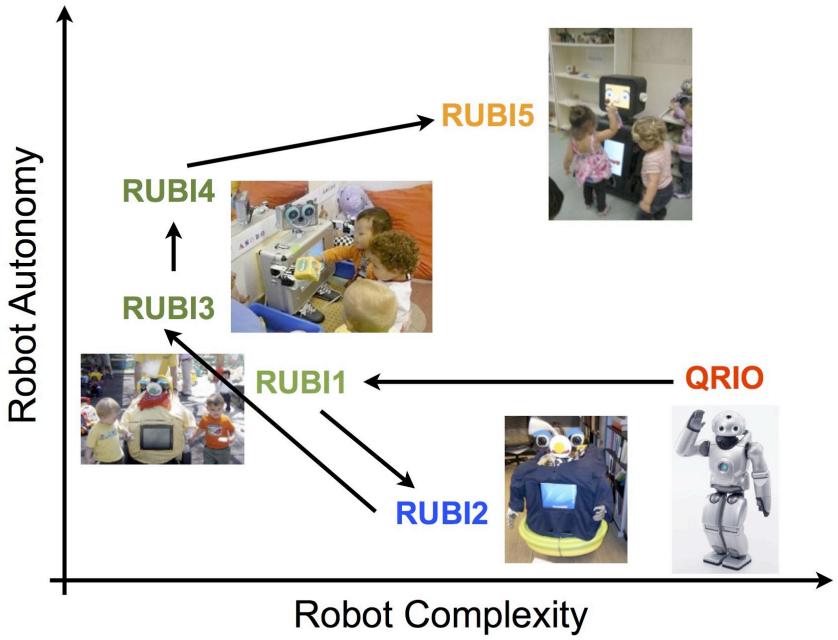


Figure 3.1. Prototypes used in the RUBI project organized in terms of their complexity (Y axis), degree of autonomy (Y axis) and quality of the HRI (red for high, blue for low).

By the time she broke our previous record, RUBI-5 was in full need of repair: a physically broken arm, and several burnt servomotors. During the 28 days of operation RUBI-5 collected a wealth of sensory data. Previously, we showed how RUBI-5 predict kids preference over different activities using facial expression recognition [43]. Here, we delve into more details about socio-emotional analysis of the environment in which RUBI-5 operated.

Robots have previously been used in classrooms for educational purposes. In a 2 week field study, Movellan et. al. [51] used a robot to teach kids English and Finnish vocabulary. It was shown that the kids that had the most persistent interaction time with the robot learned most. In another study, Kanda et. al. [31] used a humanoid robot to interact with elementary school students and teach them English. They proposed if the students have some background, education with robot might be more fruitful. Robots has also been used in schools to monitor the social structure of the environment. Also, Kanda

used Robovie to monitor the social structure of an elementary school and discover the pattern of friendship between students [32]. Tanaka and Movellan [80] analyzed behavior of toddlers interacting with QRIO robot and found 6 different categories of touch related behavior.

Not surprisingly, emotion plays a critical role in the interaction between toddlers and robots. Having something akin to emotional states that the children could understand was critical for surviving the rigors of interacting with toddlers. From the early versions of RUBI [19, 55], we also pioneered the development and testing of expression recognition technology in daily life environments, including smile detection [54, 84] and the ability to analyze and detect infants crying from sound [68, 69]. This pioneering work was influential on the development of the sophisticated facial expression recognition software, FACET 1.1 that we used in RUBI-5.

The paper is organized as follows. In Section 2, we briefly describe the RUBI-5 architecture, including face recognition and facial expression recognition. In Section 3, we describe the field study that is the focus of this document. Section 4 describes the main results of the study and is followed by a discussion section.

3.3 The RUBI-5 Prototype

3.3.1 Hardware

RUBI-5, the latest prototype of the RUBI series is shown in figure 3.2. This was the first prototype developed using modern digital fabrication methods [30]. Each of Rubi's arms has 4 DOF, independently controlled using Robotis' Dynamixel servos EX106+, RX64 and RX28. Each hand has an IR sensor inside the gripper that is used as object proximity sensor. The head has 3 DOF, a webcam for image capture and an iPad2 for the animated face. RUBI's "belly" has a touchscreen tablet PC, which is used to

Figure 3.2. RUBI-5.

display educational games and popular songs. A MacMini server with 2 GHz Intel Core i7 processor and 8 GB of RAM runs the Robot Operating System (ROS), the machine perception engines (face detection, person recognition, and expression recognition), activity scheduling, and motor control algorithms.

3.3.2 Software

ROS: RUBI-5's software architecture is based on the Robot Operating System (ROS). The entire system is distributed and works by passing ROS messages between ROS nodes that provide a variety of services. A node called RUBIScheduler is a finite state machine that schedules the activity to play with the children. This is based on the previous activities, the amount of time since the children touched RUBI's belly and the constraints of the ECEC classroom's daily schedule.

Games: RUBI performs four types of activities: (1) Sings songs ("Wheels on the Bus", and "Monkeys on the Bed") while playing animations on her belly's tablet, and dancing. (2) Educational Games targeting vocabulary development. For example, in

one game 4 images are presented on the screen and RUBI asked to touch one of them (e.g., where is the orange?). These games combine sounds and visuals presented on RUBI's belly, as well as physical actions, like clapping, looking towards the screen when the child touches it, and smiling. (3) "Give and Take" games: children give objects to RUBI. She takes the object, looks at it and gives it back to the child saying "Thank You". (4) Idle. If RUBI's belly is not touched for a period of 10 seconds the RUBIScheduler puts her in "Idle Mode", in which she makes randomly scheduled idle movements, and displays simple visuals on her belly. When children touch RUBI's belly the scheduler chooses a new activity, provided it is consistent with the classroom's daily schedule.

Person Recognition: RUBI-5 captures the ongoing scene using three cameras, located in the head, right and center of the belly's tablet and under the belly. Currently these cameras are used to identify who is playing with RUBI and what facial expressions they are making. We use the following face recognition pipeline. Each image is fed to OpenCV face detector to find the faces in the image. The detected faces are normalized to the same size and converted to 5 layer Gaussian image pyramids with a between layer downscale of 1.2. Daisy features [81] are extracted from overlapping image patches from each layer of this pyramid. PCA is then used to reduce the dimensionality of the features. A Multinomial Logistic Regression classifier is used to recognize the different participants (see Figure 3.3).

The classifier was trained using 7000 images of 28 subject collected by RUBI. Of these 28 subjects, 16 were toddlers and 12 were adults, including classroom teachers and researchers accompanying RUBI. We divided the entire dataset into two non-overlapping sets: train and test. The test set size was 35% of the entire dataset. After training using the training set we tested the system on the test set using the following procedure: For each image on the test set the system had to choose amongst 28 possible alternatives (16 toddlers, plus 12 adults). The system guessed the correct alternative with 93% accuracy.

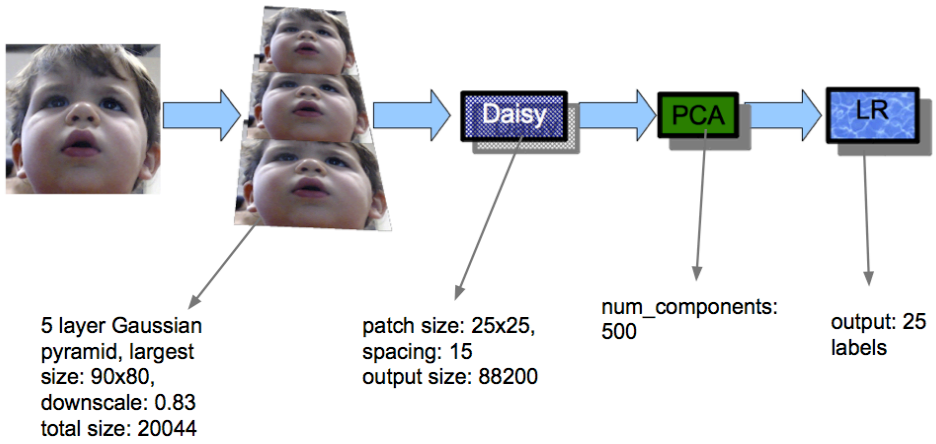


Figure 3.3. Face recognition pipeline in RUBI-5

Figure 3.4 shows the confusion matrix for our dataset.

Facial Expression Recognition: Facial expression recognition was performed using the FACET R1.1 SDK from Emotient.com. FACET is the commercial version of CERT [41], one of the most popular and accurate facial expression recognition systems. FACET R1.1 recognizes 6 primary expressions of emotion: anger, disgust, fear, joy, sadness and surprise. Since FACET R1.1 was trained to recognize adult facial expressions from faces that deviate no more than 15 degrees from frontal, it was not clear whether the system would prove useful to recognize toddler facial expressions.

3.4 Study Design

Participants: 16 toddlers (ages 11 to 23 months) from room 1B of UCSD's Early Childhood Education Center (ECEC) enrolled during the period of Jan 24 to September 11, 2013. The total number of children at any given time ranged from 9-12. Two teachers informally observed the interaction between the children and RUBI. A research assistant under the supervision of an ethnographer took notes to characterize the observed interactions between children and robot using standard ethnographic methods.

Procedure: RUBI was left alone in Room 1B of ECEC, starting on Jan 24, for

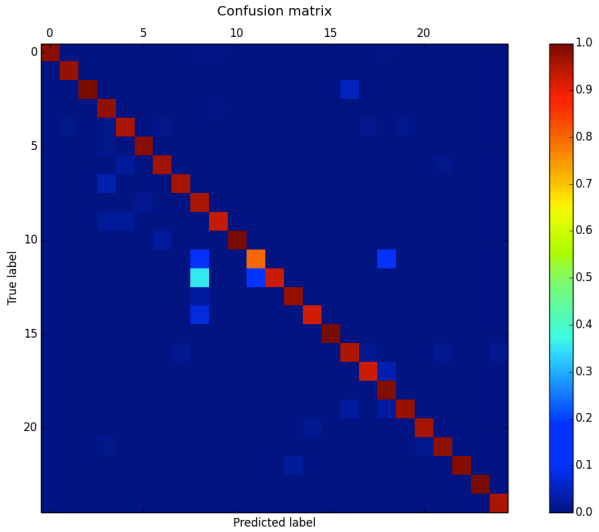


Figure 3.4. Confusion matrix for face recognition on ECEC faces dataset.

increasingly longer periods of field-testing. On Aug 12 we brought RUBI to ECEC with the intention of continuing the study until she stopped operating. This happened on September 11, 2013. During this period RUBI was relatively stationary, making only small rotational movements with the drivetrain, thus allowing to obtain power using a standard electrical outlet. RUBI-5 ran on two types of schedules: continuously while research staff were on location; and an automated schedule designed to coincide with activity periods chosen by the educational staff as curriculum appropriate. During every session, RUBI-5 was on idle state until someone touched her belly. At this time, she chose either a game or a song. The songs always end after a specific pre-determined time, while the games continue until no one touches the belly for 10 seconds. After game or song has finished, RUBI goes back to the idle state, showing the idle game on the belly and looking around while moving slightly her arms and head. This cycle continues until the session finishes.

Data: During each session, RUBI kept the log of the games and songs that were played. She also recorded images from the three RUBI cameras. The head and belly-mounted cameras captured an image when they detect a face and a game is being played.

The tablet camera captured images every 2 seconds during the game episodes. These pictures were then processed to extract the identity using the face recognition described in the previous section. Facial expressions were also extracted using the FACET SDK.

3.5 Results

3.5.1 Predicting Activity Preferences

We asked the 2 classroom teachers and a research assistant that observed the children on a daily basis to rank how much the children liked the 10 different activities they played with RUBI: 7 educational games, 2 songs, 1 give and take game. The average Pearson correlation between the three human judges was 0.68. Then we computed the correlation between the output of the different emotion channels (obtained using FACET) and the activity rankings averaged across the 3 human judges. The independent variable was the total number of images greater than 0.95 on the corresponding emotion channel (e.g. FACET was at least 95% sure about the target emotion). Amongst all the facial expression channels we found one large and statistically significant correlation (r=0.73, p¡ 0.05, 2-tails): the Joy channel. The average agreement between the Joy channel and each of the human judges was 0.73, which was very close to the average agreement between the 3 judges (0.68). Thus the facial expressions of Joy, automatically detected by the robot, provide reasonable estimates of how much the children like the different activities. Figures 3.5 shows rows of 8 faces corresponding to the highest values of joy for 3 toddlers and faces corresponding to the lowest values of joy for the same toddlers. The Figure shows that, while not perfect, on average the images that FACET chosen as being more joyful, do indeed look more joyful than those chosen as being less joyful.

We then retrieved the top pictures that were used in predicting activity preference (Figure 3.6). We found that some of these faces were actually from some of the adults in

Figure 3.5. Examples of toddlers faces with maximum and minimum Joy value. The left 8 columns are the faces with top Joy value, while the right 8 columns are the faces with least Joy value. The average of Joy channel for corresponding set of faces is written next to them.

Figure 3.6. Faces with top Joy values. Each row indicates one class according to the face recognition system. Some of the misclassified samples are adults (usually parents of toddlers) that were not in our face training dataset.

the classroom. Basically RUBI detected the general mood of the classroom, including the response of adults, while RUBI was engaging the children on different activities.

3.5.2 Detailed Temporal Analysis

During the 28 days of field study RUBI played the same activities multiple times. We synchronized the outputs of the Joy detector channel for each activity and averaged it across the 3 cameras and the different times the activity was played. In all the activities except for one, the result was that Joy was approximately constant across the activity. However for the “Wheels on the Bus” song the function had clear peaks and valleys (See Figure 3.7). The local peak in the Joy channel appeared at the beginning of the song, indicating that they were happy RUBI was playing this song, and at the points in the songs where RUBI said “all to the town”.

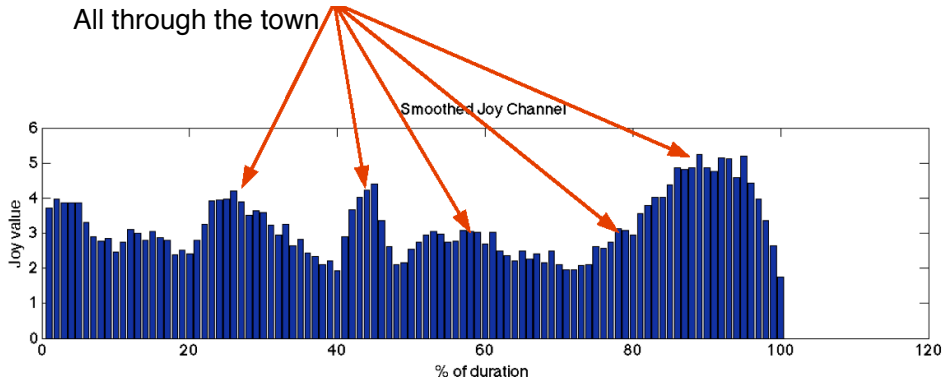


Figure 3.7. Average of Joy channel for different trials of "The Wheels on the Bus" song. A local peak is observed at the start of the verse "all to the town".

3.5.3 RUBIGrams

We also investigated whether the data collected by RUBI could reveal some aspects of the social structure in the classroom. To this end we collected the frequencies with which RUBI detected two children together during each specific game and song trial. The results are presented in Figure 3.8, using a Sociogram-like display, which we called RUBIGram: The width of the lines in figure 3.8 represents the relative amount of time each pair was seen together. The graph shows that some children play much more with RUBI than others, and that some pairs of children are seen together much more than others. However two children may be seen together often for two reasons: (1) they may like playing together with RUBI. (2) They may be playing independently and, just by chance, those children that play more with RUBI are more likely to be seen together. We then compensated for the effects of chance as follows: for each edge between x,y, denote the strength of edge with $P(x,y)$. This is the amount of time x,y were seen together. Denote by $P(x)$ the total amount of time x spent with RUBI. Then we are interested in the quantity $P(x,y) - P(x)P(y)$, which is 0 if the times x,y spent with RUBI are independent of each other. Figure 3.9 shows the edges corresponding to $P(x,y) - P(x)P(y)$. Positive values are shown by red, while negative values are in blue. Thus positive values indicate

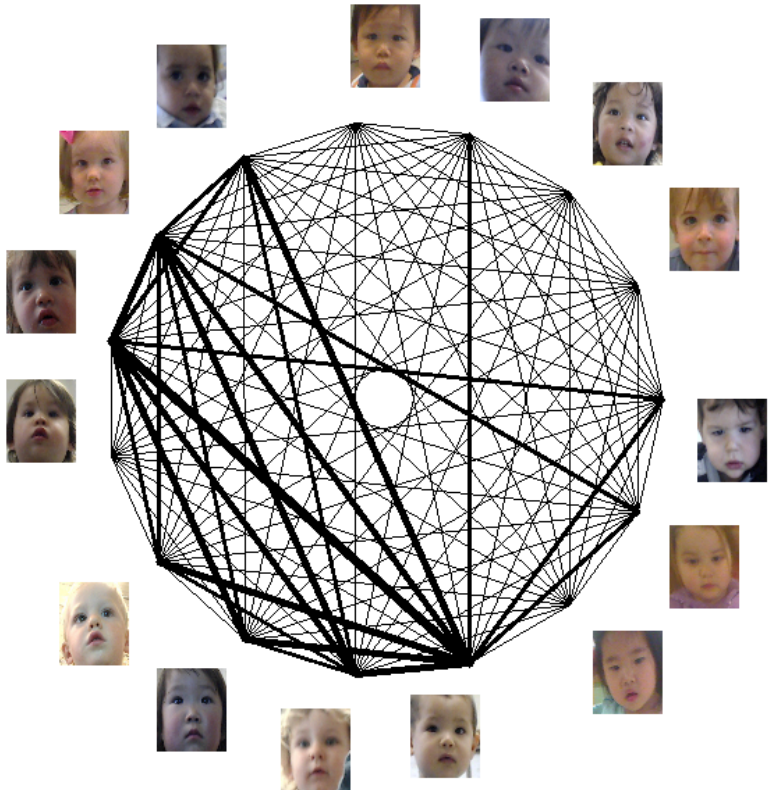


Figure 3.8. RUBIGram. Each link between two toddlers indicate the amount of time they spent together playing with RUBI. The width of the line represents the time.

that two children are seen together more than it is expected from chance. Blue lines indicate that two children are seen together less than would be expected from chance (i.e., they avoid each other).

3.6 Discussion

Advances in machine perception technologies are providing social robots with perceptual primitives that can support sophisticated forms of HRI. Because of the active, real time experience that sociable robots can provide, they are ideal tools to harvest and datamine behavioral data from daily-life environments. Here we showed some analysis of data harvested by a social robot, RUBI-5, that interacted with 16 toddlers for a period of 28 days. In particular we focused on the analysis of the facial expressions the

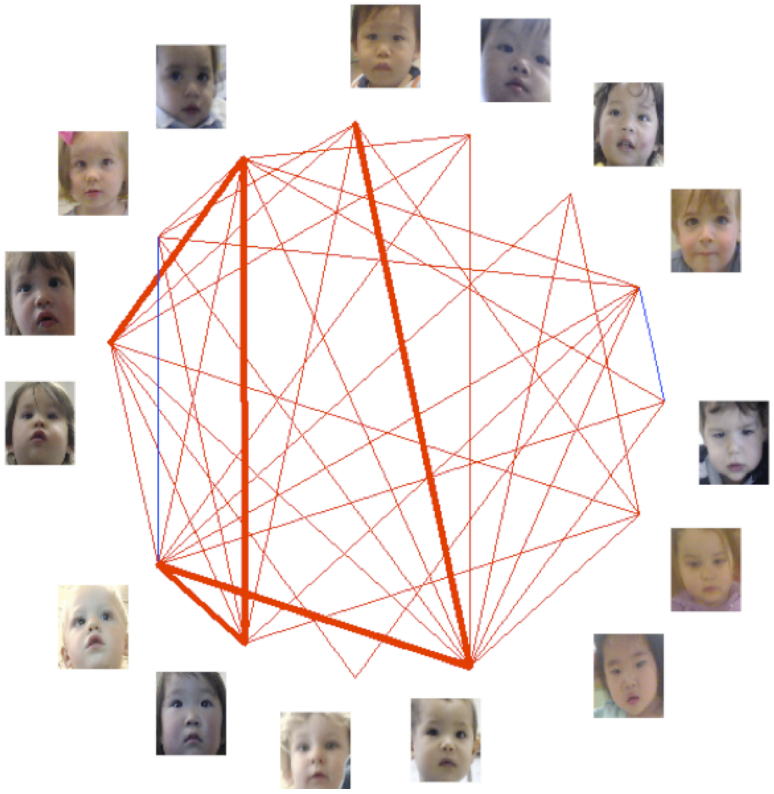


Figure 3.9. Chance corrected RUBIGram. Red edges indicate pairwise associations larger than expected from pure chance, while blue edges indicate avoidance.

children made while engaging on different activities with the robot, and on the analysis of which toddlers the robot saw playing together. The analysis revealed that automatic expression recognition (joy detection) was a very effective metric for detecting activity preferences. Using expression recognition RUBI achieved a 0.73 average correlation with the preference rankings provided by human observers. This is slightly larger than the human inter-observer correlation (0.68) for preference rankings. RUBI could also provide precise temporal information about which parts of an activity the children liked most. In addition RUBI discovered which children preferred to play alone, play with other specific children, or avoided specific children. The study illustrates that social robots could become a useful tool in early childhood education to discover socio-emotional patterns over time and to monitor their development. The data harvested by these robots could be mined to develop norms for typical socio-emotional development and to help on early detection of developmental disorders.

3.7 Acknowledgements

The research presented here was funded by NSF IIS 0968573 SoCS, IIS INT2-Large 0808767, and NSF SBE-0542013.

Chapter 3, in full, is a reprint of the material as it appears in International Conference on Human Robot Interaction Workshops (HRIW) 2014. Movellan, J avierR.; Malmir, Mohsen; Forster, Deborah. copyright IEEE, 2014. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Deep Q-learning for Active Recognition of GERMS: Baseline performance on a standardized dataset for active learning

4.1 Abstract

In this paper, we introduce GERMS, a dataset designed to accelerate progress on active object recognition in the context of human robot interaction. GERMS consists of a collection of videos taken from the point of view of a humanoid robot that receives objects from humans and actively examines them. GERMS provides methods to simulate, evaluate, and compare active object recognition approaches that close the loop between perception and action without the need to operate physical robots. We present a benchmark system for active object recognition based on deep Q-learning (DQL). The system learns to actively examine objects by minimizing overall classification error using standard back-propagation and Q-learning. DQL learns an efficient policy that achieves high levels of accuracy with short observation periods.

4.2 Introduction

Active object recognition (AOR) refers to problems in which an agent interacts with the world (e.g., via commands to servo motors of a robotic arm) and controls its sensor parameters (e.g., camera orientation, gain, sensitivity) to maximize the speed and accuracy with which it recognizes objects. A wide range of approaches have been developed since Wilkes and Tsotsos' [85] pioneering work. These approaches are designed to re-position sensors or change the environment so that the new inputs to the system become less ambiguous [1, 5] with respect to goals such as 3D reconstruction, localization or recognition of objects (see [66] for a comprehensive literature review).

Active object recognition systems include two modules: A recognition module and a control module. Given a sequence of images, the recognition module produces a belief state about the objects that generated those images. Given this belief state, the control module produces actions that will affect the images observed in the future. The controller is typically designed to improve the speed and accuracy of the recognition module. One of the earliest active systems for object recognition was developed by Wilkes and Tsotsos [85]. They used a heuristic procedure to bring the object into a "standard" view by a robotic-arm-mounted camera. In a series of experiments on 8 Origami objects, they qualitatively report promising results for achieving the standard view and retrieving the correct object labels. Seibert and Waxman explicitly model the views of an object by clustering the images acquired from the view-sphere of the object [73]. The correlation matrices between these are then used to predict the correct object label. Using three model aircraft objects, they show that the belief over the correct object improves with the number of observed transitions compared to randomly generated paths on the view sphere of these objects.

Since these pioneering efforts, more theoretically-motivated approaches have

attempted to optimize an objective function, for example the conditional entropy $H(O|M)$ between the original object O and the observed signal M, the expected entropy loss over actions, the belief uncertainty, or simply the variance among the object representations[8, 11, 14, 71]. Paletta & Pinz's work [59] is probably the most similar to our proposed model, as they treat active object recognition as a reinforcement learning problem, using Q-learning to find the optimal policy. They used an RBF neural network with the reward function depending on the amount of entropy loss between the current and the next state. Finally, the architecture for our work was inspired in part by recent work using a DCNN for representation of images in the context of learning to play Atari games with reinforcement learning[49].

A common thread in many of these approaches is the use of small, sometimes custom-designed sets of objects. One exception by Schiele and Crowley[71] used the COIL-100 dataset for their experiments, which consists of 7200 images of 100 toy objects rotated in depth [58]. This dataset is appealing for active object recognition because it provides systematically defined views of objects. However it is not an adequately challenging dataset for several reasons, including the simplicity of the image background, and the high similarity of different views of the objects due to single-track recording sessions. Indeed, by selecting the two most discriminative views of each object, Schiele and Crowley achieved almost perfect recognition accuracy.

This paper makes two main contributions: First, we present and make publicly available the GERMS dataset [1], that was specifically developed for active object recognition. The goal of this dataset is to accelerate progress on active object recognition by providing a common framework for evaluation of active object recognition algorithms. The dataset can be used to simulate the effect of robot actions without the need to have access to the physical robot. Second, we propose an architecture (DQL) for AOR based

[1] Available at http://rubi.ucsd.edu/GERMS/

on deep Q-learning. Deep Q-learning supports learning the optimal policy for action selection from raw images [49]. The proposed model sets a performance baseline on the GERMS dataset that can be improved upon by other research groups. Although there is existing work that utilizes deep convolutional neural networks (DCNN) for mapping raw image sensory to actuator values [38], to our knowledge, this is the first work employing deep Q-learning for active object recognition. In the following sections, we introduce the GERMS dataset and describe its composition, data collection procedure and proposed benchmarks. Then the DQL system is described in detail, and our baseline performance benchmarks are reported.

4.3 The GERMS Dataset

Many of the active object recognition methods are built around a specific hardware system, which makes the replication of their results very difficult. Other systems use off-the-shelf computer vision datasets, which include several views of objects captured by systematically changing object's orientation in the image. However, these datasets do not offer any active object recognition benchmark *per se*. Adapting such datasets for active object recognition ignores the challenges in the active component such as noisy actions. A well-defined benchmark should consist of an established list of training and testing images, along with the baseline results for object recognition. The dataset should also offer methods to simulate active observation with actual robotic sensory systems. In this section, we introduce the GERMS dataset, which aims to accelerate progress on active object recognition by addressing some of the shortcomings of the previous datasets.

4.3.1 Data Collection Context

The data collection procedure was motivated by the needs of the RUBI project, whose goal is to develop robots that interact with toddlers in early childhood education

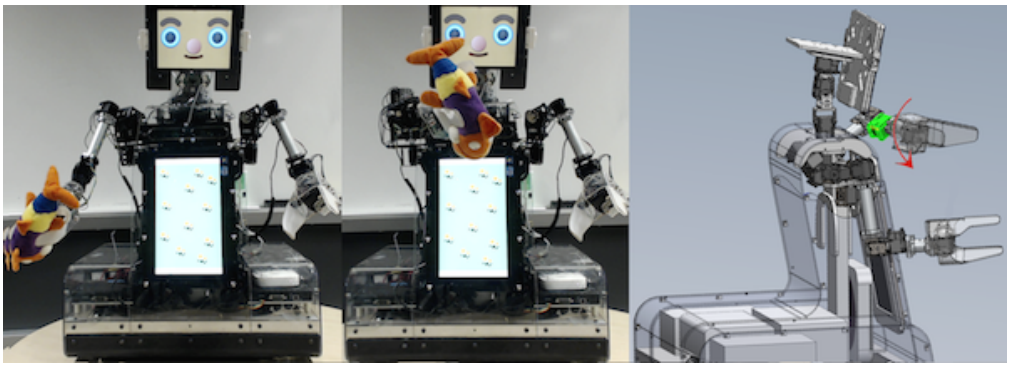


Figure 4.1. Left: RUBI holding an object. Middle: RUBI brings the object to its center of view and rotates it 180 degrees using its wrist. Right: The red arc shows the direction of rotation of the wrist. The wrist joint is shown in green.

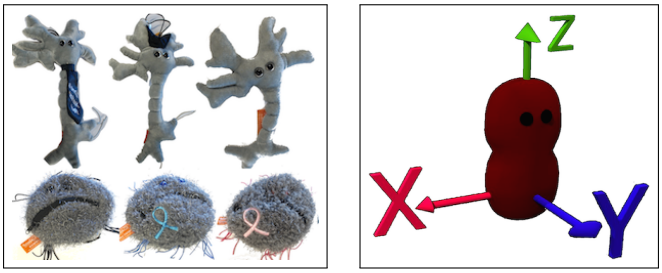
environments [51, 53, 43, 30, 19]. As part of this project, we found that one of the activities toddlers liked most was give-and-take games with the robot (RUBI). In these games the toddlers hand objects to RUBI, who then pretends to examine them and gives them back to the toddler. We found that teachers use this type of give-and-take activity as an opportunity to name the objects and teach vocabulary skills. Thus we aim for RUBI to be able to recognize and name the objects given to it. In order to minimize the effect of prior knowledge of the objects that different toddlers have, we choose a large collection of soft toys, whose names the toddlers were unlikely to know.

4.3.2 Dataset Details

The GERMS dataset consists of 1365 video recordings of give-and-take trials using 136 different objects. The objects are soft toys depicting various human cell types, microbes and disease-related organisms. Figure 4.2 shows a collage of these toys. Subsets of objects in GERMS exhibit interesting visual similarities that makes it a suitable dataset for active object recognition. For example, ambiguity between objects in figure 4.3(a) can be resolved only from certain alternative views.

A trial starts when someone hands an object to RUBI (see Figure 4.1). RUBI

Figure 4.2. Object set used in GERMS dataset. The objects represent human cell types, microbes and disease-related organisms.



(a) (b)

Figure 4.3. (a) Similar objects. Top: 3 brain cells. Bottom: 3 cancer cells. Toys in each row are not distinguishable from the rear view. (b) Coordinate system on *sore throat* (see text).

brings the grasped object to its center of view, rotates it by 180 degrees and then returns it. During each trial, RUBI continuously records images from its head-mounted camera at 30 frames per second. For each image, it also reads the positions of its joints. These data are then stored in a *track*, a collection of which constitutes the dataset.

On average, each track contains 265 snapshots of a give-and-take trial, with each snapshot consisting of an image from the head-mounted camera, the capture time, and the joint angles at capture time. These joint angles allow researchers to simulate different active observation policies. Table 4.1 summarizes the number of images in the dataset. The details of the objects used, and train and test data collection are described in the next two subsections.

Table 4.1. GERMS dataset statistics (mean±std).

	No. of tracks	Frames/track	Frames with object/track	Mean track length (sec)
Train	816	265 ± 7	157 ± 12	8.94 %
Test	549	265 ± 7	145 ± 19	8.91 %

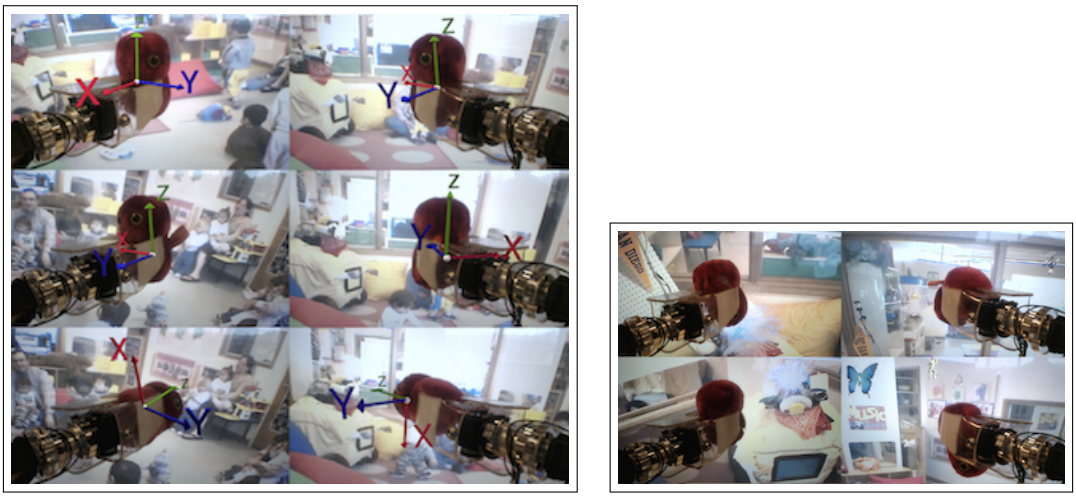
4.3.3 Data Collection

The training data consist of 6 video clips per object, for a total of 816 clips. In each clip an object was handed to RUBI in one of 6 predetermined poses. These poses are determined via an object-centered coordinate system (see Figure 4.3(b)). The Z-axis is aligned with the longest side of the object, with the positive direction determined by the orientation of the object's face. The Y-axis always comes out of the object's face plane. The 6 different poses result from the combination of 3 object orientations with respect to each of 2 grippers (left arm and right arm). Figure 4.4(a) shows the 6 different configurations for a single object. In each trial, RUBI grabs the object in one of these 6 configurations, brings the object to its center of view, and rotates the object by 180 degrees.

To collect test data, we asked a set of human subjects to hand the GERM objects to RUBI in poses they considered natural. A total of 12 subjects participated in test data collection, each subject handing between 10 and 17 objects to RUBI. They were asked to hand each object to each gripper twice, using a different pose each time. Figure 4.4(b) shows snapshots of the test data for the same object. The background of the GERMS dataset was provided by a large screen TV displaying video scenes from the classroom in which RUBI operates, including toddlers and adults moving around.

4.3.4 Annotation

The training and test data were annotated manually with the target object's



(a) (b)

Figure 4.4. (a) Six different poses of *flesh eating* object used for training data. The images are captured from RUBI's head-mounted camera, (b) Test poses for *flesh eating* object.

bounding box in frames where the object was visible. These annotations serve to limit the boundaries of objects in each image since the images are much larger than individual objects. For examples of annotations, see Figure 4.5. The annotations provide ground truth to test object segmentation algorithms and to allow testing AOR algorithms that assume ideal object segmentation.

4.3.5 Actuator Data

Accompanying each image in give-and-take trials are recordings of the joints of the robot. These joints include 2-DOF head, and two 7-DOF arms. The servos are MX-28T and MX106T type Dynamixels. Each servo is equipped with a contact-less absolute encoder with 12-bit resolution of 360 degrees. During a give-and-take trial, after each image is captured, its corresponding servo positions are also recorded simultaneously.

4.4 Active Object Recognition Using Deep Q-learning

As in previous work by Paletta & Pinz [59], we treat active object recognition

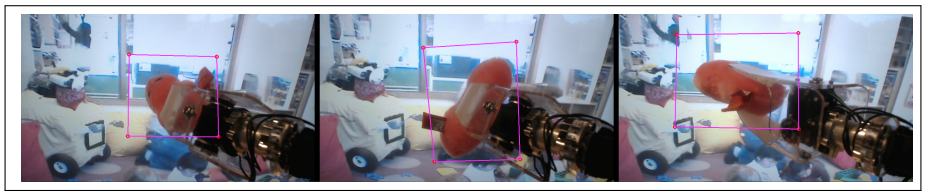


Figure 4.5. Bounding box annotation examples.

as a reinforcement learning problem, using Q-learning to find the optimal policy. We use a DCNN for object representation and policy learning and call it deep Q-learning (DQL). Our model takes a minimalist approach to encoding the state: we extract a belief vector over different object labels and use that as input to the policy learning DCNN. Our method is different from the work by Mnih et al. [49] in an important way. We use objects beliefs as representation of the current states to learn actions, while in [49] actions are learned from the raw image. Here we hypothesize that the next action to disambiguate the current view can be inferred only from the belief over different objects. Another difference between our problem and that of [49] is that object recognition at test-time may not be episodic, that is, there may be no way of knowing when the object inspection is finished. This is a difficult problem to solve automatically, for this work we use a fixed length threshold to finish the object inspection trials.

4.4.1 Deep Q-learning

The model architecture is shown in figure 4.6. An image is first transformed into a set of features using a DCNN borrowed from [16] which was trained on ImageNet. We add a softmax layer on top of this model to recognize GERMS objects; the output of this softmax layer is the belief over different GERMS objects given an image. This belief is combined with the accumulated belief from the previous images using Naive Bayes. This accumulated belief represents the *state* of the AOR system in each time step. Let I_i be the input image to the system at the ith time step, the accumulated belief over objects

given images from time steps $1,\ldots,n$ is given by,

$$P(O|I_1,\ldots,I_n) \propto \prod_{i=1}^{n} P(O|I_i) \tag{4.1}$$

where $P(O|I_i)$ is the posterior belief over object label O computed by the first softmax layer in figure 4.6. The accumulated belief is then transformed by the policy learning network into action values. This network is composed of two Rectified-Linear-Unit (ReLU) layers followed by a Linear-Unit (LU) layer. Each unit in the LU represents the action value for a given accumulated belief and one of the possible actions. We will discuss possible actions in the next subsection. In order to train this module, we employ the Q-learning iterative update:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\{R(s,a) + \gamma \max_{a^*} Q(s^*,a^*) - Q(s,a)\}. \tag{4.2}$$

In the above equation, $Q(s,a)$ is the action value for action a in state s, α is the learning rate and $0 < \gamma < 1$ is the reward discount factor. The Q-learning iterative update is turned into the following stochastic gradient descent weight update rule for the network:

$$W \leftarrow W - \lambda \left(R_t + \gamma \max_{a} Q(B_{t+1},a) - Q(B_t,a_t)\right) \frac{\partial}{\partial W} Q(B_t,a_t). \tag{4.3}$$

Here, W is the set of weights of the policy learning network, $Q(s,a)$ is the action-value learned by the network for action a in state s, γ is the reward-discount factor and R_t is the reward value at time step t. Also, λ is the learning rate for the neural network and B_t is the vector of beliefs over object labels, which is used here to represent the state of the system.

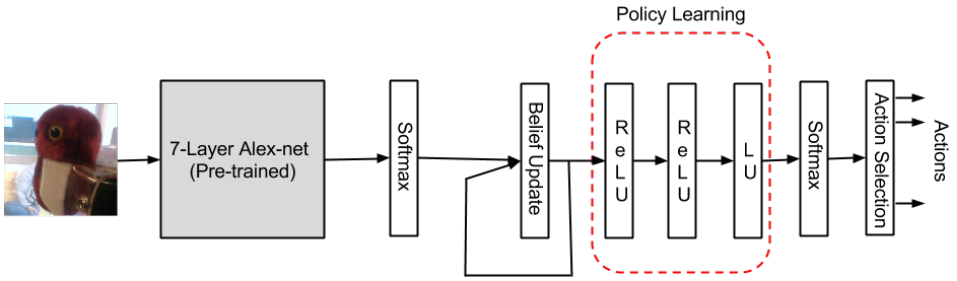


Figure 4.6. The proposed architecture for DQL. Images are first transformed into beliefs over object labels using a pre-trained Alex-net (gray block). Each block except for 7-layer Alex-net represents one layer in the network.

4.4.2 Policy Learning

The number of output units in the policy learning network is equal to the number of possible actions. Each output unit calculates the action value $Q(s,a)$ for one action a. We implemented a set of actions which rotate the robot's wrist from its current position by an offset angle. We chose the rotation offsets to be $\pm\pi/64$, $\pm\pi/32$, $\pm\pi/16$, $\pm\pi/8$, $\pm\pi/4$, for a total of ten actions. The allowable range of rotation for both robot wrists is in $[0,\pi]$. We choose these actions because they allow for both fine-grained inspection of the object at hand and large rotations to traverse the entire range of wrist rotation in few actions.

The training procedure for DQL is shown in Algorithm 1. In this algorithm, *AlexNet-Softmax* converts a single image into a belief vector over objects, and *Action-Value* converts the accumulated belief into action-values using the policy network. A *game play* is defined as the sequence of *moves* (wrist rotations) selected by the policy learning network to examine the current object at hand. Each move results in rotation of the object and thus a new belief vector over object labels. The mapping of the image to action values is shown in lines 8-11 of Algorithm 1. For the selected action then we update the parameters of the network according to Equation 4.3. For each update, we calculate the target value of the selected action by grabbing the next image based on

the current pose of the robots wrist and the selected action. The action value for this *look-ahead* image is used as the target value for the current image (lines 12-15).

In Algorithm 1 we show the training for a single image at a time. In practice, the training algorithm uses mini-batches that contain images from different give-and-take trials. In each training iteration, the procedure keeps track of different game plays and updates the neural network parameters using the sum of their gradients. A game play finishes after n moves, after which the training proceeds to the next mini-batch.

We use the same set of learning parameters to train different models for the left and right arm. The training is done using stochastic gradient descent with mini-batches of size 128. The learning rate starts at 0.01 and is multiplied by 0.1 every 1000 iterations. The training procedure is on-policy, with probabilistic action selection. The training runs for 5 epochs over the training data (3500 iterations on mini-batches), where for each mini-batch a game play of length 5 is followed to update the weights. We found no significant difference in the accuracy of models trained with longer game plays (10 and 20). After each move, a reward R_c of ± 10 was given to the network depending on whether the maximum probability label in the accumulated belief vector is equal to the target label for that give-and-take trial or not.

4.5 Baseline Results

The GERMS dataset contains two benchmark tasks, one for each arm. We report the accuracy of label prediction on test set trials. On each trial a new test object is selected for recognition. The AOR algorithm chooses a sequence of servo configurations producing a sequence of views of the test object. We report the accuracy of predicting the correct label as a function of the number of actions

We report the accuracy of the DQL active object recognition system as the baseline for GERMS. An instance of the network in Figure 4.6 is trained using game

Algorithm 1. Training Deep Q-learning network

procedure TRAIN
  $t \leftarrow 1$
  while not converged **do**
    $I_t \leftarrow$ Next-Training-Image$(t-1)$
    $I_c \leftarrow I_t$
    $B_c \leftarrow$ vector of ones of length C
    for move=1 To LengthofGamePlay **do**
      $A \leftarrow$ AlexNet-Softmax(I_c)
      $B_c \leftarrow$ Normalize(elementwise-product(A, B_c))
      $Q(B_c, a) \leftarrow$ Action-Value(B_c)
      $a_c \leftarrow$ Action-Selection$(Q(B_c, a))$
      $I_c^* \leftarrow$ Next-Image(I_c, a_c)
      $A^* \leftarrow$ AlexNet-Softmax(I_c^*)
      $B_c^* \leftarrow$ Normalize(elementwise-product(A^*, B_c))
      $Q(B_c^*, a) \leftarrow$ Action-Value(B_c^*)
      $W \leftarrow W - \lambda \left(R_c + \gamma \max_a Q(B_c^*, a - Q(B_c, a_c))\right) \frac{\partial}{\partial W} Q(B_c, a_c)$
      $I_c \leftarrow$ Next-Image(I_c, a_c)
    end for
    $t \leftarrow t+1$
  end while
end procedure

plays of length 5 using images from the training set. After the model is trained, we measure the performance on the test set as a function of the number of actions. For each action, a new accumulated belief vector is calculated and used to measure the accuracy of the model. The benchmarks are shown in figure 4.7.

We compared the performance of our model against two alternative policies: sequential and random. The random policy selects a random action with uniform probability, while the sequential strategy always starts from the same position and moves in the same direction to the next immediate position. Figure 4.7 compares the accuracy of predicting the correct object label as a function of the number of observed images. This performance is averaged over the entire set of images in the test set, that is if the system starts from any image in the test set and selects the next action according to the corresponding policy. We report the average performance of 20 models trained in

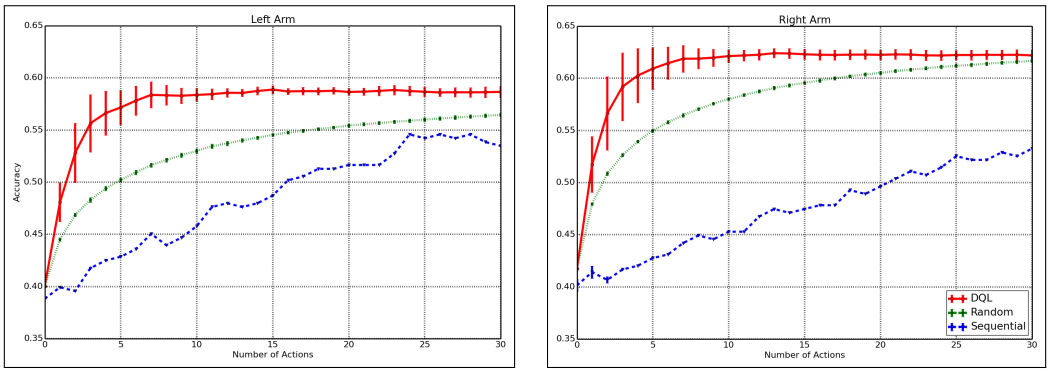


Figure 4.7. Performance comparison between DQL, sequential and random policies on test images for RUBI's (a) left and (b) right arm.

Figure 4.8. From left to right: the sequence of actions chosen by DQL on "Cancer".

separate runs of Algorithm 1.

Table 4.2 shows the required number of step to reach the same level of prediction accuracy over 136 different target classes of GERMS by the sequential, random and DQL strategies. For the samples collected with the left arm, DQL strategy achieves 55% accuracy in only 3 steps which is significantly better than 18 steps by the random and 37 steps by the sequential method. For the same arm, DQL achieves its peak performance (58%) in 7 steps while the other two methods can't achieve this in a maximum of 30 steps. For the right arm, DQL reaches 58% accuracy in 3 steps, compared to 10 steps required by the random strategy. Sequential method can't reach this level in 30 steps. DQL achieves its peak performance of 62% accuracy in 10 steps, while none of the other methods can reach the same accuracy within 30 steps.

4.6 Discussion

Active object recognition has the potential to overcome many of the difficulties

Table 4.2. Number of steps required by the sequential, random and DQL policies to reach the same level of label prediction accuracy on GERMS dataset.

	48%	53%	55%	58%	62%	
Sequential	18	30	-	-	-	
Random	2	4	6	10	-	**Right Arm**
DQL	**1**	**2**	**2**	**3**	**10**	
Sequential	15	24	-	-	-	
Random	3	10	18	-	-	**Left Arm**
DQL	**1**	**3**	**3**	**7**	-	

encountered in classical vision problems of passive object recognition from static images. While the literature on active object recognition has shown promising results, progress has been slow due to the lack of realistic datasets and benchmarks that can be easily shared by multiple research groups. In this paper, we introduced the GERMS dataset that includes a collection of videos, a set of active object recognition benchmarks and baseline results on those benchmarks. We hope that this dataset will facilitate the comparison of different active object recognition methods and accelerate progress in the field.

We also proposed an architecture for active object recognition based on deep Q-learning. Instead of the standard approach of encoding the state using a vector of visual features, we used the representation produced by a commodity deep object classification network [49]. This representation was then processed by an additional deep network trained using Q-learning for efficient action selection. The proposed approach outperforms sequential and random action selection policies and serves as baseline for future comparisons. We also observed that different lengths of game plays did not have an impact on the performance of trained model. The model achieves its peak recognition accuracy in 7-10 steps, far fewer than the random and sequential strategies, and always achieves the highest accuracy among them.

The current baseline approach relied on human-annotated bounding boxes. We are

currently baselining algorithms that include automatic object segmentation. We are also evaluating potential performance gains achievable by post-training the early perceptual layers of the object recognition network using the policy learning error signals.

4.7 Acknowledgements

Chapter 4, in full, is a reprint of the material as it appears in Proceedings of the British Machine Vision Conference (BMVC) 2015. Malmir, Mohsen; Sikka, Karan; Forster, Deborah; Movellan, Javier R.; Cottrell, Garrison W. copyright BMVA Press, 2015. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Deep Active Object Recognition by Joint Label and Action Prediction

5.1 Abstract

An active object recognition system has the advantage of acting in the environment to capture images that are more suited for training and lead to better performance at test time. In this paper, we utilize deep convolutional neural networks for active object recognition by simultaneously predicting the object label and the next action to be performed on the object with the aim of improving recognition performance. We treat active object recognition as a reinforcement learning problem and derive the cost function to train the network for joint prediction of the object label and the action. A generative model of object similarities based on the Dirichlet distribution is proposed and embedded in the network for encoding the state of the system. The training is carried out by simultaneously minimizing the label and action prediction errors using gradient descent. We empirically show that the proposed network is able to predict both the object label and the actions on GERMS, a dataset for active object recognition. We compare the test label prediction accuracy of the proposed model with Dirichlet and Naive Bayes state encoding. The results of experiments suggest that the proposed model equipped with Dirichlet state encoding is superior in performance, and selects images that lead to better

training and higher accuracy of label prediction at test time.

5.2 Introduction

A robot interacting with its environment can collect large volumes of dynamic sensory input to overcome many challenges presented by static data. A robot manipulating an object with the capability to control its camera orientation, for example, is an example of an active object recognition (AOR) system. In such dynamic interactions, the robot can select the training data for its models of the environment, with the goal of maximizing the accuracy with which it perceives its surroundings. In this paper, we focus on AOR with the goal of developing a model that can be used by a robot to recognize an object held in its hand.

There are a variety of approaches to AOR, the goal of which is to re-position sensors or change the environment so that the new inputs to the system become less ambiguous for label prediction [1, 5, 18]. An issue with previous approaches to AOR is that they mostly used small simplistic datasets, which were not reflective of challenges in real-world applications [45]. To avoid this problem, we have collected a large dataset for AOR, called GERMS[1], which contains more than 120K high resolution (1920x1080) RGB images of 136 different plush toys. This paper extends our previous work, Deep Q-learning [45], where an action selection network was trained on top of a pre-trained convolutional neural network. In this paper we extend the model to train the network end-to-end using GERMS images to jointly predict object labels and action values.

This paper makes two contributions: First, we develop a deep active object recognition (DAOR) model to jointly predict the label and the best next action on an input image. A deep convolutional neural network is trained to predict the object label and action-values from an image of the object. We use reinforcement learning to teach

[1] Available at http://rubi.ucsd.edu/GERMS/

the network to predict the action values, and minimize the action value prediction error along with the label prediction cross entropy. The visual features in early stages of this network are learned to minimize both errors. The second contribution of this work is to embed a generative Dirichlet model of objects similarities for encoding the state of the system. This model integrates information from different images into a vector, based on which actions are calculated to improve object recognition. We embed this model as a layer in the network and derive the learning rule for updating the Dirichlet parameters using gradient descent. We conduct a series of experiments on the GERMS dataset to test (1) if the model can be trained jointly for label and action prediction, and (2) how effective is the proposed Dirichlet state encoding compared to more traditional Naive Bayes approach, and (3) discuss some of the properties of the learned policies.

In the next section, we review some of the previous approaches to AOR and the datasets they used. Next we introduce the GERMS dataset and describe the training and testing data used for the experiments in this paper. After that, we describe the details of the proposed network and Dirichlet state encoding, going into the details of cost function and update rules for different layers of the network. In the results section, we report the properties of the proposed network and compare its performance in different state encoding scenarios. The final section is the concluding remarks.

5.3 Literature Review

Active object recognition methods can be divided into two groups based on how they select actions to improve object recognition. The first group uses heuristic methods to select actions, for example to bring the object to a predefined *standard* view where the recognition performance is expected to be maximized. The second group of methods are motivated by information theory, using information gain to determine the effect of actions on object label prediction uncertainty. The next action is chosen to maximize the

reduction in this uncertainty.

An early heuristic AOR system was developed by Wilkes and Tsotsos [85]. They used a heuristic procedure to change the camera's position and orientation to bring the object into a 'standard' view using a robotic-arm-mounted camera. The standard view of objects was defined to be unique among all objects with respect to their low level visual features. In a series of experiments on 8 Origami objects, they qualitatively report promising results for achieving the standard view and retrieving the correct object labels. Heuristic method clearly suffer from generalization problem, as the number of objects increases it is not possible to define standard views for each object manually. A more systematic approach is needed to define the effectiveness of different object views for label prediction.

Among the information theoretic approaches to AOR, Schiele and Crowley's work was pioneering in making an analogy between object recognition and information transmission [71]. They try to minimize the conditional entropy $H(O|M)$ between the original object O and image M, which is the object's transformation through measurement. Starting from a random view of an object, their system determines the most-likely object label and moves to the view that has the lowest conditional entropy for that label among the training data. The movement is then verified by measuring the prediction discrepancy between the first and the second views. They used the COIL-100 dataset for their experiments, which consists of 7200 images of 100 toy objects rotated in depth [58]. This dataset has been appealing for active object recognition because it provides systematically defined views of objects. Schiele and Crowley achieved almost perfect recognition accuracy on this dataset using their one-step view selection procedure.

Borotschnig et al. formulate the observation planning in terms of maximization of the expected entropy loss over actions [8]. Larger entropy loss is equivalent to less ambiguity in interpreting the image. A set of distributions are learned for different views

of each object, and used to predict the entropy loss for the next view. The novelty of this work is the use of parametric distributions for object views. With an active vision system consisting of a turntable and a moving camera, they report improvements in object recognition over random selection of next views on a small set of objects.

Paletta & Pinz search for the most discriminative views of objects by maximizing the entropy loss between two consecutive views of objects [59]. The novelty of their method is the use of reinforcement learning to discover the optimal strategies to explore the objects. Action-values in this work correspond to the decrease in entropy of view sequences of objects. A variant of Q-learning is used to train a neural network to predict the action values given the current view of the image. This work is different from our work in that in our model the visual features are learned simultaneously with the optimal policy, which allows the features to be tuned for object inspection. Paletta & Pinz showed that their model is superior in recognizing COIL100 objects compared to a random exploration strategy.

Calculating the exact value for entropy loss is computationally expensive since it requires marginalization over the observation space, and one might resort to approximations or simpler criterion to measure the uncertainty in prediction. This argument motivated Browatzki et al. to maximize a measure of variance of observations across different objects [11]. They used a particle filter approach to determine the viewing pose of an object held in-hand by an iCub humanoid robot. They show that their method is superior to random action selection on small sets of custom objects.

A common trend in these approaches is the use of small, sometimes custom-designed sets of objects. There are medium sized datasets such as COIL-100, which consists of 7200 images of 100 toy objects rotated in depth [58]. We have summarized the properties of datasets used in these studies in table 5.1. In this table, meridian denotes the great circles on the surface of view sphere of objects, moving along which camera

Table 5.1. Details of different active object recognition datasets used in the literature.

Dataset	number of objects	Meridians on view sphere	occlusion	publicly available	complex back-ground
Origami [85]	8	1(single view)	No	No	No
COIL100 [58]	100	$1 \times 2\pi$	No	Yes	No
model objects [8]	15	$3 \times 2\pi$	No	No	No
office objects [11]	18	not specified	Yes	No	No
GERMS [45]	**136**	$\mathbf{10 \times \pi}$	**Yes**	**Yes**	**Yes**

captures images of objects. We also mention the angular distance that camera traverses on the great circle while capturing images, with 2π denoting a full circle. From this table it is clear that these datasets are not challenging for recognition because of small number of objects, simple background and no occlusion of the objects in images. We collected GERMS, which includes a large number of objects with complex background, occlusion and large number of viewing pose per objects to cover the shortcoming of existing AOR datasets.

Another common trend in the existing literature is the notion of a pre-defined encoding scheme for objects appearance. In these studies, visual features extracted from objects are hand-crafted and fixed during policy learning. However, a more compelling scheme would be to learn the features for object appearance encoding along with the object exploration policy. This way we allow the visual features to be fine-tuned for better object inspection. In this paper, we train a deep convolutional neural network to jointly predict label and action-values given objects images. Deep neural networks have proven to be superior in learning visual features to hand-crafted methods. We utilize a deep network to learn the appearance and object inspection policy at the same time. This reduces the training to a single stage, as opposed to the two stage process of feature encoding and policy learning in the current AOR literature.

5.4 The GERMS Dataset

The GERMS dataset was collected in the context of the RUBI project, whose goal is to develop robots that interact with toddlers in early childhood education environments [43, 56, 45]. This dataset consists of 1365 video recordings of give-and-take trials using 136 different objects. The objects are soft toys depicting various human cell types, microbes and disease-related organisms. Figure 5.1 shows the entire set of these toys. Each video consists of the robot (RUBI) bringing the grasped object to its center of view, rotating it by 180 degrees and then returning it. The dataset was recorded from RUBI's head-mounted camera at 30 frames per second.

Figure 5.1. The GERMS dataset. The objects represent human cell types, microbes and disease-related organisms.

The data for GERMS were collected in two days. On the first day, each object was handed to RUBI in one of 6 pre-determined poses, 3 to each arm, after which RUBI grabbed the object and captured images while rotating it. The robot also captured the positions of its joints for every capture image. On the second day, we asked a set of human subjects to hand the GERM objects to RUBI in poses they considered natural. A total of 12 subjects participated in test data collection, each subject handing between 10 and 17 objects to RUBI. For each object, at least 4 different test poses were captured.

Table 5.2. GERMS dataset statistics (mean±std).

	Number of tracks	**Images per Track**	**Total Number of Images**
Day 1	816	157 ± 12	76,722
Day 2	549	145 ± 19	51,561

The background of the GERMS dataset was provided by a large screen TV displaying video scenes from the classroom in which RUBI operates, including toddlers and adults moving around.

We use half of the data collected in day 1 and 2 for training and the other half of each day for testing. More specifically, three random tracks out of six tracks for each object in Day 1 and two randomly selected tracks for each object from Day 2 were used for training the network and the rest was used for testing. Table 5.2 shows the statistics of training and testing data for the experiments in this paper.

5.5 Network Architecture

The traditional view of an active object recognition pipeline usually treats the visual recognition and action learning problems separately, with visual features being fixed when learning actions. In this work, we try to solve both problems simultaneously to reduce the training time of an AOR model. By incorporating the errors from action prediction into visual feature extraction, we hope to acquire features that are suited for both label and action prediction.

The network architecture is shown in figure 5.2. The input image is first transformed to a set of beliefs over different object labels by a classification network. The belief vector is then combined with the accumulated belief vectors over previous views to produce an encoding of the *state* of the system. This is accomplished by the *Mixture belief update* layer in the network. The new accumulated belief is then transformed into

action-values, based on which the next object view is selected.

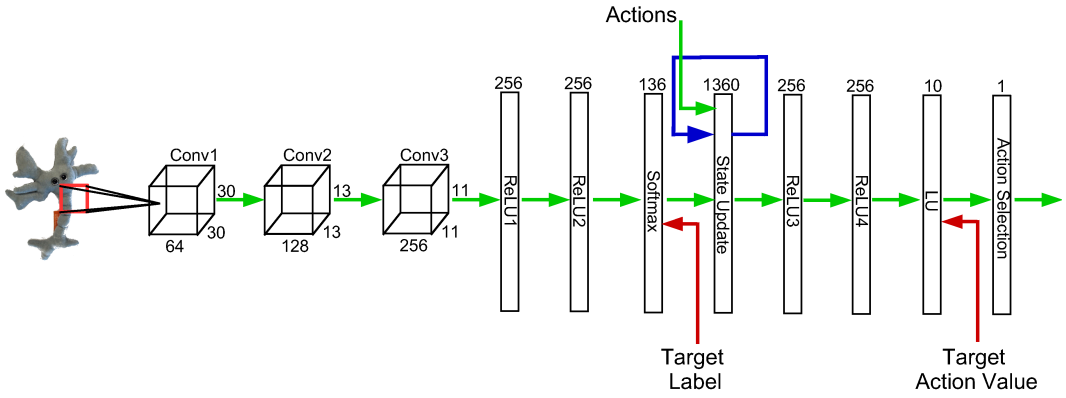


Figure 5.2. The network architecture for active object recognition. Red arrows represent target values that are used to train the network. The numbers represent the number of units in each layer of the network.

We next detail each part of the network, describing the challenges in training the layer and corresponding solutions. We first address the transformation of images into beliefs over object classes. Then we outline the belief accumulation problem over object views, followed by the action learning and,finally, present the full description of the algorithm to train this model.

5.5.1 Single Image Classification

The goal of this part of the network is to transform a single image into beliefs over different object labels. The feature extraction stage is comprised of 3 convolution layers followed by 3 fully connected layers. The dimensions of each layer are shown in figure 4.6. The convolution layers use filters of size $3 \times 7 \times 7$, $64 \times 5 \times 5$ and $128 \times 3 \times 3$ respectively for layers 1,2 and 3. The number of parameters in each layer of the network is shown in table 5.3. The operations of each layer are inspired by the model proposed in [36]. Each convolution layer is followed by rectification, normalization across channels and max pooling over a neighborhood of size 2×2 with stride of 1.

We shall denote the GERMS dataset by $D = \{I_i, y_i, P_i\}_{i=1}^{N}$, where $I_i \in \mathbb{R}^{64 \times 64 \times 3}$

Table 5.3. Number of units and parameters for the proposed network.

Layer	Number of Units	Input to Unit	Num. Parameters
Conv1	64x30x30	$3 \times 7 \times 7$	9K
Conv2	128x13x13	$64 \times 5 \times 5$	204K
Conv3	256x11x11	$128 \times 3 \times 3$	294K
ReLU1	256	30976	7M
ReLU2	256	256	65K
Softmax	136	256	34K
State Update.	1360	136	184K
ReLU3	256	1360+256	413K
ReLU4	256	256	65K
LU	10	256	2K

is the image captured by the robot camera, $y_i \in \{o_1, o_2, ..., o_c\}$ is the object label and P_i is a positive integer number denoting the pose of the robot's gripper [45]. In order to learn the weights of the single image classification part, we perform gradient decent on action prediction and cross-entropy costs, denoted by $\mathbb{C}_{RL}$ and $\mathbb{C}_{CL}$ respectively. The cross-entropy classification cost $\mathbb{C}_{CL}$ is:

$$\mathbb{C}_{CL} = -\sum_{i=1}^{N}\sum_{j=1}^{C} \mathbb{I}(y_i = c) \log B_{ij}. \tag{5.1}$$

Here $\mathbb{I}$ is the indicator function for the class of the object and $B_{ij} = P(o_j|I_i)$ is the predicted label belief for the i^{th} image belonging to the j^{th} object class. The next subsection describes the action prediction cost $\mathbb{C}_{RL}$.

5.5.2 Action Value Prediction

Active object recognition can be treated as a reinforcement learning problem, whose goal is to learn an optimal policy $\pi^* : S \rightarrow A$ from states S to actions A. The optimal policy is expected to maximize the total reward for every *interaction sequence*

$s^{\pi}_{0:T}$ with the environment,

$$s_0 \xrightarrow{\pi(s_0)} s_1 \xrightarrow{\pi(s_1)} s_2 \xrightarrow{\pi(s_2)} \dots \xrightarrow{\pi(s_{T--1})} s_T$$

where $s_i \xrightarrow{\pi(s_i)} s_{i+1}$ is the transition from s_i to s_{i+1} by performing the action $a_i = \pi(s_i)$. The *total reward* for an interaction sequence $s^{\pi}_{0:T}$ is $TR(s^{\pi}_{0:T}) = \sum_{t=0}^{T} \gamma^t R(s_t)$ where $R : S \to \mathbb{R}$ is a reward function and γ, $0 < \gamma < 1$ is a discount factor used to emphasize immediate rewards. For an AOR system, an interaction sequence starts by observing image of the object with the initial orientation in the robot's gripper. The state of the system is then updated by the observed image, and an action is selected to perform on the object to maximize the total reward. The reward in each step is determined by the accuracy of predicted label for the observed images up to that step.

In order to learn the optimal policy, we use the $Q(\lambda)$ algorithm to train the network to predict actions for improved classification [83]. This is a model-free method that learns to predict the expected reward of actions in each state. More specifically, let $Q^{\pi}(s,a)$ be the *action value* for state s and action a,

$$Q^{\pi}(s,a) = E_{\pi}\left\{TR(s^{\pi}_{0:T}) | s_0 = s, a_0 = a\right\},$$

which is the expected reward for performing action a in state s and then following policy π. Let the agent interact with the environment to produce a set of interaction sequences $\{s^{\pi}_{0:T}\}$. Then $Q(\lambda)$ learns a policy by applying the following update rule to every observed transition $TR^{\pi}(s_t, s_{t+1}) = s_t \xrightarrow{\pi(s_t)} s_{t+1}$,

$$Q^{\pi}(s_t,a_t) \leftarrow (1-\alpha)Q^{\pi}(s_t,a_t) + \alpha\left[R(s_{t+1}) + \gamma \max_a Q^{\pi}(s_{t+1},a)\right] \tag{5.2}$$

where $0 < \alpha < 1$ is the learning rate, and action a_t is selected using an epsilon-greedy

version of the learned policy. We interpret this iterative update in the following way to be useful for training a neural network. Let the output layer of the network predict $Q(s,a)$ for the learned policy π for every possible action a in s. Then a practical approximation of the optimal policy is obtained by minimizing the reinforcement learning cost,

$$\mathbb{C}_{RL} = \sum_{TR^{\pi}(s_t,s_{t+1})\in\{s^{\pi}_{0:T}\}} \Big[R(s_{t+1}) + \gamma \max_{a} Q^{\pi}(s_{t+1},a) - Q^{\pi}(s_t,a_t)\Big]^2 \tag{5.3}$$

In this network, action value prediction is performed by transforming the state of the system s_t at t^{th} step through layers ReLU3,ReLU4 and LU. We train the weights of the network in these layers by minimize $\mathbb{C}_{RL}$. In the next subsection, we go into the details of state encoding, and after that we describe the details of the set of actions.

5.5.3 State Encoding

State encoding has a prominent effect on the performance of an AOR system. Based on the current state of the system, an action is selected that is expected to decrease the ambiguity about the object label. An appealing choice is to transform images into beliefs over different target classes and use them as the state of the system. Based on the target label beliefs, the system decides to perform an action to improve its target label prediction. What we expect from the AOR system is to guide the robot to pick object views that are more discriminative among target classes.

We first transform the input image I_i into a belief vector $B_i = [B_{ij}]_{j=1}^{C}$ using the the first 7 layers of the network, where

$$B_{ij} \geq 0, \sum_{j=1}^{C} B_{ij} = 1,$$

The produced label belief vector is then combined with the previously observed belief vectors from this interaction sequence to form the state of the system. The motivation

for this encoding is that the combined belief encodes the ambiguity of the system about target classes and thus can be used to navigate to more discriminative views of objects. Active object recognition methods usually adapt a Naive Bayes approach to combining beliefs from different observations. Assume that in an interaction sequence, a sequence of images $I_{0:t} = \{I_0, I_1, \ldots, I_t\}$ have been observed and their corresponding beliefs $B_{0:t} = \{B_0, B_1, \ldots, B_t\}$ have been calculated. The state of the system at time t is calculated using Naive Bayes belief combination, which is to take the element-wise product of the individual belief vectors and then normalize,

$$\begin{aligned} s_t = P(O|I_{0:t}) &= \frac{P(O, I_{0:t})}{P(I_{0:t})} \\ &\propto P(O) \prod_{i=0}^{t} P(I_i|O) \\ &\propto \prod_{i=0}^{t} P(O|I_i) \end{aligned} \tag{5.4}$$

where O is the target label, and $P(O|I_i)$ is the vector of beliefs produced using single image classification. Here we assumed a uniform prior over images and target labels. The problem with Naive Bayes is that if an image is observed repeatedly in $I_{0:t}$, the result will change based on the number of repetitions. This is undesirable since the state of the system changes with repeated observations of an image where no new information is added to the system. If a specific image is suitable for classification, the system can visit that image more often to artificially increase the performance of the system. To avoid this problem, we adapt a generative model based on Dirichlet distribution to combine different belief vectors.

We use a generative model similar to [64] to calculate the state of the system given a set of images. The intuition behind this model is that performing an action on an object will produce a distribution of belief vectors. We model the observed belief vectors given

the object and action as a Dirichlet distribution, parameters of which are learned from the data. The model is shown in figure 5.3. Here $a \in \{a^1, a^2, \ldots, a^H\}$ is a discrete variable representing the action from the repertoire of actions, $o \in \{o^1, o^2, \ldots, o^C\}$ represents the object label and $\alpha \in \mathbb{R}^C$ is the vector of parameters of the Dirichlet distribution from which the belief vector $B \in \mathbb{R}^C$ over target labels is drawn,

$$\begin{aligned} P(B|\alpha) &= \mathrm{Dir}(B;\alpha) \\ &= \frac{\Gamma(\sum_{j=1}^{C}[\alpha]_j)}{\prod_{j=1}^{C}\Gamma([\alpha]_j)} \prod_{j=1}^{C}[B]_j^{[\alpha]_j-1} \end{aligned} \tag{5.5}$$

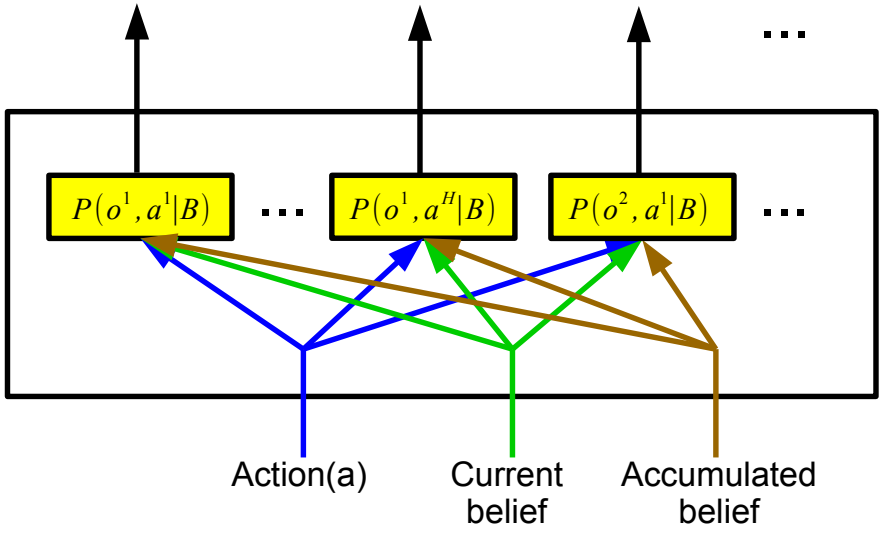


Figure 5.3. Dirichlet belief update layer. Each unit in this layer represents a Dirichlet distribution for a pair of object-action. The parameters of this layer are the vectors of Dirichlet parameters α_k^o for each unit.

The state of the system is calculated by computing the posterior probability of object-action beliefs using the model in figure 5.3. Let $P_a^o(a_i, B_i) = P(o, a|a_i, B_i)$ denote the posterior probability of an object-action pair given the performed action and the observed belief vector. Assuming uniform prior over object and α and a deterministic

policy for choosing actions,

$$
\begin{aligned}
P(o,a|B) = & \\
& = \frac{\int_\alpha P(o,a,B,\alpha)d\alpha}{P(B)} \\
& \propto \int_\alpha P(o)P(a|o)P(\alpha|o,a)P(B|\alpha)d\alpha \\
& \propto \int_{\alpha_a^o} Dir(B;\alpha_a^o))d\alpha_a^o
\end{aligned} \tag{5.6}
$$

The notation α_a^o is to make clear that there is an α for each pair of object-action. Instead of full posterior probability, we use $\hat{\alpha}_a^o$, the maximum likelihood estimate of α, and replace the integral above by ,

$$P(o,a|B) \approx Dir(B|\hat{\alpha}_a^o) \tag{5.7}$$

For an interaction sequence $B_{0:t}$ and $A_{0:t} = \{a_0, a_1, \ldots, a_t\}$, the posterior probability of object-action pair is,

$$P(o,a|A_{0:t},B_{0:t}) = \prod_{i=0}^{t} P(o,a|B_i)^{\mathbb{I}(a,a_i)} \tag{5.8}$$

The state of the system is comprised of the vector of object posterior beliefs for every object and action, plus the features and belief extracted from the latest image I_t,

$$
\begin{aligned}
s_t = \{[P(o,a|A_{0:t},B_{0:t})], B_t\}, & \\
o \in \{o^1, o^2, \ldots, o^C\} & \\
a \in \{a^1, a^2, \ldots, a^H\} &
\end{aligned} \tag{5.9}
$$

Note that $s_t \in \mathbb{R}^{CH}$ is a vector of length $C \times H$.

5.5.4 Training for Joint Label and Action Prediction

Our goal is to train the network for joint action and label prediction. We achieve this by minimizing the total cost which is the sum of label (5.1) and action prediction (5.3) costs. The errors for action value prediction are back-propagated through the entire network, reaching visual feature extraction units. The total cost function for action-value and label prediction is,

$$Cost = \mathbb{C}_{RL} + \mathbb{C}_{CL} \tag{5.10}$$

The weights of the network in the visual feature extraction layers (Conv1, Conv2, Conv3, ReLU1, ReLU2, softmax) are trained using backpropagation on (5.10), while the action prediction layers (ReLU3, ReLU4 and LU) are trained by gradient descent on the action prediction error (5.3).

We use gradient descent with respect to the network weights to minimize the cost function in (5.10). If the training converges, it will land on a local optimum point since the neural network's error surface is spiky with many local optimums. A concern may be raised that the minimization of the cost function may diverge, for example if changing the network weights to reduce the reinforcement learning cost causes the classification cost to increase. We didn't observe such behavior in practice while training the network. A counter argument against the divergence of the cost function is that learning a better classifier is in the direction of learning an optimal policy, as less confusion in label prediction simplifies the object exploration policy and can help the policy to more efficiently search for discriminative views of objects.

To learn the parameters of the belief update layer α_a^o, we use gradient descent on log-likelihood of the data. The maximum likelihood of Dirichlet distribution is a convex function of its parameters and can be minimized using gradient descent. For a

set of beliefs $B_{1:N}$ observed by performing action a on the object o, the gradient of the log-likelihood with respect to the parameters are,

$$\begin{aligned}\frac{\partial \log P(B_{1:N}|\alpha_a^o)}{\partial [\alpha_a^o]_k} &= N\frac{d}{d[\alpha_a^o]_k}\log\Gamma(\sum_{j=1}^{C}[\alpha_a^o]_j) - \frac{d}{d[\alpha_a^o]_k}\log\Gamma([\alpha_a^o]_k) + \log B_k \\ &= N\Psi(\sum_{j=1}^{C}[\alpha_a^o]_j) - N\Psi([\alpha_a^o]_k) + \log B_k \end{aligned} \tag{5.11}$$

where $\Psi(x) = d/d(x)\log\Gamma(x)$ is the digamma function. There is one unit per Dirichlet distribution $Dir(|\alpha_a^O)$ in the belief update layer of the network. These units receive the current belief and the previous state of the system, and produce an updated belief. An schematic of the belief update layer of the network is shown in figure 5.3. Learning α_k^o is carried out simultaneously with the rest of the network weights in the same training session.

5.5.5 Reward Function

Another component that has an important effect on the performance of our AOR system is the reward function which maps state of the system (5.4) into rewards. A simple choice for reward function is

$$R(s_t) = \begin{cases} +1 & \text{if } \arg\max_i [B_t]_i = \text{Target-Label}(I_t)) \\ -1 & \text{otherwise} \end{cases} \tag{5.12}$$

A reward of $+1(-1)$ is given to the system if at time step t the action a_t brings the object to a pose for which the predicted label is correct (wrong). The intention behind this reward function is to drive the AOR system to pick actions that lead to best next view of the object in terms of label prediction.

5.5.6 Action Coding

In order to be able to reach every position in the robot's joint gripper range, we use a set of relative rotations as the actions of the system. More specifically, we use 10 actions to rotate the gripper from its current position by any of the following offset values: $\{\pm\frac{\pi}{4}, \pm\frac{\pi}{8}, \pm\frac{\pi}{16}, \pm\frac{\pi}{32}, \pm\frac{\pi}{64}\}$. The total range of rotation for each of the robot's grippers is π. The actions are selected to be fine grained enough so that the robot can reach any position with minimum number of movements possible. This encoding is simple and flexible in the range of positions that the robot can reach, however we found that the policies can become stuck with a few actions without trying the rest. Encoding the states with the Dirichlet belief update helps alleviate this issue to some degree, however, it doesn't completely remove the problem. We deal with this problem by forcing the algorithm to pick the next best action whenever the best action leads to an image which has already been seen.

5.6 Experimental Results

5.6.1 Training Details

We trained the network by minimizing the costs of classification, action value prediction (5.3) and negative of log-likelihood of Dirichlet distributions (5.11). We used backpropagation with minibatches of size 128 to train the network. For $Q(\lambda)$, we used initial learning rate of 0.1 which was multiplied by 0.5 after iterations $400, 800, 1200, 1500$ and then remained constant. The total number of training iterations is 4000. For each iteration, an interaction sequence of length 5 is followed. The full training procedure is shown in algorithm 2. For $Q(\lambda)$, we used ε-greedy policy in the training stage, with ε decreasing step-wise from 0.9 to 0.1. We found that using an $\varepsilon > 0$ at the test stage hurts the performance, therefore we used $\varepsilon = 0$ during testing. The number of actions is 10 as

Algorithm 2. Training the network for joint label and action prediction.

procedure TRAIN
  $R \leftarrow 1$
  for iteration=1 To N **do**
    $I_1, y \leftarrow$ NextImage(iteration)
    $s_0 \leftarrow [0]$
    Actions $\leftarrow$ RandomActions(NumActions)
    for t=1 To NumMoves **do**
      s_t, predictedActions $\leftarrow$ FeedForward(I_t, s_{t-1}, Actions)
      $I_{t+1}, y \leftarrow$ NextImage(I_t, predictedActions)
      targetActionVals, $\hat{y} \leftarrow$ LookAhead(I_{t+1}, s_t, Actions)
      if $t =$ NumMoves **then**
        targetActionVals $\leftarrow$ targetActionVals $+ R(s_t)$
      end if
      for $W \in \{ReLU3, ReLU4, LU\}$ **do**
        $W \leftarrow W - \lambda_W \frac{\partial}{\partial W}\{\mathbb{C}_{RL}\}$
      end for
      for $W \in \{Conv1, Conv2, Conv3, ReLU1, ReLU2, Softmax\}$ **do**
        $W \leftarrow W - \lambda_W \frac{\partial}{\partial W}\{\mathbb{C}_{RL} + \mathbb{C}_{CL}\}$
      end for
      for $o \in \{o^1, o^2, \ldots, o^C\}, a \in \{a^1, a^2, \ldots, a^H\}$ **do**
        $\alpha_a^o \leftarrow \alpha_a^o + \lambda \frac{\partial}{\partial \alpha_a^o} \log P(B_t | \alpha_a^o)$
      end for
    end for
  end for
end procedure

described above, and there are a total of 136 object classes, resulting in a total of 1360 Dirichlet distributions for state encoding 5.9.

5.6.2 Learning the Parameters of Dirichlet Distributions

Figure 5.4 shows the average negative log-likelihood of the data under Dirichlet distributions for training a network. This figure shows that the neg-log-likelihood of data decreases for the first 1000 iterations, after which the rate of change is decreased but not stopped. In this figure, there are impulses that occur in the negative-log-likelihood of the data. It is observed that the magnitude of these impulses increase as the model

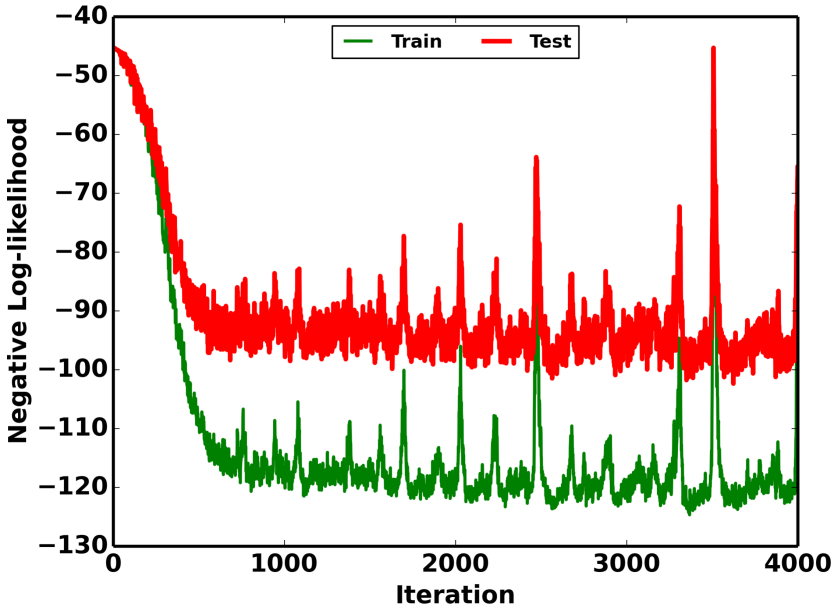


Figure 5.4. Average Negative log-likelihood of data under Dirichlet distributions. The decrease in negative log-likelihood indicates learning in the belief update layer.

fits the training data. We attribute these impulses to the glitches in the gradients of the action-value cost function with respect to the network weights. As training continues, the glitches are fixed by the image batches for which the network can predict the action-values correctly.

5.6.3 Label Prediction Accuracy

Static Label Prediction

First we report the accuracy of *static label prediction* on GERMS using a deep convolutional network that is trained to predict object labels without the active component. We train a deep convolutional network with 3 convolutional and 2 fully connected layers with the number of units shown in figure 4.6. This network is trained by minimizing the cross entropy cost in (5.1) for predicting the labels of single frames of the GERMS dataset. Unlike active methods which select different images with different probabilities for training, we select GERMS training images for static label prediction with uniform

probability. For testing with one or more images, we randomly select images from each track, and classify them with the trained network. The probabilities for multiple images are combined using the naive Bayes rule in (5.4). The average accuracy of static label prediction for the test set is shown in table 4.2.

We observe that the accuracy of static label prediction is higher than Naive Bayes active methods, but lower than Dirichlet based active models. The difference originates from the ability of active methods in selecting images that are used for training and then to choose such images at test time. Dirichlet based active methods achieve higher accuracy by focusing the training on images that are more discriminative for label prediction. The static model randomly chooses among the ambiguous and non-ambiguous views of different objects at training, which leads to lower accuracy compared to Dirichlet based methods. On the other hand Naive Bayes methods fail to visit enough training images due to overfitting in the action selection layer, and thus are unable to compete in accuracy even with static models.

Since the primary focus of this paper is active object recognition, we do not investigate further the properties of static object recognition models. Instead, we focus on Dicihlet-based and Naive Bayes active object recognition models, and compare their performance in the following sections.

Comparing Naive Bayes and Dirichlet State Encoding

In this section we compare the effectiveness of the Dirichlet and Naive Bayes state encodings for label prediction accuracy. For Naive Bayes models (NB), the state of the system is updated using (5.4), while the size and configuration of the rest of the network remain the same. Dirichlet (DR) state encoding is implemented using (5.9). For each encoding and for each arm, we train 10 different models and report the average test label prediction accuracy as a function of number of observed images, comparing the

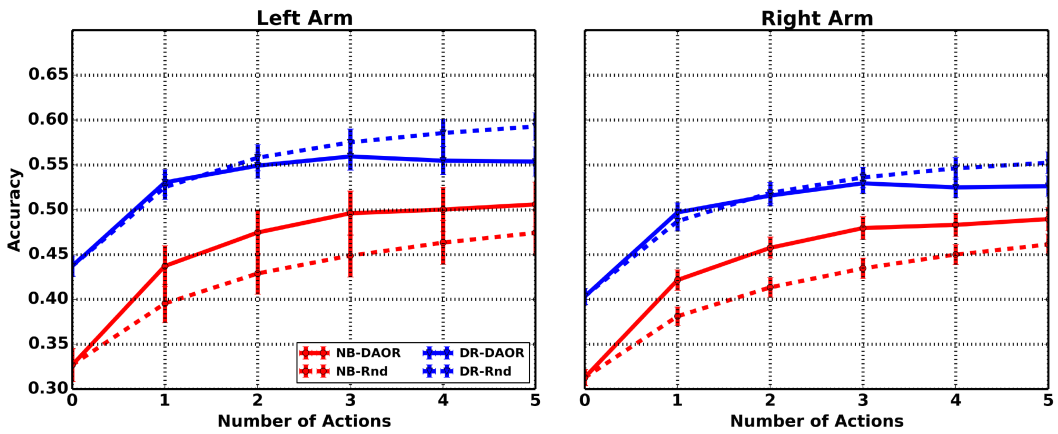


Figure 5.5. Test label prediction accuracy as a function of number of observed images for left and right arms for Dirichlet state encoding with repeated visits (DR) and non-repeated visits (DN).

Deep Active Object Recognition (DAOR) and Random (Rnd) action selection policies. Figure 5.5 plots the performance for these models. It is obvious that the Dirichlet model is superior to Naive Bayes in label prediction accuracy.

The first point to notice in figure 5.5 is the performance difference between Naive Bayes and Dirichlet belief updates on single images (action 0). NB models achieve a performance less than 35%, while Dirichlet achieves higher than 40%. One interpretation of this result is that the Naive Bayes model pick actions that bounce between a subset of train images, leading to under-fitting of the model. In the *visualizing policies* subsection, we provide some evidence for this justification. On the other hand, the performance of DR-DAOR model tends to saturate after 3 actions, while DR-Rnd keeps improving for subsequent actions. This might be due to the fact that DR-DAOR also bounces between subsets of images at the test time. We can avoid such behavior by forcing the policies to pick actions that lead to joint poses that haven't already been visited in the same interaction sequence.

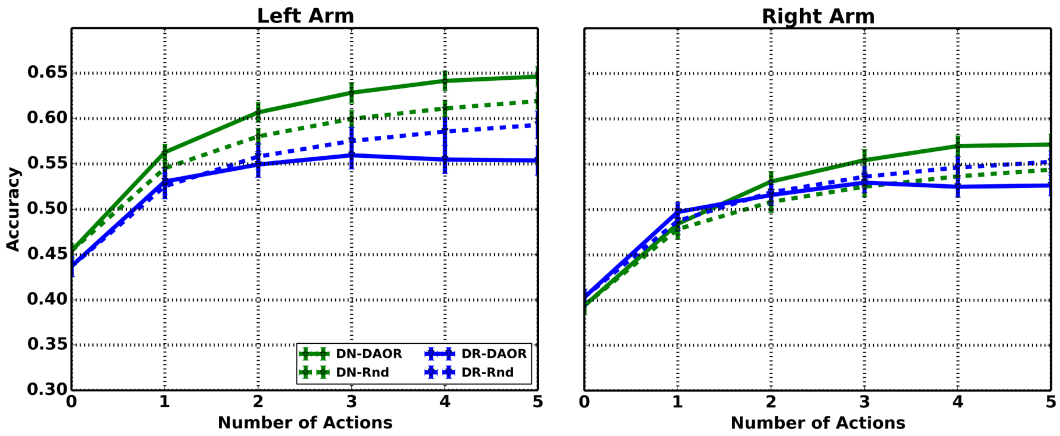


Figure 5.6. Test label prediction accuracy as a function of number of observed images for left and right arms for Naive Bayes (NB) and Dirichlet (DR) state encoding.

Removing Duplicate Visits

We train a set of models using Dirichlet state encoding, while forcing the policy to pick non-duplicate joint poses in every action of an interaction sequence. This approach is easy to implement by keeping a history of visited joint poses during an interaction sequence and picking actions with highest action value that lead to novel joint positions. We refer to this model as Dirichlet with non-repeated visits (DN). Comparison between DN and DR for Rnd and DAOR policies (both forced to visit novel poses) is shown in figure 5.6.

Comparison between the models mentioned above is shown in table 5.4. We see that the best performing model is DN-DAOR with the exception of action 1 for the right arm, which DR-DAOR achieves the best performance. For both arms, Dirichlet models perform significantly better than Naive Bayes, improving the model's performance on average by 10% for the right arm and 14% for the left arm.

Visualizing Policies

It may help us understand the weakness and strength of different models if we take a closer look into the learned policies. For this purpose, we visualize the consecutive

Table 5.4. Comparison of static, DQN, random and sequential AOR accuracy(%) as a number of observed frames.

	1	2	3	4	5	6	
Static OR	35.2	46.3	51.0	53.9	56.0	**57.1**	
NB-Rnd	31.3	38.1	41.3	43.4	45.0	46.1	
NB-DAOR	31.3	42.1	45.8	48.0	48.3	49.0	**Right Arm**
DR-RND	**40.3**	48.7	51.9	53.6	54.6	55.2	
DR-DAOR	**40.3**	**49.7**	51.6	53.0	52.5	52.6	
DN-RND	39.4	47.8	50.8	52.5	53.6	54.3	
DN-DAOR	39.3	48.4	**53.1**	**55.4**	**57.0**	**57.1**	
Static OR	35.6	46.2	50.8	53.4	55.3	56.6	
NB-Rnd	32.7	39.5	42.9	44.9	46.3	47.4	
NB-DAOR	32.7	43.7	47.5	49.6	50.0	50.6	**Left Arm**
DR-RND	43.7	52.5	55.8	57.5	58.6	59.3	
DR-DAOR	43.7	53.0	54.9	55.9	55.5	55.4	
DN-RND	**45.4**	54.5	58.0	60.0	61.1	61.9	
DN-DAOR	**45.4**	**56.3**	**60.7**	**62.8**	**64.1**	**64.6**	

actions in the interaction sequences of length 5, as shown for training data in figures 5.7 and for test data in figure 5.8. Each plot represents actions in different rows, with the magnitude and orientation of the action begin depicted by the length and direction of the corresponding arrow on the left side. Each time step of the interaction sequence is shown as a numbered column. The colored lines in each plot connect one action in column i to another action in column $i+1$ only if those actions appeared consecutively in interaction sequences at these time steps. The thickness of lines depicts the relative frequency by which two actions were observed on the data.

Figure 5.7 visualizes the policies DN-DAOR and NB-DAOR on the training data. This figure helps clarify the lower performance of NB models as described before. For NB-DAOR shown on the left side of figure 5.7, we see thick lines connecting actions that rotate the object with the largest magnitude in opposite directions. The relative thickness of these lines indicates that the model tends to go to one end of the joint's rotation range,

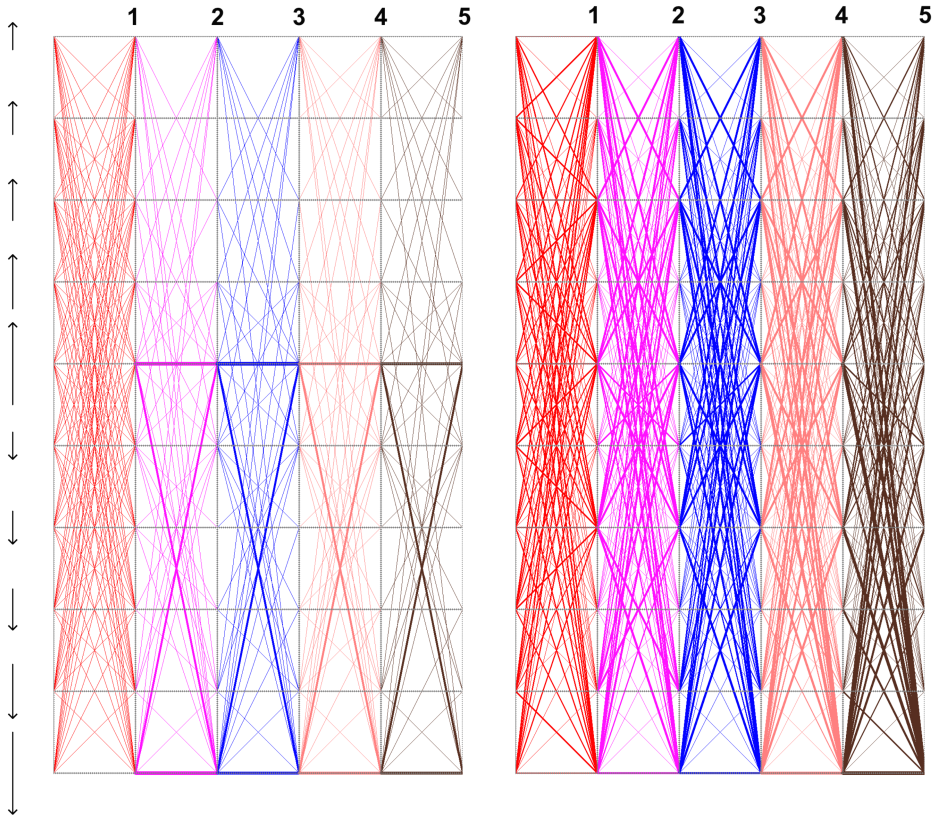


Figure 5.7. Visualization of (left) NB and (right) DN policies on training data. Each row represents an action and each column represents a time-step in object exploration performed by the policy in an interaction sequence. The color of lines connecting two columns are different for clarity for every consecutive time steps, while the thickness of these line indicate the frequency of that transition between views in interaction sequences.

go back with one large rotation and then repeat. Despite presence of other actions, this back and forth action dominates the training process, leading to lower accuracy on test label prediction for single images. On the right side of figure 5.7 we see that DN-DAOR picks a wide range of actions, which leads to better examination of training images and thus higher performance on single images.

Figure 5.8 visualizes the learned policies at test time for NB-DAOR and DN-DAOR. We see on the left side that NB-DAOR only swings between the two large rotations in the opposite direction, while DN-DAOR prefers to do a few larger actions (thick purple and blues lines connecting columns 2, 3 and 4) followed by few smaller

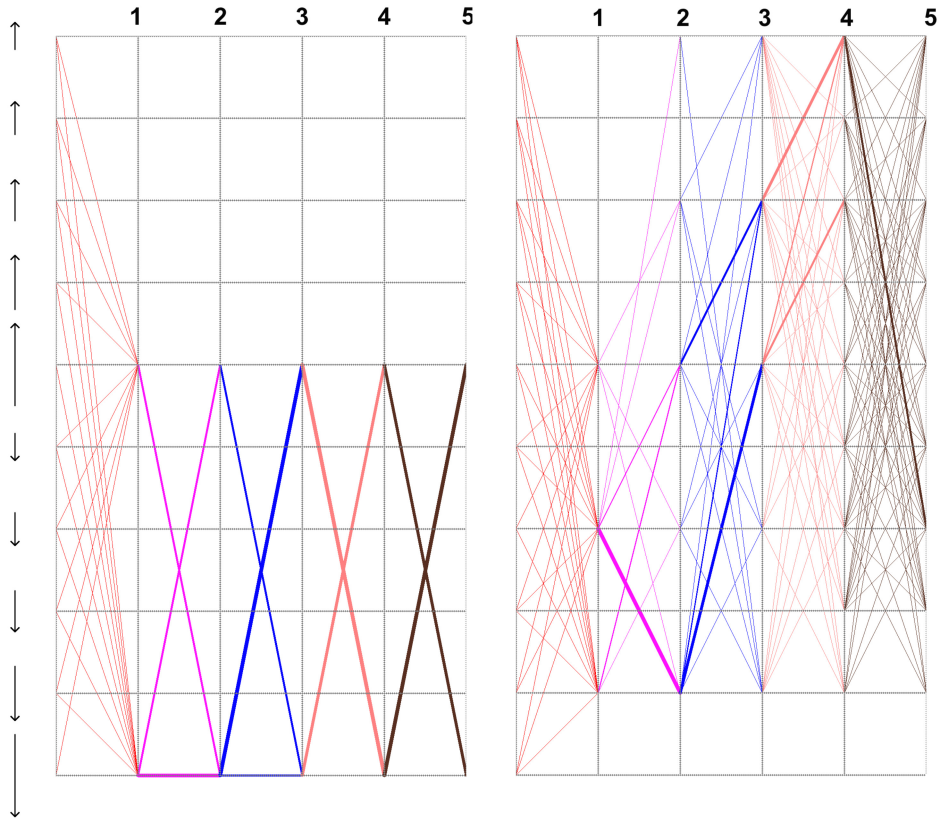


Figure 5.8. Visualization of (left) NB and (right) DN policies for test data. NB model prefers to repeats the same two actions, swinging between two joint poses at one end of the joint range. The DN model usually performs a few larger rotations on the object, followed by a few smaller rotations in different directions to inspect the objects in a fine-grained manner.

actions in different directions. There is no back and forth for DN-DAOR between visited joint positions, which leads to better performance on the test set.

5.7 Discussion

In this paper, we proposed a model for active object recognition based on deep convolutional neural networks. The model is trained by minimizing the action and label prediction costs. The visual features in early stages of this network were trained by minimizing both the action and label prediction costs. The difference between the work presented here and deeply supervised networks [17] is that in the latter, the training is

carried out by minimizing the classification error, while in our approach we minimized the action learning cost along with classification error. The joint cost minimization allows the model to learn visual features that are suitable for predicting object label and the action to be performed on the object to improve the recognition performance.

We adapted an alternative approach to the common Naive Bayes belief update rule for state encoding of the system. Naive Bayes has the potential of overfitting to subsets of training images, which could lead to lower accuracy at the test time. We used a generative model based on Dirichlet distribution to model the belief over object-action pairs. This model was embedded into the network, which allowed training the network in one pass jointly with label and action-value learning. The results of experiments confirmed that the proposed Dirichlet model is superior in test label prediction accuracy to the Naive Bayes approach for state encoding.

A common trend we observed in the models trained in this paper was the strong preference for a few actions, which led to limited examination of the objects, and thus lower performance on label prediction. This preference was strongest in the Naive Bayes state encoding models. Employing Dirichlet for state encoding helped alleviate this problem, mainly for the training data and less for the test data. We observed that the strong preference for a limited set of actions weakens for the training stage for the DR-DAOR model, and as a result the model explored the training data more efficiently and achieved higher label prediction accuracy on the test data.

A difficulty that arises in using beliefs for state encoding is the difference in distribution of beliefs over train and test data. This results in overfitting of the policies to high confidence beliefs, which may not be the case for test data. In training our models, the training accuracy reaches above 90% after 1000 iterations. This may cause the algorithm to reward every action, which finally may lead to one action taking over and always producing the highest action value. A remedy for this problem requires

the training data to be representative of the test data in prediction accuracy. However we found that the test set in GERMS is very challenging for label prediction. Another possibility is the use of outside data in training the label prediction module, which may help produce more similar distribution of beliefs over training and test data.

5.8 Acknowledgements

The research presented here was funded by NSF IIS 0968573 SoCS, IIS INT2-Large 0808767, and NSF SBE-0542013 and in part by US NSF ACI-1541349 and OCI-1246396, the University of California Office of the President, and the California Institute for Telecommunications and Information Technology (Calit2).

Chapter 5, in full, is a reprint of the material as it appears in Computer Vision and Image Understanding (CVIU) 2017. Malmir, Mohsen; Sikka, Karan; Forster, Deborah; Fasel, Ian; Movellan, Javier R.; Cottrell, Garrison W. copyright Elsevier, 2017. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Belief Tree Search for Active Object Recognition

6.1 Abstract

Active Object Recognition (AOR) has been approached as an unsupervised learning problem, in which optimal trajectories for object inspection are not known and to be discovered by reducing label uncertainty or training with reinforcement learning. Such approaches suffer from local optima and have no guarantees of the quality of their solution. In this paper, we treat AOR as a Partially Observable Markov Decision Process (POMDP) and find near-optimal values and corresponding action-values of training data using Belief Tree Search (BTS) on the AOR belief Markov Decision Process (MDP). AOR then reduces to the problem of knowledge transfer from these action-values to the test set. We train a Long Short Term Memory (LSTM) network on these values to predict the best next action on the training set rollouts and experimentally show that our method generalizes well to explore novel objects and novel views of familiar objects with high accuracy. We compare this supervised scheme against guided policy search, and show that the LSTM network reaches higher recognition accuracy compared to the guided policy search and guided Neurally Fitted Q-iteration. We further look into optimizing the observation function to increase the total collected reward during active recognition. In

AOR, the observation function is known only approximately. We derive a gradient-based update for the observation function to increase the total expected reward. We show that by optimizing the observation function and retraining the supervised LSTM network, the AOR performance on the test set improves significantly.

6.2 Introduction

Active Object Recognition (AOR) refers to the problem of predicting object label from the images while being able to change the pose of the object relative to the camera for increasing prediction certainty. A robot rotating an in-hand object to refine its label prediction accuracy is an example of an AOR system. Ambiguity in object recognition exists because of similar views of different objects. AOR aims at finding the optimal sequence of actions which decreases the label ambiguity and improves object recognition performance in smaller number of steps. Despite its wide application and performance improvement capacity, AOR has not been applied widely and has remained secluded from main-stream computer vision progress in recent years.

Existing approaches to AOR change the sensor position to reduce the ambiguity of label prediction [2, 6, 18]. Most of these methods rely on uncertainty about object label, and use greedy best next action selection [12] at the test time to decrease label probability entropy. A few methods aim for optimal action selection at the test time using dynamic programming [3] or Monte Carlo planning [61]. However these methods require a model of the object, and are computationally heavy at the test time. We propose a method that reduces optimal action selection to classification of belief at the test time and does not require a generative model of objects. We show that the proposed method generalizes well to *novel views* of familiar objects, and also to *novel* objects. Moreover, we show that AOR paradigm can be used to improve the label prediction accuracy of image classifier by emphasizing images in the train set that are more likely to result in

higher rewards.

In the first contribution of this paper, we formulate AOR as a POMDP problem and adapt a Belief Tree Search algorithm [37] to discover near-optimal values for objects poses on the training set. We infer a policy from these values, and use it to train an LSTM network to predict the best action given the current objects belief. At the test time, we use the actions predicted by this LSTM to explore the objects. We show that this supervised approach generalizes well to explore novel objects and novel views of familiar objects, and results in higher AOR accuracy compared to reinforcement learning and guided policy search methods.

In our second contribution, we derive an update rule to learn the parameters of the POMDP likelihood function with the goal of maximizing the total reward. This update rule emphasizes views of objects that will produce higher rewards in the future. We show that by retraining the likelihood function using the proposed method, the performance of the AOR system significantly improves.

In the next section, we review the previous approaches related to our method. Then we present the BTS algorithm and the observation function update rule. In the results section, we report the details of the implementation of these methods and their performance on GERMS [45] dataset. GERMS has proven to be a challenging AOR dataset, and we improve state-of-the-art performance on this dataset. The final section is the concluding remarks.

6.3 Literature Review

A large category of AOR models try to minimize the predicted label uncertainty through best next-action planning [72, 9, 15, 18, 10, 12]. These models predict the object label probabilities using the current view, and search for the best next action that minimizes the expected entropy of object labels. In these methods, learning object

appearance is performed by fitting a generative model offline, while best action selection is carried out online at the test time. Uncertainty measures such as conditional entropy and mutual information are computationally expensive to evaluate for all possible observations. Therefore these methods usually resort to approximations of these measures, which might result in poor AOR performance.

A second category of models use techniques such as REINFORCE [86] or Neurally Fitted Q-Iteration (NFQ) [65] to find a good policy or action-value function for object exploration [60, 47, 45]. A parametric function that encodes object exploration policy or action-values is learned offline by using an exploration policy and collecting rewards that depend on the label prediction accuracy. The model then updates the parameters of policy or action-value function to maximize its total expected reward. These method suffer from high variance of prediction due to sampling of actions, only guarantee convergence to a local optima and require large training periods to explore and discover suitable sequences of actions.

More recently, deep convolutional neural networks (CNN) have been applied in AOR as a tool for modeling object appearance along with action-value prediction [29, 44, 24, 27]. Malmir et. al trained a deep CNN using NFQ update rule [44]. In this work, a layer of Dirichlet distribution is embedded into the network for modeling the distribution of beliefs for different object-action pairs. Johns et. al used deep CNNs for entropy regression and action prediction for the set of next view points [29]. Finding the optimal trajectory for object inspection is then approximated by maximizing the sum of cross entropy over adjacent views pairs. Haque et al. trained LSTM networks with REINFORCE algorithm to recognize subjects from 3D point-clouds [24]. Jayaraman and Grauman modeled object exploration policy as a neural network and trained it using classification accuracy as reward [27]. They found that predicting the next state of the environment based on current state and action improves the overall AOR accuracy.

All these methods show improved performance over random exploration strategy and non-active methods. However they suffer from the same problem as previous methods, which is lack of guarantee of performance even on the training set.

There are very few approaches to AOR that aim to find optimal exploration policies. Atanasov et al. [3] adapted an active hypothesis testing approach [57] for camera view-point selection for object segmentation. This approach learns a model of object appearance, and uses that for planning a sequence of actions that minimizes the cost of motor movements, object classification and view-point prediction. A dynamic programming approach is used to discover the best sequence of actions that minimizes this cost. This method depends on the representation of object appearance for efficient planning, while our method acts in the belief space and is completely independent of object representation and classification. Patten et al. used Monte Carlo planning for active exploration and perception of outdoor objects . [61]. This method uses rollouts that depends on the point-cloud of objects for different actions. Compared to this method, our method is more intuitive, and doesn't require 3D models of objects.

Of special interest to this paper are works on belief tree search and Monte Carlo POMDP planning. Lee et. al [37] proposed clustering of beliefs in a belief tree search algorithm to reduce the width of the tree. DESPOT [76] uses sampling of observations to reduce the width of the belief tree for optimal action selection. POMCP [75] adapts Monte Carlo sampling and the Upper Confidence Trees algorithm [34] for efficient POMDP planning. We adapt an approach similar to [37] because of desirable properties such as acting on the belief space and performance guarantees on the *reachable* space of beliefs.

An important property of the proposed approach is its freedom of maintaining an object model. Our method acts in the belief space of objects, and only requires a black box simulator of objects that produces rollouts of object examination, and returns the

sequence of actions and their corresponding beliefs. Another important property of our method is that at the test time, the next action selection is achieved by (computationally simple) classification of the current belief. We show that this approach is more effective for learning object exploration policies, compared to reinforcement learning and actor-critic methods. In the next section we describe the proposed approach in more details.

6.4 Proposed Method

6.4.1 Belief Tree Search

Active object recognition is formulated as a POMDP problem denoted with tuple $< \mathscr{S}, \mathscr{A}, \mathscr{O}, \mathscr{T}, \mathscr{P}, \mathscr{R}, \gamma >$, where $\mathscr{S}$ is the set of states (object labels), $\mathscr{A}$ is the set of actions for object examination, and $\mathscr{O}$ is the set of observations (captured images of objects). We don't perceive the identity of object directly but rather collect information about it through observations. The transition function $\mathscr{T}$ marks the transition between different states and the observation function $\mathscr{P}(s,a,o)$ relates the observations to different object identities through $\mathscr{P}(s,a,o) = Pr(o|s,a)$, the probability of observing o after taking action a when the object labels is s. The reward function $\mathscr{R}(s,a)$ determines the reward for taking action a when the object label is s. Finally, γ is the reward discount factor.

In AOR POMDP, the transition function $\mathscr{T}$ reduces to the identity function. The observation space on the other hand includes all images of objects, and is prohibitively large to apply value iteration techniques [63]. Another possibility is to use Monte Carlo planning , which builds a search tree for one-step action planning [75]. However this method suffers from the curse of history. We seek to find a solution to the AOR POMDP that compactly represents history, and allows tractable search for finding the optimal values, and the corresponding action-values.

It has been shown that solution to POMDP can be found by solving the equivalent *belief* MDP, where the belief $b \in \mathbb{R}^{|\mathscr{S}|}$ denotes the posterior probability of states given the observation history,

$$b^t(s) = Pr(s|a_{0:t-1}, o_{0:t-1}) \tag{6.1}$$
$$\sum_{s\in\mathscr{S}} b(s) = 1, \;\; b(s) \geq 0 \;\; \forall s \in \mathscr{S}.$$

In belief MDP, states represent the posterior probabilities of POMDP states given the action-observation history. For this MDP, the transition between beliefs given action a is defined as,

$$\begin{aligned} &\mathscr{T}^{MDP}(b,a,b') \\ &\quad = Pr(b'|b,a) \\ &\quad = \sum_{o\in\Omega(b,a,b')} \sum_{s,s'\in\mathscr{S}} b(s)\mathscr{T}^{POMDP}(s,a,s')\mathscr{P}(s',a,o) \end{aligned} \tag{6.2}$$

where $\Omega(b,a,b')$ denotes the set of observations that can result in changing beliefs from b to b',

$$\begin{aligned} &\Omega(b,a,b') = \{o \in \mathscr{O}| \\ &\quad b'(s') = \frac{\mathscr{P}(s',a,o)\sum_{s\in\mathscr{S}} \mathscr{T}^{POMDP}(s,a,s')b(s)}{Pr(o|a,b)}\} \end{aligned} \tag{6.3}$$

The belief MDP reward is defined by calculating the expected reward over states,

$$R(b,a) = \sum_{s\in\mathscr{S}} b(s)\mathscr{R}^a_s. \tag{6.4}$$

And finally, given an initial belief b, action a and observation o the updated belief b' is

calculated by,

$$\begin{aligned} b'(s') &= Pr(s'|b,a,o) \\ &= \frac{\mathscr{P}(a,s',o)\sum_{s\in\mathscr{S}} b(s)\mathscr{T}^{POMDP}(s,a,s')}{Pr(o|a,b)}. \end{aligned} \tag{6.5}$$

Now that we defined the equivalent belief MDP, we aim at solving the planning problem using *Belief Tree Search* algorithm. This algorithm constructs a search tree for a given belief b, where different branches represent actions and the resulting observations. Each node in the tree represents a belief about the underlying POMDP states and edge captures an action and the resulting observation. The algorithms starts with b at the root and exhaustively performs all action to collect new observations and form new beliefs. These beliefs are then added as children of the root and the process iterates with new beliefs until stop states are reached in the leaves. The values are then backtracked from the leaves to the root to estimate the value of b. The belief tree search is used in online planning for POMDPs where the dynamics of the environment are known. We adapt BS to calculate the optimal values of images for the training set.

The plain belief tree search algorithm is computationally intractable to use for AOR belief MDP, since it examines all observations for each action. Instead we adapt the algorithm in theorem 1 of [37], which sacrifices optimality of the predicted values in exchange of computational tractability. This algorithm utilizes the smoothness of the optimal value function to cluster the belief space. More specifically, for a given $\delta > 0$, each node in the tree represents the approximate value of beliefs that are in δ-neighborhood of b', which we represent as $\delta(b)$. By clustering the belief space, the width of the belief tree decreases to a manageable size. This algorithm calculates the approximate optimal value of a given belief b only in $\mathscr{R}(b)$, which is the *reachable space*

of b. The reachable space is defined as the set of beliefs that are reachable by arbitrary sequences of actions from b. Finally, the algorithm utilizes the discount factor γ to limit the height of the belief tree.

We adapt this algorithm to find an approximately optimal value for each image in the training set. The algorithm is depicted in pseudo code style in algorithm 3. Using these values, training an active object recognition system reduces to a supervised learning and knowledge transfer problem. In each node of the tree, the algorithm expands all actions and receives the new observations. New beliefs are then calculated using (6.5). For each new belief b', if there is an already expanded belief b_0 in that height of the tree for which $b' \in \delta(b_0)$, the algorithm sets the value of b' equal to b_0 and backtracks. Otherwise, the search continues in the children of b'.

Our algorithm builds a belief tree over $\mathbf{R}(b_0)$ by sampling from images in the training set and maintaining a δ-packing of $\mathbf{R}(b_0)$ at each level of the tree. A belief tree with root b_0 denotes all the possible actions and observations that are encountered while inspecting an object with the initial belief b_0. A belief tree captures all the possible actions and observations, construction of full belief tree is prohibitive in case of active object recognition because the size of the observation space is extremely large. One modification that we made to algorithm 3 compared to theorem 1 of [37] is that at the root node, we calculate the value of $\delta(b_0)$. This is to reduce the overfitting of values to specific beliefs. In our algorithm, if two beliefs are very similar but result in vastly different rewards, that should be considered in calculating the value of b_0. In AOR this happens when the classifier is uncertain about some examples then their beliefs are close to each other and this should be reflected in their calculated values. Figure 6.1 demonstrates the belief tree search .

Theorem 1 provides guarantee on the optimality of the values found for images in algorithm (3).

Algorithm 3. Belief Tree Search

Belief Tree Search(Belief b_0, radius δ, max height h)
for All Images o_i in training set **do**
  $b_i \leftarrow$ Classify(o_i)
  if $b_i \in \delta(b_0)$ **then**
    Expand($o_i, b_i, 0$)
  end if
end for
Expand(Image o, Belief b, level i)
if $i = h$ **then return**
end if
for action $a \in \mathscr{A}$ **do**
  Image $o_a \leftarrow$ Simulate-Action(o, a)
  Belief $b_a \leftarrow$ Transition(b, a, o_a)
  if $b_a \in \delta$-neighborhood of any b' belief at level $i+1$ **then**
    $V(b_a) \leftarrow V(b')$
  else
    Add b_a to nodes in level $i+1$
    Expand($o_a, b_a, i+1$)
  end if
  add (o_a, b_a) to children of b
end forreturn

Theorem 1. For a given maximum error ε, and the optimal value function $V^*(b)$, algorithm (3) finds the value of a given belief b_0 such that $|V^*(b_0) - V(b_0)| \leq \varepsilon$ by setting the parameter values as,

$$\delta = \frac{\varepsilon(1-\gamma)^2}{2R_{max}} \tag{6.6}$$

$$h = \log_\gamma \frac{(1-\gamma)\varepsilon}{2R_{max}} \tag{6.7}$$

Proof. See the supplementary materials.

6.4.2 Optimizing Observation Function

In POMDP problems, usually it is assumed that the observation function is given as part of the environment. In AOR POMDP, the observation function is usually modeled using a generative model of objects. For example Borotsching et. al use Gaussian mixture on the eigenspace to model the likelihood of images from the view sphere of object under different classes [9]. Calculating the observation function value for an image usually requires feature extraction from the image and density estimation for different classes.

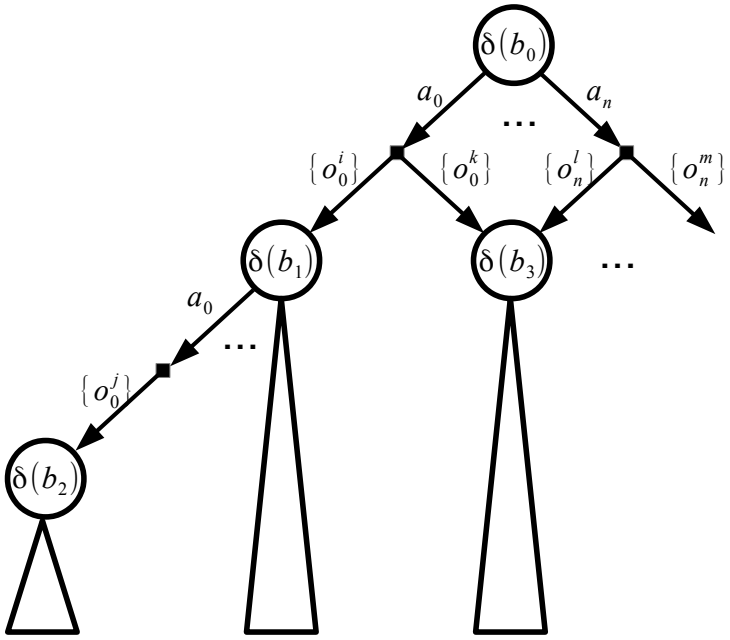


Figure 6.1. Belief Tree Search algorithm. Each node in the tree represents the value of δ-neighborhood of a belief with some error. Different observations may lead to visiting the same belief after taking an action.

We assume an observation function parameterized by ϕ,

$$\mathscr{P}(s,a,o;\phi) = Pr(o|s,a;\phi) \tag{6.8}$$

where ϕ is usually learned using maximum likelihood estimation. Different values of parameters ϕ changes the observation function. One may *improve* the observation function by using a different estimation of ϕ, which changes the feature extraction or density estimation. The improved observation function then results in a POMDP with different environment dynamics. In the ideal case, the observation function for an image is 1 for the correct object label and 0 for other labels, in which case the POMDP reduces to a trivial MDP.

For a given observation function, theorem 1 finds the image values for policies arbitrarily close to the optimal policy. However these values depend on the POMDP

dynamics, e.g. the observation function. We propose to improve the observation function by increasing the total reward collected by the near-optimal policy. For any policy π and observation function $\mathscr{P}$, the total reward ρ is defined as,

$$\begin{aligned}\rho(\pi,\mathscr{P}) &= E\left\{\sum_{t=1}^{\infty}\gamma^t r_t|s_o,\pi\right\}\\ &= \int_{b\in\mathscr{B}} d^{\mathscr{P},\pi}(b)\sum_{a\in\mathscr{A}}\pi(b,a)R(b,a)db\end{aligned} \tag{6.9}$$

Where $\mathscr{B}$ is the $|\mathscr{S}|$-dimensional belief simplex. Changing the observation function parameters may increase the likelihood of images under the correct object label. Theorem 2 presents a gradient ascent update rule to the parameters of observation function, with the goal of increasing the total reward.

Theorem 2. Given a policy π and the corresponding value function V^π the gradient of the total reward $\rho(\pi,\mathscr{P})$ with respect to the parameters of the observation function $\mathscr{P}(;\phi)$ is given by,

$$\begin{aligned}&\frac{\partial}{\partial\phi}\rho(\mathscr{P},\pi)\\ &= \int_b d^{\mathscr{P},\pi}(b)\sum_a\pi(b,a)\int_{b'}\sum_{o\in\Omega(b,a,b')}\sum_{s,s'\in\mathscr{S}}\\ &\quad b(s)T(s,a,s')V^\pi(b')\frac{\partial}{\partial\phi}\mathscr{P}(o|s',a)\mathrm{d}b'\mathrm{d}b\end{aligned} \tag{6.10}$$

Proof. See the supplementary notes.

Intuitively speaking, the update rule in (6.10) weights each parameter by the value of the belief that is reached by observing the corresponding o. Changing the observation function parameters ϕ changes the belief MDP dynamics as the transition probabilities in (6.2) depend on $\mathscr{P}$. After updating the observation function, a new belief MDP is reached, for which we can use theorem 1 to find near-optimal values of images. In

practice evaluation of (6.10) is computationally intractable. In the results section, we use a sampling approach for updating the observation function based on the values of images.

6.5 Results

6.5.1 Calculating BTS Action-Values

In this section, we implement the proposed method in algorithm (3) for active recognition of GERMS [45]. This is a medium size dataset with $\sim 120K$ images of 136 different object collected by a robot. The robot grabs each object with 10 different orientations and examines the object by rotating them in front of the camera. The goal is to recognize object for 4 test orientations, given the other 6 in-hand orientations. The actions bring the robot gripper into one of 5 pre-defined positions that are evenly spaced in the rotation range (180 degrees) of the gripper. GERMS is proved to be a challenging dataset since separation of objects in this dataset requires extraction of fine-grained visual cues.

We extract visual features from GERMS images using ResNet deep CNN model [25]. A softmax layer is trained on top of these features to predict the object label. Then we convert train and test images into belief vectors, and train our AOR method in the belief space. We normalize the output of the softmax layer for each class to sum to 1 over all GERMS images and use that as the observation probability. This is to ensure that the deep CNN outputs the likelihood and to maintain the integrity of (6.2) and (6.5).

After calculating the likelihood of train images, we use algorithm 3 to obtain the value of the near-optimal policy for them. In order to use these values in planning, [37] proposes a sampling approach that repeatedly executes algorithm 3 for different simulations and augments the tree with newly discovered beliefs and finally uses the

action-values of the root of the resulting tree for planning. The proposed BTS algorithm is similar to the work in [37] in that we use belief vectors in δ-neighborhood of the root to run the simulations. We found the action-values of the root of the tree to be very effective for AOR.

After we extract the action-values for training images, we transfer the knowledge of these action-values to the test set using three different approaches. In the first and second approaches, we use *Neurally Fittred Q-learning* (NFQ) [65] and *Actor Critic* (AC) [62], guided by a probabilistic policy that uses the action-values from BTS. We show that guiding NFQ and AC results in slight improvement of average performance on the test set, compared to the plain version. In the third approach, we use an LSTM network to learn to predict the best action given the objects belief.

6.5.2 Guided Neurally Fitted Q-learning

Neurally Fitted Q-learning (NFQ) trains a neural network to predict the action-values using the reward signal from the environment [65]. This algorithm has been successfully applied to reinforcement learning benchmarks [65], Atari games [50] and active object recognition [45]. At the heart of this approach is an iterative update rule for the network parameters θ,

$$\theta_{t+1} \leftarrow \theta_t + \alpha \frac{\partial}{\partial \theta}(R_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t))^2 \tag{6.11}$$

where R_t is the reward at time step t, γ is the reward discount factor, α is the learning rate and $Q(s,a)$ is the network output denoting action-values for action a in state s. In the above, the gradient operator on the right hand applies only to $Q(s_t, a_t)$. It has been observed that the plain NFQ algorithm may fail to discover optimal policies for active object recognition [44]. Instead, we employ the n-step extension of this algorithm,

proposed in [46], in which the update rule in (6.11) is applied to action sequences of length n. The n-step NFQ speeds up learning by updating n action-values in each iteration, compared to a single action-value update in the original NFQ. All experiments reported here are obtained using 10-step rollouts.

We improve the performance of NFQ by applying the *importance sampling* framework for policy search [28]. The idea is to use an auxiliary policy $\pi_{\theta'}$ to acquire sequences of actions and states $\tau = \{s_0, a_0, s_1, a_1, \ldots\}$, and update the parameters of the target policy π_θ using these sequences. In order to obtain an unbiased estimate of the policy gradients, update terms are weighted by their importance,

$$\frac{P(\tau|\pi_\theta)}{P(\tau|\pi_{\theta'})} = \frac{\prod_j \pi_\theta(a_j|s_j)}{\prod_j \pi_{\theta'}(a_j|s_j)} \tag{6.12}$$

Where $P(\tau|\pi)$ is the probability of rollout sequence τ under policy π, and $\pi(a|s)$ is the probability of action a in state s under policy π. We implement the guided NFQ (GNFQ) by drawing sequences of actions from an stochastic policy acquired by performing softmax on BTS action-values. The gradients in (6.11) are then multiplied by their importance in (6.12) and applied to the network. Figure 6.2 shows the performance of NFQ and GNFQ on the GERMS test set. For both approaches, we show $1-\sigma$ interval of object recognition accuracy. We also report the performance of random policy (RND), where at each time step, a previously *unseen* view of the object is explored. We see that the plain NFQ algorithm fails to surpass the random policy, while GNFQ performs slightly better than random. The advantage of GNFQ over NFQ and RND is most significant over the first action, and it gradually fades over the next four actions. After the last action where the majority of evidence has been accumulated, all three methods reach the same label prediction accuracy.

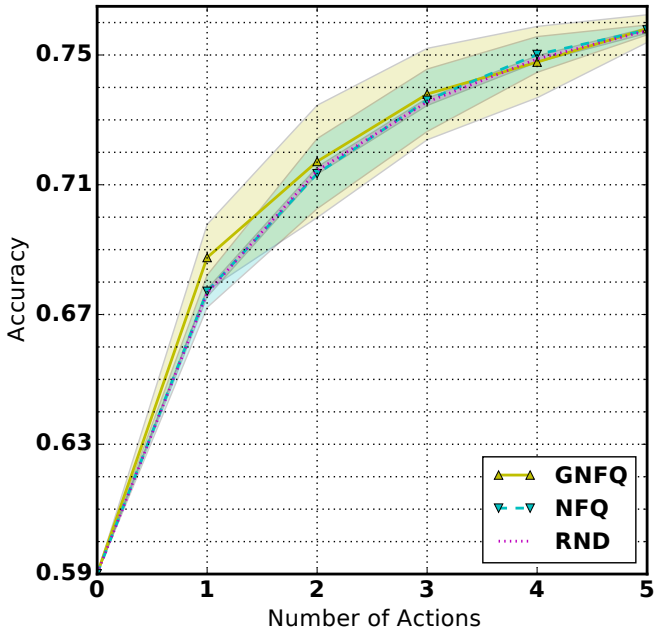


Figure 6.2. Comparison of AOR performance for NFQ, guided NFQ and random policies. The shaded areas show accuracy mean $\pm$ std.

6.5.3 Guided Actor-critic

Actor-critic is a policy learning method which updates the policy parameters using gradients of the total expected reward [62]. To reduce the variance of gradient estimation, a bias which is the prediction of the current state value, is decreased from the reward,

$$\theta_t \leftarrow \theta_t + \alpha \frac{\partial}{\partial \theta} \log \pi_\theta(a_t|s_t)(R_t - V_\psi(s_t)) \tag{6.13}$$

$$\psi_t \leftarrow \psi_t + \alpha \frac{\partial}{\partial \psi}(V_\psi(s_t) - R_t)^2 \tag{6.14}$$

where α is the learning rate, R_t is the reward at time t, $V_\psi(s_t)$ is the predicted value for s_t parameterized by ψ, and $\pi_\theta(a_t|s_t)$ is the probability that policy π_θ, parameterized by

θ, assigns to action a_t in state s_t. In figure 6.3, we show the performance of 10-step actor-critic method with (ACG) and without (AC) guiding. We used the same guiding scheme as described above, by multiplying the gradient terms of (6.13) and (6.14) by their importance (6.12). We see that the plain AC fails to perform better than random, while ACG shows slight performance improvements over RND after the second and third actions.

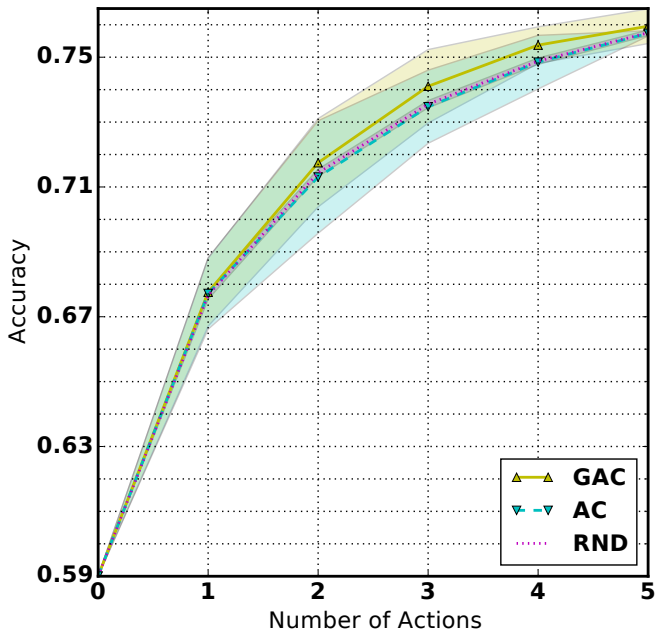


Figure 6.3. Comparison of AOR performance for actor-critic, guided actor-critic and random policies. The shaded areas show accuracy mean $\pm$ std.

6.5.4 Supervised Learning of Action-Values

To transfer the knowledge of extracted action-values from training to test set, we train a neural network to directly predict the best next action given a belief. We use a stack of 3 LSTM layers with 128 units in each layer, followed by a softmax layer that predicts action (see Figure 6.4). The training sequences are produced by following a probabilistic policy derived from BTS action-values. For each belief vector, the target

action is the action selected in the rollout by the BTS-driven policy. We found that using both rollouts and LSTM is crucial to the AOR performance in supervised action prediction. Figure 6.5 compares the performance of LSTM and random policies. LSTM has a clear advantage in performance over the random policy. Moreover, the variance of the learned policy is significantly smaller compared to actor-critic and NFQ methods. Figure 6.6 compares the average performance of all methods. We see that supervised learning for action-value prediction is clearly superior to the policy learning methods. Table 6.1 compares the performance of all methods in more details.

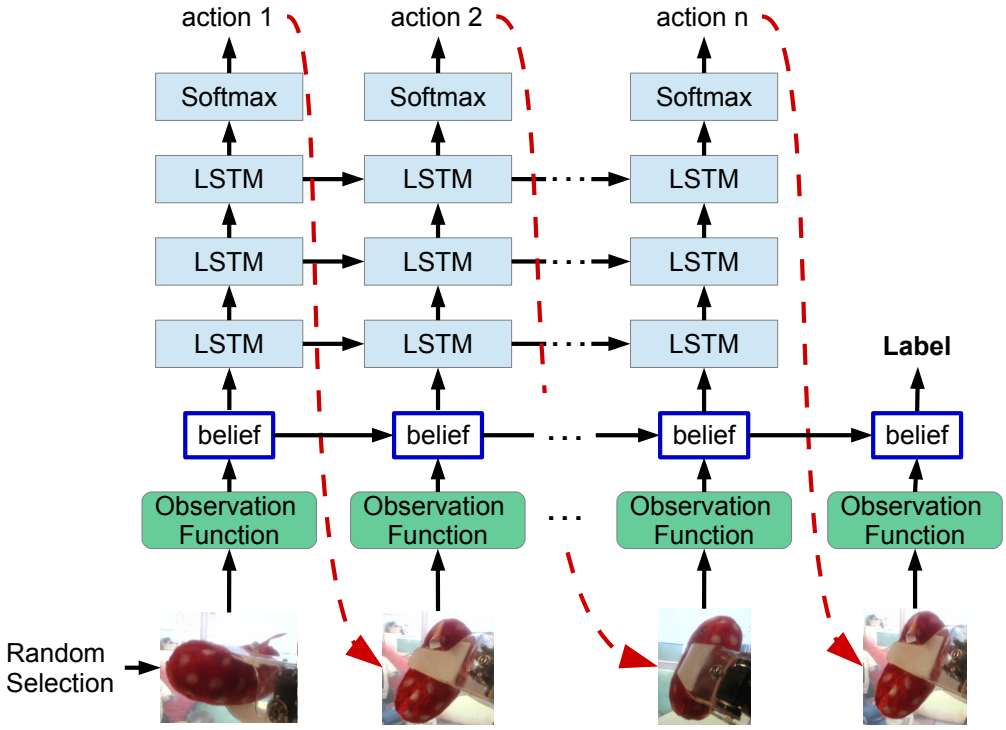


Figure 6.4. The proposed supervised learning method for action prediction using LSTM network. The observation function is fixed for training the LSTM layers.

6.5.5 Generalization to Novel Objects

In this section, we test the generalization of the proposed AOR method to novel object. The goal of this experiment is to understand how much object inspection knowl-

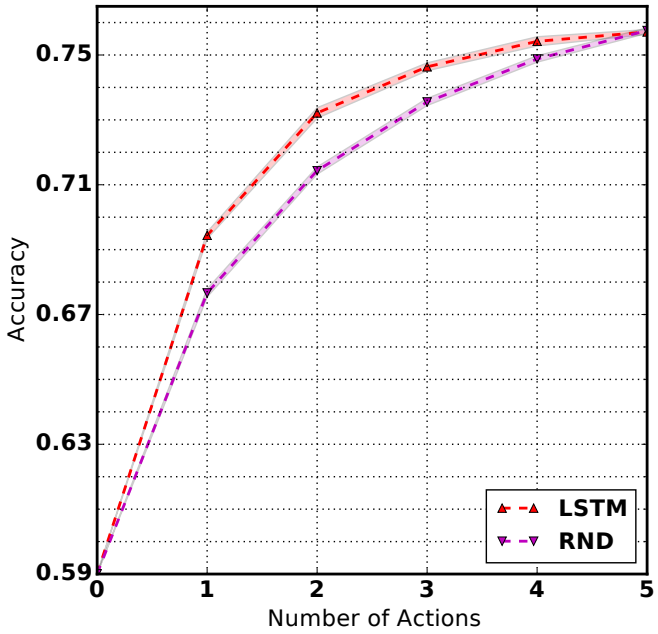


Figure 6.5. Comparison of AOR performance for LSTM and random methods. The shaded areas show accuracy mean $\pm$ std.

edge is transferrable between GERMS objects. For this purpose, we use 60% of objects in GERMS for supervised training of action prediction, and the rest of objects for testing. The results averaged over 20 different experiments are shown in figure 6.7. Overall, the variance of results is high because of the large variations in the recognition accuracy of GERMS objects. We see that the supervised LSTM method achieves significantly higher performance compared to the random strategy.

6.5.6 Improving Observation Function

We retrain the observation function using the proposed gradient update rule in (6.10) to increase the AOR total collected reward. In order to implement this update rule, we adapt a sampling strategy by generating a set of rollouts using policy π derived from BTS action-values. Let b_i denote the belief vector corresponding to image I_i. Let $\{b_j, a_j\}, j = 1, 2, \ldots, n$ denote the set of beliefs and actions that resulted in b_i in the

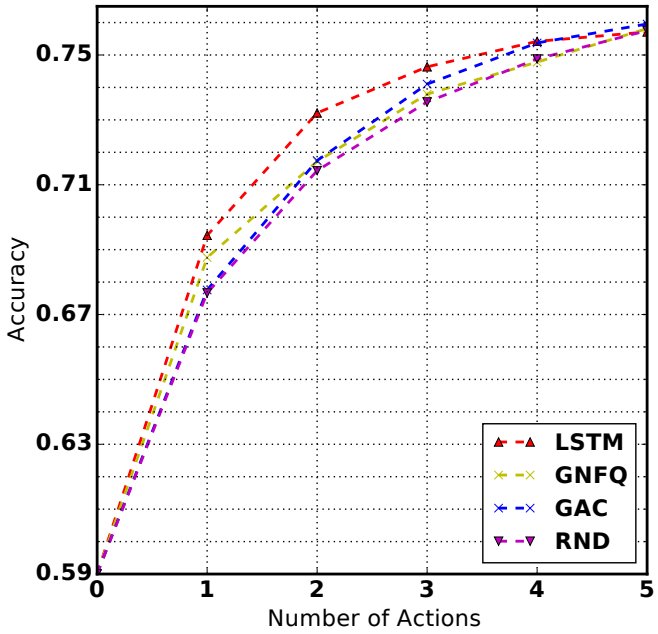


Figure 6.6. Comparison of AOR performance for LSTM, AC, NFQ and RND policies.

rollouts. The retraining weight of I_i is then calculated using a sample average of (6.10) on these rollouts,

$$Weight(b_i) \propto \frac{1}{n} \sum_{j=1}^{n} \pi(b_j, a_j) b_j(s) V^{\pi}(b_i) \tag{6.15}$$

where V^{π} values are calculated using algorithm 3, $b(s)$ denotes the entry corresponding to object label s in belief vector b, and $\pi(b,a)$ is the probability assigned to action a for belief b by the BTS-driven policy. We retrain the softmax over visual features, by weighting the cross entropy cost of each image by (6.15). After retraining, we run the BTS algorithm on the resulting beliefs and train the LSTM model on the resulting action-values. We show the performance of the retrained LSTM as *LSTM-i2* in figure 6.8. The retrained LSTM achieves significantly higher accuracy compared to the original LSTM. We repeat this procedure and observe slight improvements, shown as *LSTM-i3* in figure 6.8. In practice, the performance of the LSTM starts to decline after the second

Table 6.1. AOR performance comparison on the GERMS test data based on the number of actions.

Method	0	1	2	3	4	5
RND	0.590	0.677	0.714	0.736	0.749	0.758
AC	0.590	0.677	0.713	0.735	0.748	0.757
ACG	0.590	0.678	0.717	0.741	0.754	0.760
NFQ	0.590	0.677	0.713	0.736	0.750	0.758
NFQG	0.590	0.688	0.717	0.738	0.748	0.758
LSTM	0.590	0.694	0.732	0.746	0.754	0.757
LSTM-i2	0.614	0.715	0.751	0.769	0.778	0.785
LSTM-i3	**0.617**	**0.718**	**0.758**	**0.776**	**0.790**	**0.793**

retraining. The performance of the retrained models are shown in more details in table 6.1.

6.6 Discussion

Active object recognition has received little attention from mainstream computer vision and machine learning communities despite its potential for improving recognition performance. Progress on AOR has been slow due to reliance of the AOR models on semi supervised or heuristic methods for learning object inspection policy. In this work, we proposed a method that learns the object exploration policy in a supervised manner from the training data. The proposed method has desirable properties, for example it provides guarantee of optimality of calculated values on training set, it is very fast at the test time and generalizes well to novel objects and novel views of familiar objects since it does not require a generative model of objects.

Recently there has been a renewed interest in attention models mostly for reasons beyond object recognition. These models rely on reinforcement learning [48] or variation inference [4] to guide the system to discover suitable exploration policies. In contrast, our method benefits from reduced training time, which is a large problem in these approaches.

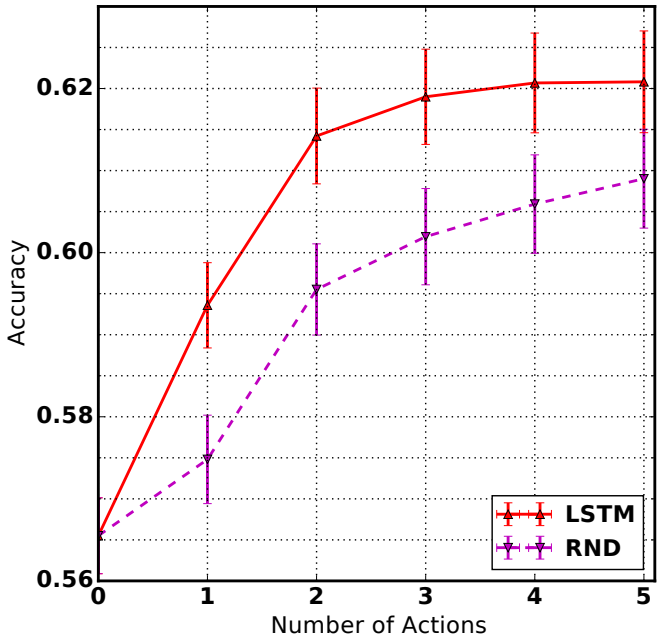


Figure 6.7. Generalization of object inspection policies of LSTM model to novel objects. The reported numbers are mean± standard error.

By reducing the optimal policy inference to a supervised learning problem, we can use recent advances in supervised visual recognition for learning policies and knowledge transfer to test data.

We developed a weighting scheme for training classification of single images, that puts emphasize on images with higher values during exploration. The weight of each image is a measure of how useful it is in inferring the correct label of object. Such weighting scheme potentially reduces overfitting of the image classification model to the training data that have little or no discriminative information. Because of the complex background in GERMS images and blocking of in-hand object by the robot gripper, there is high chance of overfitting to background cues during training. By performing BTS, the AOR has the opportunity to look ahead a few steps in active inspection of objects. Image values are then used to guide the single image classification model to discover more discriminative features.

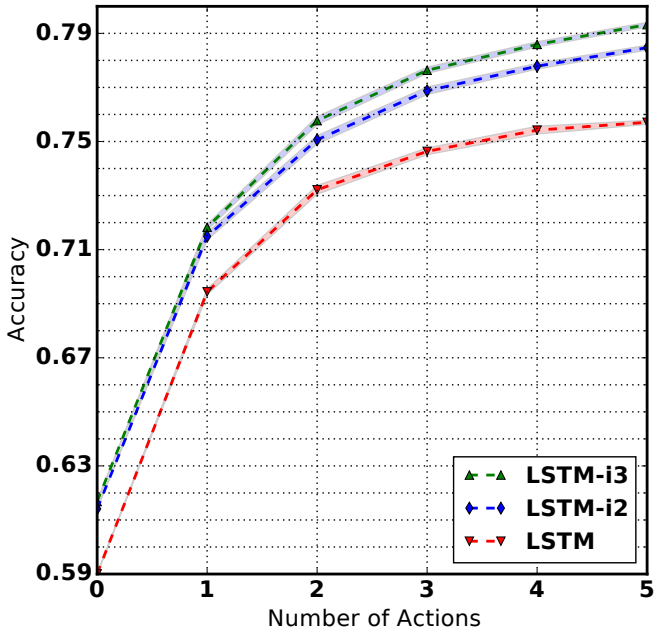


Figure 6.8. Comparison of the accuracy of the retrained LSTM models, denoted as LSTM-iteration2 (LSTM-i2) and LSTM-iteration3 (LSTM-i3) with the original model.

6.7 Appendix

6.7.1 Proof of Theorem 1

Theorem 1. Let $V^*(b)$ be the optimal value function for the POMDP AOR on $\Re(b)$. For a given maximum error $\varepsilon \geq 0$, algorithm 1 finds the value b such that $|V^*(b) - V(b)| \leq \varepsilon$ by setting the maximum height h and partitioning radius δ to

$$h = \log_\gamma \frac{(1-\gamma)\varepsilon}{2R_{max}} \tag{6.16}$$

$$\delta = \frac{\varepsilon(1-\gamma)^2}{2R_{max}} \tag{6.17}$$

where R_{max} is the maximum reward and γ is the discount factor in AOR POMDP.

Proof. We use similar steps to the proof of theorem 1 in [37]. The main difference in our proof is that the error at the root comes from approximation of policy and δ-packing of

beliefs. Let $\varepsilon'_i = |V^*(b') - V(b)|$ be the approximation error for belief b at level i of the tree where We assign beliefs b and b' the same value through δ-packing and where V^* is the optimal value function and V is the approximation. This error can be divided into two parts,

$$\begin{aligned}
\varepsilon'_h &= |V^*(b') - V(b)| \\
&\leq |V^*(b') - V^*(b) + V^*(b) - V(b)| \\
&\leq |V^*(b') - V^*(b)| + |V^*(b) - V(b)| \\
&\leq \frac{R_{max}}{1-\gamma}\delta + |V^*(b) - V(b)| \qquad (6.18)
\end{aligned}$$

We used lemma 1 of [37] in the last step above for an upper bound on the difference of optimal value function for b and b'. Let $\varepsilon_i = |V^*(b) - V(b)|$, then the error at root, assuming the height of tree is h, is,

$$\varepsilon'_h \leq \frac{R_{max}}{1-\gamma}\delta + \varepsilon_h \qquad (6.19)$$

Let ε'_{i-1} be the largest error of children of a node at level i. We have,

$$\varepsilon_i \leq \gamma\varepsilon'_{i-1} \qquad (6.20)$$

Then we can complete the recursion in (6.18), starting from error at the root of tree

$$\begin{aligned}\varepsilon'_h &\leq \frac{R_{max}}{1-\gamma}\delta + \varepsilon_h \\ &\leq \frac{R_{max}}{1-\gamma}\delta + \gamma\varepsilon'_{h-1} \\ &\leq \frac{R_{max}}{1-\gamma}\delta + \gamma\{\frac{R_{max}}{1-\gamma}\delta + \varepsilon_{h-1}\}\end{aligned} \tag{6.21}$$

after several expansion of ε_{h-1} in the above,

$$\varepsilon'_h \leq \frac{R_{max}}{1-\gamma}\delta + \frac{R_{max}\gamma}{(1-\gamma)^2}\delta + \gamma^h\frac{R_{max}}{1-\gamma} \tag{6.22}$$

If we set,

$$h = \log_\gamma\left\{\frac{1-\gamma}{R_{max}}\frac{\varepsilon}{2}\right\} \tag{6.23}$$

$$\delta = \frac{(1-\gamma)^2}{R_{max}}\frac{\varepsilon}{2} \tag{6.24}$$

we get the desired result.

6.7.2 Proof of Theorem 2

Theorem 2. Given a policy π and the corresponding value function V^π, the gradient of the total reward $\rho(\pi, \mathscr{P})$ with respect to the parameters of the observation

function $\mathscr{P}(;\phi)$ is given by,

$$\begin{aligned}
&\frac{\partial}{\partial\phi}\rho(\mathscr{P},\pi)\\
&=\int_b d^{\mathscr{P},\pi}(b)\sum_a \pi(b,a)\int_{b'}\sum_{o\in\Omega(b,a,b')}\sum_{s,s'\in\mathscr{S}}\\
&\quad b(s)T(s,a,s')V^{\pi}(b')\frac{\partial}{\partial\phi}\mathscr{P}(o|s',a)\mathrm{d}b'\mathrm{d}b
\end{aligned} \tag{6.25}$$

Proof. We extend theorem 1 in [78] to the POMDP case. For a given observation function, algorithm 1 approximates the image value for a policy that is arbitrarily close to the optimal policy. However these values depend on the POMDP dynamics, that is, the observation function. Once these values are extracted, we can improve the observation function with the goal of increasing the total reward collected by the optimal policy. For any policy $\pi(b,a)$ and observation function $\mathscr{P}$, the total reward ρ is defined as,

$$\begin{aligned}
\rho(\pi,\mathscr{P}) &= E\left\{\sum_{t=1}^{\infty}\gamma^t r_t|s_o,\pi,\mathscr{P}\right\}\\
&=\int_{b\in\mathscr{B}} d^{\mathscr{P},\pi}(b)\sum_{a\in\mathscr{A}}\pi(b,a)R(b,a)db
\end{aligned} \tag{6.26}$$

Where $\mathscr{B}$ is the $|\mathscr{S}|$-dimensional belief simplex and $d^{\mathscr{P},\pi}(b)$ is the stationary distribution of observing belief b in the belief MDP. Note that this stationary distribution depends on the POMDP observation function and the policy π. Theorem 2 presents a gradient ascent update rule to the parameters of observation function with the goal of increasing this total reward.

Note that we cannot directly apply the gradient operator to 6.26 because $d^{\pi,\mathscr{P}}(b)$

depends on $\mathscr{P}$. We start by writing the equation for the value function,

$$V^{\pi,\mathscr{P}}(b) = \sum_a \pi(b,a) Q^{\pi,\mathscr{P}}(b,a) \tag{6.27}$$

$$= \sum_a \pi(b,a) \left\{ R(b,a) - \rho(\pi,\mathscr{P}) + \int_{b'} Tr(b,a,b') V^{\pi,\mathscr{P}}(b') db' \right\} \tag{6.28}$$

If we take the gradient of the above w.r.t to the paramters ϕ of the observation function $\mathscr{P}$, we have,

$$\frac{\partial}{\partial \phi} V^{\pi,\mathscr{P}}(b) = -\sum_a \pi(b,a) \frac{\partial}{\partial \phi} \rho(\pi,\mathscr{P}) \tag{6.29}$$

$$+ \sum_a \pi(b,a) \int_{b'} \sum_{o \in \Omega(b,a,b')} \sum_{s,s'} [\frac{\partial}{\partial \phi} V^{\pi,\mathscr{P}}(b') \mathscr{P}(o|s',a) \tag{6.30}$$

$$+ \frac{\partial}{\partial \phi} \mathscr{P}(o|s',a) . V^{\pi,\mathscr{P}}(b')] db' \tag{6.31}$$

$$= -\frac{\partial}{\partial \phi} \rho(\pi,\mathscr{P}) \tag{6.32}$$

$$+ \sum_a \pi(b,a) \int_{b'} \sum_{o \in \Omega(b,a,b')} \sum_{s,s'} V^{\pi,\mathscr{P}}(b') \frac{\partial}{\partial \phi} \mathscr{P}(o|s',a) db' \tag{6.33}$$

$$+ \sum_a \pi(b,a) \int_{b'} \sum_{o \in \Omega(b,a,b')} \sum_{s,s'} \mathscr{P}(o|s',a) \frac{\partial}{\partial \phi} V^{\pi,\mathscr{P}}(b') db' \tag{6.34}$$

Now if we take the expectation of the above w.r.t b, we get,

$$\int_b d^{\pi,\mathscr{P}}(b)\frac{\partial}{\partial\phi}V^{\pi,\mathscr{P}}(b)db = \tag{6.35}$$

$$\begin{aligned}
&-\int_b d^{\pi,\mathscr{P}}(b)\frac{\partial}{\partial\phi}\rho(\pi,\mathscr{P})db \\
&+\int_b d^{\pi,\mathscr{P}}(b)\sum_a \pi(b,a)\int_{b'}\sum_{o\in\Omega(b,a,b')}\sum_{s,s'} V^{\pi,\mathscr{P}}(b')\frac{\partial}{\partial\phi}\mathscr{P}(o|s',a)db'db \\
&+\int_b d^{\pi,\mathscr{P}}(b)\sum_a \pi(b,a)\int_{b'}\sum_{o\in\Omega(b,a,b')}\sum_{s,s'} \mathscr{P}(o|s',a)\frac{\partial}{\partial\phi}V^{\pi,\mathscr{P}}(b')db'db
\end{aligned} \tag{6.36}$$

However note that,

$$d^{\pi,\mathscr{P}}(b') = \int_b d^{\pi,\mathscr{P}}(b)\sum_a \pi(b,a)\sum_{o\in\Omega(b,a,b')}\sum_{s,s'}\mathscr{P}(o|s',a). \tag{6.37}$$

As a result, the last term on the right side of 6.35 cancels with the term on the left side, and the proof of theorem1. concludes by noticing that $\rho(\pi,\mathscr{P})$ is independent of the initial b.

6.8 Acknowledgements

The research presented here was funded by NSF IIS 0968573 SoCS, IIS INT2-Large 0808767, and NSF SBE-0542013 and in part by US NSF ACI-1541349 and OCI-1246396, the University of California Office of the President, and the California Institute for Telecommunications and Information Technology (Calit2).

Chapter 6, in full, is a reprint of the material as it submitted to IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2017. Malmir, Mohsen; Cottrell, Garrison W.; The dissertation author was the primary investigator and author of this paper.

Chapter 7

Conclusion

7.1 Perceptual Primitives for RUBI

Advances in machine perception technologies are providing social robots with perceptual primitives that can support sophisticated forms of HRI. In this thesis, we developed perceptual primitives for RUBI, a social robot in early childhood education. We developed a face recognition model for RUBI using classic computer vision pipeline. We demonstrated example applications of face recognition for RUBI in an Early Childhood Education Center (ECEC) classroom at UCSD. Furthermore, we developed active object recognition models for RUBI that enables her to teach object names to the children in the classroom.

In the first part of this thesis, we developed a supervised face recognition model for RUBI, to enrich the analysis of social structure of the classroom. We created a dataset of faces of toddlers and adults using images collected by RUBI in the classroom. The dataset contains face images taken over a period of 6 months, and exhibits a variety of face poses and facial expressions which make face recognition in the classroom difficult. To make the face recognition robust to these variations, we trained the classifier on the densely extracted features from the Gaussian pyramid of face images. We presented RUBI-gram by analyzing the social structure of the classroom using the face recognition

model, and the data from the "home-alone" study. RUBI-gram summarizes the children's preference for playmates during the time they play in front of RUBI.

We presented the results of facial expression recognition in the classroom, and predicted the children's preference for different activities and games using the "joy" channel from the facial expression recognition tool. Face recognition and facial expression recognition enable RUBI to present the results of the data analyses, RUBI-gram, preferred games and typical facial expressions to the teachers on a periodic base, to keep track of the children's social, affective and cognitive state.

We developed models for active object recognition for RUBI to enable her to teach object names to the children during give-and-take. In this game, a toddler hands in an object to RUBI, after which she examines the object and says it name. This game is an opportunity to teach object naming to the children, and compare learning from 3D in-hand objects to the on-screen 2D images of objects. By holding an object in her gripper, RUBI has the opportunity to actively improve the object recognition accuracy by examining the object from different views.

In chapter 4, we developed a model for active in-hand object recognition using deep Q-learning. While the literature on active object recognition has shown promising results, progress has been slow due to the lack of realistic datasets and benchmarks that can be easily shared by multiple research groups. We introduced the GERMS dataset that includes a collection of videos, a set of active object recognition benchmarks and baseline results on those benchmarks. We hope that this dataset will facilitate the comparison of different active object recognition methods and accelerate progress in the field. We also trained a model for active object recognition using deep Q-learning. We encoded the state of the system as the belief over different object labels. This representation was then processed by an additional deep network trained using Q-learning for efficient action selection. The proposed approach outperforms sequential and random action selection

policies and serves as baseline for future comparisons. We showed that training the system with different lengths of game plays did not have an impact on the performance of trained model.

In chapter 5, we trained a deep neural network for joint object classification and action prediction. The model is trained end-to-end, by minimizing the action and label prediction errors. The visual features in early stages of this network are learned by minimizing both errors simultaneously. The joint cost minimization allows the model to learn visual features that are suitable for predicting object label and the action to be performed on the object to improve the recognition performance.

We proposed a Dirichlet-based encoding of the state of the system as an alternative to the common Naive Bayes approach. Naive Bayes belief update leads to overfitting the action prediction to subsets of training images for which classification is confident. We used a generative model based on Dirichlet distribution for each object-action pair, to encode the beliefs observed while examining the object. This model was added to the network as a layer before the action selection layer, to enable training the network end-to-end for label and action-value prediction. The results of experiments confirmed that the proposed Dirichlet model is superior in test label prediction accuracy to the Naive Bayes approach.

We discovered a few challenges in training active object recognition systems using deep Q-learning. First, there is no guarantee on the performance of discovered policies in deep Q-learning, even on the training set. Second, semi-supervised learning of policies may fail to converge to a suitable solution due to numerous local optimums. We observed that deep Q-learning converges to the grid search policy for all objects, while intuitively we expect a "good" policy to prefer different actions for different objects. Another problem is the difference between the distribution of beliefs in the training set and the test set. The training beliefs are usually peaked on the correct object label, while

the test beliefs are more flat.

In chapter 6, we address these challenges by formulating active object recognition as a Partially Observable Markov Decision Process (POMDP) and calculating the near-optimal values on the training images using Belief Tree search. We used leave-one-out strategy in training the observation function, in order to increase the similarity of training and test beliefs. We adapted a Belief Tree search method to calculate near-optimal values of beliefs in the training set, and used the corresponding action values to train active object recognition policy. The proposed method learns the object exploration policy in a supervised manner from the training data, it provides guarantee of optimality of calculated values on training set, it is very fast at the test time and generalizes well to novel objects and novel views of familiar objects since it does not require a generative model of objects.

We developed a weighting scheme for training classification of single images, that emphasizes the more useful images for active recognition. This weighting scheme potentially guides the training of image classification model to learn from more discriminative images. Because of the complex background in GERMS images and blocking of in-hand object by the robot gripper, there is high chance of overfitting to background cues during training. By performing Belief Tree search, active object recognition has the opportunity to look ahead a few steps in inspection of objects to determine the value of images. These values are then used to guide the single image classification model to discover more discriminative features.

7.2 Future Work

7.2.1 Semi-supervised Person Recognition

The next step for enabling RUBI to recognize people in the classroom is to develop algorithms that can recognize faces, including detecting the presence of new faces, with very little supervision. The learning is done mostly with unlabeled or weakly-labeled data, where RUBI is able to ask the teachers for the labels of particular faces. The development of a method for RUBI to learn faces without hand-labeling hundreds of frames will make it practical for RUBI to learn the childrens appearances, and to learn their names, with just a small amount of input from their teachers. She can then record the performance of children and track their social interactions. The system will also be able to flexibly add identities, as new toddlers join the classroom.

RUBI typically records many hours of video, labeling these frames is a time-consuming and expensive process. The Next step is to use semi-supervised and active approaches to train classifiers for person recognition by leveraging this large amount of data. Temporal properties of the collected data provides additional constraints that can be used in semi-supervise learning of people, for example two faces that appear in the same frame belong to different people. Additional cues such as clothing or gait can also be used to help recognize people.

7.2.2 Active Object Recognition using Shift of Attention

We can extend the set of actions in active object recognition to include attending to specific regions of image. This has the benefit of reducing the input size to the recognition module, by selecting a small region of interest, while being able to attend to visual features that are important for fine-grained visual categorization. The proposed framework in chapter 6 can be used to extract near-optimal policies, that can be learned

in a supervised manner.

7.2.3 Agent RUBI: social perceptual primitives in the wild

RUBI provides the opportunity to collect data on human social interactions and culture from the real world outside of controlled lab environments. This data will enable us to understand in-the-wild human-robot interactions. Consider a scenario in which RUBI is placed in the hallway of the library, to freely interact with people. In this scenario, RUBI is a "social agent" with the goal of collecting social behavior data. As people gather around RUBI to interact with her, we can analyze their social posture and behavior and improve RUBI's social behavior by learning from these data.

In order for RUBI to engage in social interaction with people, we must provide a set of perceptual primitives for recognizing the social behavior, such as posture, gaze and speech. We can extend the sensory repertoire of RUBI to include Kinect®, which simplifies extraction of human skeleton. categorizing social behavior then requires fitting temporal classification models to the observed data.

Bibliography

[1] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[2] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[3] Nikolay Atanasov, Bharath Sankaran, Jerome Le Ny, George J Pappas, and Kostas Daniilidis. Nonmyopic view planning for active object classification and pose estimation. *Robotics, IEEE Transactions on*, 30(5):1078–1090, 2014.

[4] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

[5] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[6] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[7] Marian Stewart Bartlett, Gwen Littlewort, Claudia Lainscsek, Ian Fasel, and Javier Movellan. Machine learning methods for fully automatic recognition of facial expressions and facial actions. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 1, pages 592–597. IEEE, 2004.

[8] Hermann Borotschnig, Lucas Paletta, Manfred Prantl, and Axel Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[9] Hermann Borotschnig, Lucas Paletta, Manfred Prantl, and Axel Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[10] Björn Browatzki, Vadim Tikhanoff, Giorgio Metta, Heinrich H Bülthoff, and Christian Wallraven. Active object recognition on a humanoid robot. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2021–2028. IEEE, 2012.

[11] Bjorn Browatzki, Vadim Tikhanoff, Giorgio Metta, Heinrich H Bulthoff, and Christian Wallraven. Active in-hand object recognition on a humanoid robot. *Robotics, IEEE Transactions on*, 30(5):1260–1269, 2014.

[12] Bjorn Browatzki, Vadim Tikhanoff, Giorgio Metta, Heinrich H Bulthoff, and Christian Wallraven. Active in-hand object recognition on a humanoid robot. *Robotics, IEEE Transactions on*, 30(5):1260–1269, 2014.

[13] Nicholas J Butko, Lingyun Zhang, Garrison W Cottrell, and Javier R Movellan. Visual saliency model for robot cameras. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2398–2403. IEEE, 2008.

[14] Francesco G Callari and Frank P Ferrie. Active object recognition: Looking for differences. *International Journal of Computer Vision*, 43(3):189–204, 2001.

[15] Francesco G Callari and Frank P Ferrie. Active object recognition: Looking for differences. *International Journal of Computer Vision*, 43(3):189–204, 2001.

[16] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[17] L Chen-Yu, X Saining, G Patrick, Z Zhengyou, and T Zhuowen. Deeply-supervised nets. *CoRR, abs/1409.5185*, 3(4):93, 2014.

[18] Joachim Denzler, Christopher M Brown, and Heinrich Niemann. Optimal camera parameter selection for state estimation with applications in object recognition. In *Pattern Recognition*, pages 305–312. Springer, 2001.

[19] Bret Fortenberry, Joel Chenu, and JR Movellan. Rubi: A robotic platform for real-time social interaction. In *Proceedings of the International Conference on Development and Learning (ICDL04), The Salk Institute, San Diego*, 2004.

[20] Linton C Freeman. Visualizing social networks. *Journal of social structure*, 1(1):4, 2000.

[21] Goren Gordon and Cynthia Breazeal. Bayesian active learning-based robot tutor for children's word-reading skills. In *AAAI*, pages 1343–1349, 2015.

[22] Jeong-Hye Han, Mi-Heon Jo, Vicki Jones, and Jun-H Jo. Comparative study on the educational use of home robots for children. *Journal of Information Processing Systems*, 4(4):159–168, 2008.

[23] Jeonghye Han and Dongho Kim. r-learning services for elementary school students with a teaching assistant robot. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 255–256. ACM, 2009.

[24] Albert Haque, Alexandre Alahi, and Li Fei-Fei. Recurrent attention models for depth-based person identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[26] Andri Ioannou, Emily Andreou, and Maria Christofi. Pre-schoolers interest and caring behaviour around a humanoid robot. *TechTrends*, 59(2):23–26, 2015.

[27] Dinesh Jayaraman and Kristen Grauman. Look-ahead before you leap: end-to-end active recognition by forecasting the effect of motion. In *European Conference on Computer Vision*, pages 489–505. Springer, 2016.

[28] Tang Jie and Pieter Abbeel. On a connection between importance sampling and the likelihood ratio policy gradient. In *Advances in Neural Information Processing Systems*, pages 1000–1008, 2010.

[29] Edward Johns, Stefan Leutenegger, and Andrew J Davison. Pairwise decomposition of image sequences for active multi-view recognition. *arXiv preprint arXiv:1605.08359*, 2016.

[30] Daniel Johnson, Mohsen Malmir, Deborah Forster, Morana Alac, and Javier Movellan. Design and early evaluation of the rubi-5 sociable robots. In *Development and Learning and Epigenetic Robotics (ICDL), 2012 IEEE International Conference on*, pages 1–2. IEEE, 2012.

[31] Takayuki Kanda, Takayuki Hirano, Daniel Eaton, and Hiroshi Ishiguro. Interactive robots as social partners and peer tutors for children: A field trial. *Human-computer interaction*, 19(1):61–84, 2004.

[32] Takayuki Kanda, Rumi Sato, Naoki Saiwaki, and Hiroshi Ishiguro. A two-month field trial in an elementary school for long-term human–robot interaction. *IEEE Transactions on robotics*, 23(5):962–971, 2007.

[33] James Kennedy, Paul Baxter, and Tony Belpaeme. Comparing robot embodiments in a guided discovery learning interaction with children. *International Journal of Social Robotics*, 7(2):293–308, 2015.

[34] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[35] Tsuyoshi Komatsubara, Masahiro Shiomi, Takayuki Kanda, Hiroshi Ishiguro, and Norihiro Hagita. Can a social robot help children's understanding of science in classrooms? In *Proceedings of the second international conference on Human-agent interaction*, pages 83–90. ACM, 2014.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[37] Wee S Lee, Nan Rong, and Daniel J Hsu. What makes some pomdp problems easy to approximate? In *Advances in neural information processing systems*, pages 689–696, 2007.

[38] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *arXiv preprint arXiv:1504.00702*, 2015.

[39] G. Littlewort, M. Bartlett, J. Chenu, I. Fasel, T. Kanda, H. Ishiguro, and J. Movellan. Towards social robots: Automatic evaluation of human-robot interaction by face detection and expression classification. *Advances in Neural Information Processing Systems*, 16:1563–1570, 2004.

[40] Gwen Littlewort, Marian Stewart Bartlett, Ian Fasel, Joshua Susskind, and Javier Movellan. Dynamics of facial expression extracted automatically from video. *Image and Vision Computing*, 24(6):615–625, 2006.

[41] Gwen Littlewort, Jacob Whitehill, Tingfan Wu, Ian Fasel, Mark Frank, Javier Movellan, and Marian Bartlett. The computer expression recognition toolbox (cert). In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 298–305. IEEE, 2011.

[42] Rosemarijn Looije, Anna van der Zalm, Mark A Neerincx, and Robbert-Jan Beun. *Help, I need some body the effect of embodiment on playful learning*. IEEE, 2012.

[43] Mohsen Malmir, Deborah Forster, Kendall Youngstrom, Lydia Morrison, and Javier R Movellan. Home alone: Social robots for digital ethnography of toddler behavior. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 762–768. IEEE, 2013.

[44] Mohsen Malmir, Karan Sikka, Deborah Forster, Ian Fasel, Javier R. Movellan, and Garrison W. Cottrell. Deep active object recognition by joint label and action prediction. *Computer Vision and Image Understanding*, pages –, 2016.

[45] Mohsen Malmir, Karan Sikka, Deborah Forster, Javier Movellan, and Garrison W Cottrell. Deep q-learning for active recognition of germs: Baseline performance on a standardized dataset for active learning. In *Proceedings of the British Machine Vision Conference (BMVC), pages*, pages 161–1. BMVA, 2015.

[46] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016.

[47] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014.

[48] Volodymyr Mnih, Nicolas Heess, Alex Graves, and koray kavukcuoglu. Recurrent models of visual attention. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc., 2014.

[49] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[50] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[51] Javier Movellan, Micah Eckhardt, Marjo Virnes, and Angelica Rodriguez. Sociable robot improves toddler vocabulary skills. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 307–308. ACM, 2009.

[52] Javier R Movellan. An infomax controller for real time detection of social contingency. In *Development and Learning, 2005. Proceedings. The 4th International Conference on*, pages 19–24. IEEE, 2005.

[53] Javier R Movellan, Mohsen Malmir, and Deborah Forester. Hri as a tool to monitor socio-emotional development in early childhood education, 2012.

[54] Javier R Movellan, Fumihide Tanaka, Ian R Fasel, Cynthia Taylor, Paul Ruvolo, and Micah Eckhardt. The rubi project: a progress report. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 333–339. ACM, 2007.

[55] Javier R Movellan, Fumihide Tanaka, Bret Fortenberry, and Kazuki Aisaka. The rubi/qrio project: origins, principles, and first steps. In *Development and Learning, 2005. Proceedings. The 4th International Conference on*, pages 80–86. IEEE, 2005.

[56] JR Movellan, M Malmir, and D Forester. Hri as a tool to monitor socio-emotional development in early childhood education, in proc. In *HRI 2nd Workshop on Applications for Emotional Robots, Bielefeld, Germany*, 2014.

[57] Mohammad Naghshvar and Tara Javidi. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703–2738, 2013.

[58] Sameer A Nene, Nayar, and Hiroshi Murase. Columbia object image library (coil-100). Technical report, Technical Report CUCS-006-96, 1996.

[59] Lucas Paletta and Axel Pinz. Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1):71–86, 2000.

[60] Lucas Paletta and Axel Pinz. Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1):71–86, 2000.

[61] Timothy Patten, Wolfram Martens, and Robert Fitch. Monte carlo planning for active object classification. *Autonomous Robots*, pages 1–31, 2017.

[62] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

[63] Joelle Pineau, Geoff Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *IJCAI*, volume 3, pages 1025–1032, 2003.

[64] Antons Rebguns, Daniel Ford, and Ian R Fasel. Infomax control for acoustic exploration of objects by a mobile robot. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[65] Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer, 2005.

[66] Sumantra Dutta Roy, Santanu Chaudhury, and Subhashis Banerjee. Active recognition through next view planning: a survey. *Pattern Recognition*, 37(3):429–446, 2004.

[67] Paul Ruvolo, Ian Fasel, and Javier Movellan. Auditory mood detection for social and educational robots. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3551–3556. IEEE, 2008.

[68] Paul Ruvolo and Javier Movellan. Automatic cry detection in early childhood education settings. In *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on*, pages 204–208. IEEE, 2008.

[69] Paul Ruvolo, Jacob Whitehill, Marjo Virnes, and Javier Movellan. Building a more effective teaching robot using apprenticeship learning. In *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on*, pages 209–214. IEEE, 2008.

[70] Martin Saerbeck, Tom Schut, Christoph Bartneck, and Maddy D Janse. Expressive robots in education: varying the degree of social supportive behavior of a robotic tutor. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1613–1622. ACM, 2010.

[71] Bernt Schiele and James L Crowley. Transinformation for active object recognition. In *Computer Vision, 1998. Sixth International Conference on*, pages 249–254. IEEE, 1998.

[72] Bernt Schiele and James L Crowley. Transinformation for active object recognition. In *Computer Vision, 1998. Sixth International Conference on*, pages 249–254. IEEE, 1998.

[73] Michael Seibert and Allen M. Waxman. Adaptive 3-d object recognition from multiple views. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):107–124, 1992.

[74] Masahiro Shiomi, Takayuki Kanda, Nicolas Miralles, Takahiro Miyashita, Ian Fasel, Javier Movellan, and Hiroshi Ishiguro. Face-to-face interactive humanoid robot. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1340–1346. IEEE, 2004.

[75] David Silver and Joel Veness. Monte-carlo planning in large pomdps. In *Advances in neural information processing systems*, pages 2164–2172, 2010.

[76] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. Despot: Online pomdp planning with regularization. In *Advances in neural information processing systems*, pages 1772–1780, 2013.

[77] J Sparling, SL Ramey, CT Ramey, and RG Lanzi. The transition to school: Building upon preschool foundations and preparing for lifelong learning. *The Head Start Debates. Yale University Press, Connecticut*, 2004.

[78] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, volume 99, pages 1057–1063, 1999.

[79] Fumihide Tanaka, Aaron Cicourel, and Javier R Movellan. Socialization between toddlers and robots at an early childhood education center. *Proceedings of the National Academy of Sciences*, 104(46):17954–17958, 2007.

[80] Fumihide Tanaka and Javier R Movellan. Behavior analysis of children's touch on a small humanoid robot: Long-term observation at a daily classroom over three months. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 753–756. IEEE, 2006.

[81] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):815–830, 2010.

[82] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[83] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge England, 1989.

[84] Jacob Whitehill, Gwen Littlewort, Ian Fasel, Marian Bartlett, and Javier Movellan. Toward practical smile detection. *IEEE transactions on pattern analysis and machine intelligence*, 31(11):2106–2111, 2009.

[85] David Wilkes and John K Tsotsos. Active object recognition. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 136–141. IEEE, 1992.

[86] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.