

**GUIDED TEACHING INTERACTIONS WITH ROBOTS:
EMBODIED QUERIES AND TEACHING HEURISTICS**

A Thesis
Presented to
The Academic Faculty

by

Maya Cakmak

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology
August 2012

Copyright © 2012 by Maya Cakmak

GUIDED TEACHING INTERACTIONS WITH ROBOTS: EMBODIED QUERIES AND TEACHING HEURISTICS

Approved by:

Professor Charles L. Isbell,
Committee Chair
School of Interactive Computing
Georgia Institute of Technology

Professor Andrea L. Thomaz, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor Henrik I. Christensen
School of Interactive Computing
Georgia Institute of Technology

Professor Ayanna M. Howard
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Professor Manuela M. Veloso
School of Computer Science
Carnegie Mellon University

Date Approved: 10 May 2012

ACKNOWLEDGEMENTS

To my wonderful and loving husband Raffay Hamid — your existence has made the last five years more exiting than I could have ever imagined and your support has taken me through all the challenges of graduate school. Thank you for “being there” even when you were on the other side of the planet. Five years ago, you were on a student panel in one of my first classes at Georgia Tech and you gave an advice to all of us: PhD is not about getting to the end, it is about the experience. You were very right.

My time at Tech has been an amazing experience and it is the people around me that have made it so. The most important things I am taking away with me are the role-models who will keep guiding me after I have left. I could not have asked for a better role-model than my advisor Andrea Thomaz. She has set an incredible example of the researcher, teacher, advisor, colleague, manager, wife and mother that I will strive to be in the future. I am confident I will succeed in whatever challenges I face, as long as I remember to think “what would Andrea do?” I am very thankful to her for allowing me to grow and stand on my two feet as a researcher, by giving me just-the-right-amount of guidance. To the absolute best advisor ever — thank you for everything!

I want to thank all my thesis committee members, Henrik Christensen, Ayanna Howard, Charles Isbell and Manuela Veloso for raising the standard of my thesis through their constructive and practical feedback, and for their invaluable advise about the future.

Every professor I have interacted with at Tech has been more inspiring than the other and has influenced me in unique ways. Especially, I would like to thank Henrik Christensen for all the opportunities to demo my research, for creating a better future for young roboticist across the continents and for starting the Robotics program at Tech (and making sure we had a coffee machine); Charles Isbell, for convincing me to serve on the chair search

committee which was a great learning experience; Rosa Arriaga and Santosh Vempala, for taking me under their wings very early on in the program, for making me feel welcome in a foreign land and for guiding me towards the path that led me here; Gregory Abowd, for changing the way I think about classroom teaching through his passion and talent; and Charlie Kemp, for reminding me that Robotics is “fun fun fun” at my time of despair.

I could have never come to Tech without the help of my MSc advisor Erol Sahin, who has given me the privilege to be a part of Kovan Lab, and taught me everything about research from the ground-up. Also, my internship mentor Sidd Srinivasa has very generously treated me like his own student even after I left, and has taught me a lot on how to be a well-rounded, ever-curious researcher.

My collaborators have greatly contributed to this thesis. I thank all past and present members of the Socially Intelligent Machines Lab for creating a collaborative and stimulating environment. It was a pleasure to debug code, hack demos and travel to conferences with my brilliant lab mate Crystal Chao. The profound discussions over coffee with my academic brother Baris Akgun have helped me think much more clearly about many things related to my thesis. The long days and nights working on experiments were made much easier by Jaewook Yoo and Nick DePalma. I also thank Manuel Lopes for initiating a wonderful overseas collaboration and for sharing his expertise. The staff at RIM and CoC has had a crucial behind-the-scenes role in keeping things running smoothly and never letting me worry about them.

My friends take all the credit for making Atlanta my second home. This thesis would not have been possible without their support and the balance they brought to my life here. My heroes Misha Novitzky and Kate Wagner have been extremely generous and had my back at critical times— taking care of Pisu while I was traveling or coming to the lab in the middle of the night for my practice talk, just to name a few. I owe a big thank you to Arya Irani, Richard Roberts and Carlos Nieto who have let me know that I could always ask for their help on anything, and to Jaeun Shim for taking the time to teach me an important

survival skill: driving. I thank Jonathan Scholz for being the “organizer” of the many good times: group dinners, bike rides, swing dances, and barbecues. Sooraj Bhat has been like a brother who understands me better than anyone else. I could always rely on Charlie Brubaker and Zsolt Kira for sharing their wisdom. Two very special roommates, Dana Ionita and Ana Huaman, have been like my elder and younger sisters. I am grateful to all my peers who came to my practice talks and participated in my user studies over the years, particularly the 2008 qualifiers for their cooperative spirit. Few dearest friends have been my support system from far away: Mehmet and Nursel Dogar, Cetin Mericli and Munire Sagat.

As strange as it may be for a roboticist to have emotions towards robots, I admit that I have been angry at Simon when he spilled all the coffee beans, sad for him when he broke his wrist cables, and happy every time he successfully learned something. Now I am feeling thankful for having had the chance to work with such an awesome robot — thank you Meka robotics and Carla Diana.

Most importantly, I want to thank all my family members, who make my work sound like the coolest thing on earth, for their encouragement. Particularly, my grandparents Rosette and Raymond Lepine for never forgetting to ask how Simon is doing and for checking the weather in Atlanta every morning; and my late grandparents Ziya and Turkan Cakmak for encouraging me to become an academic. I am forever grateful to my parents, Mariepaul and Kenan Cakmak, for nurturing me to be creative, encouraging me to do what I love, never pressuring me about anything and only caring about my happiness; and to my younger brother Kaan, who has surprised me with his wisdom many times, for his advise that has kept me sane through this journey.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
LIST OF TABLES	xii
LIST OF FIGURES	xv
SUMMARY	xx
I INTRODUCTION	1
1.1 Motivation	2
1.1.1 Robots that learn from end-users	2
1.1.2 Guiding user interactions with robots	4
1.2 Approach	5
1.2.1 Guidance by Embodied Queries	6
1.2.2 Guidance by Teaching Heuristics	7
1.3 Thesis Overview	8
1.3.1 Thesis statement	8
1.3.2 Summary of contributions	8
1.3.3 Document outline	9
II RELATED WORK	11
2.1 Learning from Demonstration	11
2.2 Interactive Machine Learning	13
2.3 Active Learning	14
2.3.1 Active Learning in Robotics	15
2.4 Algorithmic Teaching	16
2.5 Human Learning and Teaching	18
2.6 Relation to present work	20
III EXPERIMENTS ON HUMAN TEACHING	21
3.1 Robot platforms	22
3.1.1 Simon	22

3.1.2	Junior	23
3.1.3	The Creatures Software Architecture	23
3.2	Task Learning	25
3.2.1	Concept Learning with Discrete Features	26
3.2.2	Active Concept Learning	28
3.2.3	Domains	28
3.3	Skill Learning	34
3.3.1	Keyframe-based Learning from Demonstration	38
3.3.2	Domains	39
3.4	Overview of Experiments	43
3.5	Summary	45
IV	UNGUIDED HUMAN TEACHING	46
4.1	Experiment 1: Unguided Teaching of Tasks	47
4.1.1	Experimental Design	47
4.1.2	Findings	53
4.2	Experiment 2: Unguided Teaching of Skills	54
4.2.1	Experimental Design	55
4.2.2	Findings	57
4.3	Experiment 3: Unguided teaching of action models	61
4.3.1	Experimental Design	62
4.3.2	Findings	67
4.4	Issues in Unguided Teaching	74
4.4.1	Humans provide sub-optimal examples	74
4.4.2	Humans do not complete teaching	77
4.5	Useful Patterns in Unguided Teaching	79
4.5.1	Humans provide balanced datasets	79
4.5.2	Humans partially use teaching heuristics	81
4.6	Summary	82

V	EMBODIED ACTIVE LEARNING	84
5.1	Experiment 4: Active task learning	86
5.1.1	Experimental Design	87
5.1.2	Findings	89
5.2	Issues in Embodied Active Learning	92
5.2.1	The balance-of-control problem	92
5.2.2	The question clarity problem	93
5.2.3	The compliance problem	94
5.2.4	The noisy teacher problem	95
5.3	Summary	95
VI	INTERMITTENTLY ACTIVE LEARNING	97
6.1	Teacher-triggered queries	97
6.2	Conditional queries	98
6.2.1	Conditional queries for task learning	98
6.3	Experiment 5: Intermittently-active task learning	99
6.3.1	Experimental design	99
6.3.2	Findings	102
6.4	Summary	111
VII	EMBODIED QUERY TYPES	112
7.1	Query Types for Skill Learning	113
7.1.1	Label queries	114
7.1.2	Demonstration queries	114
7.1.3	Feature queries	115
7.2	Experiment 6: Skill learning with different query types	116
7.2.1	Experimental Design	117
7.2.2	Findings	120
7.3	Experiment 7: Human question asking	124
7.3.1	Experimental Design	124

7.3.2	Findings	128
7.4	Discussion	134
7.5	Summary	135
VIII ACTIVE KEYFRAME-BASED LfD		136
8.1	Embodied Queries in Keyframe-based LfD	137
8.1.1	Label queries	137
8.1.2	Demonstration queries	149
8.1.3	Feature queries	150
8.2	Experiment 8: Evaluation of Active Keyframe-based LfD	157
8.2.1	Experimental design	158
8.2.2	Findings	162
8.3	Summary	166
IX TEACHING HEURISTICS FOR CLASSIFICATION		167
9.1	Optimal Teaching in Classification Problems	169
9.1.1	Provably Optimal Teaching	170
9.1.2	Empirically Optimal Teaching	173
9.1.3	Approximately Optimal Teaching	174
9.2	Experiment 9: Instructed teaching of tasks	179
9.2.1	Experimental Design	179
9.2.2	Findings	181
9.3	Web-based experiments	186
9.4	Experiment 10: Instructed teaching of face categories	188
9.4.1	Experimental Design	188
9.4.2	Findings	192
9.5	Experiment 11: Instructed teaching of linear separators	196
9.5.1	Experimental Design	196
9.5.2	Findings	198
9.6	Experiment 12: Instructed teaching of mouse gestures	199

9.6.1	Experimental Design	200
9.6.2	Findings	202
9.7	Discussion	203
9.8	Summary	205
X	TEACHING HEURISTICS FOR SEQUENTIAL DECISION TASKS	207
10.1	Optimal Teaching in MDPs	207
10.1.1	Inverse Reinforcement Learning	208
10.1.2	Teaching IRL agents optimally	209
10.1.3	Evaluation of optimal teaching algorithm	212
10.2	Experiment 13: Instructed teaching of navigation tasks	215
10.2.1	Experimental Design	216
10.2.2	Findings	218
10.3	Summary	221
XI	TEACHING HEURISTICS FOR KEYFRAME-BASED LFD	222
11.1	Experiment 14: Instructed teaching of skills	224
11.1.1	Experimental Design	224
11.1.2	Findings	226
11.2	Summary	231
XII	COMBINING TEACHING HEURISTICS AND QUERIES	233
12.1	Teaching Heuristics for Intermittently-Active Learning	234
12.2	Experiment 15: Instructed Teaching in I-AL	236
12.2.1	Experimental Design	236
12.2.2	Findings	238
12.3	Summary	239
XIII	CONCLUSIONS AND FUTURE WORK	240
13.1	Contributions	240
13.1.1	Experimental findings about teaching-learning interactions	240
13.1.2	Intermittently-Active Learning	241

13.1.3	Active Keyframe-based Learning from Demonstration	241
13.1.4	Experimental findings on human question asking	242
13.1.5	Teaching heuristics	242
13.2	Open questions and future work	242
APPENDIX A	— PROTOCOL FOR EXPERIMENT 2	245
APPENDIX B	— PROTOCOL FOR EXPERIMENT 5	249
APPENDIX C	— LEARNING CURVES FOR INDIVIDUALS IN EXP. 5	251
APPENDIX D	— SURVEY IN EXPERIMENT 6	255
APPENDIX E	— PROTOCOL FOR EXPERIMENT 7	257
APPENDIX F	— MECHANICAL TURK HIT FOR EXPERIMENT 10	259
APPENDIX G	— MECHANICAL TURK HIT FOR EXPERIMENT 11	261
APPENDIX H	— MECHANICAL TURK HIT FOR EXPERIMENT 12	262
APPENDIX I	— MECHANICAL TURK HIT FOR EXPERIMENT 13	263
APPENDIX J	— PROTOCOL FOR EXPERIMENT 14	264
REFERENCES	265

LIST OF TABLES

1	Object categories of fixed specificity for task learning in the six feature toy objects domain.	30
2	Object categories of varying specificity for task learning in the nine feature toy objects domain.	31
3	Progress of the version space in the concept learning algorithm used in task learning experiments, illustrated for a sequence of three labeled examples for the HOUSE concept.	32
4	Label prediction based on the version space for the concept learning algorithm used in task learning experiments exemplified for the HOUSE concept.	33
5	Overview of all experiments in this thesis.	44
6	Performance of participants in Experiment 1 on different metrics of good teaching, compared to optimal values.	52
7	Number of participants who achieved different levels of success for goal-oriented skills in Experiment 2.	61
8	Expert ratings of means-oriented skills (median and coefficient of dispersion) and comparison of two demonstration types in in Experiment 2.	62
9	Number of examples presented by individuals for each object and action in Experiment 3.	71
10	Accuracy, time taken to teach, and efficiency for the three post-hoc groups in Experiment 4.	91
11	Learning performance of four the conditions in Experiment 5. $ V = 1$ corresponds to having a single hypothesis in the version space, <i>i.e.</i> exact identification of the target.	103
12	Mean and standard deviation over the subjective ratings given by participants in five different 7-point Likerd-scale questions in Experiment 5.	104
13	Repeated measures ANOVA test results indicating the effect of condition on subjective rating scales in Experiment 5.	104
14	Comparison of the NQ condition with the three interactive conditions (UQ, CQ and TTQ) on subjective ratings in Experiment 5.	105
15	Accuracy of the learned concept as predicted by the teacher, compared to the true accuracy, in Experiment 5.	106
16	Average time taken to teach concepts, number of steps and efficiency as defined by how much the F_1 -score was increased per step, in Experiment 5.	106

17	A selection of positive comments about the three interactive conditions (UQ, CQ, TTQ) from the open-ended questions in the survey in Experiment 5.	107
18	A selection of negative comments about the three interactive conditions (UQ, CQ, TTQ) from the open-ended questions in the survey in Experiment 5.	107
19	Comparison of the UQ condition with the other conditions in terms of balance of control ratings given by participants in Experiment 5, measured with t -tests.	108
20	Overall compliance by participants in answering the robot's queries in Experiment 5.	109
21	Compliance on answering queries, split according to informativeness of the query in Experiment 5 for the UQ and CQ conditions.	110
22	Forced ranking results in Experiment 6 presented in terms of the number of participants out of 18 who voted the query type as best or worst.	121
23	Average time (in seconds) taken for the robot to make a query and for the participants to answer the query for the three types of queries in Experiment 6.	122
24	Coverage of the taught skills and the number of demonstrations provided within 5 minutes in the six conditions of Experiment 8. For conditions that involve queries, the number of demonstrations indicate the sum of the two initial demonstrations and the number of queries made in the rest of the time.	164
25	Success of learned skills taught in the six conditions of Experiment 8, tested in five different goal configurations. The numbers indicate the percentage of tests out of five that were successful, failed, or N/A. N/A means that the learned skill did not cover the goal configuration in which it was being tested.	165
26	Summary of all teaching performance metrics in Experiment 9.	182
27	Wall clock time (in seconds) spent on an example or test (t_{avg}), expected total time to get to $F = 1$, $t_{avg} * T_t _F$, and number of tests used by participants in Experiment 9.	184
28	Description of the conjunction concepts in the Chernoff faces domain used in Experiment 10.	188
29	Responses to the open-ended question in the <i>Natural</i> condition for Experiments 10, 11 and 12. The question asks the participants to characterize their teaching strategies. The responses are coded in terms of their overlap with the guidance given in the <i>Guided</i> condition. The responses are coded independently by the two authors as one or none of the given rows; the reported numbers are for responses where both coders agreed. Inter-coder agreement is reported with Cohen's Kappa (κ).	195

30	Responses to the open ended question in the <i>Guided</i> condition for Experiments 10, 11 and 12. The question asks the participants whether they had any problems following the guidance, and if so, what they found most challenging. The responses are coded independently by the two authors as one or none of the given rows. Inter-coder agreement is reported with Cohen’s Kappa (κ).	196
31	Results from Experiment 13. First two columns show the number of participants who taught with an optimal sequence. The next two columns show the average uncertainty in the reward estimation achieved by the demonstrations given by the participants. The optimal value is given for comparison. The last column reports <i>t</i> -test results between the two conditions. . . .	219
32	Coverage of the taught skills and the number of demonstrations provided within 5 minutes in the two conditions of Experiment 14.	227
33	Success of learned skills taught in the two conditions of Experiment 14, tested in five different goal configurations. The numbers indicate the percentage of tests out of five that were successful, failed, or N/A. N/A means that the learned skill did not cover the goal configuration in which it was being tested.	227

LIST OF FIGURES

1	The two robot platforms used in the experiments across this thesis.	23
2	Snapshots from the robot models animated in C6. (a) Simon (b) Junior	24
3	The different Machine Learning problems involved in different skill models.	36
4	Overview of Keyframe-based Learning from Demonstration.	38
5	Skills in the single-handed goal-oriented skills domain.	41
6	Skills in the single-handed means-oriented communicative skills domain. . .	42
7	Skills in the two-handed goal-oriented skills domain.	43
8	Simon’s workspace with the different object parts arranged around the demonstration area, while a human teacher is providing a demonstration in Experiment 1.	48
9	An optimal teaching sequence example for target concept HOUSE.	50
10	Distribution of <i>good</i> and <i>bad</i> positive/negative examples given by participants in Experiment 1. For good examples, breakdowns of how much of the version space was pruned is shown (<i>i.e.</i> by a factor of 2, 4, or 8). Bad negative examples are broken down as uninformative or too varied.	53
11	A participant kinesthetically interacting with Simon in Experiment 2.	55
12	Histogram of number of demonstrations provided by participants in KD and TD conditions.	58
13	An example of the temporal alignment issue in trajectory demonstrations while teaching the <i>Close-box</i> skill in Experiment 2. The resulting skill does not close the box (the horizontal arm movement is not sufficient) due to the attenuation of the movement in the Shoulder-Z joint caused by averaging temporally misaligned demonstrations. Demonstrations (upper four panels) and reproduced motions (lower four panels) are shown on four joints.	59
14	An example of forgetting obstacle avoidance keyframes in the first demonstrations (dashed line), and providing them in the second demonstration (solid line) in the KD condition while teaching the <i>Push-button</i> skill in Experiment 2. Demonstrations (upper four panels) and reproduced motions (lower four panels) are shown on four joints.	60
15	Five objects for the playground domain and a screenshot from Experiment 3 in which a human teacher is interacting with Junior.	65
16	All configurations of non-spherical objects considered in the systematic condition in Experiment 3.	66

17	Distribution of starting object choice of participants for the grasping action in Experiment 3.	68
18	Comparison of human usage of objects and object complexity in Experiment 3. (a) Distribution of examples given for each object averaged across participants (b) Number of affordances of each object (c) Number of configurations of each object.	69
19	Comparison of object reachability and object placements by participants in Experiment 3. (a) Histogram of reachable distances, taken from systematic data (b) Histogram of distances where participants placed objects.	70
20	Distribution of effects in the <i>Non-social</i> and <i>Social</i> conditions for each action in Experiment 3.	73
21	Impact of learning social versus non-social learning on the performance of the learned classifier in Experiment 3. Graphs present the prediction accuracy of classifiers trained with the non-social (systematic) data and the social data, on (a) social test data and (b) systematic test data. Values are averaged over 10 randomly sampled test sets.	73
22	Snapshots from Experiment 4 showing gestures by Simon for (a) pointing during a query and (b) shrugging to indicate that he is uncertain about the answer.	87
23	Snapshots from Experiment 5 involving human teachers demonstrating different toy object concepts to Simon.	100
24	Participants demonstrating three different skills to Simon in Experiment 6.	117
25	The queries made by the robot for each type of query in Experiment 6.	118
26	The end-effector trajectories provided by all participants in Experiment 6 (thin lines). (a) (Pour-cereal) Superimposed with the label queries made by the robot (thick lines). (b) (Add-salt) Simon's pose shows the second demo query starting point. Starts of the demonstrations are shown as dots.	124
27	Description of the four tasks used in Experiment 7 and snapshots from the two demonstration videos shown to the participants.	125
28	Distributions of human questions into categories in Experiment 7. (a) Distribution of questions into three types of queries for each task in the first experiment averaged across participants. (b) Distribution of feature queries into sub-categories. (c) Average distribution of common actions and gestures that physically ground questions across tasks.	127
29	An example partial-label query from Experiment 7 where the participant elicits a label after each step of the skill execution.	130

30	Question forms [73] and example instances from Experiment 7 (percentage of average occurrence shown in parentheses).	130
31	Examples from Experiment 7 of the four common ways in which humans use their embodiments to support the communication of their questions. . .	132
32	Plot of the utility function $U_k(s)$ for different values of n_k in 1D.	138
33	Progress of the fit between normal distributions learned through queries and the true distribution. Three different query variance constants, including the one proven to be optimal, are compared with uniform sampling.	142
34	Illustration of the label and partial label query execution process.	146
35	Snapshots from a sample execution of label queries that receive (a) positive and (b) negative labels while learning the <i>Pour</i> skill.	147
36	Snapshots from a sample execution of partial label queries for learning the <i>Pour</i> skill.	149
37	Snapshots from a sample execution of demonstration queries for learning the <i>Pour</i> skill.	151
38	Fixed ordering logic for feature queries.	152
39	Snapshots from sample executions of feature queries.	157
40	Snapshots from Experiment 8 while participants in the baseline data collection demonstrate the two benchmark skills to Simon.	158
41	Layout of possible goal configurations \mathcal{G} relative to Simon and the five canonical goal configurations in which the robot is tested to assess the success of the skills.	163
42	Motivating example from [47] which illustrates the power of optimal teaching as compared to active learning.	170
43	Graphical interface used for teaching Simon in Experiment 9.	180
44	Distribution of <i>good</i> and <i>bad</i> positive/negative examples given by participants in Experiment 9. For good examples, breakdowns of how much of the version space was pruned is shown (<i>i.e.</i> by a factor of 2, 4, or 8). Bad negative examples are broken down as uninformative or too varied.	183
45	Illustrations of the three domains used in Experiments 10, 11 and 12. (a) Two extreme faces in the Chernoff face domain with 16 binary features. (b) Instances from the shape domain. (c) Examples from the mouse gesture domain.	189

46	Screenshots of the teaching interfaces used in Experiments 10, 11 and 12. (a) Interface for teaching conjunctions in the Chernoff faces domain (16 binary features, <i>Angry face</i> concept) (b) Interface for teaching a linear classifier in 1-D in the <i>Shapes</i> domain. (c) Interface for training a nearest-neighbor classifier to classify mouse gestures.	190
47	Example optimal teaching sequences for two conjunction concepts in the <i>Chernoff faces</i> domain used in Experiment 10.	192
48	Results for teaching conjunctions in Experiment 10. The graphs show the progression of the average accuracy on the whole sample space over the first $ \tau^* $ examples for human teachers teaching the four different target concepts. Each graph shows the accuracy curves for <i>Natural</i> and <i>Guided</i> human teaching as well as the optimal teaching sequence. The error bars denote standard error.	194
49	Accuracy of the (a) linear separators in \mathbb{R}^1 (<i>Shapes</i>) in Experiment 11 and (b) nearest-neighbor classifiers (<i>Mouse gestures</i>) in Experiment 12 over the first few examples of the teaching sequences provided by the human teachers in the <i>Natural</i> and <i>Guided</i> conditions. (a) Compared to the empirically optimal teaching sequence. (b) Compared to the average of teaching sequences produced by the greedy approximate algorithm (indicated as <i>Possible</i>). Error bars indicate standard error.	199
50	Additional results from the mouse gesture teaching domain used in Experiment 12. (a) Sample non-alpha gestures chosen by the greedy algorithm as part of the first two examples. (b) Non-alpha gesture examples given by participants using the <i>borderline</i> teaching heuristic.	203
51	The two maps and the associated reward weights for the four tasks used in evaluating the optimal MDP teaching algorithm and used in Experiment 13.	213
52	The start states and trajectories chosen by the MDP teaching algorithm for Tasks 1 and 2. The gray-scale color of the squares on the map indicate the value function according to the optimal policy (light colors have high value).	214
53	The start states and trajectories chosen by the MDP teaching algorithm for Tasks 3 and 4. The gray-scale color of the squares on the map indicate the value function according to the optimal policy for corresponding reward functions.	215
54	Comparison of learning from demonstrations selected by the MDP teaching algorithm and demonstrations whose start states are randomly selected. The top row shows the decrease in the uncertainty on the rewards, and the bottom row shows the change in the percentage of correctly chosen actions with the policy obtained from the estimated rewards. The x-axes are the increasing number of demonstrations.	216

55	Images used to supplement the teaching heuristics for the instructed teaching of skills in Experiment 14. (a) Explanation of the collision problem due to missing obstacle avoidance keyframes. (b) Guidance for preferring keyframes that are closed to the goal.	226
56	Example starting keyframes provided by participants in Experiment 14, as a result of the teaching heuristic that advises the teachers to vary the position and orientation of keyframes when possible.	229
57	Teaching sequences for the HOUSE concept produced by (a) the optimal teaching algorithm and (b) by the learner through queries once the initial positive example was given.	235
58	Average informativeness of examples provided by human teachers in five conditions: <i>Natural</i> , <i>Motivated</i> and <i>Guided</i> conditions from Experiment 9 and <i>TTQ</i> and <i>Guided TTQ</i> from Experiment 15.	238
59	F-score over time for individuals in Experiment 5 including tests made by the participant.	251
60	F-score over time for individuals in Experiment 5 excluding tests made by the participant.	252
61	Size of the version space over time for individuals in Experiment 5 including tests made by the participant.	253
62	Size of the version space over time for individuals in Experiment 5 excluding tests made by the participant.	254

SUMMARY

The vision of personal robot assistants continues to become more realistic with technological advances in robotics. The increase in the capabilities of robots, presents boundless opportunities for them to perform useful tasks for humans. However, it is not feasible for engineers to program robots for all possible uses. Instead, we envision general-purpose robots that can be programmed by their end-users.

Learning from Demonstration (LfD), is an approach that allows users to program new capabilities on a robot by demonstrating what is required from the robot. Although LfD has become an established area of Robotics, many challenges remain in making it effective and intuitive for naive users. This thesis contributes to addressing these challenges in several ways. First, the problems that occur in teaching-learning interactions between humans and robots are characterized through human-subject experiments in three different domains. To address these problems, two mechanisms for guiding human teachers in their interactions are developed: *embodied queries* and *teaching heuristics*.

Embodied queries, inspired from Active Learning queries, are questions asked by the robot so as to steer the teacher towards providing more informative demonstrations. They leverage the robot's embodiment to physically manipulate the environment and to communicate the question. Two technical contributions are made in developing embodied queries. The first is *Active Keyframe-based LfD* — a framework for learning human-segmented skills in continuous action spaces and producing four different types of embodied queries to improve learned skills. The second is *Intermittently-Active Learning* in which a learner

makes queries selectively, so as to create balanced interactions with the benefits of fully-active learning. Empirical findings from five experiments with human subjects are presented. These identify interaction-related issues in generating embodied queries, characterize human question asking, and evaluate implementations of Intermittently-Active Learning and Active Keyframe-based LfD on the humanoid robot Simon.

The second mechanism, teaching heuristics, is a set of instructions given to human teachers in order to elicit more informative demonstrations from them. Such instructions are devised based on an understanding of what constitutes an optimal teacher for a given learner, with techniques grounded in Algorithmic Teaching. The utility of teaching heuristics is empirically demonstrated through six human-subject experiments, that involve teaching different concepts or tasks to a virtual agent, or teaching skills to Simon.

With a diverse set of human subject experiments, this thesis demonstrates the necessity for guiding humans in teaching interactions with robots, and verifies the utility of two proposed mechanisms in improving sample efficiency and final performance, while enhancing the user interaction.

CHAPTER I

INTRODUCTION

This thesis aims at allowing end-users to program new capabilities on their robot. Learning from Demonstration (LfD) is an intuitive method that achieves this through demonstrations of what is required from the robot. Research in LfD over the last three decades has yielded various representations, learning algorithms and interaction modalities. However, user studies involving LfD systems have been very few and many challenges remain in making LfD intuitive and effective for end-users. Average users do not have a good understanding of how the robot represents world states and tasks, and how it learns from the provided demonstrations. As a result they present sub-optimal or uninformative demonstrations to the robot. In addition, there are constraints in learning from end-users, such as limited time, impatience or limited memory capacity. Therefore an important challenge for LfD-capable robots is to obtain the most informative demonstrations possible, as fast as possible, through limited interactions with humans.

This thesis seeks to address this challenge by augmenting teaching-learning interactions in order to *guide* the human teacher. Two mechanisms that directly attempt to improve the informativeness of human demonstrations are developed:

- *Embodied queries* are questions asked by the robot, leveraging its embodiment, to steer the human teachers towards providing more informative demonstrations.
- *Teaching heuristics* are instructions given to human teachers, in order to influence their choices towards more informative demonstrations.

With a diverse set of experiments involving human subjects, the necessity for such guidance and the utility of these two mechanisms are empirically demonstrated.

The rest of this chapter motivates the problems addressed in this thesis, summarizes the approach of the thesis, overviews the statement and contributions of the thesis and provides a document outline.

1.1 Motivation

This thesis is motivated by the vision of personal robots that assist humans in everyday tasks. This can involve assisting humans in various contexts such as homes, hospitals, offices or factories; performing various tasks like house chores, delivering items, or collaborating on an assembly line. These robots will increase the quality of life, give persons with motor impairments or older adults more independence, and increase productivity. While there are a number of challenges towards achieving this vision, this thesis focuses on learning from end-users which we motivate in the following.

1.1.1 Robots that learn from end-users

While there has been tremendous progress in robotics over the years, the few robots that have succeeded as commercial products have had limited purpose with very specific tasks to perform (*e.g.* vacuuming, entertainment). As robot actuators and sensors become more sophisticated and new techniques are developed for perception, navigation and manipulation, general-purpose robots which can be used for a variety of tasks are becoming realistic. However, the increase in the scope of a robot's capabilities presents several challenges that can only be addressed through learning. In particular, three types of learning are essential.

- **Adaptivity:** Robots need to be *adaptable* because of the variability in the environments that they will be required to operate in. They should be able to discover how their actions influence the world and how certain goals can be achieved in a new environment. It is not feasible for robot designers to predict the variability that the robots will encounter when they are deployed in different environments.

- **Programability:** Robots should be *programmable* because of the variability in the tasks that their owners will need assistance with. As robots become more capable, the number of task that they can carry out will increase and it will not be realistic to program-in all possible uses of a robot. Therefore it is crucial to have efficient and intuitive ways that people can teach new tasks to a robot.
- **Customizability:** In many tasks, there are more than one way of achieving the same goal. While some parameters of the robot's behavior might not affect their success in achieving a goal, users might prefer one way over the other for reasons that are not detectable or represented by the robot. Therefore robot behaviors should be *customizable* by the user to meet their preferences.

Consider a high-end robot such as the Personal Robot 2 (PR2) by Willow Garage. This robot has previously been programmed to fetch drinks from the fridge, fold the laundry, prepare pancakes or help a person with motor-impairments in personal hygiene tasks. As highly capable robots like the PR2 become commercial, we can expect that they will be equipped with a basic set of capabilities and that we would be able to upgrade them with new capabilities over time. Nevertheless, this limits the capabilities of a robot to what engineers or experts program on it. Some examples of the three types of learning for this robot are as follows. The drink fetching skill programmed by engineers might have assumed that the drink is in a bottle, whereas in some homes drinks are in cans or cartons. In this case the robot will need to *adapt* its existing skill to be able to accomplish it in this novel environment. The t-shirt folding skill might fold in a way that results in the folded clothes not fitting into the closet, so the user might want to *customize* this skill to add an extra fold. A user might want the robot to play with his or her cat, by grasping a cat toy and dragging it around to make the cat run. It is unlikely that the robot has this capability built in, or there is a downloadable app for it — so the user should be able to *program* this new capability.

The importance of learning from end-users is clear for programability and customizability: only the end-user knows the target behavior that the robot is going to learn. The user has to interact with the robot to program a new capability on the robot or to customize its existing capabilities. Adaptivity, on the other hand, is something the robot can achieve on its own. As long as the robot has a way to evaluate its own success, it can autonomously explore new environments and experiment with different actions to adapt to new situations. Nonetheless, a human can have an important role in speeding up the adaptation process by guiding the robot's exploration and creating environments that will provide the robot with useful learning experiences.

1.1.2 Guiding user interactions with robots

Learning from humans has unique challenges and advantages. Humans have limited time, attention, memory and patience; they are unpredictable and varied in many dimensions; and they have limited understanding of how the machine perceives, acts and learns. Nevertheless, learning from humans can be beneficial. Humans can intelligently select examples for the learner and can easily adapt to new ways of teaching. In addition, teaching can be an inherently rewarding social interaction for humans [135]. Therefore, robots that learn from humans should exploit the potential benefits of learning from humans. At the same time, they need ways to cope with the drawbacks and minimize their influence on the quality of the data obtained from them.

The challenge in learning from humans can often be boiled down to one common objective: obtaining the most informative examples possible from humans, as fast as possible. As humans cannot spend long time training their robots, time is limited. Besides the time limitation, getting sufficiently informative demonstrations from humans is challenging for several reasons. The way that robots learn is often fundamentally different from how humans learn. Humans strongly draw on their existing knowledge and common sense in their generalizations. Thus they might make wrong assumptions about what the robot already

knows and not demonstrate the full range that is required for the robot to generalize. Secondly, human learning often requires repetition, which is redundant for robots, hence a waste of time. Another cause of redundant examples is the limitations of human memory capacity. Humans might forget that they have already demonstrated something and would not be able to keep track of what they have covered in their previous examples.

These challenges point towards a need for guiding humans in their interactions with the robot. Robots need ways to influence their teachers' behavior in a way that results in more informative examples or demonstrations.

As an example, consider again skills on a home assistant robot like PR2. Assume the laundry folding skill represents clothes as topological graphs, and that users demonstrate how they want their clothes to be folded. Based on the robot's representation, long and short pants might be exactly the same, and having learned either, the robot could fold the other. Users on the other hand might consider these to be different and provide demonstrations for each, causing a waste of time. Similarly after showing several categories, the user might forget if they have already shown how to fold towels, and show it again. The other problem discussed above is not providing sufficiently varied demonstrations. For example, the user may think that demonstrating how to fold t-shirts covers folding sweaters; while the robot is not able to make that generalization.

In the described scenario, both problems of redundant and missing demonstrations could be avoided by allowing the robot to choose the cloth that the human will demonstrate folding on. Alternatively, the problems could be avoided by giving instructions to the users to demonstrate folding on one pant, one shirt, and etcetera.

1.2 Approach

This thesis addresses the following broader research questions:

1. What are the challenges in learning new tasks and skills from naive users?
2. How can we support or augment the interaction in order to address these challenges?

Elaborations about the challenges in learning from humans (Sec. 1.1.2) suggest general directions to explore in addressing the first question. However, a much more concrete characterization of the issues is essential in order to propose solutions. This thesis takes an empirical approach to making these challenges concrete. The problems that occur when the input to a learning robot is provided by a naive user are characterized through three baseline human-subject experiments.

Experiments in this thesis focus on two types of robot learning problems. *Task learning* involves learning a representation of a task goal from demonstrations of the task. This corresponds to learning *what* to do as part of a task. *Skill learning*, on the other hand, focuses on *how* to do a task. This involves learning a policy or a model of the skill that allows the robot to produce sequences of actions that achieve a task goal. While certain issues may be specific to each of these learning problems, or even specific to the particular methods and domains used in our experiments, we seek to identify common issues and propose methods or recommendations that can be applied to a diverse range of domains.

To address the issues identified through baseline experiments, two approaches motivated in Sec. 1.1.2 are proposed.

1.2.1 Guidance by Embodied Queries

The first approach is guiding human teachers with questions asked by the robot using its embodiment. The intuition in this approach is that the learner is often better informed on what would be most informative for it, than the teacher. This idea comes from an extensively studied method in Machine Learning, called Active Learning (AL) [5, 42, 122]. In passive supervised learning, the input to the learning algorithm is a set of paired instances and labels. The learner needs to produce a classifier that can predict the label given a new instance. In AL, the learner has access to a pool of unlabeled instances and it iteratively picks an instance, for which the teacher provides a label. Such requests for labels are called *queries*. By selecting queries based on a criterion of informativeness, such as uncertainty

about the predicted label, the learner can significantly speed up the learning process [122].

Active learning has attracted attention in the Robotics community and has been applied to different problems [40, 61, 58]. Differing from these, we take a Human-Robot Interaction (HRI) approach and evaluate the use of queries in teaching-learning interactions with naive humans. We empirically identify issues that arise in these interactions and propose solutions that address them without sacrificing the benefits of AL. Secondly, we relax the idea of AL from “requesting labels for instances”, to the general notion of the learner having more control in the learning process. This leads to different types of questions that the robot can ask in learning interactions to increase the informativeness of the teacher’s demonstrations. Another approach taken in this thesis is getting inspiration from how humans ask questions. This leads to novel query types that are observed in humans, and gives insights into how robot questions can be made more natural for interactions with humans.

1.2.2 Guidance by Teaching Heuristics

The second approach for guiding humans in their teaching interactions is providing specific instructions on how to teach, which we refer to as *teaching heuristics*. This simple method attempts to directly influence the teacher to provide more informative demonstrations. The motivation for this approach comes from a field of ML called Algorithmic Teaching (AT) [13, 57]. AT formally studies the problem of teaching, that is, given a *known* concept, how can a teacher produce a set of examples that teaches the concept with as few examples as possible. Research in AT investigates the upper bound on how fast a concept can be taught and devises efficient algorithms that can produce optimal teaching sets. In many cases an optimal teacher can teach with much fewer examples than the number of queries made by an active learner to learn the same concept [57].

This means that a good teacher can be much more powerful than a good learner — hence our approach is to directly influence the teacher. For this we use instructions that try to elicit the behavior of an optimal teaching algorithm. Such instructions are derived either

directly from these algorithms, or from observations of the types of examples they produce. We test these instructions through user studies, comparing them to un-instructed teaching to verify their utility and comparing them to theoretically optimal teaching to identify ways they can be improved.

1.3 Thesis Overview

1.3.1 Thesis statement

The sample efficiency and final performance of a robot in skill and task learning problems, can be greatly improved by guiding the teacher with *embodied queries* from the robot and *teaching heuristics* provided to the teacher.

1.3.2 Summary of contributions

The contributions made by this thesis can be summarized as follows.

- Experimental findings from three domains that characterize issues and useful patterns that occur when non-expert humans teach tasks and skills to a robot.
- Experimental findings that characterize human-interaction related issues in applying Active Learning to robot skill and task learning.
- *Intermittently-Active Learning (I-AL)* — a method for creating balanced teaching-learning interactions while maintaining the benefits of fully-active learning; two alternative I-AL techniques, *Teacher-triggered Queries* and *Conditional Queries*, and their evaluation through human-subject experiments.
- *Active Keyframe-based Learning from Demonstration* — a framework for human-segmented representations of skills; implementation and evaluation of four different types of embodied queries in this framework.
- Experimental findings that characterize *human question asking* during skill and task learning.

- Novel types of queries for skill learning, namely *Partial-label*, *Demonstration* and *Feature* queries, based on human-question asking and state-of-the-art query methods in other applications of Active Learning.
- *Teaching heuristics* — a technique for improving the quality of examples obtained from humans, and methods grounded in Algorithmic Teaching for creating teaching heuristics for a given learner.
- Experimental findings that demonstrate the utility of teaching heuristics in six different domains.

1.3.3 Document outline

This thesis is organized as follows.

Chapter 2 – Related Work provides a survey of related research, positions the contributions of this thesis in relation to these works and provides background on the areas of Machine Learning, such as Active Learning and Algorithmic Teaching, that this thesis draws from.

Chapter 3 – Experiments on Human Teaching provides preliminaries about the experiments presented in this thesis, describes the common learning algorithms and domains and overviews the list of all experiments presented in the thesis.

Chapter 4 – Unguided Human Teaching presents findings from experiments involving naive human teachers and highlights the problems and benefits of learning from humans.

Chapter 5 – Embodied Active Learning introduces *Embodied queries*, discusses the differences between conventional applications of Active Learning and learning problems in robotics, presents an observational experiment comparing active and passive task learning and highlights interaction related issues in Embodied AL.

Chapter 6 – Intermittently-Active Learning introduces two approaches for I-AL and an experiment that evaluates these approaches against passive and fully-active learning.

Chapter 7 – Embodied Query Types explains three alternative query types, presents an experiment that compares these types in skill learning from the human teacher’s perspective, and presents an experiment that analyzes query types, forms and embodiment in humans.

Chapter 8 – Active Keyframe-based LfD introduces a framework for representing human-segmented skills, presents methods for autonomously generating different query types from such skill representations and evaluates them through a comparison with skills taught by naive humans.

Chapter 9 – Teaching Heuristics for Classification introduces teaching heuristics, and presents methods grounded in Algorithmic Teaching for devising teaching heuristics in classification problems and presents four experiments that demonstrate the impact of teaching heuristics.

Chapter 10 – Teaching Heuristics for Sequential Decision Tasks extends Algorithmic Teaching to sequential decision tasks, and presents an experiment that evaluates teaching heuristics for Inverse Reinforcement Learning agents.

Chapter 11 – Teaching Heuristics for Keyframe-based LfD presents an experiment that evaluates teaching heuristics for skills in the KLfD framework.

Chapter 12 – Combining Teaching Heuristics and Queries discusses the strengths and weaknesses of the two mechanisms and presents an experiment that demonstrates the synergy between the two mechanisms.

Chapter 13 – Conclusion and Future work re-iterates the contributions of this thesis and discusses open questions in guided interactions with robots.

CHAPTER II

RELATED WORK

This section presents an overview of five related research fields and situates this thesis in relation to each of these fields.

2.1 Learning from Demonstration

Learning from Demonstration (LfD) is a method for programming new capabilities on a robot by providing demonstrations [10, 16]. *Skill learning* involves learning a *policy* that maps a state to an appropriate action in order to achieve or maintain a goal state. Actions can be continuous [59] or discrete [39, 117, 19]. A recent survey [8] gives an extensive review of LfD problems formulating them as policy learning.

A learned policy can lead the robot to the goal without explicitly representing the goal. The goal can be encoded in the policy using goal-directed features in the state representation [31, 98, 97], or learning a function parametrized with the goal [119, 107]. In other cases the goal is learned from examples of states where the goal is achieved or not. *Goal learning* involves learning a generalized representation of states in which a task is deemed successful [19, 108, 118], or learning a value function that measures the quality of a state [21, 1].

Role of the teacher. In LfD systems the teacher can provide different levels of input to the underlying learning algorithm. Three types of demonstrations are common:

- Teacher presents a set of (*state*, *label*) pairs
- Teacher presents a set of (*state*, *value*) pairs
- Teacher presents a sequence of (*state*) instances

In skill learning, a *label* would be a discrete action and a *value* would be a continuous action parameter. In addition to providing the *label* or the *value*, the teacher might be

directly or indirectly manipulating the *state* in the demonstration. We differentiate between the state of the robot and the state of the environment. Usually actions of the robot will have direct effects on the state of the robot, sometimes with a known relation (*e.g.* forward kinematics).

The third type of demonstration mentioned above is particularly interesting and common in the literature. Problems of this type can also be considered as policy learning, assuming the state sequence is produced using the optimal policy [94]. However in most LfD problems this data is sequential/temporal and the learner needs to reproduce the trajectory of states. One approach is to encode the trajectory globally over time [17, 29, 137].

The level at which a policy encodes actions has important implications. At one extreme, the learned policy needs to pick between alternative actions given a state. At the other extreme, the policy determines what the torque should be on all motors of an articulated robot given their current position. For the first type of policies it is easy to isolate a single demonstration as a state-action pair, whereas for the second type a demonstration consists of running through the whole trajectory of robot states. This is reflected in how the learned skills can be tested. The first type can be tested with individual states by the correctness of the chosen action, while the second type would be tested by the success of the reproduced trajectory in achieving the goal.

Interaction. Most LfD systems are evaluated using a set of demonstrations provided by one teacher (the creator of the LfD system) in a batch mode [16]. However recently, different mechanism that augment the interaction have been developed. One mechanism, similar to embodied queries developed in this thesis, are *state queries* which consist of the learner asking for an action suggestion in a certain state for which is not confident [40, 69]. Another mechanism is *corrective feedback*, which allows the teacher to provide demonstrations during the learner’s execution [95, 7].

Underlying learning problem. Despite the differences in how the teacher contributes to the input of the learning algorithm, the underlying ML problems in LfD is either a classification or a regression problem. Skill encoding can be considered as a special case of classification where the learner only gets positive examples (*e.g.* trajectories that achieve the goal), and the task is to reproduce or synthesize a trajectory, rather than label a new trajectory as positive or negative. Thus, skill encoding problems usually use generative models that let the learner retrieve prototypical positive examples.

Quality of examples in LfD. The concern about the limitations of demonstrations in LfD is elaborated in [8]. The two issues of undemonstrated state and poor quality data are pointed out and the approaches in the literature addressing both issues are reviewed. This thesis addresses the problem of increasing the accuracy of the learned model over all possible states as well as reducing the number of demonstrations needed to achieve a certain accuracy. This mostly overlaps with the issue of undemonstrated state.

2.2 *Interactive Machine Learning*

A recently growing community within ML and Human-Computer Interaction investigates the interaction between humans and systems that involve ML. The term Interactive Machine Learning (I-ML) was introduced in [53] to refer to the loop in which (i) the teacher provides new examples, (ii) the learner builds/update the classifier, (iii) the learner gives feedback in the form of classifications on a large unlabeled dataset. The feedback from the learner affects the teacher’s examples towards corrections of wrong classifications. Subsequent research in this area has extended I-ML to a great number of learning algorithms and systems [125, 54, 81, 65, 45]. A survey of different types of information that can be provided to the human teachers to increase their accuracy is given in [113].

Some efforts in I-ML are directed towards letting humans influence a classifier in ways other than presenting examples and labels. One example is the ManiMatrix system [72], that lets the user interact with the confusion matrix for a multi-class classifier to specify

preferences about errors. It automatically reflects these preferences on the underlying classifier. Another example is users changing weights of features of a classifier [128]. This involves an email classification system that provides *explanations* of a classification by highlighting the features (keywords) with high weight that contributed to the classification of an instance. Users can directly manipulate these weights and observe the results of their modifications. This work has been extended to *rule-based* and *similarity-based* explanations [127]. The authors analyze user critiques of the learner’s explanation of a classification, and propose and evaluate several new ways that the user can directly influence the classifier, such as the ability to add different features (*e.g.* new words), or feature combinations (*e.g.* two consecutive words as one feature).

A different line of research focuses on interactive games that learn from human players [66, 132, 78, 139]. These make observations about common patterns in human teaching (*e.g.* more positive feedback than negative, rewards diminishing over time) in order to augment the learning algorithms to work better with human teachers.

Thomaz introduced Socially Guided Machine Learning (SG-ML) [132, 133, 131] as a framework for designing agents that learn from humans. SG-ML advocates designing learners to be more amenable to human teaching. This involves designing interfaces to accommodate the ways that humans want to teach, as well as designing transparency mechanisms that lets the teacher monitor the state and progress of the learner.

2.3 Active Learning

Active Learning (AL) is a Machine Learning paradigm in which the learner chooses the examples from which it is going to learn [5, 42, 122]. This assumes an *oracle* who provides the labels for queried examples. Queries are chosen from potential unlabeled examples. In the original formulation by Angluin [5] it is assumed that any point in the feature space is a viable query (membership query synthesis), whereas in practice only parts of the feature space are meaningful. Thus queries are often chosen from a pool of unlabeled examples.

AL techniques are often evaluated in terms of the reduction in number of labels required to learn a sufficiently accurate classifier over a passive learner that receives random examples. There is substantial evidence that AL can achieve significant gains in this respect, especially in applications involving text, image, and video classification where label acquisition is costly but unlabeled data is abundant [122]. For several simple concept classes it is theoretically shown that AL provides benefits in the sample complexity. However for more complex concept classes (*e.g.* non-homogeneous linear separators) it is shown that AL gives no or only asymptotic advantage over random sampling [47, 14].

For evaluation of new techniques in AL literature it is common to use an *automated oracle* rather than an actual person [136, 43, 46]. However a few user studies have been performed to evaluate AL systems, *e.g.* for estimating the cost of different detail levels of annotations or queries [121, 138], the feasibility of different types of queries (*e.g.* feature queries or multiple-instance queries) [51], or the effectiveness of different user interfaces while using non-expert annotators as oracles [51, 45].

Gervasio *et al.* investigated question asking in procedure learning with a range of question types [55, 56].

2.3.1 Active Learning in Robotics

The potential of AL for making learning efficient has been noticed in the Robotics community; evidenced by recent a special issue [88], and a workshop on AL at the Robotics Science and Systems conference [90]. *Confidence based autonomy* [40] and *dogged learning* [61], use the principle of uncertainty sampling to select states in which the learning agent requests a demonstration while learning a policy. In [58], the robot actively selects points outside the region of stability of a learned policy, and requests demonstrations from these states. Similarly, Lopes *et al.* use AL to select the states in which an expert human should be queried for an appropriate action assuming an Inverse Reinforcement Learning agent [87].

The evaluation of robotic systems that ask questions to humans has been limited, particularly in LfD settings. Rosenthal *et al.* investigate how augmenting questions with different types of additional information improves the accuracy of human teachers' answers [112, 113]. In later work, they explore the use of humans as information providers in a real-world navigation scenario [114]. Other related work explores the idea of learning actions on a robot through dialog [33].

A different notion of AL in robotics is to consider the environment as the teacher. In this case one data point is obtained by executing an action in the environment and perceiving an outcome, *i.e.* performing experiments. Applying AL in this context corresponds to intelligently selecting the actions such that the model can be learned faster than randomly executing actions (a.k.a. motor babbling). Some examples of this include learning to choose grasping points [82], sensorimotor learning of primitive actions such as bashing or biting [106] or locomotion skills on an Aibo robot [105], or learning a model of the robot's dynamics for control [111].

2.4 Algorithmic Teaching

ML algorithms are often studied assuming that data comes from a fixed but unknown distribution or an adversarial teacher. However, a good teacher can produce a dataset that results in a more accurate output with less examples. Producing such a data set for an arbitrary concept is a challenging problem and has been formally studied within the field of algorithmic teaching [13, 92, 57]. Having a helpful teacher can considerably improve the learning rate. For instance, even an active learner can never learn faster than when a passive learner is taught optimally [5, 57].

One goal of algorithmic teaching is to identify metrics that indicate the teachability of concepts [99, 57, 6, 11]. The most common measure of teachability is the *teaching dimension* [57], which is the minimum number of labeled examples needed to uniquely identify *any* concept in a concept class. A sequence of examples that uniquely identifies

a target concept is called a *teaching sequence*; the shortest such sequence is called an *optimal teaching sequence* for a given concept. Finding an optimal teaching sequence for an arbitrary concept is NP-hard (by reduction to the minimum cover set problem [57]) however efficient algorithms have been proposed for particular concept classes.

[99] uses a different measure of teaching complexity which is the number of examples that must be given until the most specific hypothesis consistent with the data is the target concept. As in the teaching dimension, the difficulty of teaching a concept class is measured by the concept in the class that is *hardest* to teach. The *average teaching dimension* [6] captures the overall difficulty of teaching a concept class. [11] propose two variants of the teaching dimension where (i) the learner is biased towards less complex solutions or (ii) the learner assumes that the teacher is optimal.

Concepts for which efficient teaching algorithms have been proposed include conjunctions, monotone decision lists, orthogonal rectangles, monotone K-term DNFs among others [57]. Having certain assumptions about the learner [11, 143], information exchange prior to the teaching session [67], or feedback from the learner during teaching [12] can make certain classes teachable in polynomial time or reduce their teaching dimension. However for more expressive concepts there are no efficient teaching algorithms. For such concepts it is more natural to talk about approximately optimal teaching algorithms. Consequent research in the field has illustrated that learning from a good (but not necessarily optimal) teacher is advantageous. The notion of *helpful distributions* is proposed in [48], where a distribution is helpful if all examples in the optimal teaching set of a concept have non-zero probability to be drawn in this distribution. In [11], a greedy teacher that picks an example that is inconsistent with the most probable hypothesis of the learner is shown to be optimal in some cases.

2.5 *Human Learning and Teaching*

Human Teaching. Different forms of learning and adaptation are observed in a variety of species in nature. The *teaching behavior*, on the other hand, is not very common, and is mostly confined to humans with a few exceptions [44]. Teaching was defined by [34] as having three properties: (i) it occurs only in the presence of a naive observer, (ii) it is altruistic (it is costly to the teacher and does not provide immediate benefits), and (iii) it facilitates *knowledge acquisition* or *skill learning* in the observer. Human teaching has been studied experimentally on different teaching tasks and a number of theoretical accounts and formalisms have been proposed.

The *pedagogical sampling* framework introduced by Shafto and Goodman assumes a benevolent teacher and learner [124]. In this framework, as in the Teaching Dimension model of Goldman [57], the teacher chooses data that will be most informative for the learner in inferring the target hypothesis. The learner tries to infer the correct hypothesis assuming that the teacher is benevolent. Shafto *et al.* present an experiment in which participants teach axis-parallel rectangles by showing a specified number of examples. They find a strong correlation between the examples predicted by pedagogical sampling and the examples chosen by human teachers. Note that this does not imply humans were optimal teachers in these experiments, because the examples given by the participants are compared to specific pedagogical sampling cases (*e.g.* the most informative examples given that there are two positive examples), rather than a globally optimal teaching sequence for axis-parallel rectangles. However the strong correlation suggest that the examples chosen by humans were highly informative to the learner. Warner *et al.* [141] extends the framework to cases where the benevolent teacher assumption is removed and the intent of the teacher needs to be inferred by the learner.

Khan *et al.* performed an experiment about teaching a threshold in 1D [75]. They characterize three teaching strategies used by humans: *linear* (starting from one end of the range and moving towards the other linearly), *positive-only* and *extreme* (starting from the

ends of the range and alternating between classes). They observe that most humans use the *extreme* strategy. Consistent with our observation that spontaneous optimal teaching is rare, they observed no instances of a *boundary* strategy. They present a theoretical account of the predominant strategy based on the assumption that humans represent examples in high dimensional spaces rather than just their 1D projection. This justifies the presented example sequences as most informative. The teaching strategy of starting with easy examples (far from the boundary) and gradually increasing the difficulty of examples (getting closer to the boundary) is also referred to as scaffolding [91] or *curriculum design* [15, 126, 130]. The example sequences produced with this strategy are indeed favorable for certain learners [52, 49, 84].

Human Learning. There are several recent related works analyzing human learning with respect to ML. [142] measures the Rademacher complexity of humans and shows their learning error is bounded by this metric in similar ways as ML algorithms. [74] proposes a model of human concept complexity, and show that this model predicts people’s ability to learn and make use of a concept. This notion of human concept complexity is complementary to the human teaching complexity we explore here. In particular, we see that people’s ability to make use of teaching guidance varies depending on the teaching dimension of the concept (Sec. 9.4.2), which may be related to its subjective complexity. Similarly, [35] analyzes human active learning, and shows a significant improvement over passive learning, but not as large as the improvement predicted theoretically.

These principled characterizations of human learning are highly relevant to the approach proposed in this thesis, as humans’ teaching is inevitably influenced by their learning. Human learning is thought to be mediated by multiple qualitatively distinct systems, depending on the target concept [9]. For example, *rule-based tasks* involve concepts that are easy to verbally describe, while *information integration tasks* involve concepts that are

difficult to articulate even though humans can learn them (*i.e.* correctly predict concept membership). Different learning problems considered in this thesis fall into different categories.

2.6 Relation to present work

This thesis primarily contributes to research on Learning from Demonstration in Robotics. While previous work in LfD has focused on representations, learning algorithms and demonstration modalities, the focus in this thesis is on the interaction through which demonstrations are acquired and the quality of demonstrations obtained through this interaction. The developed interaction techniques are potentially applicable to different LfD systems. In addition, unlike most previous work, this thesis advocates evaluating LfD systems with users other than the programmer of the system. The methods developed in this work are potentially useful for learning agents other than robots, and are complementary to many of the interaction methods developed within the field of I-ML. In particular, this thesis strongly adheres to the principles of SG-ML. The interaction mechanisms proposed in this work are based on theoretical and applied research on AL and Algorithmic Teaching, and might also contribute some extensions to the methods borrowed from these fields.

CHAPTER III

EXPERIMENTS ON HUMAN TEACHING

This thesis focuses on the scenario of a human user teaching a new skill or task to a robot. The main research method used in studying problems in this scenario is human subject experiments. This involves two types of experiments.

- *Observational experiments* involve observing interactions between humans and robots within a certain context to identify patterns. These are *explorative*, *i.e.* their outcomes cannot be predicted, and they are *formative*, *i.e.* they guide the design of follow up experiments and reveal new research questions.
- *Controlled experiments* involve comparisons of multiple conditions so as to measure the effect of certain parameters. These are associated with hypotheses that can be tested.

The experiments involve a variety of robot or virtual agent platforms, learning problems, domains and interaction modalities. This chapter presents preliminaries that support the experiments presented throughout the subsequent chapters. First, the robot and software platforms are introduced. Then the two main learning problems considered throughout this thesis, task learning and skill learning, are motivated and explained. This includes the particular learning algorithms and the domains for each learning problem. Finally an overview of all experiments is given.

Note that not all experiments involve the two main learning problems described in this section. Any learning problem or domain that is used only within a single experiment is described as part of the experimental design of that experiment.

3.1 Robot platforms

Two robot platforms were used in the experiments, Simon and Junior. Both robots are controlled through an agent behavior software called Creatures-6 (C6). These platforms are detailed next.

3.1.1 Simon

Simon is an upper-torso humanoid with two 7 degree of freedom (DoF) arms, two 4-DoF hands, a 2-DoF torso, and a socially expressive head, including a 4-DoF neck, two 2-DoF eyes, coupled 1-DoF eyelids on both eyes, and two 2-DoF ears with full RGB spectrum LEDs (Fig. 2(a)). Simon is specifically designed for face-to-face social interaction with humans [50].

Unlike many of the common commercial robot arms, Simon’s 7-DoF arms have a close-to-human layout, with three approximately aligned DoFs each on the shoulder and the wrist, and one DoF on the elbow. The actuators on the arm are series-elastic actuators [109], which make the arms compliant to external forces and safer for human-interaction. Simon’s arms are used in two different modes: in the position mode the arms maintain a commanded position or follows a trajectory with high stiffness; and in the gravity compensated torque mode the arms are passive and can be directly controlled by a human.

Simon’s hand are also controlled in two modes. A constant torque is applied for grasping objects. The compliant hands naturally match the shape of the object in this mode. For gesture such as pointing the fingers are position controlled.

Simon has several sensors both egocentric and environmental. The ones used within the experiments in this thesis are the cameras in the eyes for tracking faces or gazing at objects on a table top, and a top-down facing camera on top of the Simon’s workspace for tracking objects in teh workspace. Note also that Simon has proprioceptive sensors such as torque and position sensors on the motors, that allow it to sense external forces while in the position control mode.

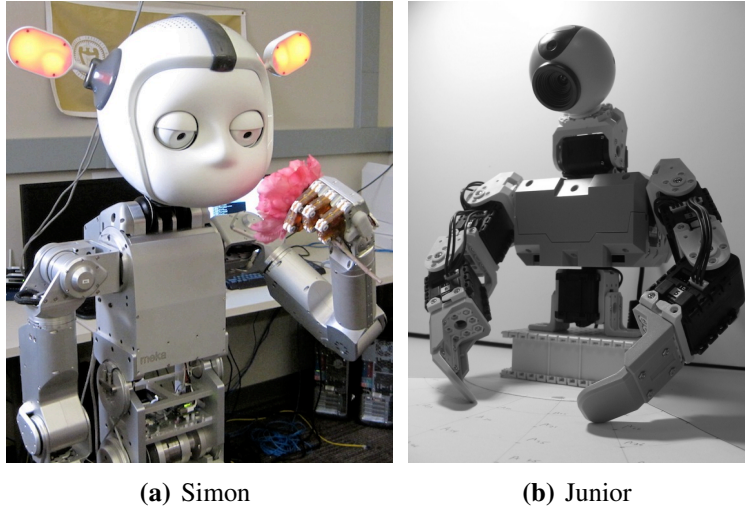


Figure 1: The two robot platforms used in the experiments across this thesis.

3.1.2 Junior

Junior is a Bioloid robot configured as an upper torso humanoid with a Logitech Webcam head (Fig. 2(b)). It is approximately 10 inches high. It has a total of 8 DoFs: 3 DoFs in each arm, and a torso rotation DoF and a neck tilt DoF.

The torso rotation and the neck tilt allow Junior to look around at objects on the table. The two arms allow manipulating objects that are within a certain range. With single arm actions, Junior can poke objects from either side to move, roll or tip them. With coordinated two arm actions, Junior can pick up objects and drop them.

3.1.3 The Creatures Software Architecture

Simon and Junior are controlled through a behavior system implemented in the Creatures-6 software architecture, a descendant of [18]. The behavior system takes input from sensory modules, such as speech recognition or vision; and it outputs motor commands to the robot. Communication between modules is achieved with packets using the Intra-Robot Communication Protocol (IRCP). C6 uses accurate 3D kinematic models of the robot platform which are animated in an OpenGL window. This allows viewing what is commanded to the robot for debugging, before actually commanding the physical robot.

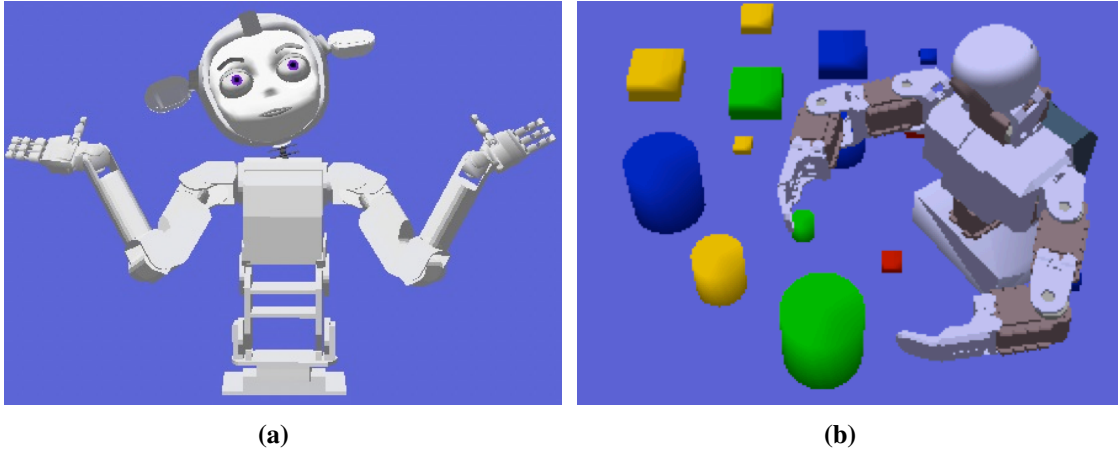


Figure 2: Snapshots from the robot models animated in C6. (a) Simon (b) Junior

The behavior system involves a pipeline of sub systems. The perception and belief systems label and fuse sensory input into beliefs, represented as a bag of features. The action system implements a state machine whose transitions may be controlled by these beliefs and trigger new actions. Actions are converted by the motor system, to sequences of motor commands determined by a pre-designed animation, or by learned or planned trajectories. Specific learning algorithms and representations will be introduced as needed throughout the thesis.

For the Simon platform, movements can also be specified in the configuration space (*i.e.* with end-effector trajectories). C6 computes the wrist configuration for a desired end-effector configuration and then computes the inverse kinematics for the shoulder and elbow to achieve the desired wrist configuration using Cyclic-Coordinate Descent (CCD) [140]. Finally the values for the three wrist joints are computed analytically. Once an end-effector trajectory is converted to a joint trajectory, it is checked for collisions between the two arms and the torso.

3.2 *Task Learning*

In order to have a robot perform a new task, the user needs a way to communicate the goal of the task. A goal is a world state that has certain desired properties. In some cases communicating the goal is trivial; for instance a navigation task only requires a target location which can be referred to with a coordinate or a reference name (*e.g.* the kitchen, the copy room). However, in many cases communicating the goal with pre-specified commands is not possible due to the differences in the way that the robot and the humans represent a task. Consider the task of emptying the dishwasher. Every home has unique rules as to what should go where, and in many cases it is difficult for a person to articulate these rules. Furthermore, every home has a different set of kitchenware. It is challenging to have universal recognition and referencing of items and places (*e.g.* the sentence “place the small bowls into the lower left cupboard” could mean something different in different kitchens), as well as to cover the set of objects that robots will encounter (*e.g.* “the whisk adapter of the hand blender”). Therefore, robots are likely to have a representation of objects based on features that have no meaning or name for humans, and communicating a goal becomes a challenge.

When the goal cannot be directly commanded to the robot, it can be taught to the robot through demonstrations. This allows the robot to have a representation of the goal in its own feature space. This is the problem of learning *what* to do in order to accomplish a task, which we refer to as *task learning*. The challenge in this problem is to form a general representation of the goal given a few demonstrations. This involves discovering the features of the world state that are relevant to the task. The robot should learn that things that are varied across the demonstrations are not relevant and form a model of what needs to be true of the world state for the goal to be accomplished. A number of different concept or function representations can be used for this model.

3.2.1 Concept Learning with Discrete Features

Several experiments in this thesis investigate the problem of teaching task goals by providing positive and negative examples of states that satisfy the goal. This problem is also referred to as *concept learning*. In particular, goals are represented as a *conjunction* of feature values required for the goal to be accomplished.

3.2.1.1 Problem Description

The problem of learning a task goal is essentially learning a concept description or classifier that labels a world state as positive or negative in order to indicate whether it satisfies the goal. Given a set of examples, $\{\mathbf{s}_i, \ell_i\}_{i=0}^N$, where \mathbf{s}_i is a d -dimensional vector of discrete features (s_1, s_2, \dots, s_d) and $\ell_i \in \{0, 1\}$, the learner constructs a function $g : \mathbf{s}_i \rightarrow \ell_i$ that predicts whether a world state satisfies the goal or not.

A conjunction is a list of feature values that must all be true in a state, for it to be labelled as positive. Features that appear in the conjunction are referred to as *relevant*, while all the other dimensions of the sample space are called *irrelevant* features. *Monotone* conjunctions allow only one value of each feature to be present in the conjunction; *e.g.* in the case of binary variables all elements of the conjunction need to be positive. We focus on *non-monotone* conjunctions, which allow more than one value of a feature to be in the conjunction. World or object states represented with a set of discrete features are considered.

The learning of conjunctions from examples is a well-studied concept learning problem. A concept learning algorithm produces a *hypothesis* based on a set of labeled examples consistent with an unknown target concept. In our experiments the learner's hypotheses are of the same form as the target concepts (*i.e.* non-monotone conjunctions). This is referred to as learning in the *realizable case* (as opposed to the *agnostic case*) and it allows exact learning of the target concept. The learner tries to estimate the target concept by producing a hypothesis that is consistent with the given examples. The learner chooses its hypothesis

from the *hypothesis space* — the set of all possible hypotheses.

3.2.1.2 Algorithm

We use a concept learning algorithm based on version spaces. The *version space* is traditionally defined as the subset of the hypothesis space that contains hypotheses consistent with all labeled examples provided to the learner [85]. In order to accommodate noisy data and errors in labeling, we quantify the *consistency* of any given hypothesis as the number of seen examples with which it is consistent, and we define the *version space* to be the set of hypotheses having the highest consistency value among all members of the hypothesis space. This relaxes the requirement that hypotheses be consistent with every previously seen labeled example, allowing the learner to recover from labeling mistakes or perception errors.

The learning algorithm updates the consistency value of each hypothesis after receiving a new example and re-expands the version space. This algorithm is summarized in Algorithm 1. For efficiency, our implementation does not expand the complete hypothesis space but rather considers all hypotheses that are consistent with positive examples. A generalized version of the algorithm is presented here for clarity.

Algorithm 1 The concept learning algorithm used in Experiment 1. \mathcal{H} denotes the set of all hypotheses, and \mathcal{V} denotes the version space.

```

loop
  Obtain new labeled example  $(\mathbf{s}_i, \ell_i)$ 
  for  $h_j \in \mathcal{H}$  do
    if  $h_j$  is consistent with  $(\mathbf{s}_i, \ell_i)$  then
       $\text{consistency}(h_j) \leftarrow \text{consistency}(h_j) + 1$ 
    end if
  end for
  Initialize  $\mathcal{V} \leftarrow \emptyset$ 
  for  $h_j \in \mathcal{H}$  do
    if  $\text{consistency}(h_j) = \max(\text{consistency}(\mathcal{H}))$  then
       $\mathcal{V} \leftarrow \mathcal{V} \cup h_j$ 
    end if
  end for
end loop

```

At any time during learning, the current version space can be used to predict the label of a new instance by serving as a committee. The decision is based on the majority in the predictions of the hypotheses in the version space. The confidence in the prediction is defined as the distance of the majority label to the average label. Thus for the degenerate case of having no majority label, the confidence is 0.

3.2.2 Active Concept Learning

The core of all AL algorithms is the method for selecting the query from a set of candidate unlabeled samples. Ideally, this mechanism selects samples that are maximally informative for the learner and thus significantly reduce the number of required labels to create a good model.

The query selection method used for the concept learning algorithm for conjunctions, is called *query-by-committee* [5, 123]. This method uses a committee of competing hypotheses and selects the example that results in maximal disagreement between the hypotheses in terms of predicted labels. The committee in this implementation is the version space. For a two class classification problem, the most uncertain queries are the samples for which the committee is split in half. This means that when the label is acquired, the learner guaranteed that half of the version space will be pruned.

3.2.3 Domains

3.2.3.1 Toy-objects domain

In the first teaching scenario considered for task learning with Simon platform, the robot works with a human partner at a tabletop workspace. The learned tasks involve colorful paper cutouts, which are called *object parts*. Each object part has a *color* (pink, green, yellow, or orange), a *shape* (square, triangle, or circle), and a *size* (small or large) feature, for a total of 24 possible unique parts. Tasks involve objects that are composed of two parts, a top and a bottom part. Therefore an object has six features: $color_{top}$, $shape_{top}$, $size_{top}$, $color_{bottom}$, $shape_{bottom}$, and $size_{bottom}$. Top and bottom are from Simon’s perspective. A nine

feature version of this domain involves three additional features. These are binary features indicating whether the top and the bottom have the same color, shape or size (e.g. $size_{same}$ is *true* if $size_{top}$ is equal to $size_{bottom}$, and *false* otherwise). The distance between the parts and their orientation are ignored.

Although no task executions are involved in task learning experiments, performing a given task would consist of assembling a target toy object from two parts available in the environment. Simon’s workspace contains exactly one of each part, so there are 552 (24×23) possible compound objects.

Humans teach Simon different object categories, which are also referred to as object concepts. An object category is represented as a non-monotone conjunction of compound object attribute values that must hold true to be a member of that category. For instance, a HOUSE is a compound object that has a pink and triangular top piece and a square bottom piece. The size of either piece and the color of the bottom piece do not matter. Thus, the HOUSE category is represented with the conjunction $\{color_{top} = pink \wedge shape_{top} = triangle \wedge shape_{bottom} = square\}$. This can also be denoted with $\langle pink, triangle, *, *, square, * \rangle$ where $*$ means that the value does not matter, assuming the order of features to be as given in above.

While teaching object categories, humans interact with Simon through a fixed set of speech commands. When a command is given, Simon expects to find exactly two object parts in the demonstration area. Object parts are detected through a fixed overhead camera and segmented using background subtraction. The shape of the part is recognized by the number of corners of the simplified polygon contour of the segmented object (square: 4, circle: 8, triangle: 3). Size is based on the area within the contour and color is recognized using the color histogram of the segmented object. The parts are localized in robot world coordinates using a fixed homography from the image plane to the table plane.

Two sets of object concepts are considered in our task learning experiments. The first set consists of four concepts that all have the same specificity in the six-feature state space.

Table 1: Object categories of fixed specificity for task learning in the six feature toy objects domain.

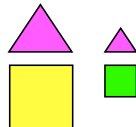
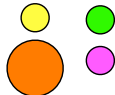
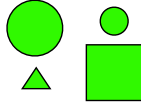
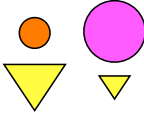
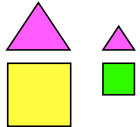
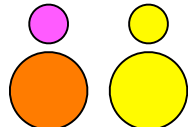
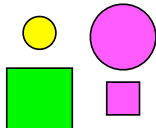
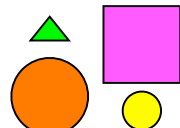
Category	Conjunction	Examples	# of Instances
HOUSE	$shape_{top} = triangle \wedge$ $color_{top} = pink \wedge$ $shape_{bottom} = square$		16
SNOWMAN	$shape_{top} = circle \wedge$ $size_{top} = small \wedge$ $shape_{bottom} = circle$		28
ALIEN	$shape_{top} = circle \wedge$ $color_{top} = green \wedge$ $color_{bottom} = green$		10
ICE CREAM	$shape_{top} = circle \wedge$ $shape_{bottom} = triangle \wedge$ $color_{bottom} = yellow$		16

Table 2: Object categories of varying specificity for task learning in the nine feature toy objects domain.

Category	Conjunction	Examples	% of instance space
HOUSE	$shape_{top} = triangle \wedge$ $color_{top} = pink \wedge$ $shape_{bottom} = square \wedge$ $size_{same} = true$		1.45%
SNOWMAN	$shape_{top} = circle \wedge$ $size_{top} = small \wedge$ $shape_{bottom} = circle \wedge$ $size_{bottom} = large$		2.89%
PERSON	$shape_{top} = circle \wedge$ $shape_{bottom} = square$		10.87%
RAINBOW	$color_{same} = false$		75.00%

They all involve three relevant features. These concepts are shown in Table 1. The second set of concepts consist of four concepts of varying specificity in the nine-feature state space. These have different numbers of relevant features in the conjunction that defines the concept. These concepts are shown in Table 2.

3.2.3.2 Examples

The learning algorithm described in Sec. 3.2.1 is exemplified in Table 3 to illustrate how the version space changes as labeled examples are received in the toy objects domain. Table 4 shows examples of labels predicted by the version space committee after the learner has seen the three labeled examples shown in Table 3. In the first example, all of the hypotheses in the version space agree on the label of the example, so the confidence is 1. In the second

Table 3: Progress of the version space in the concept learning algorithm used in task learning experiments, illustrated for a sequence of three labeled examples for the HOUSE concept.

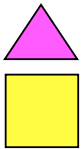
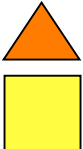

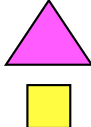
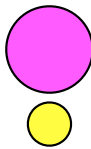
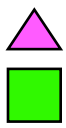
Step	Example	Label	Version Space
1	 <pink,triangle,large, yellow,square,large>	+	<i>Most specific</i> <pink,triangle,large yellow,square,large>
			<i>Most general</i> <pink,*,*,*,*,*> ... <*,*,*,*,*,large>
2	 <orange,triangle,large, yellow,square,large>	-	<i>Most specific</i> <pink,triangle,large yellow,square,large>
			<i>Most general</i> <pink,*,*,*,*,*>
3	 <pink,triangle,small, yellow,square,small>	+	<i>Most specific</i> <pink,triangle,* yellow,square,*>
			<i>Most general</i> <pink,*,*,*,*,*>

Table 4: Label prediction based on the version space for the concept learning algorithm used in task learning experiments exemplified for the HOUSE concept.

Example	Version Space Predictions	Total Votes	Prediction
	$\langle \text{pink, triangle, *}$ $\text{yellow, square, *} \rangle \rightarrow +$		
	...	+: 8 -: 0	+
$\langle \text{pink, triangle, large}$ $\text{yellow, square, small} \rangle$	$\langle \text{pink, *, *, *, *, *} \rangle \rightarrow +$		
	$\langle \text{pink, triangle, *}$ $\text{yellow, square, *} \rangle \rightarrow -$		
	...	+: 2 -: 6	-
$\langle \text{pink, circle, large}$ $\text{yellow, circle, small} \rangle$	$\langle \text{pink, *, *, *, *, *} \rangle \rightarrow +$		
	$\langle \text{pink, triangle, *}$ $\text{yellow, square, *} \rangle \rightarrow -$		
	...	+: 4 -: 4	?
$\langle \text{pink, triangle, small}$ $\text{green, square, small} \rangle$	$\langle \text{pink, *, *, *, *, *} \rangle \rightarrow -$		

example, 6 out of 8 hypotheses label the example as a non-member, so the predicted label is negative, and the confidence is smaller than 1. In the last example, there are equal votes for positive and negative, so the confidence is 0, and the predictor outputs no label.

The query mechanism can also be exemplified in the toy objects domain. Consider the state of the version space after seeing the three labeled examples given in Table 3. As shown in Table 4, some of the examples in the instance space are classified with full confidence. For other examples, the version space is split equally in terms of the predicted label — that is, half of the hypotheses are inconsistent with the example. When the version space is split in such a manner, the result of labeling the example is to halve the size of the version space.

In other words, there is maximal disagreement on this example, and therefore it is a good query candidate.

3.3 *Skill Learning*

In order to accomplish goals assigned by humans, robots need a set of skills. In some cases pre-coded skills are sufficient in accomplishing the range of goals that a robot is assigned. In other cases, the robot has an accurate model of the effects of its actions, and it can plan a sequence of actions that achieves the goal (*e.g.* using motion planning for a pick and place with a manipulator). However for many skills that users might want robots doing, it is not trivial to specify a goal that allows planning. Instead the user should be able to directly program a skill by demonstrating it. This is the problem of learning *how* to accomplish a task goal, referred to as *skill learning*. This complements the problem discussed earlier of learning *what* the goal is.

Skill learning involves constructing a model of the skill based on demonstrations of successful executions of the skill [8]. Experiments in this thesis involve kinesthetic teaching, which lets humans provide demonstrations to the robot by physically guiding the robot. Kinesthetic teaching has several advantages when compared to other demonstration techniques, such as avoiding the correspondence problem and being restricted to the robot's capabilities. The teaching process involves humans moving the robot's arm to demonstrate the skill, while the arm is passive in a gravity compensation mode.

State spaces. A skill encodes the trajectories of states that the robot needs to go through to accomplish a certain goal. The state space can differ based on the domain and properties of the robot, however it is assumed that the robot has controllers that allows it to attain a desired state such that skills can be executed. Alternative state spaces exist for representing the state of a robotic manipulator.

- *Joint space*: The state of a manipulator is fully specified with the states of its joints (position, velocity and acceleration). This state representation has the advantage of easy control since it involves directly setting joint torques that achieve the desired joint states. The disadvantage is that it has no reference to the environment and it is robot specific. As a result it is not possible to learn generalizable skills that involve a target object, *i.e.* skills that work for different configurations of the target.
- *Task space*: For many skills only the state of the end-effector is important in achieving the goal. Therefore the state of the manipulator can be represented with the position, velocity and acceleration of the end-effector. The coordinate frame is often chosen as a fixed frame on the robot, *e.g.* the root of the kinematic chain. Skills represented in this state space require the ability to control the end-effector. For instance, the robot may use an inverse kinematics solver which calculates the joint positions that achieve a certain end effector configuration. Skills represented with this state space are independent of the particular robot manipulator, provided that the end-effector configurations required by the skill are reachable for the robot. However, these skills are still not generalizable to different target configurations.
- *Relational task space*: A third alternative is to represent the state of the manipulator with the end-effector configuration in a reference frame that is tied to a target in the robot's environment. Such targets may move independently relative to the robot. These skills are generalizable to different target configurations as long as the configurations required by the skill are reachable for the robot. In addition these skills are independent of the particular robotic manipulator, if reachability is granted. Skills represented using this state space require robust perception of the target configuration, as well as the capability to change the robot's base configuration relative to the target (*e.g.* by manipulating the target, or by moving the robot's base).

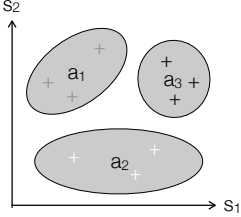
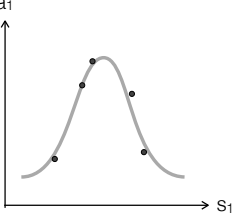
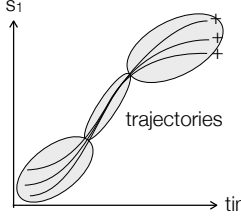
Policy learning with discrete actions: Classification	Policy learning with continuous actions: Regression	Trajectory encoding: Time series modeling
Demonstration: (state, label) Usage: (state, ?)	Demonstration: (state, value) Usage: (state, ?)	Demonstration: (state) ₁ , ..., (state) _t Usage: (?) ₁ , ..., (?) _t
		

Figure 3: The different Machine Learning problems involved in different skill models.

This thesis focuses on skills represented with relational task spaces for the benefits mentioned above. For illustrative purposes, skills are represented in the joint space in Experiment 2, however methods developed in Chapter 8 assume a relational task space representation. Only the position and rotation of the end-effector are used in the representation, excluding velocity and acceleration of these values. This limits the types of skills that can be learned. For instance, skills like playing ping pong or throwing a ball into a basket cannot be learned with only static properties of the manipulator. It is assumed that there is a single target for a given skill and that the target does not change during demonstrations or executions.

Skill models. A skill model can have different forms. For instance, it can be a policy that maps the current state of the manipulator to a next state (*e.g.* [76]). Trajectories of states can be obtained by iteratively applying the policy. Actions in this representation can be discrete primitive actions [39, 61] or continuous actions [76, 10]. Another representation of a skill is a generative model over the space of trajectories which allows sampling trajectories directly (*e.g.* [30]). This thesis involves the second type of skill models. The different skill models involve characteristically different Machine Learning problems illustrated in Fig. 3.

Demonstration types. A demonstration is a successful execution of a skill in which the state or the robot (or equally the actions of the robot) is controlled by a human teacher. Two alternative ways of demonstrating skills are considered.

- *Trajectory demonstrations (TD)* involve the teacher indicating the start and end of a demonstration and providing a continuous time series of states in between. The sequence is obtained by sampling the state at a relatively high constant frequency.
- *Keyframe demonstrations (KD)* involve the teacher providing a sparse sequence of targets (*i.e.* keyframes) that the robot needs to go through in order to accomplish the skill. The robot only records the states associated with keyframes and ignores any movement in between.

Both types of demonstrations provide a sequence of states. The main difference between the two types is that TDs have time information associated with states, while KDs do not. In addition KDs are typically much sparser than TDs.

To represent skills provided in the form of TDs, Gaussian Mixture Models (GMMs) are used with the learning algorithm introduced in [32]. As a preprocessing step, the time series subsampled to a constant length. The GMM is learned in a $d + 1$ dimensional space, where d is the dimensionality of the state space, to include an additional dimension of time. First, all points in the demonstration are clustered in this $d + 1$ dimensional space, using the K-means algorithm with a constant k . The resulting clusters are used to calculate initial means and covariance matrices for the Expectation-Maximization (EM) algorithm that estimates the GMM that best fits the data. To reproduce the learned skill, the learner can sample the learned GMM model using Gaussian-Mixture Regression (GMR). This involves obtaining a conditional distribution over states given time, and then sampling a state at each time step (or directly using the mean of the conditional model).

Skills learned from KDs consist of a sequence of keyframe models. This thesis contributes a novel method for learning from KDs, which is described in the following. A

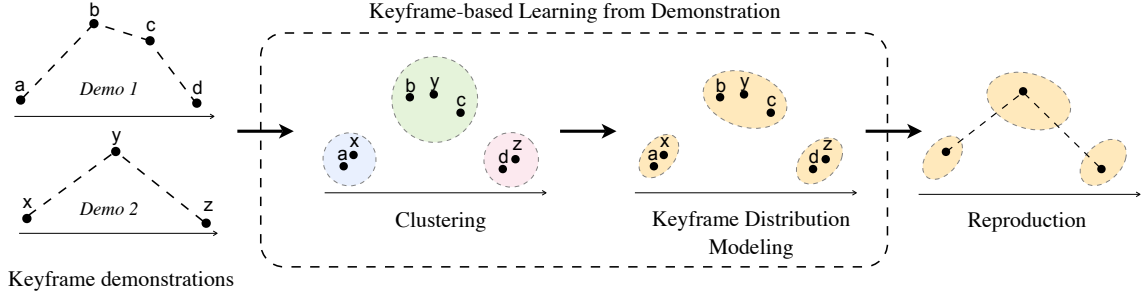


Figure 4: Overview of Keyframe-based Learning from Demonstration.

method for learning from *hybrid* demonstrations, *i.e.* demonstrations that involve both trajectory segments and keyframes, is described in [3].

3.3.1 Keyframe-based Learning from Demonstration

The input to the learning algorithm is a set of N keyframe demonstrations, $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^N$. Each demonstration is a sequence of n_i states (*i.e.* keyframes), $\mathcal{D}_i = (s_1, \dots, s_{n_i})$. It is assumed that the state space is the end-effector configuration relative to a target frame of reference, however other state spaces are possible. The output of Keyframe-based Learning from Demonstration (KLfD) is a sequenced set of Normal or Gaussian Distributions, $\mathcal{S} = \{\mathcal{N}_k : k = 1, 2, \dots, n\}$. These are referred to as *keyframe distributions*. This representation captures the number of keyframes of the skill and how each keyframe is distributed. A trajectory for the skill can be generated from this representation by sampling one point from each keyframe distribution and moving through these points in sequence. An overview of the KLfD framework is given in Fig. 4.

3.3.1.1 Alignment and Clustering

Given a set of demonstrations that all have the same number of keyframes, it is clear that the skill should also have that number of keyframe distributions. Assuming that the teacher intends to give a consistent set of keyframes across demonstrations, the median of the demonstrations is taken as the number of keyframes of the skill, *i.e.* $K = \text{median}(\{n_i : i = 1, \dots, N\})$. In order to model the keyframe distribution, a cluster corresponding to each

keyframe is created and keyframes of each demonstration are assigned to a cluster.

Clustering and alignment are initialized with the demonstrations that have K keyframes. The i th keyframe in these demonstrations are assigned to the i th clusters. For demonstrations that have more or less keyframes, we iterate over the keyframes in sequence and assign them to one of the clusters. To maintain ordering constraints in this process, a keyframe can only be assigned to a cluster that is the same as or comes after the cluster of the previous keyframe in the demonstration. After this process the set of demonstrations \mathcal{D} is re-grouped into K clusters as $\{\mathcal{D}_k : k = 1, \dots, K\}$.

3.3.1.2 Keyframe models and skill reproduction

Data from each cluster is used for modeling the corresponding keyframe distribution. This is simply the empirical mean and covariance of the cluster, $\mathcal{N}_k = P(s|D_k) = \mathcal{N}(\mu_{D_k}, \Sigma_{D_k})$. A reproduced skill is a sequence of points (s_1, \dots, s_K) sampled from the keyframe distributions, *i.e.* $s_k \sim \mathcal{N}_k : k = 1, \dots, K$.

To reproduce the skill, the robot moves between keyframes with a constant velocity on a straight line in the Euclidian state space. If the distance between two consecutive keyframes is smaller than a threshold the speed is reduced with a factor that is inversely proportional to the distance between the keyframes. As a result, keyframes placed close to one another are traversed with slower movements. The velocity has a fixed value between any keyframes that are separated with more than the threshold. This choice in the skill reproduction was based on an attempt to make the reproduction of individual demonstrations successful.

3.3.2 Domains

Three skill domains used across skill learning experiments are described in the following.

3.3.2.1 Single-handed goal-oriented skills

The first type of skill involves manipulation skills that try to achieve a particular goal state, *i.e.* goal-oriented skills. These use a single manipulator of the Simon robot, possibly while

certain objects or tools are being held by the compliant hand. The skills from this category used in skill learning experiments in this thesis involve the following. Snapshots from the execution of these skills are shown in Fig. 5.

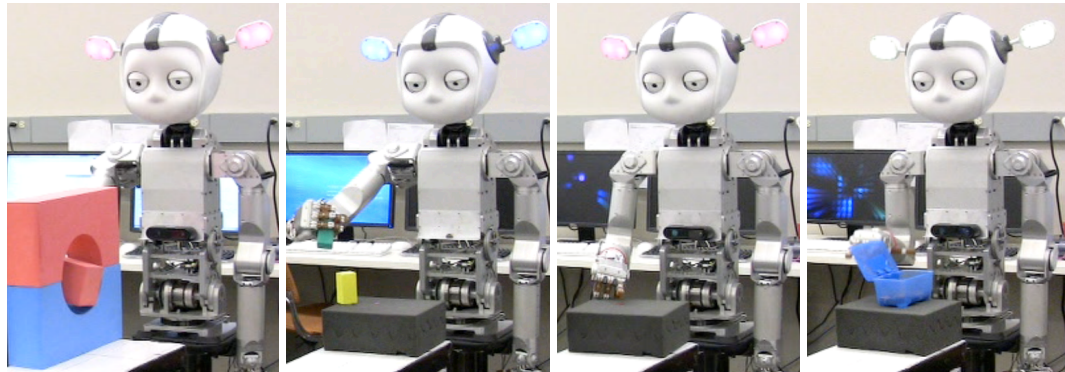
- *Insert*: A block in the hand is inserted through the hole without touching other blocks.
- *Stack*: A block in the hand is stacked on top of another block on the table.
- *Push-button*: A button on the table is pressed with the finger tip.
- *Close-box*: The lid of a box is closed without moving it.
- *Pour-cereal*: Move the cereal box in the hand as to pour cereal into a bowl on the table.
- *Add-salt*: Move the salt shaker in the hand as to pour salt into a salad bowl on the table.
- *Pour-coke*: Move the coke bottle in the hand as to pour coke into a cup on the table.

In all experiments that involve these skills the joint space is used to represent the state of the manipulator. The location of the target object is kept constant across demonstrations and across participants. This is done to keep the experiment times manageable and to avoid the inaccuracies in the localization of the target object through sensors.

3.3.2.2 *Single-handed means-oriented skills*

The second type of skill, referred to as *means-oriented* skills, include a gesture or communicative intent. Unlike goal-oriented skills, the emphasis in these skills is in the arm trajectory rather than the end-state. The means-oriented skills used in our experiments are as follows. Snapshots from executions of these skills are shown in Fig. 6.

- *Salute*: A soldier's open hand salute gesture.

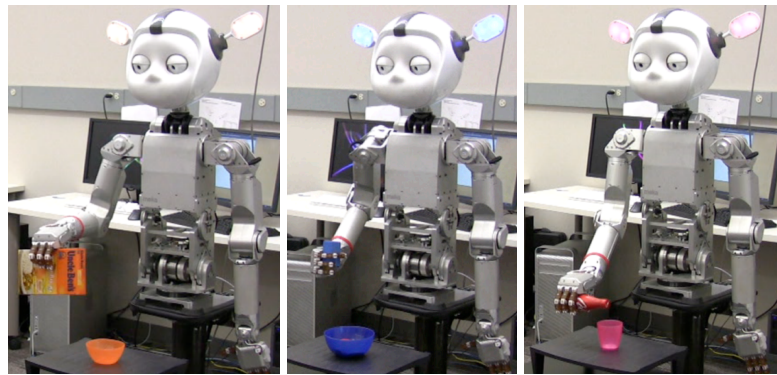


(a) Insert

(b) Stack

(c) Push-button

(d) Close-box



(e) Pour-cereal

(f) Add-salt

(g) Pour-coke

Figure 5: Skills in the single-handed goal-oriented skills domain.

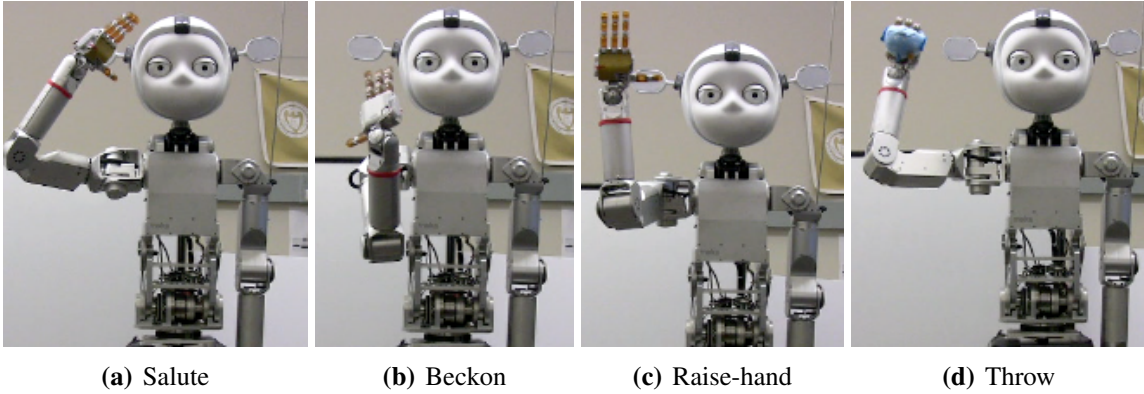


Figure 6: Skills in the single-handed means-oriented communicative skills domain.

- *Beckon*: A gesture asking someone to come closer.
- *Raise-hand*: Raising the robot's hand as if it is asking for permission.
- *Throw*: A throwing gesture with a ball, but without actually releasing it.

Experiments that involve these skills use the joint space as the state of the manipulator.

3.3.2.3 *Two-handed goal-oriented skills*

The third type of skill involves goal-oriented manipulation skills in which the goal is held in the other hand of the robot. The skills are still learned only on a single manipulator of the Simon robot, again possibly while holding objects. In these skills, that state of the manipulator is represented with the end-effector configuration in the coordinate frame of the secondary end-effector. In other words, it is the relative configuration of the two end-effectors.

The skills from this category used in skill learning experiments in this thesis involve the following. Snapshots from the execution of these skills are shown in Fig. 7.

- *Pour*: A block in the hand is inserted through the hole without touching other blocks.
- *Close-box*: A block in the hand is stacked on top of another block on the table.
- *Unscrew*: A button on the table is pressed with the finger tip.

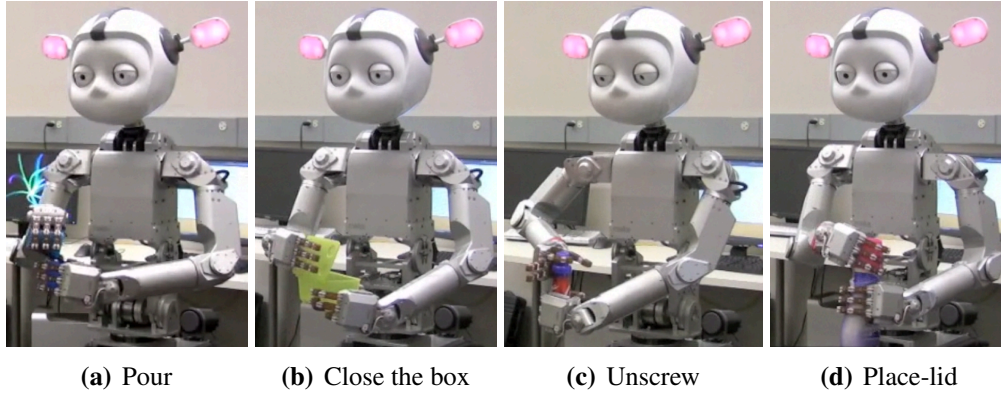


Figure 7: Skills in the two-handed goal-oriented skills domain.

- *Place-lid*: The lid of a box is closed without moving it.

Unlike single handed skills, these allow changing the location of the target object. To keep demonstrations manageable only the primary manipulator is moved during a demonstration, however the other manipulator can be moved between demonstrations.

The models for these are skills are also represented in the relative manipulator configuration space. Therefore, once a trajectory is obtained from such skill models, it needs to be transformed into the robot’s coordinate frame, given the current configuration of the goal (*i.e.* the secondary end-effector). After this, inverse kinematics are used for converting the end-effector trajectory into a joint trajectory that can be replayed on the robot.

3.4 Overview of Experiments

This thesis reports results from 15 different experiments. An overview of these experiments is given in Table 5. This table provides a reference to each experiment and the chapters in which they are presented. It indicates the experiment type, the learning problem and the domain used in the experiment. Note that most experiments involve one of the two learning problems described in this chapter, however some experiments have specific problems and domains that are described in respective chapters.

Table 5: Overview of all experiments in this thesis.

#	Name	
1	Unguided teaching of tasks <i>Type: Observational, Problem: Task learning, Domain: Six-feature toy objects</i>	Chapter 4
2	Unguided teaching of skills <i>Type: Observational, Problem: Skill learning Domain: 1-handed goal and means oriented skills</i>	Chapter 4
3	Unguided teaching of action models <i>Type: Observational, Problem: Action model learning, Domain: Playground</i>	Chapter 4
4	Active task learning <i>Type: Observational, Problem: Task learning, Domain: Nine-feature toy objects</i>	Chapter 5
5	Intermittently-active task learning <i>Type: Controlled, Problem: Task learning, Domain: Six-feature toy objects</i>	Chapter 6
6	Skill learning with different query types <i>Type: Controlled, Problem: Skill learning, Domain: 1-handed goal-oriented skills</i>	Chapter 7
7	Human question asking <i>Type: Observational, Problem: Skill and task learning, Domain: Abstracted tasks</i>	Chapter 7
8	Evaluation of Active KLfD <i>Type: Controlled, Problem: Skill learning, Domain: 2-handed goal-oriented skills</i>	Chapter 8
9	Instructed teaching of tasks <i>Type: Controlled, Problem: Task learning, Domain: Six-feature toy objects</i>	Chapter 9
10	Instructed teaching of face categories <i>Type: Controlled, Problem: Classifier learning, Domain: Chernoff faces</i>	Chapter 9
11	Instructed teaching of linear separators <i>Type: Controlled, Problem: Classifier learning, Domain: Shapes</i>	Chapter 9
12	Instructed teaching of mouse gestures <i>Type: Controlled, Problem: Classifier learning, Domain: Mouse gestures</i>	Chapter 9
13	Instructed teaching of navigation tasks <i>Type: Controlled, Problem: Inverse Reinforcement Learning, Domain: Maps</i>	Chapter 10
14	Instructed teaching of skills <i>Type: Controlled, Problem: Skill learning, Domain: 2-handed goal-oriented skills</i>	Chapter 11
15	Instructed teaching in Intermittently-active learning <i>Type: Controlled, Problem: Task learning, Domain: Six-feature toy objects</i>	Chapter 12

3.5 *Summary*

This chapter overviews the experiments presented in this thesis. First the different types of experiments are discussed. Then the two main types of learning problems considered in these experiments are introduced.

- *Task learning* consists of inferring a general representation of a task goal.
- *Skill learning* consists of learning a model of manipulator motions to achieve a certain goal.

The learning algorithms used for each problem are explained and domains for each problem are described. Lastly, all experiments in this thesis are overviewed in a table.

CHAPTER IV

UNGUIDED HUMAN TEACHING

Humans have the unique ability to teach one another. While different forms of imitation learning are observed in nature, the explicit act of teaching is mostly confined to humans [44]. Parents intelligently structure the surroundings of their child and organize new tasks into manageable steps to provide learning opportunities [91]; professors design curriculums, assignments and tests to teach a new topic; and coaches provide feedback and exercises to improve a motor skill. Teaching comes so naturally that, even peers at pre-school or siblings can teach one another at a very young age. The crucial ingredient of humans' aptitude to teach is the ability to put oneself in the other's shoes [135] and maintain a mental model of the learner's state in terms of what has been learned and what remains uncertain [83]. Therefore, it can be a challenge to teach anything, when the mental model is flawed. This is one of the key challenges in allowing humans to teach robots, since Machine Learning algorithms typically learn very differently than humans.

While we can expect that teaching robots will not be intuitive for everyday people, it is difficult to predict the nature of the problems that will occur. Even getting humans to provide input that is meaningful to the learner, might require several iterations of interaction design. Therefore, an empirical analysis of how humans naturally teach robots is essential in characterizing the issues encountered in learning from humans.

This chapter presents observations from three human-subject experiments that involve unguided or natural human teaching. While each experiment presents distinct challenges, they all illustrate that humans provide sub-optimal data sets from a Machine Learning perspective, resulting in slow and incomplete learning on the robot.

The experiments presented in this section involve *unguided teaching* of the robot by

study participants. The participants are naive teachers who do not know how the robot learns. They present examples of their choice, in order to teach a task or skill, given minimal instruction on how to teach. The experiments involve three learning problems and domains that are characteristically different. Experiments 1 and 2 involve *task learning* and *skill learning* that were introduced earlier (Sec. 3.2 and Sec. 3.3). A different learning problem, referred to as *action modeling*, is introduced in Experiment 3.

4.1 *Experiment 1: Unguided Teaching of Tasks*

This experiment seeks to make observations about the types of examples naturally provided by human teachers when teaching a task goal with positive and negative examples. In particular, unguided human teaching is compared to theoretically optimal teaching in terms of speed of teaching and quality of the final concept¹.

4.1.1 Experimental Design

In this experiment, a human teaches Simon an object category from the six-feature toy objects domain described in Sec. 3.2.3. In particular the HOUSE category, described in Table 1 is used.

4.1.1.1 Interaction

Participants interact with Simon through a fixed set of speech commands which are recognized using the Sphinx speech recognition system. These commands and their purpose are as follows.

- *Positive label*: [Simon], this is a HOUSE.
- *Negative label*: [Simon], this is *not* a HOUSE.
- *Test*: [Simon], is this a HOUSE?

¹This experiment appears in [25].

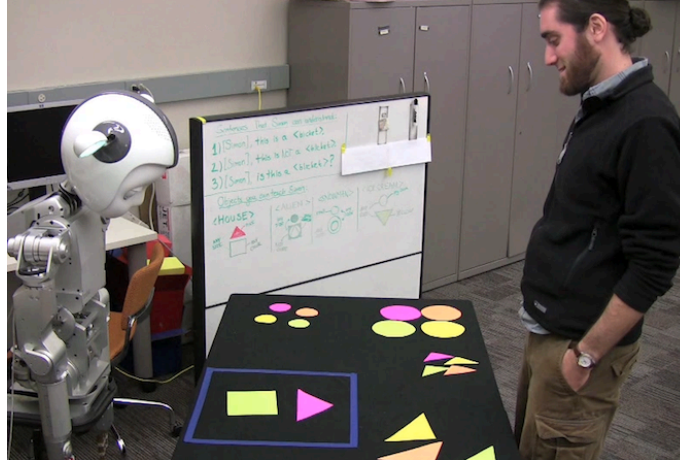


Figure 8: Simon’s workspace with the different object parts arranged around the demonstration area, while a human teacher is providing a demonstration in Experiment 1.

Before using any of these commands the human configures an object by placing two parts at the top and bottom of the demonstration area in front of Simon. Simon uses speech synthesis and gaze directions to interact with the human teacher. When the commands are used Simon gazes to the demonstration area to indicate attention, and then responds. If the person labels the instance, Simon adds the new labeled example to its data set and then acknowledges the example. If the person tests Simon, the instance is classified based on the current version space and responds to the test. To acknowledge that the example given by the teacher has been processed, Simon gives a verbal confirmation such as “okay” or “thank you.” Simon uses a gesture (head nod or shake) in conjunction with speech (“Yes, this is a house”, “No, it’s not” or “I don’t know”) to respond to tests. After this response, Simon looks up and blinks his ears to indicate that he is ready for another example.

4.1.1.2 Procedure

Simon’s workspace is a table covered by a black tablecloth, with the 24 object parts arranged on the perimeter of the table. The center of the table immediately in front of Simon is the demonstration area and is demarcated to the human with a rectangular tape boundary

(Fig. 8). A whiteboard near Simon provided reminders about the object category descriptions and the sentences that the teacher could say.

Before starting the interaction, participants are told about the high-level goal of the experiment and their role in it. They are instructed about the speech commands and they are told about the concept that they will teach Simon.

During the interaction, human teachers stand on the opposite side of the table from Simon for face-to-face interaction and provide examples in the demonstration area from Simon’s perspective. The learning/teaching process involves a turn-taking interaction between Simon and the human teacher. Participants were instructed to wait for Simon to blink the lights on his ears before continuing. They were told to continue teaching the category until they were satisfied that Simon had learned it well or thought that he had stopped making progress. In order to reduce errors, both vision and speech perception are monitored and corrected by an experimenter during the interaction.

4.1.1.3 Evaluation

In this experiment, human teaching is evaluated by comparing to a theoretically optimal teacher. Optimality of teaching is very well defined for the considered teaching problem. As described in Sec. 2.4, there exists a polynomial time optimal teaching algorithm for conjunctions. This algorithm is guaranteed to teach a conjunction with the smallest possible number of examples, identified as $\min(n+1, r+2)$, where n is the number of features (*i.e.* the dimensionality of the state space), and r is the number of relevant features in the particular conjunction that is being taught. The HOUSE concept that is taught in this experiment, has $n = 6$ and $r = 3$. Thus there exists at least one teaching sequence of size 5, that uniquely identifies the correct conjunction. One such optimal teaching sequence is shown in Fig. 9.

In order to characterize how well people teach compared to an optimal teaching algorithm, the following metrics are used.

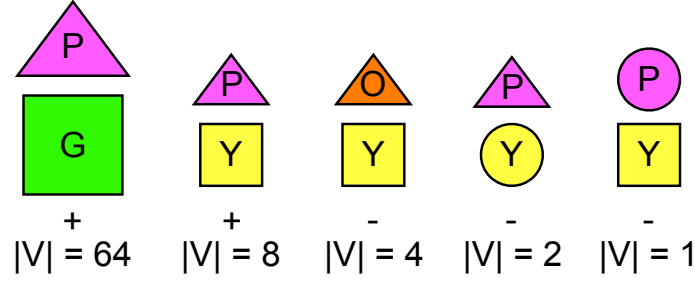


Figure 9: An optimal teaching sequence example for target concept HOUSE.

Quality of teaching. Optimal teaching requires exact identification, however human teachers often stop teaching without uniquely identifying the target concept. We measure how close someone gets to exactly identifying a concept with the number of hypotheses that remain in the version space after the example sequence is given (denoted by $|V|$). Note that the learner can achieve perfect classification performance without exact identification². Classification performance is measured with the F -value:

$$F = 2 * \frac{\textit{precision} * \textit{recall}}{\textit{precision} + \textit{recall}}$$

The extent to which a concept is fully taught is measured with both $|V|$ and F values. The percentage of participants that achieve $|V|=1$ and $F=1$ at the end of their example sequence are also used (denoted by $\%(|V|=1)$ and $\%(F=1)$).

Speed of teaching. Optimal teaching requires the teacher to give as few examples as possible. If a teacher is able to exactly identify a concept their speed of teaching can be measured as the length of their teaching sequence, denoted by $|T|$. Similarly, the number of examples to get to $F=1$ is a measure of how fast one teaches. This is denoted by $|T|_F$.

Since most teaching sessions do not result in exact identification of the target, a better measure is the rate at which the version space is pruned or the rate at which the learner's prediction performance improves. To capture this, the average $|V|$ and the average F -score over the first few examples are also measured (denoted by $avg(|V|)$ and $avg(F)$). This is

²This happens when V contains a set of hypotheses where all but one are split 50/50 in their prediction, and the last hypothesis is h^* .

equivalent to measuring the area under the learning curve. Since $\min(|T|) = 5$, only the first five examples are considered.

A measure of the speed of teaching can include or exclude the tests performed by the teacher between labeled examples. Ideally a teacher asks no questions, however in practice human teachers prefer to do tests to verify what the learner knows because (i) they might not have an accurate mental model of the learner, (ii) they might forget what they have already taught or (iii) out of habit from their experiences with human learners. An efficient teacher from a ML perspective would ask as few questions as possible. To account for questions, both the *number of examples* (excluding tests) and *number of steps* (including tests) are considered to identify how fast people teach. The inclusion of *tests* in the teaching sequence or the averaging is denoted by the subscript t (T_t or avg_t). In addition to the number of examples, the average wall-clock time taken to provide an example or test are also measured.

Similarity to optimal teaching strategy. In order to better characterize the optimality of how humans teach, we try to describe the *informativeness* of examples given by the person according to the optimal teaching algorithm given in Sec. 2.4. Remember that the optimal strategy starts with a positive example, then gives another positive example in which all the irrelevant features have been varied and then gives r negative examples in which the relevant features are varied one by one. There are three tricks that this strategy uses to prune the wrong hypotheses in order to achieve accurate and fast learning.

The first trick is to start with a positive example, drastically pruning the hypothesis space (from all possible concepts to all that are consistent with the example, *i.e.* from 576 to 64).

The second trick is to vary as many irrelevant features as possible while keeping the second example positive. This has the potential to reduce the size of the version space by a factor of $2^{(n-r)}$ where $(n - r)$ is the number of irrelevant features. Thus, in our case the best possible positive example reduces the size of the version space by a factor of $2^3 = 8$.

Table 6: Performance of participants in Experiment 1 on different metrics of good teaching, compared to optimal values.

	Metric	Human teachers	Optimal
Quality of Teaching	$\%(V =1)$	25% (6/24)	N/A
	$\%(F=1)$	58.33% (14/24)	N/A
	$ V $	6.62 (S=11.79)	1
	F	0.78 (SD=0.31)	1
Speed of Teaching (excluding tests)	$ T $	10.67 (SD=4.59)	5
	$ T _F$	6.93 (SD= 4.43)	2
	$avg(V)$	48.87 (SD=68.56)	15.80
	$avg(F)$	0.36 (SD=0.19)	0.82
Speed of Teaching (including tests)	$ T_i $	19.17 (SD=7.41)	5
	$ T_i _F$	10.86 (SD=6.99)	2
	$avg_i(V)$	73.90 (SD=125.80)	15.80
	$avg_i(F)$	0.25 (SD=0.21)	0.82

Positive examples that vary by 2 or 1 irrelevant features will be less effective, with a pruning factor of 4 or 2, but still informative. If a positive example does not prune the hypothesis space at all it is a redundant (*bad*) positive example.

The final trick of the optimal teaching strategy is to vary relevant features one at a time. This is effective since it proves that a feature is relevant and prunes out all hypotheses that do not include this feature. This results in pruning half of the version space. When two or more features are varied at once, it is not possible to judge which subset of the features that are varied causes the instance to be a negative example. Therefore, the learner can only eliminate the hypotheses that have all of the varied features. Negative examples that halve the version space are considered *good* examples. The ones that do not change the version space (*i.e.* uninformative) or that change the version space by a factor less than 2 (*i.e.* too varied) are *bad* negative examples.

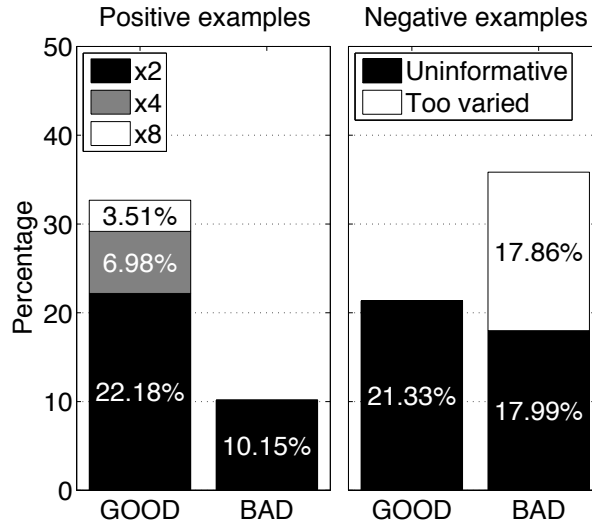


Figure 10: Distribution of *good* and *bad* positive/negative examples given by participants in Experiment 1. For good examples, breakdowns of how much of the version space was pruned is shown (*i.e.* by a factor of 2, 4, or 8). Bad negative examples are broken down as uninformative or too varied.

4.1.2 Findings

For this experiment 24 participants (5 females and 19 males) taught Simon the HOUSE concept. Participants were recruited from the Georgia Institute of Technology campus community through mailing list announcements and flyers around campus.

Performance of the participants on the described metrics is given in Table 6. The distribution of good and bad examples given by participants is shown in Fig. 10. The following observations are made.

Failed to achieve exact identification. The majority of the participants ended teaching without achieving exact identification. A higher percentage reached an F -measure of 1, however there were still some participants that ended teaching when the robot could not correctly classify some instances. When very few hypotheses are left in the version space, disagreement among the hypotheses occurs only for few instances and the majority prediction is correct for most instances. As a result it is difficult for participants to identify the example that will further prune the version space.

Speed of teaching is suboptimal. The speed at which the participants taught the concept in both experiments was slower than optimal. Participants used twice the number of optimal examples to achieve $|V|=1$ and three times the number of optimal examples to achieve $F=1$. When including tests, the teaching sequence has 3-5 times more examples and tests than the optimal teaching sequence. The rate at which participants teach at the beginning of the sequence is also suboptimal. For instance, the average F -measure throughout the first five examples is about half of the optimal value, which means that the robot will make many more mistakes if it is tested during this phase of teaching.

Example distribution is suboptimal. The underlying reasons for the suboptimal performance of natural teaching is better characterized by the distribution of good and bad examples (see Fig. 10). We observe that very few participants gave a positive example that prunes the version space by a factor of 8 (only 2 out of 24). Thus people do not seem to intuitively vary as much of the irrelevant space as possible. Second, we observe that only about one third of negative examples given by people is maximally informative.

Positive examples more optimal than negative. Human teachers seem naturally inclined to start with a positive example when teaching a concept; 91.67% (22/24) of participants started teaching with a positive example. Additionally, looking at the distribution of examples, it is observed that positive examples given by teachers are mostly informative (more good examples than bad). Thus, people are able to induce variance in the irrelevant part of the space as part their effort to give “different” positive examples. In addition, the ratio of positive and negative examples in the data provided by people is very close to the ratio in an optimal teaching sequence (40% positive and 60% negative, Fig. 9).

4.2 Experiment 2: Unguided Teaching of Skills

The second experiment seeks to observe demonstrations provided by human teachers while they teach skills. The teaching process involves humans moving the robot’s arm to

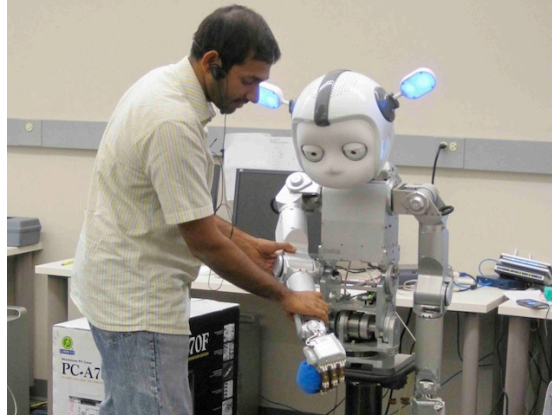


Figure 11: A participant kinesthetically interacting with Simon in Experiment 2.

demonstrate the skill, while the arm is passive in a gravity compensation mode³.

4.2.1 Experimental Design

This experiment involves the skill learning problem described in Sec. 3.3. It involves both trajectory and keyframe demonstrations. The domains used in the experiment are the single-handed goal and means oriented skills shown in Fig. 5(a-d) and Fig. 6 (Sec. 3.3.2).

Participants in this experiment teach four different skills with trajectory and keyframe demonstrations. These are referred to as TD and KD conditions. All skills are taught kinesthetically, *i.e.* by physically moving the arm while it is in a passive mode (Fig. 11). The ordering of the two conditions is counterbalanced across participants. In addition the particular skills taught in each condition are varied. Each participant taught one means-oriented skill, and one goal-oriented skill in each condition. At the beginning of each condition, participants taught a pointing skill to the robot for familiarization with the condition. Participants were also allowed to move the robot's arm to practice before recording demonstrations.

The skills that the participants are asked to teach the robot are described verbally, and then performed by the experimenter to provide an example.

³This experiment appears in [4] with additional results on iterative modification of learned skills.

4.2.1.1 Interaction

The interaction with the robot is regulated through a fixed set of speech commands. In the TD condition, the teacher is informed that the robot will record all the movements they make with its right arm between the start and the end of a demonstration. The teacher initiates the demonstration by saying “New demonstration”, moves the arm to make the robot perform the skill and finishes by saying “End of demonstration.” This process is repeated to give as many demonstrations as the person desires. After a demonstration, the teacher can use the speech command “Can you perform the skill?” to have the robot perform the current state of the learned skill and adjust his/her demonstrations to address errors. In the KD condition, the teacher is informed that the robot will only record the arm configuration when they say “Record frame”, and it will not record any movements between these keyframes. The teacher can use the speech commands “New demonstration”, “End of demonstration” and “Can you perform the skill?” in the same way as the TD condition.

4.2.1.2 Evaluation

Two different methods were used to measure the quality of the different types of learned skills. Goal-oriented skills are evaluated based on the success of reproduced skills, and the means-oriented skills are evaluated by expert ratings of reproduced skills taught by participants.

The performance of goal-oriented skills were scored separately by the two coders. A skill execution was categorized as one of the success levels *(i)* success, *(ii)* partial success, or *(iii)* fail. The scoring was based both on the recorded videos of the experiment and on the skill performances recreated on the robot. In the few cases where there was disagreement, the two coders revisited the example and reached a consensus on the scoring.

Unlike the goal-oriented skills that have an objective measure of success, means-oriented skills are better evaluated subjectively. Expert ratings of the recreated skills are used for

evaluating performance on these skills. The experts are specialized in computer animation. They are asked to answer the following three 7-point Likert-scale questions for all means-oriented skills taught by all participants.

- *Appropriate emphasis*: How appropriate was the emphasis/exaggeration of the gesture? (1: very inappropriate – 7: very appropriate)
- *Communicating intent*: How well the animation reflects the intended gesture in your opinion? (1: very bad – 7: very well)
- *Closeness to perfection*: If you were to perfect this animation, how much modification do you think is necessary? (1: significant amount – 7: not much)

4.2.2 Findings

The experiment was completed by 34 participants (6 females, 28 males between the ages of 19-47), who were undergraduate and graduate Georgia Institute of Technology students with no previous machine learning and robotics experience. The means-oriented skills were rated by two animation experts from the Georgia Institute of Technology Graphics group.

The following observations are made.

Number of demonstrations used. In the experiment users were able to see what the robot has learned after each demonstration and decide whether to move on or give another demonstration. Fig. 12 shows the number of demonstrations provided by participants in TD and KD conditions. Teaching with a single demonstration was common in both conditions. For goal-oriented skills, a larger portion of the participants provided a single demonstration in the TD condition than in the KD condition (19 versus 10). It was common in the KD condition to forget to provide keyframes that allow the robot to avoid obstacles while trying to achieve the goal. These frames were provided by participants in subsequent demonstrations after observing the performed skill collide with obstacles (*e.g.* see Fig. 14). For

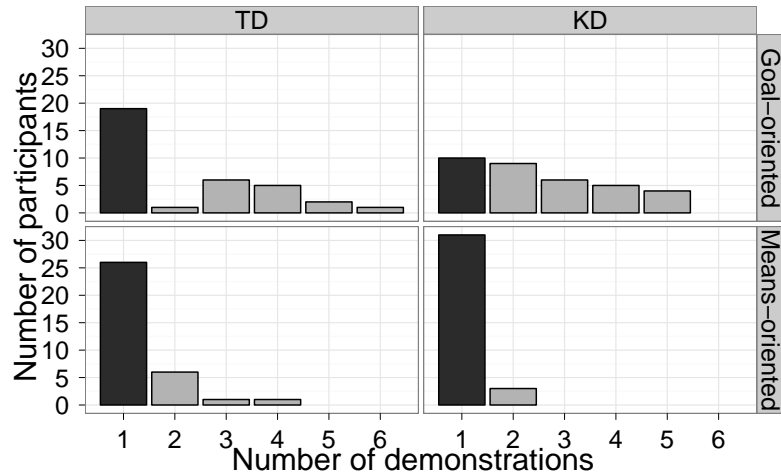


Figure 12: Histogram of number of demonstrations provided by participants in KD and TD conditions.

means-oriented skills, teaching with a single demonstration was more common in the KD condition than in TD (31 versus 26).

Success of goal-oriented skills. The success of goal-oriented skills taught by participants is summarized in Table 7. Teaching with TDs results in successful skills more often (46% of participants taught a successful skill), as compared to teaching with KDs (27% of participants taught successful skills). The percentage of failing skills is around 30% for both types of demonstrations, which is considerably high. 41% of skills taught with KDs are partially successful, as they often achieve the goal but collide with an obstacle on the way.

Only one participant out of the 14 who provided multiple demonstrations in the TD condition was able to achieve success with the goal-oriented skill (Table 7). Participants often have trouble providing multiple trajectory demonstrations that have proper temporal alignment. As a result, parts of the demonstrations that are intended as different parts of the skill get averaged together across demonstrations. An example is shown in Fig. 13. On the other hand, the learning algorithm for KDs handles the alignment problem, as seen in

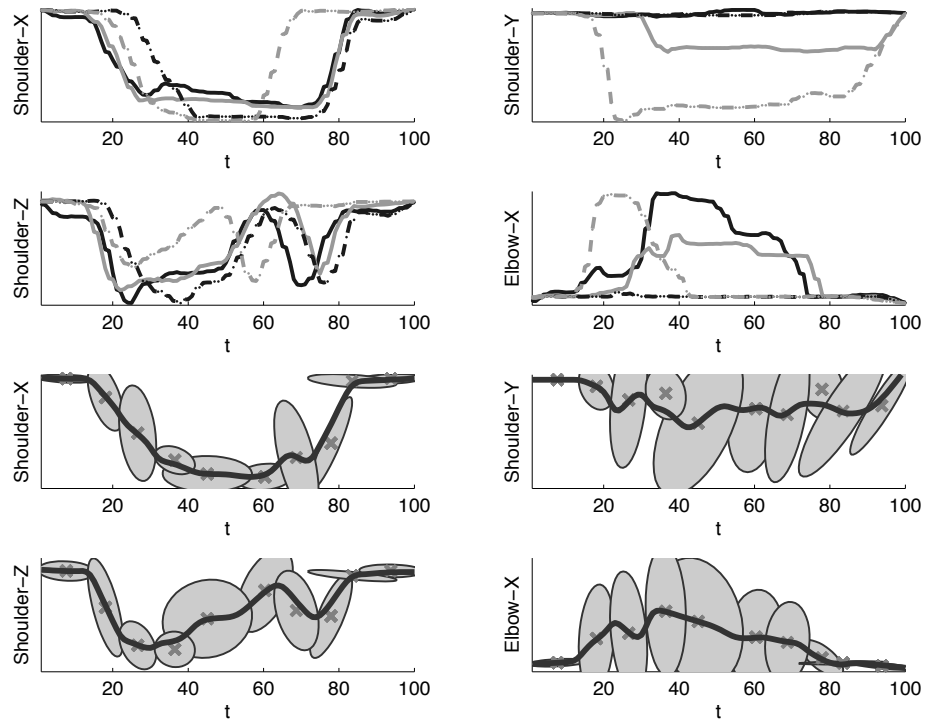


Figure 13: An example of the temporal alignment issue in trajectory demonstrations while teaching the *Close-box* skill in Experiment 2. The resulting skill does not close the box (the horizontal arm movement is not sufficient) due to the attenuation of the movement in the Shoulder-Z joint caused by averaging temporally misaligned demonstrations. Demonstrations (upper four panels) and reproduced motions (lower four panels) are shown on four joints.

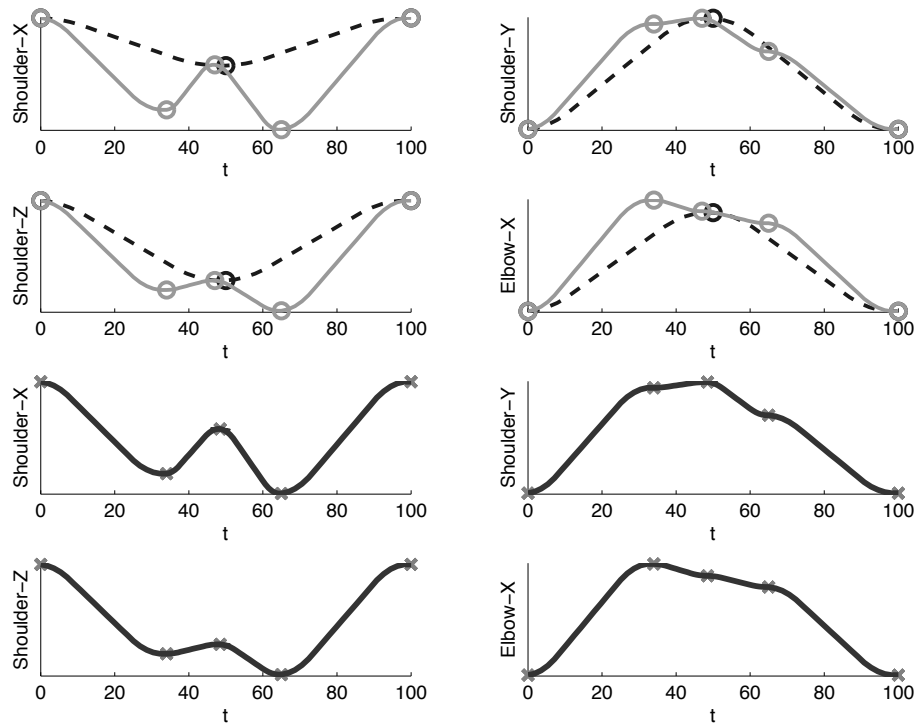


Figure 14: An example of forgetting obstacle avoidance keyframes in the first demonstrations (dashed line), and providing them in the second demonstration (solid line) in the KD condition while teaching the *Push-button* skill in Experiment 2. Demonstrations (upper four panels) and reproduced motions (lower four panels) are shown on four joints.

Table 7: Number of participants who achieved different levels of success for goal-oriented skills in Experiment 2.

Condition	# of demos	Success	Partial Success	Fail
TD	Single	15	4	1
	Multiple	1	5	8
	Total (%)	16 (46)	9 (27)	9 (27)
KD	Single	5	5	1
	Multiple	4	9	10
	Total (%)	9 (27)	14 (41)	11 (32)

Fig. 14. As a result more participants were able to achieve success with multiple demonstrations in the KD condition. While this issue can be reduced with temporal alignment algorithms or with practice by the teacher, it cannot be fully eliminated for non-expert users. Furthermore, teaching with a single demonstration is unrealistic in settings outside of a simple experiment domain, when learning needs to generalize across different goals (e.g. learning to insert an object through a hole at any location or orientation).

Success of means-oriented skills. The evaluation results for mean-oriented skills are shown in Table 8. Both experts rated the means-oriented skills learned in KD condition higher in all three dimensions on average. The difference was only significant for *closeness to perfection*, and the difference is marginally significant when the three scales are averaged ($Z=2740$, $p=0.06$ on Wilcoxon signed rank test). This distinction is partly related to the difficulty of moving a 7-DoF arm smoothly in the TD condition. In addition, the temporal alignment issue mentioned earlier for goal-oriented skills, also exists for the few participants who provided multiple demonstrations for means-oriented skills.

4.3 Experiment 3: Unguided teaching of action models

Human environments are filled with objects that serve different purposes. Many of these are relevant for tasks that personal robots might carry out in assisting humans. When

Table 8: Expert ratings of means-oriented skills (median and coefficient of dispersion) and comparison of two demonstration types in in Experiment 2.

Cond.	Expert	<i>Emphasis</i>	<i>Intent</i>	<i>Perfection</i>
TD	1	5.5 (0.27)	5 (0.33)	5 (0.35)
	2	3 (0.29)	3.5 (0.38)	4 (0.3)
KD	1	6 (0.21)	6 (0.17)	6 (0.2)
	2	4 (0.21)	4 (0.24)	5 (0.22)
TD versus KD (Wilcox s.r. test)		Z=2679, p=0.10	Z=2677, p=0.11	Z=2796, p=0.03

faced with a novel object, a robot needs to learn what it can do with it. It is important for the robot to learn this by directly interacting with objects rather than just observing, since the robot’s embodiment and available actions determine what it can achieve with objects. The problem of learning to predict the effects of actions applied on objects is referred to as affordance learning [96, 116] or *action model learning* in model-based Reinforcement Learning [70]. The learned information can later be used in planning how to achieve goals assigned by the user.

Action model learning requires the robot to explore and learn from experience; *i.e.* perform actions on objects and observe the effects. This experiment investigates action model learning in situations where the exploration is controlled by a human teacher⁴.

4.3.1 Experimental Design

4.3.1.1 Learning problem

The learning problem in this experiment consists of learning a classifier that allows predicting the effects of actions on objects. These classifiers are learned from interaction experiences, represented by a *context*, *action* and *outcome*, where performing the action in the contexts results in the outcome. A robot interacts with objects during an exploration

⁴This experiment appears in [134] with additional results on using gaze for transparency about error states.

phase to obtain such experience tuples, which are later used to train a classifier. In this experiment the context is a perceived object state, the actions are discrete open-loop actions applied on objects, and the outcomes are discrete categories identifying the effect of the action on the object.

Formally, a robot represents an object state \mathbf{s} with a vector of features (s_1, s_2, \dots, s_d) and it has a set of discrete actions $\{a_1, a_2, \dots, a_M\}$. When an action a_i is applied on an object with state \mathbf{s}_t , the state of the object changes to \mathbf{s}_{t+1} . The change in the object state is discretized with a *known* function $g : (\mathbf{s}_t, \mathbf{s}_{t+1}) \rightarrow e_j$, where e_j belongs to the set of possible effects $E = \{e_1, e_2, \dots, e_K\}$. Thus an interaction consists of the tuple (\mathbf{s}_t, a_i, e_j) . Several representations are possible for learning the relationship between the elements of this tuple, such as modeling the probability of occurrence for each effect over the spaces of object states, *i.e.* $P(e_i|a_i, \mathbf{s}_t)$, or directly learning a function that maps the state to an effect for a given action, *i.e.* $f_{a_i} : \mathbf{s}_t \rightarrow e_i$. The one used in this experiment is as follows. A separate function is learned for each action-effect pair. The input of the function is the state of the object and the output is whether or not the effect will occur when the action is executed, *i.e.* $f_{a_i, e_i} : \mathbf{s}_t \rightarrow \{0, 1\}$.

4.3.1.2 Learning Algorithm

For predicting the occurrence of an effect, given an object-state and action, two-class Support Vector Machines (SVM) are used. A separate SVM is trained for each type of effect, using the state of the object as the feature space and outcome (*i.e.* whether or not the action resulted in the corresponding effect) as the target value. Also, separate SVMs are trained with the social and non-social data sets.

SVMs are widely used discriminative classifiers. SVM classification is based on a linear separator constructed from a set of representative points close to the margin (support vectors). The input space is implicitly projected to a higher dimensional space with the help of kernels. A linear kernel was used in this experiment. The used implementation is the

open source library LibSVM [36] integrated with the WEKA data mining software [63].

4.3.1.3 Domain

The robot used in this experiment is Junior (Sec. 3.1.2). The robot sits on a tabletop, and its workspace for this experiments is a 5 inch straight line in front of it. Junior's action set consists of two actions.

- *Poke*: A single arm swing (e.g., for batting or pushing objects).
- *Grasp*: A coordinated swing of both arms.

Both actions are parametrized with the height and distance of the object to which the action is directed.

Junior learns about five simple objects with different geometrical shapes and bright, distinct colors (Fig. 15). This is referred to as the *playground* domain. Objects are tracked using the OpenCV Library implementation of Continuously Adaptive Mean Shift based blob tracking, for pre-defined colors. The state of an object in the workspace is specified with several properties of the tracked blob. This includes the following measured and derived features.

- *Distance*: Neck position required to center the blob in the image.
- *Color*: Discrete indicator of the unique object color.
- *Area*: Number of pixels of the blob.
- *Orientation*: Angle of the major axis of the blob.
- *Height* and *Width*: Length of major and minor axes.
- *Eccentricity*: Ratio of major and minor axes.
- *Squareness*: Ratio of areas of the blob to the minimum enclosing rectangle.

Junior's interaction with the objects is regulated with two behaviors. When there is no object in the visual field a *search* behavior randomly changes the head tilt until an object is in view. Then a *fixation* behavior centers the object in the visual field. When an object is stationary in the center of visual field for about two seconds, one of the two actions is

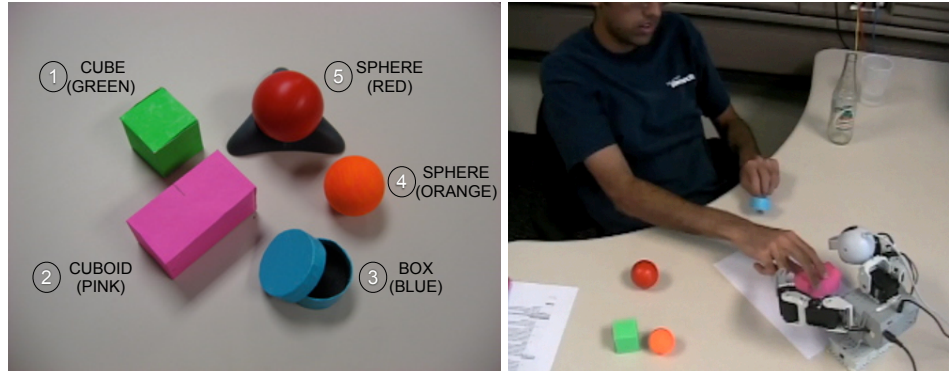


Figure 15: Five objects for the playground domain and a screenshot from Experiment 3 in which a human teacher is interacting with Junior.

triggered.

Action effects are perceived by the robot as changes in the object state. Each interaction of the robot is manually labelled as one of the following effect categories.

The effects for the *grasp* action are:

- *Lifted*: The object moves upwards until it is dropped.
- *Opened*: The cover of the box separates from the bottom.
- *Other*: The object slips, falls, moves away or is thrown away.

The effects for the *poke* action are:

- *Rolled*: The object keeps moving in the poking direction after contact is lost.
- *Moved*: The object moves until contact with the object is lost.
- *Tipped*: The object falls over around one point.

Both actions have a fourth category (*no effect*) in which the object does not move at all.

The baseline or ground-truth for the described domain is obtained through the following procedure. For each object, the space of all possible configurations is uniformly discretized. The object is moved at 0.25 inch intervals on the workspace in several possible orientations. There are 2 canonical orientations for the cube, 5 for the cuboid, 9 for the box and one for each sphere as shown in Fig. 16. Both actions are executed in each configuration and the effects of the action are recorded. This results in 756 object interactions which characterize

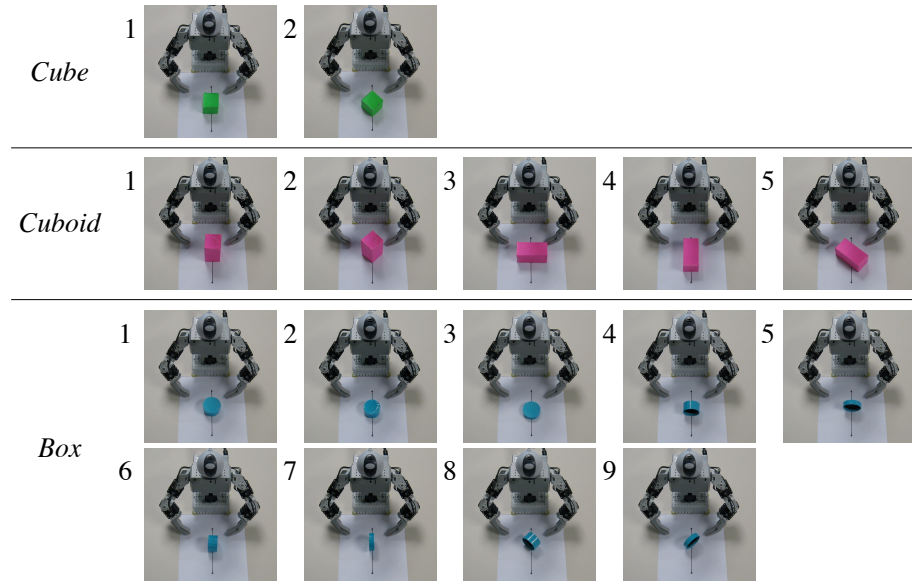


Figure 16: All configurations of non-spherical objects considered in the systematic condition in Experiment 3.

the domain.

4.3.1.4 Procedure

In this experiment, the role of the human, as a teacher, is to guide the robot’s exploration. This impacts the interactions that the robot experiences, and consequently what it learns. The human teacher controls which objects the robot interacts with, and decides where and how to place the object in the workspace (Fig. 15).

In the instructions for the experiment, participants are first informed about Junior’s exploratory behavior and told that their goal is to help Junior learn what it can do with the five objects. They are told not to move the object once Junior has started an action, and that the action ends when Junior’s arms are back to their idle position. They are asked to place one object at a time, horizontally centered on the line indicating the workspace. Each participant starts experimenting with one action and decides when to switch to the second action (the action presented first is counter balanced across participants). The experiment duration or total number of object interactions for each action is decided by the participant.

The interaction is video recorded for later analysis. In addition participants are asked to answer 25 questions at the end of the experiment.

4.3.2 Findings

This teaching experiment was completed by 14 participants, recruited from the campus community (78% male). Test data sets are obtained by randomly sampling equal numbers of positive and negative examples from either the systematic or the social data set. Thus, learned classifiers are compared with two separate tests, one social test and one systematic test. Findings from this experiment are outlined with the following observations.

Humans start with simple examples. Most of the time participants start with prototypical examples in which the effect has a very high or very low likelihood of happening. For example, the distance of the object from the robot is a determining state variable for most action effects (*e.g.* lifted, moved, rolled). We see that early in the teaching session, human teachers provide more examples in which the object directly in front of the robot, or obviously outside the reachability of the robot. Later on they provide more examples in which the object is placed around the reachability boundary. We observe the very first example provided by 86% of the participants had an object placed within Junior’s reach.

This behavior in humans seems to be a form of scaffolding [91]. People start with simple examples, not complex ones. This is a useful teaching strategy for human learners; and can be favorable for certain machine learners [52, 49, 84]. However in our experiment it is a sub-optimal teaching strategy that results in a slower learning curve. We see a manifestation of this behavior also in the choice of objects chosen at the beginning of the teaching session. Fig. 17 gives the distribution across participants of starting object choice for the grasp action. The green cube followed by the pink cuboid are most popular, both of which have flat surfaces making them relatively easy to grasp. The orange sphere also has a higher rate than the other two objects for being light and having a rough surface, which makes it easier to manipulate than the large red sphere that has a slippery surface.

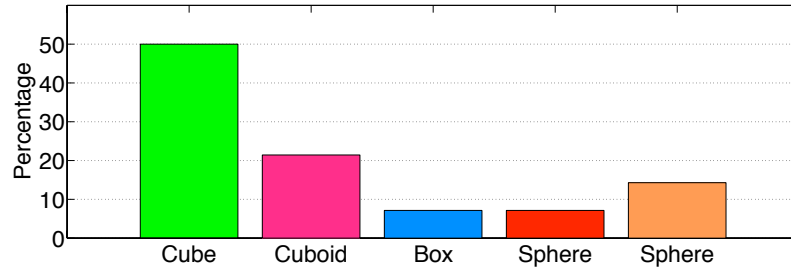


Figure 17: Distribution of starting object choice of participants for the grasping action in Experiment 3.

In Experiment 3 participants gave many examples that were at the border of having some effect and having no effect. This is reflected in the distance at which they configured the object in front of the robot. As shown in Fig. 19 a high number of examples given by participants were around the distance where the object is barely reachable. A common teaching pattern was to first present an easily reachable, close example and an obviously unreachable, far example; and then present many examples around the border where the robot seemed to have the difficulty in reaching the object. Such examples are useful since they refine the classifier around the border.

In addition, the quantity of examples people gave for an object was proportional to its complexity (number of possible effects on the object and number of configurations of the object). Participants were able to assess that there is more to learn about some objects than others. In addition, most participants started teaching with an “easy” example. This supports that, humans can evaluate how interesting an example is for the learner.

Distribution of human examples is proportional to complexity. The quantity of examples people gave for an object in Experiment 1 was proportional to object complexity. The top graph in Fig. 18 shows the distribution of examples for each object, and the other graphs show that this distribution was proportional to the object’s affordance and configuration complexity.

Number of examples is primarily aligned with the number of affordances an object has. For instance, the blue box has a much larger number of configurations compared to other

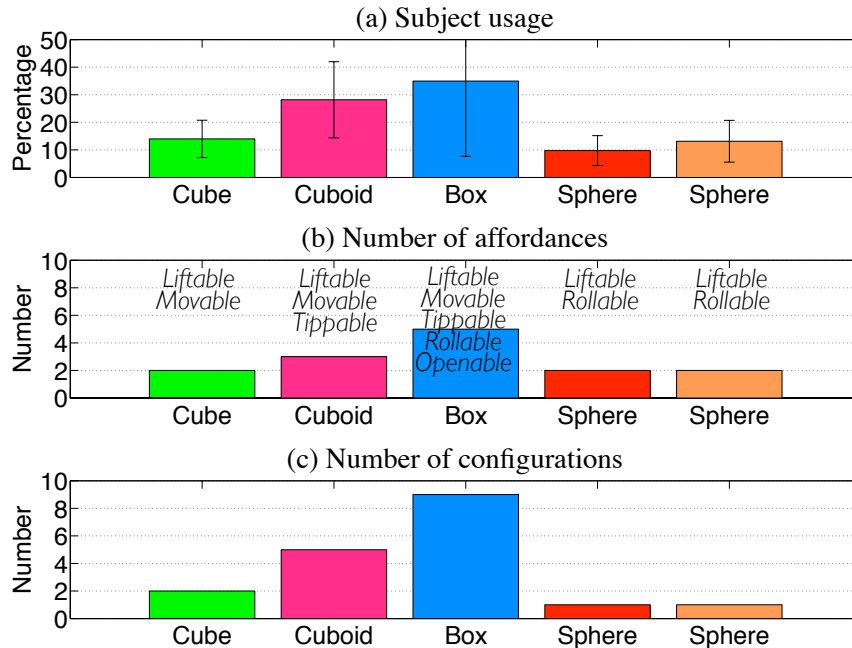


Figure 18: Comparison of human usage of objects and object complexity in Experiment 3. (a) Distribution of examples given for each object averaged across participants (b) Number of affordances of each object (c) Number of configurations of each object.

objects (Fig. 18(c)); however, the extra configurations do not add proportionally as many affordances. Accordingly, people gave the most examples for the box object, but in relation to number of affordances not configurations.

Another observation of the social data set, is that there are more samples of the orange sphere compared to the red, though they both have just one configuration and afford rolling and lifting. Information not captured in the distribution of affordances is how easily they are afforded. The two spheres differ not only in size/color but also in texture and weight. The small size and rough texture of the orange sphere makes it liftable in a larger range compared to the polished surface and high weight of the red sphere. Therefore, people's distribution of examples reflects their perception of affordances through properties not observable to the robot. They take a broader view, *i.e.* not just afford/not afford, but how easily it is afforded. One measure of how easily an effect is afforded is the number of times it is observed in the systematic experiment. For instance the orange sphere was lifted in 55%

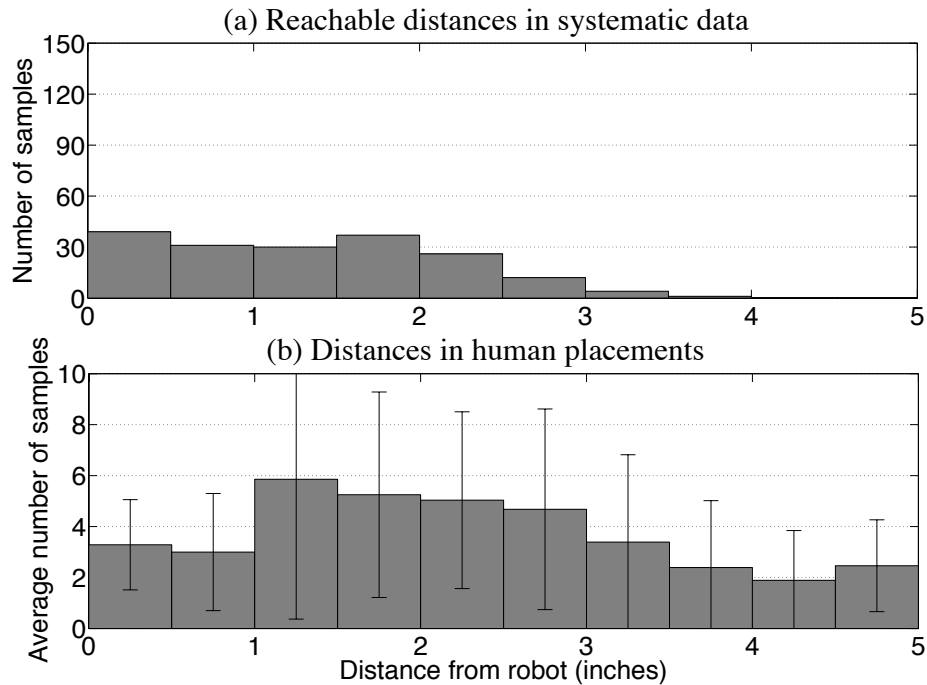


Figure 19: Comparison of object reachability and object placements by participants in Experiment 3. (a) Histogram of reachable distances, taken from systematic data (b) Histogram of distances where participants placed objects.

of the systematic grasp trials performed on it, while the red one was lifted in 35%. Note that the difference observed in the graph is caused by the differences in object preference mostly for the grasp action (the orange sphere was used almost twice as much as the red one in the grasping trials, while it was used about the same number of trials in poking). A similar observation can be made for poking, where large objects are perceived to be more easily poked than smaller ones. For instance participants presented the cuboid (which is twice the size of the cube) about twice as many times as the cube in poking trials, while they presented them equally during grasping.

The questionnaire supports this attention to affordance complexity. When asked open questions about whether they followed a certain order or emphasized any objects, over half of the participants said no. However, several talked about objects being “boring,” “interesting,” or “fun.” Thus, structuring complexity seems to be an intuitive approach to

Table 9: Number of examples presented by individuals for each object and action in Experiment 3.

		GRASP													
		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14
Cube		8	8	18	9	2	18	1	3	11	4	7	2	7	6
Cuboid		8	4	9	12	5	17	10	11	11	13	2	16	11	3
Box		26	18	12	11	3	20	14	10	15	3	2	8	10	11
L.Sphere		3	7	2	4	0	5	6	4	8	3	1	5	7	3
S.Sphere		3	3	2	4	0	10	9	5	34	10	3	9	7	5
Total		48	40	43	40	10	70	50	33	79	33	15	40	42	28
		POKE													
Cube		3	5	6	1	0	5	8	6	4	7	4	7	7	3
Cuboid		11	2	12	10	11	12	11	10	11	7	4	21	9	2
Box		17	7	9	3	6	6	4	31	17	5	7	9	11	33
L.Sphere		4	2	2	2	3	6	6	3	1	4	5	1	6	1
S.Sphere		3	3	3	3	11	3	6	8	3	5	2	3	7	2
Total		38	19	32	19	31	32	35	58	36	28	22	41	40	41

the task of teaching, even if it was not a conscious effort.

Humans provide balanced datasets. Data sets in the *Social* condition were more balanced in terms of positive/negative examples than the *Non-social* condition. Fig. 20 gives the distribution of effects seen for actions on objects in both conditions. For both actions the percentage of the *no effect* category is much lower in the *Social* condition, and common effects such as *lifted* or *moved* is higher. Rare effects like *rolled* and *opened* are approximately doubled in the social condition.

This is a result of people’s choice of orientation and location. They presented more examples of objects in orientations that have more affordances. For example, for grasping, presenting the cube in a parallel orientation (61%) rather than diagonally; and presenting the box with the cover on top (43%) as opposed to the other 8 configurations. Similarly, people mostly placed objects easily within reach (Fig. 19).

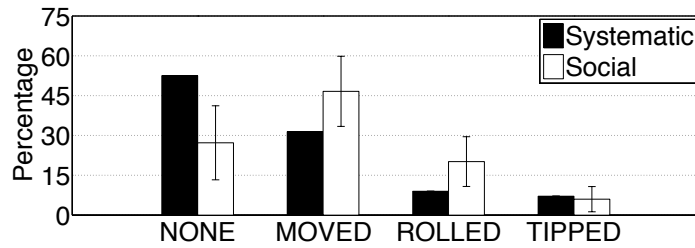
Uniformly sampling object states results in a large number of interactions in which the

action has no effect on the object. In the datasets provided by participants the average percentage of interactions in which the action has no effect is much less (Fig. 20). Participants provide more examples of interactions in which the action has some effect. This is especially useful in learning rare outcomes (like an object being *opened*), which are completely ignored by the learned classifier when learned from a uniformly sampled dataset.

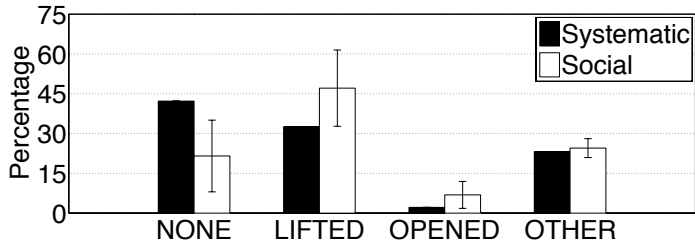
Fig. 21 compares the average successful prediction rate for classifiers with four different training data sets: (i) the complete set of examples collected systematically, (ii) the combination of examples provided by all 14 individuals, (iii) random subsets of the systematic data (size equal to the average number of examples given by one participant), and (iv) the data sets obtained by individual participants in the experiment.

The social training sets perform better on rolling and opening affordances in both test conditions. This is a result of the balanced nature of the data provided in the *Social* condition. As these affordances are rare in the systematic data set, the non-social training results in pessimistic classifiers that mostly predict a negative outcome. With more frequent affordances such as lifting and moving, the social data set is on par with the systematic training set or one is slightly better depending on the test set.

Individual teachers do not provide complete datasets. A first observation is that the complete data sets (systematic and everyone) generally perform better than the smaller data sets (random subsets and individuals). This shows that the number of samples given by one individual in a single sitting may not be sufficient for learning everything. This points to the importance of self exploration (for collecting large data sets with systematic experiments) as well as long-term training by individuals (multiple sessions) or by multiple teachers. Nonetheless, the variance in the individuals' performance indicates that some individuals were able to get closer to the performance of the complete data sets.



(a) Poke



(b) Grasp

Figure 20: Distribution of effects in the *Non-social* and *Social* conditions for each action in Experiment 3.

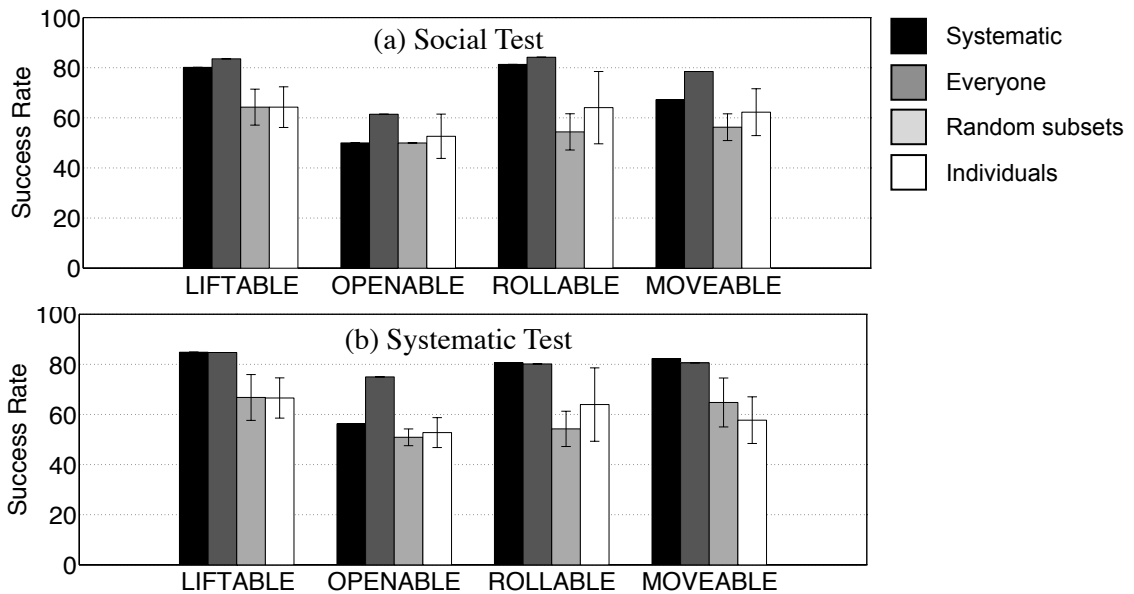


Figure 21: Impact of learning social versus non-social learning on the performance of the learned classifier in Experiment 3. Graphs present the prediction accuracy of classifiers trained with the non-social (systematic) data and the social data, on (a) social test data and (b) systematic test data. Values are averaged over 10 randomly sampled test sets.

4.4 Issues in Unguided Teaching

This chapter describes three experiments with scenarios that involve a human interacting with a robot to teach a different target concept. The three experiments have characteristically different learning problems and domains. The rest of this chapter highlights findings from all three experiments grouping them into common patterns in human teaching. This section provides observations about the issues that arise in the different teaching scenarios, resulting in unsatisfactory skills or tasks. The underlying reasons for these issues are discussed. Then, Sec. 4.5 provides observations on useful patterns in human teaching, that can be used in the robot's advantage. References to other experiments presented later in this thesis that further support results from the three experiments, as well as support from the literature are also provided.

4.4.1 Humans provide sub-optimal examples

The first issue highlighted by the three experiments is the sub-optimality of the demonstrations given by humans. The term sub-optimal is used to indicate that a demonstration was chosen against other potential ones that would have been more informative for the learner, and potentially optimal.

4.4.1.1 Support from Experiment 1

The first experiment involved teaching task goals that are represented as conjunctions, such as the HOUSE concept used in the particular experiment. For this problem, there was a concrete description of what it means to be an optimal teacher. It was observed that humans were often sub-optimal. Participants were much slower in both reaching a 100% F-score and achieving unique identification of the target concept; when compared to an optimal teacher. Several reasons for the sub-optimality in human teaching were observed, including (i) changing irrelevant features one or two at a time rather than changing them all at once, (ii) changing more than one relevant feature at once rather than one at a time,

(iii) providing repeated or redundant examples (*i.e.* examples that can already be correctly classified or repeating examples) that were not informative.

4.4.1.2 Support from Experiment 2

Optimality of examples is harder to define for the types of skills that were taught in the second experiment. Nevertheless, we can assess the quality of the provided demonstration in several aspects. For trajectory demonstrations (TD), the experiment demonstrated that participants had trouble providing consistent and well-aligned demonstrations when they chose to provide more than one demonstration (Fig. 13). For keyframe demonstrations (KD) participants often neglected certain necessary keyframes (*e.g.* keyframes that avoid collisions with the target object) until they saw the execution of the unsuccessful skill (Fig. 13). As a result of such problems in both types of demonstrations, the percentage of successful goal-oriented skills taught by the participants is low (46% for TD and 27% for KD).

These problems also contribute to the efficiency of teaching. Due to the way that skills are encoded, correcting a bad demonstration (*e.g.* non-smooth motion due to the difficulty of simultaneously controlling 7 DoFs) requires several more demonstrations in the TD-based teaching. Similarly for KDs, peoples' failures to provide obstacle avoidance keyframes, costs them another demonstration for correcting their mistake.

4.4.1.3 Support from Experiment 3

For a discriminative classifier, borderline examples are more informative than prototypical examples that are further away from the category boundaries. Experiment 3 involved teaching binary classifiers that predict whether an effect will occur when an action is executed, given state and properties of the object. Thus, most informative examples for this learning problem are ones that are at the boundary of the effect occurrence, *i.e.* in which the effect *barely happened* or *almost happened*. Due to the way that discriminative classifiers generalize, after only seeing these borderline examples, the learner will have correct

predictions on the prototypical examples as well. A good teacher should therefore directly present borderline examples. Human teachers in this experiment do the opposite. They start with prototypical examples and then move on to borderline examples. As a result the progress of the learning is slower. The classifiers trained with data obtained by individual participants did not have a high accuracy (around 60%, Fig. 21). This shows that the time they allocated for teaching, was not sufficient in reaching a reasonable performances due to the slow progress.

4.4.1.4 Additional support

Other experiments in this thesis provide additional support for the sub-optimality of examples provided by human teachers. Experiments presented in Chapter 9 and 10 compare unguided human teaching with instructed human teaching. All experiments demonstrate that unguided teaching is sub optimal as in Experiment 1 in this chapter. In Experiment 10, humans are far from the optimal teaching algorithm, when they are teaching face categories represented as conjunctions in the Chernoff faces domain. In Experiment 11 which involves teaching a threshold that discriminates two categories of alien eggs over a single dimension of spikiness, people fail to provide the two examples that are at the border of the threshold. In Experiment 12 which involves teaching mouse gesture categories, classifiers trained by humans perform much worse than a greedy (non-optimal) teaching algorithm on a test set. In Experiment 13 which involves teaching navigation tasks where there is a highly limited number of possible demonstrations, people are unlikely to pick the optimal one.

Argall et al.'s survey on LfD provides an extensive discussion of issues related to demonstration datasets, which they group as *undemonstrated state* (discussed in Sec. 4.4.2) and *poor quality data* [8]. Poor quality data refers to demonstrations of skills that are unsuccessful in achieving the goal, sub-optimal with respect to some cost, or inconsistent across demonstrations. In addition they point out issues in the smoothness of trajectories

provided by human teachers, as well as the common occurrence of noise in the form of unintended wrong labels.

Kiaser et al. also characterize sub optimality of human teachers in two categories [71]. A demonstration that is sub optimal with respect to the robot, is a demonstration that is slower or has higher energy consumption, than an optimal execution permitted by the robot platform. A demonstration that is sub optimal with respect to the learning system, has missing information about the skill. They point out five sources of sub-optimality in the provided demonstrations: (i) unnecessary actions that are irrelevant for achieving the goal, (ii) Incorrect actions, (iii) unmotivated actions whose preconditions depend on world states that are not available to the robot, (iv) bad choices of scenarios in the provided demonstrations, and (v) the specification of wrong intentions. While some of these are specific to the skill and tasks representations used in their system, the identified issues are of similar nature to the ones discussed in this chapter.

4.4.2 Humans do not complete teaching

The second issue that pointed out by the three experiments is about the completeness of teaching. In all experiments the decision of when to stop teaching was left to the teacher. In both Experiment 1 and 2 the teacher could test the robot on what it had learned and in Experiment 3 they fully observed what experiences the robot had as a result of their scaffolding.

4.4.2.1 Support from Experiment 1

Only 25% of the participants were able to uniquely identify the target concept and only 58.33% were able to achieve a perfect F-score. This means that the teacher stopped giving more examples while the robot could still not correctly classify a number of objects. This seemed to be mostly due to the difficulty of maintaining a mental model of what the robot has learned. A number of participants tried to extensively test the robot to find gaps in its knowledge, however the chance of randomly doing tests that will reveal the robot's failures

is low and intentionally doing these tests also requires a good mental model. Another factor is the inefficiency of testing the robot. Given that there are 552 possible object instances, it is not possible for the person to test the robot on every instance.

4.4.2.2 *Support from Experiment 2*

As pointed out in Sec. 4.4.1.2, the success of the skills taught in this experiment were low. This means that human teachers stopped teaching before success was achieved.

In addition, teaching with a single demonstration was common. This means participants did not show any variance in the skills. In other words, they did not cover the space of possible ways of performing the skill. This results in skills that are restricted to a particular trajectory. Such skills can easily become inapplicable when there are obstacles due to clutter in the environment, whereas skills that have higher variance whenever possible are more flexible and generalize to new situations.

4.4.2.3 *Support from Experiment 3*

Most data sets provided by individuals were not sufficient in training highly accurate classifiers. However when combined, the data set by multiple individuals were on par with having a systematic data set that uniformly covers the complete space (Fig. 21). This indicates that participants stopped the training before they had completed teaching all affordances of the objects.

In addition, it was observed that some rare action effects were never demonstrated by some participants, resulting in a completely negative data set. For example, some individuals provided data sets that had no instances of the *opened* effect of the small box, or the *tipped* effect of the cuboid. This was either because they did not think of the effect and intentionally provide configurations in which the effect would occur, or because they did not present sufficient configurations through which they would accidentally discover the effect.

4.4.2.4 Additional support.

The second issue observed in the three experiments are also supported by other experiments in subsequent chapters. The findings of Experiment 1 which involves task learning in the toy objects domain, were replicated in the baseline of Experiments 9 and 10 which compare unguided teaching of conjunctions with instructed teaching.

Adding to the findings from Experiment 2 on teaching skills, other experiments also demonstrate that humans do not insert sufficient variance in their skill demonstrations even when they are constrained to provide multiple demonstrations. In Experiment 6, the trajectory demonstrations provided by humans are confined to a certain region of the robot’s reachable space and even a precoded trajectory used for making queries largely deviates from the average demonstrations provided by participants (Fig. 26). Experiment 8 and 14 explicitly compare the coverage of skills learned through unguided teaching, with respectively (i) learning with embodied queries and (ii) learning through instructed teaching. In the common baseline dataset for these experiments, it is observed that unguided teaching leaves large portions of the state space uncovered.

This problem is pointed out in the literature by Argall et al. who refer to the problem as *undemonstrated states* [8] for which they reference methods for generalizing to new states from existing states, and methods for acquiring new demonstrations (*i.e.* Active Learning).

4.5 Useful Patterns in Unguided Teaching

Data sets provided by human teachers also have certain properties that are favorable for machine learners. This section highlights some of the advantages of learning from human input.

4.5.1 Humans provide balanced datasets

Learning a classifier requires data from multiple classes. When the data set involves very few examples from a particular class, it is difficult to model it correctly. In the experiments

that involve binary classification problems (Experiment 1 and 3), the uniform sampling of the space results in a highly unbalanced data sets. However it is observed that humans sample the space in a way that balances positive and negative examples.

Unbalanced data is a common issue in ML and several approaches have been proposed to deal with it. The solution provided by humans resembles one an approach called *random under-sampling* which randomly selects a subset of the larger set in order to reduce it to the size of the smaller set. Having teachers do this for the learner is preferable since it avoids wasting the teacher’s time and since humans can subsample intelligently to provide a more informative set of examples than a randomly subsampled subset.

4.5.1.1 Support from Experiment 1

In Experiment 1, around 3% of the instance space is positive (*e.g.* 16 instances of *HOUSE* out of the possible 552 object instances), whereas participants provide about equal number of positive and negative examples (45% +, 55% -). This shows that humans are able to adaptively sample the instance space so as to present a balanced dataset.

4.5.1.2 Support from Experiment 3

Similarly in Experiment 3, the uniformly sampled data set has a higher percentage examples with *no effect* (45-50%) than the average data set provided by human teachers (20-30%) (Fig. 20). This largely reduces the ratio of negative examples to examples in the other categories, increasing the correct detection rate of particularly rare categories (*opened, rolled*, Fig. 21). This was reflected in the performance of the trained classifiers (Fig. 21).

4.5.1.3 Additional support

Balanced datasets are also observed in the unguided teaching baseline of other experiment. For instance in Experiment 11, a uniform sampling of the sample space would results in a 1 to 4 ratio of positive and negative examples, whereas the example sets provided by the

participants have an average ratio of 1 to 1.1.

4.5.2 Humans partially use teaching heuristics

Although humans are in general sub-optimal teachers for machine learners, there are occurrences of useful patterns in their teaching that correspond to parts of an optimal teaching algorithm or that demonstrate an intuition for the priority of examples. This corresponds to a naturally occurring, partial use of the teaching heuristics discussed in Chapters 9 through 11, which explicitly instruct the teacher to teach in a particular way that is known to be useful for the learner.

4.5.2.1 Support from Experiment 1

Humans naturally perform some parts of the optimal teaching algorithm for the problem considered in Experiment 1. For example, 91.67% (22/24) of participants started teaching with a positive example, as in the optimal teaching sequence. Positive examples given by teachers are mostly informative (more good examples than bad in Fig. 10). Thus, people were able to partially follow the first part of the optimal teaching algorithm that requires showing positive examples varied in the irrelevant dimensions.

In addition, the ratio of positive and negative examples (45% +, 55% -) is not only balanced in terms of positive and negative examples (as pointed out in Sec. 4.5.1.1), but it is also close to the ratio in an optimal teaching sequence which indicates 40% positive and 60% negative examples (Fig. 9).

4.5.2.2 Support from Experiment 3

As pointed out in Sec. 4.4.1.3, useful examples in this Experiment are the ones that are at the boundary of the two categories. Humans provide more examples that are at the boundary of reachability as shown in Fig. 19. The problem that was highlighted in Sec. 4.4.1.3 is that these examples were not provided first. People started teaching with what they taught were simple examples. In other words, they were right in their assessment of the example

being simple, however they did not correctly predict that complex examples would have been more informative than simple ones.

Another evidence from this experiment that humans had a correct intuition about the complexity of the learning problem was that the number of examples they provided for particular objects were proportional with the number of affordances that object had (Fig. 18). They could correctly assess that more examples needed to be given for objects with more affordances, in order to learn all possible affordances equally.

4.5.2.3 *Additional support*

As in Experiment 1, the partial use of the optimal teaching algorithm is seen in participants in the baseline unguided teaching condition of Experiment 10. Participants' description of their own teaching strategy can also contain elements of teaching heuristics or describe steps of optimal teaching algorithms (29). Similarly in the baseline of Experiment 11, humans show a similar behavior as in Experiment 3, where they slowly converge towards the decision boundary while teaching a threshold on a single dimension. While partial use of teaching heuristics is common in unguided human teaching, rare occurrences of fully optimal teachers or apparent uses of teaching heuristics discussed in Chapter 9, are observed in Experiments 10, 12, 13 and 14.

4.6 *Summary*

This chapter presents three experiments that explore natural, unguided human teaching. The experiments involve three types of learning problems. Experiment 1 involves task learning, Experiment 2 involves skill learning and Experiment 3 involves *action model learning*. This consists of learning classifiers that predict the effect of performing a discrete action on an object in a given state. The experiments reveal similar issues in all three learning problems, which were discussed under two categories.

- *Humans provide sub-optimal examples.* Although the notion of optimality varied across domains, sub-optimality was a problem revealed in all three experiments. In Experiment 1, human teaching was compared to a well-defined optimal teaching algorithm, revealing sources of sub-optimality such as making no progress due to redundant or repeated examples, or making progress that is much slower than the optimal teacher due to bad choices. In Experiment 2, participants provided corrupted and inconsistent demonstrations that resulted in unsuccessful skills. In Experiment 3, the ordering of examples by participants was from prototypical to borderline, whereas presenting borderline directly is much more efficient.
- *Humans do not complete teaching.* In all three experiments the final performance of the learner was unsatisfactory. In Experiment 1 participants could not uniquely identify the target hypothesis and did not reach a reasonable prediction accuracy. In Experiment 2 participants often provided only a single demonstration and did not show any variance, and the resulting skills did not have a high success rate. In Experiment 3 classifiers trained by participants did not reach the performance of a uniformly sampled dataset or the performance of the data set that combines multiple participants.

These observations motivate the necessity of guiding human teachers in their teaching interactions with robots.

CHAPTER V

EMBODIED ACTIVE LEARNING

The three initial experiments in Chapter 4 demonstrate that examples provided by naive human teachers are not ideal inputs for the associated machine learners. They result in slow progress and they do not teach the concepts fully. In addressing these problems for machine learners, Active Learning (AL) is a promising direction. As detailed in Sec. 2.3, Active Learning (AL) is a technique, that speeds up the learning by allowing the learner to choose the examples that it will learn from, *i.e.* by making queries. The first mechanism studied in this thesis is based on this idea.

Embodied queries are questions asked by the robot by leveraging its embodiment to change the world state or to facilitate the communication of the question. The primary objective of the question is to improve the learned skill and task, either by directly requesting information or by steering the teacher towards providing more informative demonstrations. Gaining useful information is not granted by the act of asking a question. To achieve its purpose, the question must be understood by the teacher, and correctly answered.

A primary goal of this thesis in developing embodied queries, is to effectively apply existing AL techniques to robot learning problems in which the source of the data is a human teacher. This presents several challenges that arise from the differences of this scenario from conventional applications of AL. Some of these differences are highlighted in the following.

Selecting a question. In typical applications of AL, asking a question corresponds to selecting an unlabeled instance from a set of candidates (*i.e.* a pool of unlabeled samples), and answering a question is providing a label for the instance. For example, in an image based object classification problem, the question would be showing an image on the screen

and the answer would be the category label of the object in the image. In spam detection the question corresponds to displaying a certain email, and the answer is saying whether the email is spam or not. For many robot learning problems, particularly in skill learning, there is no pool of unlabeled samples from which to select a query for the human to label. Thus the robot needs to select samples from infinite state spaces (*e.g.* instantiating a pose or trajectory in the joint space of the robot manipulator), or from the set of possible world states in the physical environment of the human and the robot.

Instantiating a question. Secondly, once a sample has been selected, it needs to be physically instantiated by changing the robot's own state (*e.g.* moving its arm to a particular pose) or by changing the world state (*e.g.* moving an object somewhere else). These states cannot be directly attained; the robot needs to move through a sequence of states to attain the state for which it wants to ask a question about. It also needs to make sure that these intermediate states are safe. This stands in contrast with the ease and speed of displaying an image or email on a screen to make a query.

Communicating a question. The robot needs a way to communicate what the question is and correctly parse the answer. This requires correctly referencing parts of the state, and using vocabulary that is meaningful for humans. In conventional applications of AL, the question fixed; "Is this spam?" or "Name the object in the image"; and does not need to be repeated. On the other hand, a robot should be able to take advantage of the different types of questions it can ask. Unless the robot has a computer screen, the questions need to be communicated through speech and gestures, and should allow the human to request a repetition.

Number of questions. Conventional applications of AL employ a large number of teachers who are often paid for their service. Although, active learning allows reducing the number of labels needed, the scale of an interaction completed by these humans may still involve providing a large number of labels (*e.g.* hundreds). In other words, rather than reducing the

number of labels obtained by one person, AL allows reducing the number of people that need to be employed to perform the labeling task. On the other hand it is feasible for a robot to acquire a large number of labels within a single interaction. A long stream of questions within a social interaction can be inappropriate. Therefore, the robot's questions need to take into account the limited budget of questions that can be asked within an interaction, which is likely limited to one to two dozen questions.

Turn taking. In conventional applications of AL, the human labeler dominates the interaction. As soon as one label is provided, the next instance to be labelled is made available. This is often viewed as a task for the labeler, for which they might get paid for. In contrast, generating queries on a robot takes significant time (instantiating states, utterance of questions) during which the human waits and observes the robot. This creates a reciprocal interaction between the human and the robot, which can be viewed more as a social interaction rather than a task for the human. It is therefore important to create a balanced interaction and time questions appropriately to maintain the fluency of the interaction.

All of these differences contribute to the challenges in making robots ask questions to guide their teacher. While some of these challenges require developing new techniques for producing embodied queries, others are related to the dynamics of the interaction and need to be studied empirically. To this end, an initial observational experiment was conducted. This involves directly applying existing AL techniques for generating queries. This experiment reveals interaction related issues that arise in having robots ask questions in a human-robot interaction scenario.

5.1 Experiment 4: Active task learning

The goal of this experiment is to observe human-robot interactions that arise from having an active learner in a task learning scenario. Through a small scale user study an active and passive learner are analyzed and compared descriptively. This experiment highlights

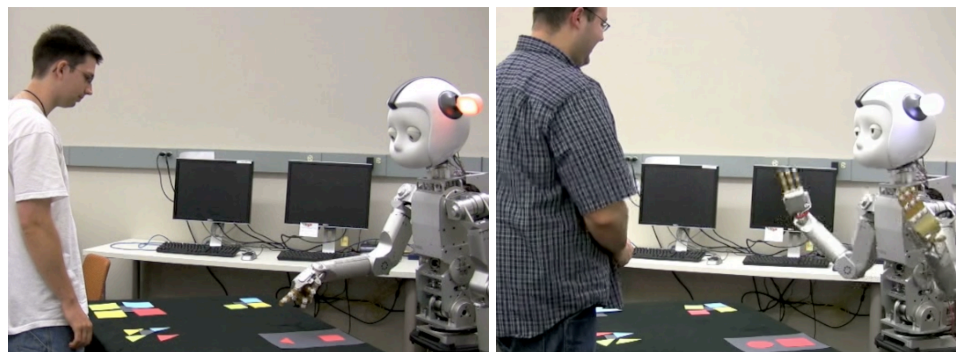


Figure 22: Snapshots from Experiment 4 showing gestures by Simon for (a) pointing during a query and (b) shrugging to indicate that he is uncertain about the answer.

three issues in applying AL to human-robot interaction scenarios, discussed in the subsequent sections¹.

5.1.1 Experimental Design

The domain for this experiment is the nine-feature toy objects domain in which object categories are described as non-monotone conjunctions (Sec. 3.2.3). As in the first observational experiment that involved task learning (Experiment 1), this experiment involves teaching object categories to Simon by presenting positive and negative examples.

The query mechanism for this experiment selects the most uncertain sample from the space of all possible objects, based on the votes of the possible hypothesis (*i.e.* the version space). This method is described in Sec. 3.2.2. In order to make a query, Simon needs to configure the sample object for which it wants a label. Since Simon is not able to manipulate the flat object parts, he asks the teacher to configure the sample. To limit these requests, Simon selects a query that differs from the sample that is already in the demonstration area by only one part, while also being maximally informative based on the query selection mechanism. Different from Experiment 1, Simon did not use speech in this experiment. The query consist of pointing to the part that is different in the query

¹This experiment appears in [37].

object and then pointing to the location that it needs to be placed (*i.e.* top or bottom) in the demonstration area. To disambiguate the referred object parts, Simon’s ears displayed the color of the object it was pointing to.

Sometimes examples cannot be queried because Simon can only request that the human change the current example by a single part, either the top or the bottom. If there are no useful queries that differ by a single object, Simon will attempt to query the preferred example in two requests. The intermediate compound object that is requested first can therefore be uninformative for learning. This was done to simplify the interaction and to keep the cognitive load reasonable for the human teacher.

Participants were tasked with teaching Simon four different object categories, referred to as HOUSE, PERSON, SNOWMAN and RAINBOW. The ground truth representation for each category is shown in Table 2. These concepts were chosen so as to vary specificity/generalizability. Specific concepts involve very few instances, while general concepts cover a large portion of the set of all possible compound objects.

When the human teacher tests the robot on an example, Simon has five possible responses: *yes*, *uncertain yes*, *uncertain*, *uncertain-no*, and *no*. The *yes* animation is the head nodding, and the *no* animation is the head shaking from side to side. The *uncertain* animation is designed to communicate lack of confidence. It tilts the head sideways and lifts the arms upwards in a bent-elbow configuration to create a shrugging gesture. By playing the *uncertain* animation simultaneously with either *yes* or *no*, Simon’s response can communicate that he is leaning towards one answer without being fully confident in that answer.

5.1.1.1 Procedure

Participants interact with the robot through three speech commands for labeling a configured instance as positive or negative, and for testing the robot.

The experiment is a between subjects study with two conditions. The first condition

is *Passive* learning in which the teacher selects all the examples and labels it. The second condition is *Active* learning in which the learner chooses the examples to be labelled by the teacher. Since the robot requires an object to be already configured in the demonstration area, the first example is chosen by the teacher. In addition the two conditions differ in the transparency level of the robot's answers to the tests. The *Passive* condition only has the three possible responses (yes, no, uncertain), whereas the *Active* condition involves the five-level graded responses. Participants were not instructed about the robot's transparency mechanisms or queries, but it was emphasized that the robot may communicate with them about the provided example.

Since Simon did not speak, participants were instructed to watch for Simon's response to their speech commands. Simon blinked the lights on his ears when he was finished responding, to indicate to the teacher that he was ready for another demonstration or for an answer. Participants were told to put any pieces used for demonstrations back in their original locations in the workspace when not currently in use. Simon used these hard-coded locations for pointing to desired object parts.

The feature space and the concepts were explained to the participants prior to the interaction and reminded by a whiteboard near Simon. Participants were allowed to teach HOUSE, PERSON, and SNOWMAN in any order, but RAINBOW was always last. This was done to allow participants to become familiar with the objects before teaching a more general and abstract concept. Participants were told to continue teaching each concept until they were satisfied that Simon had learned it well or thought that Simon had stopped making progress. Upon completion of the interaction, they were interviewed with several questions.

5.1.2 Findings

This experiment was run with nine participants, two in the *Passive* condition and seven in the *Active* condition. A diverse range of interactions were observed. The following presents

the salient characteristics of these interactions in a descriptive manner.

Ambiguity in Queries. The study demonstrated that it is important for queries to be clearly understandable. 5 of the 7 people in the *Active* group did not correctly interpret Simon's nonverbal gestures as queries. People had a variety of interpretations as to what the gestures specifically meant (such as pointing at a shape that was relevant or displaying red color on the ears when angry), but essentially the only information they understood from the robot's communication was the fact that it needed more examples, not specifically what kind of examples were needed. This seems to have led to worse performance than the *Passive* group. An explanation for this, is that the robot's gestures caused participants to form a variety of different inaccurate mental models about the learning process that in turn led to inefficient teaching, whereas in the *Passive* case, people devised a teaching strategy to take the lead and systematically explore the feature space.

As a result of this artifact of the experiment, the findings are analyzed in three groups rather than two. In the rest of this section, the five participants in the *Active* condition who did not correctly interpret the robot's queries are referred to as the *Failed-Active* group. The other two participants that correctly interpreted the queries are referred to as the *Successful-Active* group.

Teaching Performance. In order to compare the relative performance between the groups, the accuracy of taught concepts is investigated, comparing the trained model's predictions against the ground truth labels for the 552 possible compound objects. Performance is measured with F_1 -score.

Of the two participants in the *Successful-Active* group, one created and labelled every example suggested by the robot and did not begin testing the robot at all until the robot stopped making queries. This participant taught concept models with 100% accuracy for all four concepts. The other participant did not recognize the queries until the third concept was learned, but his third and fourth concepts were 100% accurate. The *Failed-Active* and

Table 10: Accuracy, time taken to teach, and efficiency for the three post-hoc groups in Experiment 4.

Group	# of participants	Accuracy	Time	Efficiency
<i>Successful-Active</i>	2	93.75%	20.33 mins	18.79%/min
<i>Passive</i>	2	79.52%	24.56 mins	13.45%/min
<i>Failed-Active</i>	5	70.76%	31.94 mins	10.74%/min

Passive groups had an average F_1 -score of 70.76%.

Additionally, the efficiency of teaching in the three groups can be characterized. Efficiency is defined as the accuracy divided by the wall clock time taken to teach. As shown in Table 10, the *Successful-Active* group had the best efficiency, followed by *Passive*, and then *Failed-Active*.

Balance of Control. Active learning has the potential to improve performance, but this experiment indicates that an interaction dictated entirely by active learning may be undesirable from the human teacher’s perspective. Both of the participants in the *Successful-Active* group expressed a desire to take a more active role in structuring the learning process rather than blindly answering all of the robot’s questions.

The participant who cooperated fully with the robot’s active learning was mostly reliant on the robot to guide the interaction. He stated that he was glad that the robot was able to figure out for itself what it needed to know, and that he stopped bothering to keep track of what the robot could be caught on. Although he said he trusted the robot more as time went on, he also said that he was skeptical when the robot stopped asking questions and thus asked many test questions to the robot. On multiple occasions he also inserted an example that the robot did not request because he wanted to give demonstrations that he believed to be representative, “just in case”. The result was that he actually gave 12.19% uninformative examples; these instances were ultimately unnecessary. So although he felt compelled to

take control of the interaction occasionally, these actually decreased efficiency.

The participant who correctly interpreted the queries while teaching the last two concepts had a different situation. When he recognized the robot's active learning strategy, he considered it inefficient due to the constraint of varying only a single part of the object; he thought that varying both the top and the bottom would be more efficient. Knowing the limitation of his own memory, he said he wished he could trust the robot's selected example but did not trust it because of this perceived deficiency. He also said he felt like he wanted to "disobey" what the robot wanted to be taught, and that he sometimes "gave in" to the robot out of "pity." One participant from the *Failed-Active* group, upon learning the purpose of the gestures, exclaimed that she felt bad for Simon that she did not answer any of his questions. These remarks suggest that active learning exerts a strong influence on the interaction in swaying the balance of control towards the robot, which could actually be an undesirable effect in an interaction.

The opposite extreme of the spectrum is the *Passive* group, in which the human teacher has full control over the learning process. In their exit interviews, these participants explicitly stated either that they wished the robot could ask questions, or that they wished they could ask the robot what exactly it did not understand. In essence, these participants felt that the robot did not participate enough in its own learning process and needed to maintain a higher amount of control in the interaction.

5.2 Issues in Embodied Active Learning

5.2.1 The balance-of-control problem

In conventional Active Learning, every label provided by the human is a response to a query made by the learner. When considered as a social interaction, this results in an unbalanced interaction dominated by the learner constantly asking questions. The experiment presented in this chapter (Experiment 4) demonstrates that this is undesirable. It can result

in annoying the person and affect longer term interactions negatively. This finding was confirmed in the literature, through a study involving an active recommendation system [62]. Secondly, it can cause the human to stop responding to the robot’s queries in order to take control of the interaction. Thirdly, by fully complying to the robot’s queries humans can end up with vague and possibly incorrect mental models of the learner.

Giving full control to the learner also results in missed opportunities for more efficient teaching. For instance, in the experiment just described (Experiment 4), the robot was limited to queries that differed from the previous example by a single part, which resulted in inefficiencies. In addition, a good teacher can teach with less examples than the number of queries needed to learn the same concept actively. For the task learning representation used in this experiment, an optimal teaching sequence has $\min(n + 1, r + 2)$ examples, whereas an active learner requires $n + 1$ provided that the very first example is positive. Therefore when $r \ll n$, the teacher can teach much more efficiently by taking more control.

A solution to the balance of control problem is proposed in the next chapter. The proposed approach, called Intermittently-Active Learning (I-AL), is to selectively make queries and allow the teacher to have control of what examples to present when no queries are made. The two baselines of the experiment that evaluates I-AL (Experiment 5), *i.e.* passive learning and fully-active learning, also support the observations from Experiment 4 with quantitative evaluation metrics. Both experiments suggest that balance of control is an important issue to be addressed in designing interactions that involve an active learner.

5.2.2 The question clarity problem

A second concern in designing robots that ask questions is to make sure the question is communicated properly. Most applications of AL involve a teacher interacting with a desktop computer, *e.g.* labeling images or documents. These allow displaying the question, or the information required from the human, in text format. Robots that have screens as part of their embodiment, can leverage this non-ambiguous form of questions asking. However

social robots like Simon, need to communicate the question in the way that humans do. Even if the question is always the same, or the teacher is instructed that the robot may ask a particular question, the person needs to detect when the robot is asking a question. This creates room for ambiguities and misunderstandings which should be carefully handled through interaction design.

Secondly queries in task or skill learning involve a reference to a world or robot state. Such references should be unambiguous. In the experiment presented in this chapter, the non-verbal gestures were designed with the belief that participants would have no trouble understanding what the robot was communicating, whereas a majority of the participants proved this wrong. This points towards the importance of iterative, user-centered design of interaction behaviors of robots, to ensure precise communication between the robot and the human in active learning interactions.

5.2.3 The compliance problem

Compliance refers to how likely the teachers are to answer the learner's queries. Asking a question that does not get answered is one of the worst cases for an active learner. Since asking a question might require the robot to change the state world state, use gestures like pointing, or utter a sentence; the time spent on these actions is wasted. Secondly, this is often an indication of another problem. For instance, the other two problems mentioned earlier, *i.e.* imbalanced interactions that involve a constant stream of questions or questions that are not correctly understood by the robot, could result in non-compliance.

Another problem that results in non-compliance is the intelligibility of the question from a human perspective. In Experiment 4 one of the participants perceived asking questions by changing a single object part at a time as inefficient, and did not respond to them. A similar result is revealed in the follow up experiment presented in Chapter 6 (Experiment 5). It is observed that queries that involve asking a label for a non-informative example (due to the limitation to change one part at a time) have much lower compliance than queries

that are actually informative.

These observations demonstrate that sources of non-compliance should be carefully addressed in designing embodied active learners.

5.2.4 The noisy teacher problem

Even when a question has been accurately communicated, and the teacher is willing to answer it, there is still a chance that they will not provide the right answer. Teachers may be benevolent, however they can make mistakes. This is a common issue in classical applications of AL and points to the importance of having learning algorithms that are robust to noise, and do not 100% trust answers provided to questions.

In the presented experiment, this problem did not occur very frequently, however when it did, it delayed the robot's progress by requiring more examples to compensate for added uncertainty due to the mistake. Therefore it is important for the questions to reduce the probability of mistakes in the answer in whichever ways possible. This problem has been extensively addressed by Rosenthal et al. in scenarios such as a humanoid robot that learns an assembly task from a human [112] or a human labeling their own actions recorded through sensors on a mobile phone [113]. They demonstrate that providing additional types of information, such as the context perceived by the learning agent, the agents uncertainty or its intents, provide differing benefits in improving the accuracy of labels provided by humans.

5.3 Summary

This chapter outlines the differences between conventional applications of AL with robot learning problems, relating to how questions are selected, instantiated and communicated, how long a question takes and how many questions can be asked within an interaction. Then a small scale observational experiment explores the use of existing AL techniques in a scenario where a human teaches a robot task goals from examples. The experiment compares interactions with passive and active learners in several dimensions. Four issues associated

with interactions are highlighted: balance of control, question clarity, compliance and labeling noise. While developing new methods to allow the generation of embodied queries are needed, the observations from this experiment emphasize the importance of interaction design in the way these queries are instantiated and communicated.

CHAPTER VI

INTERMITTENTLY ACTIVE LEARNING

In passive supervised learning the teacher selects all samples to label, while in standard active learning the learner selects all samples for the teacher to label. These are two extremes of a spectrum, in which the responsibility of choosing samples is completely on one agent versus the other. As seen in Chapters 4 and 5, both extremes are problematic. Completely teacher-directed learning results in slow progress and incomplete learning, while completely robot-directed learning results in unbalanced, unnatural interactions, potentially causing compliance problems.

Instead of these two extreme cases, this thesis proposes sharing the responsibility of choosing learning samples between the teacher and the learner. In this interaction the robot makes queries only intermittently; and when it does not make a query the sample is chosen by the teacher. This framework is referred to as Intermittently-Active Learning (I-AL).

I-AL requires a mechanism for the learner to decide *when* to make queries. Two approaches are proposed and evaluated in this chapter. The objective for these mechanisms is to maintain the guidance power of AL, while creating a more balanced interaction.

6.1 Teacher-triggered queries

The first approach gives control of the query timing to the teacher. That is, the teacher decides when the learner will make a query and the learner will never take the initiative to make a query. This type of query is referred to as a *teacher triggered query*. As demonstrated in Experiment 4 which compared a fully-active learner with a passive learner, humans prefer to have more control on the interaction. Thus a reasonable hypothesis is that having control over the timing of the robot's questions would be desirable from a human's perspective. Note that the teacher might choose to never trigger a query in this framework.

6.2 Conditional queries

The second approach lets the robot decide when to make a query based on its progress in learning. After each example, the robot needs to decide whether to query the next sample or whether to let the teacher continue teaching. This decision is based on the comparison of the expected informativeness of an example that will be provided by the human teacher, denoted by U_t^0 , and the expected informativeness of the example that would be queried if the robot was to make a query, denoted by U_t^q . The subscript t indicates the step of the interaction. If $U_t^q > U_t^0$, then the robot decides to make a query at step t .

U_t^0 should capture both the informativeness of the teacher’s history of examples, and the informativeness of possible examples that the teacher might choose to give. This could potentially model the teacher and predict their next example. U_t^q should capture the informativeness of the query suggested by the query selection mechanism at step t . Specific examples of both measures are given in the next experiment (Experiment 5), for the conjunction learning problem.

6.2.1 Conditional queries for task learning

The informativeness of an example is represented with the fraction with which the version space is reduced by the example. As explained in Sec. 3.2.2, once the robot has seen a positive example, all of its queries are guaranteed to reduce the version space by a factor of 2 (*i.e.* halve the version space). Thus as long as the expected informativeness of the predicted human example is at least 2, the robot should not make a query. For this experiment, instead of exactly estimating the expected informativeness of the future human example, the learner heuristically decides to make a query when either of two events occur: (*i*) if the example given by the teacher is uninformative (*i.e.* does not reduce the version space) for two consecutive steps, or (*ii*) if the probability of a randomly chosen example making non-zero progress is lower than a certain threshold.

The threshold is selected such that, if the teacher provided an optimal teaching sequence

for the given concept, only the last example in the sequence would be queried by the learner. For instance, for the particular concepts used in the next experiment, the learner makes a query when it cannot confidently predict the label of less than 2.17% of all examples. This corresponds to a less than 2.17% probability of the teacher randomly presenting an informative example.

6.3 *Experiment 5: Intermittently-active task learning*

This experiment compares the two approaches for I-AL, namely teacher-triggered queries (TTQ) and conditional queries (CQ) with the standard active learning protocol that always makes a query (*i.e.* unconditional queries (UQ)) as well as the passive learner that does not make queries (*i.e.* no queries (NQ)). The I-AL approaches are designed to achieve the benefits of unconditional queries over no queries, while creating more balanced interactions than unconditional queries. Thus, this experiment seeks to verify whether this goal is achieved by either approach.

The following describes the conditional query mechanisms implemented for this problem. Note that the implementation of teacher triggered queries is trivial, in that it only involves changing the interaction¹.

6.3.1 Experimental design

The experiment involves task learning in the six-feature toy objects domain (Sec. 3.2.3). The learning algorithm is the improved halving-algorithm described in Sec. 3.2.1. The query selection mechanism is the query-by-committee method described in Sec. 3.2.2. Simon learns four different conjunction concepts in this state space, described in Table 1. All concepts are chosen to have three terms in the conjunction. The actual number of possible instances for each concept differs due to the number of values that features can have and the uniqueness of objects.

¹This experiment appears in [22].

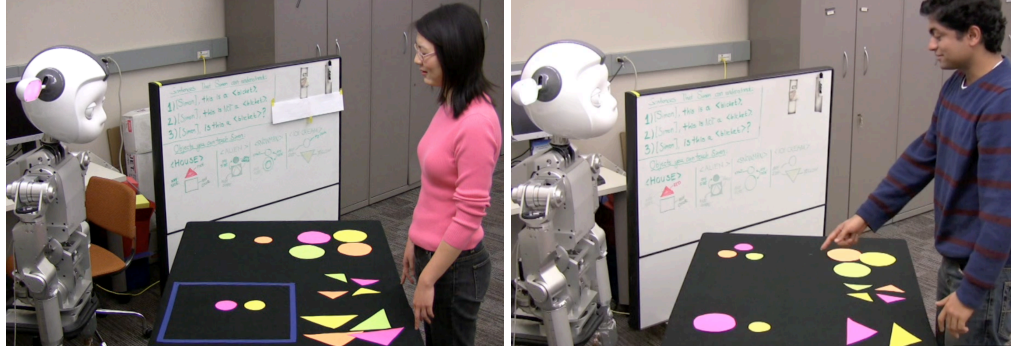


Figure 23: Snapshots from Experiment 5 involving human teachers demonstrating different toy object concepts to Simon.

6.3.1.1 Interaction

The interaction in this experiment is similar to the initial experiment investigating embodied queries (Experiment 4). Simon makes queries by requesting to replace the top or bottom piece of the compound object that is currently on the demonstration area. The request is communicated through speech synthesis using a sentence such as, “Can you replace the bottom piece with a *large pink circle*?”. During this utterance, Simon also gazes towards the group on the workspace that contains the requested object (*e.g.* the group of large circles) and changes the color of his ears to the color of the requested object (*e.g.* pink).

In this experiment, participants teach concepts to Simon through a reciprocal interaction. First, the teacher prepares a compound object in the demonstration area. Then, the teacher can do one of two things: *(i)* label the compound object as positive or negative based on the concept that is being taught, or *(ii)* test Simon on the compound object by asking him to predict a label. The teacher accomplishes these by saying sentences from a predefined grammar. When a sentence is heard, Simon gazes towards the demonstration area and perceives the compound object. If the teacher provides a label, Simon learns from this new labeled example and then acknowledges the example. If the teacher tests Simon, he responds with a predicted label based on what he has learned so far. The teacher’s next turn starts when Simon blinks his ears indicating that he is ready to see another example.

In the two conditions that involve questions initiated by the robot (UQ, CQ), Simon makes a query after the person provides a labeled example. In the TTQ condition Simon makes a query only if the teacher says “Simon, do you have any questions?” In all cases, Simon blinks his ears after finishing the query.

6.3.1.2 Procedure

The experiment is a between-groups study with four conditions corresponding to passive learning, standard active learning and the two proposed approaches for intermittently-active learning. These four conditions are respectively referred to as NQ, UQ, CQ and TTQ as indicated earlier. The latter three are also collectively referred to as the *interactive* conditions. The four concepts from Table 1 are paired with the four conditions and the pairing is kept constant across all participants. This pairing is as follows:

- No Queries (NQ) — HOUSE
- Unconditional Queries (UQ) — SNOWMAN
- Conditional Queries (CQ) — ALIEN
- Teacher-triggered Queries (TTQ) — ICE CREAM

Participants are first instructed about the speech commands for teaching and the four concepts. Every participant begins by teaching in the NQ condition as a baseline, but the order of the interactive conditions is counter-balanced.

Between the NQ condition and the first interactive condition, the experimenter explained to the participant that Simon was able to request specific examples by asking for either the top or the bottom piece in the demonstration area to be switched with a different piece. The experimenter also emphasized that the participant was not required to do what Simon asked for if they did not want to. Participants were not asked to comply strictly with the robot’s queries, so that the experiment could also reveal how well people naturally comply with the queries in different conditions.

6.3.2 Findings

Findings from this experiment are summarized in the following in terms of learning performance, subjective preference, balance of control and compliance.

6.3.2.1 Learning Performance

There is ample evidence in the literature supporting the use of AL over passive supervised learning. AL techniques can produce classifiers with better performance using a limited number of labeled examples, or reduce the number of examples required to reach certain performance [122]. One question posed in this experiment is whether or not a naive human serving as the robot’s labeler in an interactive setting can achieve such performance gains.

The experiment shows that AL significantly improves performance compared to passive supervised learning (NQ), while no significant differences are observed between the three different interactive conditions (UQ, CQ, TTQ) in which active learning was used. These results are summarized in Table 11. This holds true across the following metrics of performance:

- Overall accuracy at the end of the learning session, using the F_1 -score.
- Percentage of participants who taught the concept correctly and precisely (*i.e.* participants who reached a version space consisting of a single hypothesis that was the true hypothesis for the concept).
- Speed of convergence to the correct concept hypothesis (*i.e.* number of labeled examples given until the version space consisted only of the true hypothesis).

The results confirm that AL has better performance than passive learning both in terms of the accuracy of the learned classifier and the number of examples required to converge to a high accuracy. AL proved especially useful in terms of coverage of the instance space as indicated by the low percentage of participants who could teach the concept precisely in

the NQ condition (6 out of 24 participants). As pointed out in Experiment 1 (which serves as the baseline of this experiment), it can be difficult for human teachers to keep track of how much of the space they have covered even for a relatively small instance space. This experiment demonstrates that queries are a powerful guidance mechanism that addressed this issue.

There was no significant difference in performance between the three interactive conditions that used active learning. Since compliance with the robot’s queries was not enforced, the achieved performance is a result of both examples elicited through queries and examples given by the participants independent of the queries. Compliance of the participants with the robot’s queries is discussed in more detail later. Nevertheless, more complex tasks might reveal differences in performance between the three interactive modes.

Table 11: Learning performance of four the conditions in Experiment 5. $|V| = 1$ corresponds to having a single hypothesis in the version space, *i.e.* exact identification of the target.

Condition	Final F_1 -score	Participants who achieved $ V = 1$	Number of Steps to reach $ V = 1$
No Queries (NQ)	77.81%	6/24 (25%)	M = 10.67, SD = 4.59
Unconditional Queries (UQ)	100.00%	24/24 (100%)	M = 8.29, SD = 2.29
Conditional Queries (CQ)	98.61%	23/24 (96%)	M = 8.35, SD = 1.56
Teacher Triggered Q. (TTQ)	97.50%	20/24 (83%)	M = 9.30, SD = 3.53

6.3.2.2 Subjective preference

From the human partner’s perspective, all of the interactive conditions were more enjoyable and preferred over the interaction with no queries. Ratings from the survey are shown in Table 12, and a selection of subjective comments are shown in Table 17 and Table 18. Table 14 gives the significance of the effect of the different interaction conditions on these ratings as measured by 1-way ANOVA tests, and comparison of the NQ condition with the

Table 12: Mean and standard deviation over the subjective ratings given by participants in five different 7-point Likerd-scale questions in Experiment 5.

Scale	NQ		UQ		CQ		TTQ	
	M	SD	M	SD	M	SD	M	SD
B. of Control (7:robot)	2.50	1.38	4.46	1.67	3.25	1.45	3.25	1.39
Difficulty (7:hardest)	2.92	1.28	2.17	1.24	3.00	1.35	2.88	1.57
Mental Model (7:clearest)	4.08	1.82	5.08	1.77	5.17	1.40	5.17	1.55
Intelligence (7:best)	4.08	1.64	5.54	1.53	5.12	1.57	5.75	1.29
Enjoyability (7:best)	4.92	1.79	5.46	1.77	5.46	1.50	6.04	1.16

Table 13: Repeated measures ANOVA test results indicating the effect of condition on subjective rating scales in Experiment 5.

Scale	ANOVA
Balance of Control	F(3,23)=7.69, p<.001*
Difficulty	F(3,23)=2.69, p>.05
Mental Model	F(3,23)=3.39, p<.05*
Intelligence	F(3,23)=13.02, p<.001*
Enjoyability	F(3,23)=4.57, p<.01*

three interactive conditions using *t*-tests.

Perceived Intelligence. Simon’s intelligence was rated by participants as being higher in the three interactive conditions than in the NQ condition. Participants commented that Simon’s questions were “relevant” and “made him seem smarter,” and that he came up with “combinations that [they] had not considered or had forgotten to consider.” One participant stated that Simon’s ability to come up with examples himself was “definitely a sign of more intelligence than just passively listening to my instructions.”

In reality, not all of the robot’s queries to the teacher were informative. This was due to the nature of only requesting the top or bottom half of the example. Several participants

Table 14: Comparison of the NQ condition with the three interactive conditions (UQ, CQ and TTQ) on subjective ratings in Experiment 5.

Scale	NQ vs. UQ	NQ vs. CQ	NQ vs. TTQ
B. of Control	t(23)=-4.15, p<.001*	t(23)=-1.94, p>.05	t(23)=-1.72, p>.05
Difficulty	t(23)=2.16, p<.05*	t(23)=0.10, p>.05	t(23)=-0.25, p>.05
Mental Model	t(23)=-2.06, p>.05	t(23)=-2.57, p<.05*	t(23)=-2.41, p<.05*
Intelligence	t(23)=-5.05, p<.001*	t(23)=-5.95, p<.001*	t(23)=-2.99, p<.01*
Enjoyability	t(23)=-2.01, p>.05	t(23)=-3.58, p<.005*	t(23)=-1.88, p>.05

who perceived this shortcoming tested the robot with the queried intermediate example instead of labeling it, only to find out that the robot knew the answer already. One participant also stated that he “couldn’t be sure then if [Simon] was asking for the right examples,” showing his distrust of the informativeness of the queries. However, overall the participants still took the robot’s contributions to the learning process as demonstrating higher intelligence than never asking any questions at all.

Ease of teaching. The fully active UQ condition was considered the easiest out of the four conditions. Participants claimed that Simon’s asking of questions made it “easy to know what he knows and doesn’t know.” One participant considered it “easier to have Simon let me know if he needed to see an example,” as compared to doing all of the teaching himself in the NQ condition. Another participant commented that the UQ condition was “really easy because I didn’t have to think.”

One observation based on the comments was that many people rated the ease of the condition based on how conceptually intuitive the concept for that condition was to them. Although the concepts were designed to be the same in terms of specificity and the number of examples required to teach them, many participants perceived the complexity of the concepts as varying for the different concepts. The perceived relative complexity also varied across participants. This could be a reason that the ratings did not show a significant difference between the conditions in the way that the comments did.

Table 15: Accuracy of the learned concept as predicted by the teacher, compared to the true accuracy, in Experiment 5.

Condition	F_1 -score	Estimated	Discrepancy
NQ	77.81%	88.08%	-10.27%
UQ	100.00%	95.33%	+4.67%
CQ	98.61%	94.33%	+4.28%
TTQ	97.50%	91.38%	+6.12%

Table 16: Average time taken to teach concepts, number of steps and efficiency as defined by how much the F_1 -score was increased per step, in Experiment 5.

Condition	Time	Steps	Efficiency
NQ	6.82 mins	19.79 steps	5.12%/step
UQ	5.48 mins	20.42 steps	5.48%/step
CQ	5.70 mins	20.96 steps	5.17%/step
TTQ	6.30 mins	20.25 steps	5.42%/step

Enjoyability. Despite a number of negative comments about each of the interactive conditions (see Table 17), all three interactive conditions were ranked higher than the NQ condition in enjoyability. TTQ was the favorite among many participants. A t -test between TTQ and CQ shows a significant preference for TTQ, as shown on Table 14. Participants said it was “fun to answer his questions” and that it was “extremely enjoyable getting feedback from Simon.” This was in contrast to the NQ condition, which one participant even described as “very dull.”

6.3.2.3 Balance of control

As shown in Table 12, the rankings people gave for balance of control are as hypothesized. From the order of most to least human control are NQ, TTQ, CQ, and UQ. The data also

Table 17: A selection of positive comments about the three interactive conditions (UQ, CQ, TTQ) from the open-ended questions in the survey in Experiment 5.

Condition	Comments
UQ	"I didn't have to think" "greatly simplified the learning" "encouraging to observe his rate of learning"
CQ	"more interactive" "more of a dialog" "required less of my time"
TTQ	"easier to work with" "richer interaction" "felt very comfortable... felt complete" "enjoyed being able to direct the process more" "very clear when Simon has no questions" "easier than being asked questions more often"

Table 18: A selection of negative comments about the three interactive conditions (UQ, CQ, TTQ) from the open-ended questions in the survey in Experiment 5.

Condition	Comments
UQ	"I didn't feel like there was real interaction" "asked questions several times when I didn't want to be asked" "made me feel unnecessary" "lost track of what I was trying to teach" "turned my brain off" "constantly asked questions, it was annoying"
CQ	"wasn't confident when I was done teaching" "frustrating" "hard for me to keep straight" "confusing to determine whether he had learnt enough" "distracted by the robot"
TTQ	"less easy because I had to prompt Simon" "took more effort on my part than his" "required substantial teaching on my side"

Table 19: Comparison of the UQ condition with the other conditions in terms of balance of control ratings given by participants in Experiment 5, measured with *t*-tests.

Compared conditions	<i>t</i> -test result
UQ vs. NQ	$t(23)=4.15, p < .001^*$
UQ vs. TTQ	$t(23)=3.03, p < .01^*$
UQ vs. CQ	$t(23)=3.14, p < .005^*$

show a significant effect of the constant stream of queries in the UQ condition on sense of control, as shown in Table 19. Participants rated the UQ condition to have significantly less human control than all other conditions and the significance level decreased in the order of NQ, TTQ, and CQ.

Overall, it is observed that the participants' responses were skewed towards human control (closer to 1 than 7). That is, even in the UQ case when the robot attempted to direct all of the examples, people reported approximately equal balance of control with an average rating of 4.46. However, several participants mentioned feeling that they held a peripheral role in the learning process during the UQ condition, as shown in the negative comments in Table 18. Letting the robot control the interaction can cause a teacher to stop maintaining a mental model of the process, preventing the learner from achieving maximum efficiency when learning from a good teacher.

The two intermittently-active approaches of conditional queries (CQ) and teacher-triggered queries (TTQ) were an attempt to bring more balance of control to the interaction without sacrificing the benefits offered by AL. When presented with a forced choice between CQ and TTQ, one third of the participants preferred the CQ condition because it required less effort and seemed more balanced. The rest preferred the increased control over teaching offered by requiring permission to ask questions.

The comments shown in Table 17 and Table 18 are a representative selection of those given in the open-ended response boxes. Positive comments about UQ were characterized by triviality of teaching and speed of Simon's learning, and negative comments described

Table 20: Overall compliance by participants in answering the robot’s queries in Experiment 5.

Condition	Average number of queries	Percent answered
UQ	6.46	78.82%
CQ	3.04	68.61%
TTQ	3.29	65.55%

displeasure at being bombarded by questions or having a less significant role. Positive comments about CQ cited a balanced and efficient interaction, while negative comments cited confusion at the seeming randomness of the robot queries. The TTQ condition had the least negative comments, which mentioned that explicitly allowing Simon to ask questions took longer, and forgetting to allow him to ask questions made the condition too much like the NQ condition. The many positive TTQ comments described the interaction as feeling the most natural and easy, and enjoying the control of teaching in conjunction with the support of Simon’s queries.

6.3.2.4 Compliance

As described in Sec. 5.2.3, compliance refers to the likelihood of the teacher answering the active learner’s queries. This is an issue has not been explored in the literature, but will be highly important for active learning systems that need to learn from everyday people. In order to reap the benefits of active learning, the robot needs the human teacher to answer the questions it asks.

As a measure of compliance in answering Simon’s queries, the percentage of queries made by Simon that were labeled by the human teacher were examined. It is expected that I-AL approaches (CQ and TTQ) lead to better compliance due to being less demanding and leaving more control to the teacher compared to fully-active learning (UQ). However, the results refuted this hypothesis; teachers responded to a higher fraction of the UQ queries

Table 21: Compliance on answering queries, split according to informativeness of the query in Experiment 5 for the UQ and CQ conditions.

Condition	Informative queries		Uninformative queries	
	Answered	Not answered	Answered	Not answered
UQ	78.23%	14.76%	0.60%	6.42%
CQ	64.10%	9.51%	4.51%	21.88%

compared to the CQ or TTQ queries (Table 20).

This finding could be due to two issues. As mentioned previously, some of the queries made by the robot were uninformative and could already be classified by the robot. These queries were nevertheless necessary as intermediate steps to attain an informative query. The percentage of queries that are uninformative tends to be higher in I-AL conditions. In the CQ condition, the robot waits to make a query until the number of informative queries in the instance space becomes sparse. When very few examples are informative for pruning the version space, the probability that the example on the demonstration area has one common part with an informative example is low.

Table 21 compares compliance on informative and uninformative queries. These results are consistent with the original expectation: a higher percentage of *informative* queries are declined by the teacher in the UQ condition. Most of the queries that were declined in the CQ condition were *uninformative*. These results also show that participants were successful in detecting uninformative queries, either by maintaining a good mental model of what the robot knew, or by testing the robot with the queried examples. Note that the occurrence of uninformative queries could also have made the teacher more reluctant to respond to subsequent informative queries.

The other potential issue has to do with the human’s teaching strategy. In both CQ and TTQ, the lack of queries initiated by the robot early in the learning process seemed to establish a balance of control dominated by the teacher. A human teacher who seized

control early in the interaction may be more reluctant to give it up later on. The teacher might be following a certain strategy, and even when informative, a robot's query could seem out of place. This points to the importance of the questions asked by the robot being perceived as smart questions by the teacher. The best query might not just be an informative query, but also one that is appropriate with respect to the teacher's strategy, since it has higher likelihood of being complied with. For this, the robot may be able to model the teaching strategy in order to form the most *appropriate* informative queries. The issue of generating smart questions is further explored in Experiment 6 where multiple types of questions are available to the robot.

6.4 Summary

This chapter presents Intermittently-Active Learning (I-AL), in which the learner makes queries selectively, in contrast to the conventional Active Learning paradigm that involves making queries unconditionally at every step. Two approaches for I-AL are presented.

- In *teacher-triggered queries* the control of question timing is given to the human teacher.
- In *conditional queries* the learner decides to make a query only when certain conditions are met. These conditions are based on a comparison of the expected utility of making a query, versus allowing the human teacher to pick the next example.

Both approaches were evaluated in a controlled experiment, comparing them with passive supervised learning and fully-active learning. The experiment demonstrates that both I-AL approaches achieve the benefits of fully active learning, while addressing the balance-of-control issue that was raised Chapter 5. The experiment also reveals the importance of the intelligibility of questions in ensuring compliance.

CHAPTER VII

EMBODIED QUERY TYPES

Experiments on embodied queries presented in this thesis, have so far focused on task learning in which the robot is taught a general representation of a task goal through demonstrations. By representing the task goal as a concept, this learning problem can directly employ existing AL techniques to generate questions.

Standard AL methods involve questions that consist of asking for a label for an instance. In the case of an embodied robot, the learner needs to create the state for which it has a question about. Experiments 4 and 5 involved an example where the robot used natural language and gestures to get the teacher to configure the state that it needed. In different task domains, it might be possible for the robot to manipulate the environment to create a state for which it has a question. As pointed out in Chapter 5, such examples of embodied AL differ from conventional AL in the way that query samples are instantiated and presented to the teacher. However, the information gathered from the question is the same — a label for the sample state.

Interactions with humans afford the transfer of a range of information other than labels. This potential has been explored in more depth within the Machine Learning community, with novel active learning protocols mentioned earlier in Sec. 2.3. One example is *feature relevance queries*, which ask whether a feature is relevant for a particular concept that is being learned. These could easily be used in the task learning domain from our earlier experiments. This would allow the learner to ask questions like “Does the color of the top part matter for HOUSE?”, and directly gather an important piece of information that otherwise requires multiple labelled examples to be shown to the learner. A guided interaction that involves these alternative types of queries can therefore be much more effective.

There are several considerations in extending embodied AL to novel types of queries. The learner needs a way to compare the different types of information obtained from the different queries, in order to choose the most informative question. Thus a common framework in which all types of queries can be generated is essential. Secondly, certain types of questions might be preferable from the humans perspective, for being easier to understand or require less mental or physical effort to answer.

This chapter establishes new query types for skill learning problems that involve continuous action spaces. These query types are evaluated within a human robot interaction setting. Next, the extent to which humans use these query types is investigated in an experiment on human question asking.

7.1 Query Types for Skill Learning

Active Learning has been applied to robot skill learning problems [40, 61], however these had an important limitation. They involve discrete action spaces where the learner requests an action suggestion for a state where its prediction confidence is low. These are referred to as *state queries*. This is essentially the same as requesting a label for an instance, and thus allows for using existing query selection techniques.

When the action is continuous, it becomes infeasible for the human to provide a response to a query asking what action to do in a given state. Continuous actions form patterns that are meaningful to humans only when they are chained. For example, one way to represent manipulator skills is using the six-dimensional configuration (position and orientation) of the end-effector as the state, and the change in this state (*i.e.* the velocity) as the action. In this setting, a query would require the teacher to provide a very small position and angular displacement at a given arm configuration, and the provided displacement would need to be in accordance with an overall motion that achieves the goal of the skill that is being taught. This is clearly not a feasible interaction.

This does not mean that robots cannot ask questions to gain useful information when

they are learning skills with continuous actions. Three types of questions that are at a scale that is meaningful to humans are identified in the following. These are inspired by recently proposed methods in the AL literature, as alternatives to the conventional AL queries.

7.1.1 Label queries

Robot skill learning involves modeling/encoding a skill from a set of demonstrations, such that it can be reproduced correctly. The demonstrations provided by the teacher are inherently labelled as *positive*. One way the robot could make queries is to execute a motion and ask whether the skill was performed correctly. We refer to this as a *label query*.

The information provided by label queries depends on the answer. If the response is positive, then the motion can be used as another demonstration. What to do with negative examples is not as straightforward. LfD methods are designed to encode a skill from positive examples only, since it is unnatural to demonstrate “what not to do”. One way to make use of negative examples that arise from label queries, is to update the learned model such that the probability of the negative data being generated by the model is minimized while the probability of the positive data being generated is maximized. The main issue with this idea is the attribution of negativity to the whole trajectory, while only parts of the trajectory might be responsible for the failure. A second approach for making use of negative examples is to guide the learner’s future queries towards positive examples (*e.g.* [60]).

Methods for generating label queries depend on the particular framework used for representing the skill. However a general approach applicable to most frameworks is to sample trajectories from the learned skill and evaluate them with a certain criterion. For instance, the robot can choose to query the trajectory that it is least certain about or the trajectory that is most likely to increase the applicability of the skill.

7.1.2 Demonstration queries

The second type of query, called *demonstration* or *demo queries*, involves creating a new scenario and requesting a demonstration from the teacher. Demo queries give less control

to the learner over what information is acquired from a query, as compared to label queries. In label queries, the learner specifies the whole trajectory and the teacher only provides a label. In demonstration queries, the learner only specifies certain constraints, while the trajectory is still produced by the teacher. Nevertheless, this gives some control to the learner such that useful demonstrations can be acquired. Demo queries are analogous to a method known as *active class selection* [86], which consist of requesting an example from a certain class.

One way to constrain trajectories provided by the teacher is to specify the starting state. Trajectories are often represented with a sequence of end effector configurations relative to a goal object. Thus the robot can configure its end effector in a certain way relative to the goal and request the demonstration to start in this configuration. A larger range of queries can be made by manipulating the target object. A different way of constraining the demonstrations provided by the teacher is to allow the teacher to control only a subset of the robot's joints while the robot executes a certain trajectory on the rest of the joints, as in [31].

7.1.3 Feature queries

The third type of query is inspired from a relatively recent technique in AL that involves asking whether a feature is important or relevant for the target concept that is being learned [110, 51]. A particularly successful application of this method is document or email classification. Learning approaches to this problem involve the teacher reading documents one by one and providing category labels for them. Even in an AL setting, providing a label for the queried documents is cumbersome and time consuming. On the other hand a feature query directly asks whether a word is a strong indicator of a certain category. This allows the learner to directly modify its model for categorizing new documents and drastically reduces the time spent by the teacher.

Note that the crucial element in the success of feature queries in the given example is

that the features (words) with which the instances (documents) are represented are meaningful for humans and the way they contribute to the classification of the instance (being a strong indicator) is intuitive. Robot skill learning is at a unique position for taking advantage of this method: while features might not be as human friendly (*i.e.* feature names might be too technical and feature values might be arbitrary numbers) the robot’s embodiment can be used to refer to them by instantiating different values of the feature.

Methods for choosing feature queries are also dependent on the framework used for representing and learning skills. One framework that allows feature queries to be directly integrated is the *task space selection* problem [97, 68]. This involves representing demonstrations in high dimensional spaces that involve features that might or might not be relevant (*e.g.* relative position and orientation of different points on the robot to different objects in the environment). Methods try to identify a subspace or assign weights to each feature such that the skill is best represented. In this context a feature query is to directly ask whether a feature is important for a skill. These queries can also be used for directly manipulating the skill representation or for guiding other queries.

7.2 *Experiment 6: Skill learning with different query types*

Chapter 5 highlighted the issues encountered when queries are used within a human-robot interaction. In particular, it was pointed out that compliance with the robot’s questions is not guaranteed. Human teachers sometimes ignore the robot’s questions, particularly when they judge it to be a *bad* question. Thus, it is important to operationalize what it means for a robot learner to ask *good* questions. In particular, when the robot has multiple query types that all provide different types of information, it might be helpful to know the utility of each question from the human’s perspective. Questions that are perceived as *good* are more likely to be answered, and thus have higher utility.

This experiment evaluates, the use of the three types of queries described in Sec. 7.1

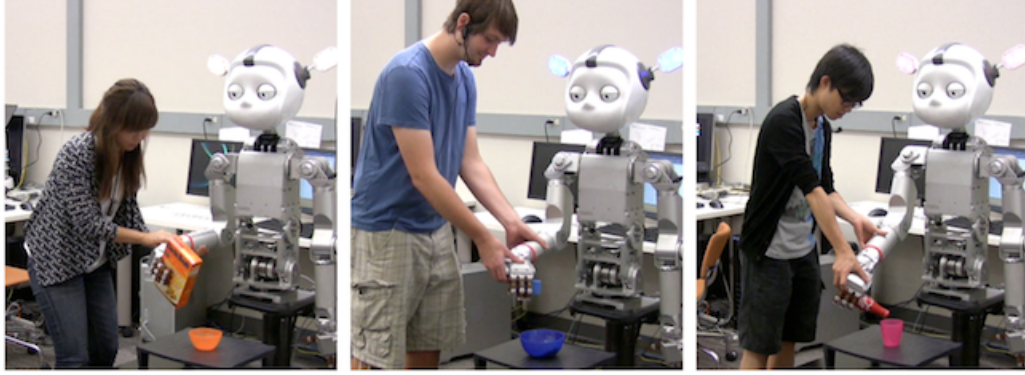


Figure 24: Participants demonstrating three different skills to Simon in Experiment 6.

within a human-robot interaction experiment that involves teaching skills to a robot and answering its questions. The query types are evaluated both subjectively by participants, and objectively in terms of how long they take to answer or how easy they are to answer¹. The impact of the queries on learning is evaluated later in Chapter 8 with an implementation in the Keyframe-based LfD framework.

7.2.1 Experimental Design

This experiment involves learning skills from trajectory demonstrations. The taught skills are from the single-handed goal oriented skills domain (Fig. 5 (e-g)): (i) pouring cereal into a bowl, (ii) adding salt to the salad and (iii) pouring soda into a cup. The skills are demonstrated kinesthetically on the right arm, while the robot holds the respective objects in the right hand. The location of the target is kept constant. Snapshot from the experiment while participants demonstrate the different skills is given in Fig. 24.

Each skill is paired with one query type and the pairings are not varied, since the three skills are very similar in nature. The particular queries made by the robot are pre-scripted, to avoid the large variance that would occur in queries automatically generated based on the teacher’s demonstrations. This is due to the stochasticity inherent to query methods as well as the unpredictable variance in the provided demonstrations.

¹This experiment appears in [24]. Query types were conceptually proposed in [26]

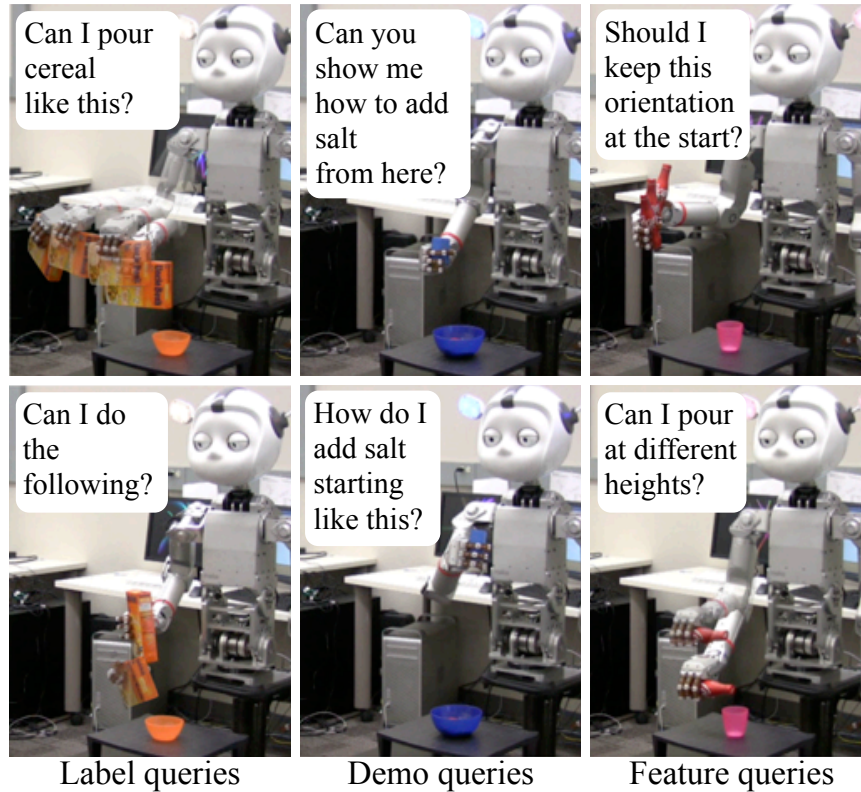


Figure 25: The queries made by the robot for each type of query in Experiment 6.

The skill-query type pairing and the two particular queries made for each type of query are shown in Fig. 25. The first label query involves a trajectory for pouring cereal that is likely to be viewed as a negative example since the cereal box is tipped too early. The second label query involves a valid cereal pouring trajectory from a direction that is less likely to be demonstrated by the participants (because of the difficulty in posing the joints to make the cereal box properly oriented). Demo queries involve one starting position that is slightly outside the range of expected common starting points closer towards the robot, and another starting position that is high above the bowl (which is unlikely to be a starting point in the humans' demonstrations). Feature queries involve one query about the invariance of the starting orientation of the coke bottle, and another asking about the height of the bottle at the pouring location. The communication of both feature queries is facilitated by creating multiple instantiations of the referred features, while uttering the

question. Participants teach all three skills and the order of the skills is counter-balanced.

7.2.1.1 Procedure

Each participant first watches an instructional video that describes the study. The video introduces the speech commands and goes over an example interaction with a skill that is different from the ones taught in the experiment. After the video participants are equipped with the microphone and referred to the whiteboard next to Simon for speech command reminders. They are also asked to move Simon's arm around to get a sense of how the arm behaves.

The experiment proceeds with one session for each skill. At the beginning of each session the experimenter places the related object in Simon's hand. Then the participant provides two demonstrations of the skill, using the commands "New demonstration" and "End of demonstration" to denote the start and end of the movement. Simon uses speech and head nods to indicate that the person's commands are heard. Additionally, Simon looks at his hand during a demonstration, and gazes back to the teacher at the end.

The participant is instructed to step away from the robot after the two demonstrations and use the command "Do you have any questions?" to trigger Simon's query. Participants are told there is no restricted vocabulary or grammar for answering and they can say anything in response to Simon's questions. Participants are not instructed about demo queries in advance, however they are prompted by the experimenter if they do not know what to do in response to demo queries. The experimenter advances the interaction with a keyboard stroke once the participant answers Simon's question. This is repeated twice for each skill, thus each participant sees both queries for the query type. Then they rate the questions in a pen-and-paper questionnaire.

7.2.1.2 Evaluation

Participants rate Simon's questions on two 7-point Likert scale questions addressing *informativeness* of the question for the robot and *ease of answering* for the teacher. Each

question has a comment box for explaining the particular ratings. Due to the sequential nature of the task these questions do not provide a reliable comparison of the three query types, however they allow the participant to elaborate about the evaluation criteria. For a more informed comparison of the query types, we ask three forced-ranking questions addressing *smartness* of the question, *informativeness* for the robot and *ease of answering* for the teacher at the end of the experiment. These questions are:

- *Informativeness*: How informative do you think the question was for Simon?
- *Unexpectedness*: How unexpected was the question? (An expected question is one that asks for information that the teacher would have provided if they were to give more demonstrations)
- *Ease of answering*: How easy or difficult is it for the teacher to answer the question?

In addition to the subjective evaluation by participants, the variance in the provided demonstrations in comparison to the information acquired through queries are analyzed. Also, the time taken to pose and respond to queries are compared across query types. The open-ended of answers provided by the participants in response to Simon's queries are also examined.

7.2.2 Findings

This experiment was completed by 18 participants (4 female, 14 male). Findings are summarized in the following.

7.2.2.1 Subjective evaluation

Results from the forced ranking questions comparing the three query types are given in Table 22. A significant ranking is observed in terms of *smartness* of the questions asked by Simon. 72% of participants thought that feature queries were the smartest, while 61% thought that demonstration queries were the least smart questions. Reasons stated by participants

Table 22: Forced ranking results in Experiment 6 presented in terms of the number of participants out of 18 who voted the query type as best or worst.

	Smartness		Informativeness		Ease	
	<i>best</i>	<i>worst</i>	<i>best</i>	<i>worst</i>	<i>best</i>	<i>worst</i>
Label	3	6	4	6	10	4
Demo	2	11	8	8	3	7
Feature	13	1	6	4	5	7
Friedman test	$\chi^2=13,$ p=0.001		$\chi^2=0.12,$ p=0.94		$\chi^2=3.11,$ p=0.21	

for choosing feature queries as the smartest include: being “more abstract”, demonstrating that “Simon understands task constraints at a high level, rather than just trajectories” and not involving “repeating the whole process of pouring coke.”

Although the query types were not found to be significantly different in terms of *ease of answering*, 56% of participants thought label queries were the easiest to answer. A number of participants noted the fact that Simon’s questions were yes/no questions and thus easy to answer (7 mentioned for label queries, 4 for feature queries). Two participants explained why they thought feature queries were not as easy to answer as label queries: “The first [label] one was a yes-no (obvious), the coke [feature] question actually made me think” and “I could not answer the second question [pouring coke at different heights] with yes/no, it depended.” A similar remark was made about label queries: “close to pouring correctly but not perfect – hard to answer with a yes or no.” Other participants answered the robot with a plain yes or no, but explained the reasons for their answer (“because the cereal would spill”) or provided additional information (“would only work for a full box of cereal”) in the survey, mentioning that they did not trust that Simon would understand these.

We do not see any consistency in the ratings in terms of informativeness of the questions for the robot. This seems to be a result of different interpretations in informativeness, which

Table 23: Average time (in seconds) taken for the robot to make a query and for the participants to answer the query for the three types of queries in Experiment 6.

	Query	Answer
Label queries	14.11 (SD=2.88)	3.31 (SD=2.96)
Demo queries	7.36 (SD=1.08)	24.07 (SD=5.38)
Feature queries	9.56 (SD=0.49)	7.17 (SD=11.99)

is revealed by participant comments: (label queries) “good question, in that the timing of when the box is tipped is important”, (demo queries) “the salt question gives more freedom for the robot to choose new actions”, (feature queries) “more general information, than a 3rd or 4th demonstration.”

7.2.2.2 Task measures

The problem of choosing between different types of queries in AL is often addressed by defining a *utility* for each query. This reflects the payoff between the benefits of the query (*e.g.* informativeness) and the costs of the query. The most common measure of cost used in the literature is wall-clock time. For queries in the LfD setting, this will depend on how fast the robot can move and how fast it speaks. Although the values are specific to the implementation and choice of parameters used in the experiment, a comparison of the time taken to make each type of query, and the time taken for the teacher to complete their answers is given in Table 23.

The ordering of the three types in terms time taken to make the query is as expected: label queries take the most time as they require a complete execution of the skill, feature queries take less time (determined by the longest of the verbal question or the action performed for physical grounding), and demo queries take the least (determined by the longest of the verbal questions and the movement to reach the desired query start pose). For the answer, demo queries take much longer than the other two types of queries, as they require a whole demonstration. Feature queries have a longer response time than label queries, even

though both are yes/no questions and a very large variance for feature queries. This was partly due to a subset of the participants having trouble understanding Simon's question and the interruption by the experimenter to repeat the question. This happened with four participants on the first feature query and one time on both feature queries. Although part of the reason was that the participants in all interruption cases were a not native English speakers, this result highlights the fact that feature queries are harder to interpret and might require repetitions that adds to their cost.

Since the experiment did not involve any particular learning method, it is not possible to make a direct comparison of the informativeness of each type of query. The following discussion tries to illustrate how each type of query is informative for the learned skill. Fig. 26(a) provides a comparison of all demonstrations made by participants with the label queries made by Simon. The second label query (Fig. 25), which was chosen to be a valid way of pouring, is distinctly isolated from all the demonstrations provided by participants. The reason that the queried trajectory was not covered by teacher demonstrations, could have been the difficulty of kinesthetic manipulation of the joints around this region. This highlights one benefit of label queries: certain ways of performing a skill might be valid even though they are unlikely to be demonstrated. Label queries can uncover such possibilities. Fig. 26(b) shows how the robot's second demo query (Fig. 25) is different from the starting points of the demonstrations provided by humans. These illustrate how the different types of queries can quickly expand the applicability of a skill to unexplored regions.

Overall, results from the experiment suggest that all query types are acceptable from a human-interaction perspective. Feature queries are the best option in terms of asking questions that are seen as the smartest, however answering these questions might not be trivial and might potentially bring ambiguous or inconsistent information. Label queries are preferable when the objective is to make it easy for the human to answer the question.

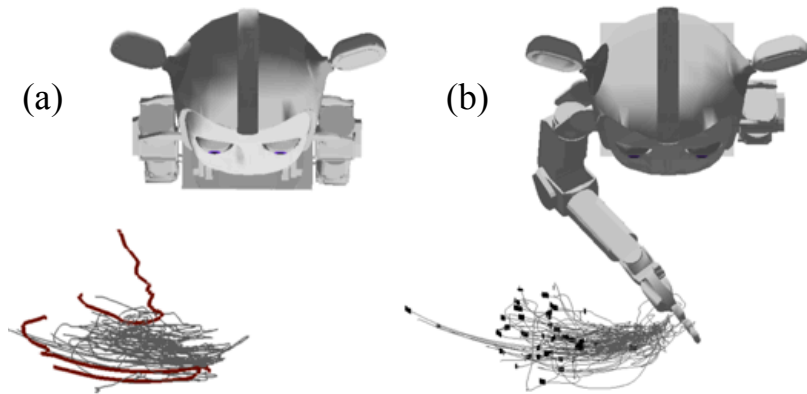


Figure 26: The end-effector trajectories provided by all participants in Experiment 6 (thin lines). (a) (Pour-cereal) Superimposed with the label queries made by the robot (thick lines). (b) (Add-salt) Simon’s pose shows the second demo query starting point. Starts of the demonstrations are shown as dots.

7.3 Experiment 7: Human question asking

In designing embodied queries for a robot learner, the Active Learning literature gives one perspective. Another source of inspiration in this thesis is human question asking, given that humans questions are inevitably embodied. The intuition is that questions people typically use will be easy for them to answer. While human question asking has been investigated and modeled in different cognitive tasks such as text understanding [73, 104], there are no existing studies on human questions in the context of skill or task learning by demonstration. This experiment investigates how humans ask questions in scenarios similar to the ones robots are faced with. It involves providing the participants with demonstrations of a task, letting them ask questions about it and then asking them to perform it in a new situation².

7.3.1 Experimental Design

7.3.1.1 Domain

Four tasks (described in Fig. 27) are considered. The first two tasks are focused on learning a task goal (desired end state) whereas the latter two emphasize learning skills to perform

²This experiment appears in [24].

TASK 1 (Sandwich): Construct four layers. Bottom is a large green circle, second layer is a large purple circle, third layer has 3 or more non-overlapping small circles placed within the large circle, and top layer is a large green circle. Large circles are aligned.

TASK 2 (Laundry): Place all blue objects on the table into the large yellow square bin.

TASK 3 (Salt): Grasp the yellow block from the table, make it horizontal with its top above the blue bin. Move it down a few inches and back up. Number of lowering movements and length of the movement depends on the size of the bin. Place the block on the table vertically.

TASK 4 (Scoop&pour): Grasp the green block from the table. Dip it into the yellow bin and rotate it clockwise around 90°. Then move it to the blue bowl maintaining its orientation. Dip it and turn it counter-clockwise around 90°. Place the block back on the table.



Figure 27: Description of the four tasks used in Experiment 7 and snapshots from the two demonstration videos shown to the participants.

the task (desired actions or movements to accomplish the task). Each of these are designed to be an abstraction of a real-world skill, to allow the teacher to answer unexpected questions based on the real-world counterpart. The tasks or skills involve a fixed set of objects.

7.3.1.2 Procedure

Two demonstrations, varied in certain aspects, were recorded for each skill. The recordings were used for demonstrating the skill to the participants. This is done to mitigate unintended variances between the demonstrations given to different participants. Participants

learn all four tasks and the order is counterbalanced across participants. The interaction between the participant and the experimenter is recorded by a camera that oversees the table (Fig. 31).

Throughout the experiment, the participant sits in front of a table with a computer screen to their right for displaying the demonstration videos. For each task, the experimenter first brings the involved objects onto the table. Then the participant watches the video. Participants are told that they can watch the video as many times as they want, at any time during the experiment. This is to reflect the fact that a robot has access to the demonstrations provided by a teacher before they ask questions and forgetting what happened in the demonstration is not an issue. Thus questions that address information present in the videos are avoided. After watching the video at least once, the participant is probed for questions. There is no upper limit on number of questions, but a lower limit of three questions.

When the participant has no additional questions, the experimenter configures the environment in a certain way and asks the participant to perform the task. The participant is reminded that they can keep asking questions, during and after their execution. The execution is repeated twice with different starting configurations.

In the instructions, participants are encouraged to ask any type of question that they can think of and that they are not restricted to yes/no questions. They are encouraged to refer to the objects on the table in their questions. Participants are told that the purpose of the study is to take inspiration from their questions to allow robots to ask questions. Thus they are told not to ask questions about the underlying purpose of the tasks since explaining these to a robot would be impossible. Such questions are not answered by the experimenter if they occur.

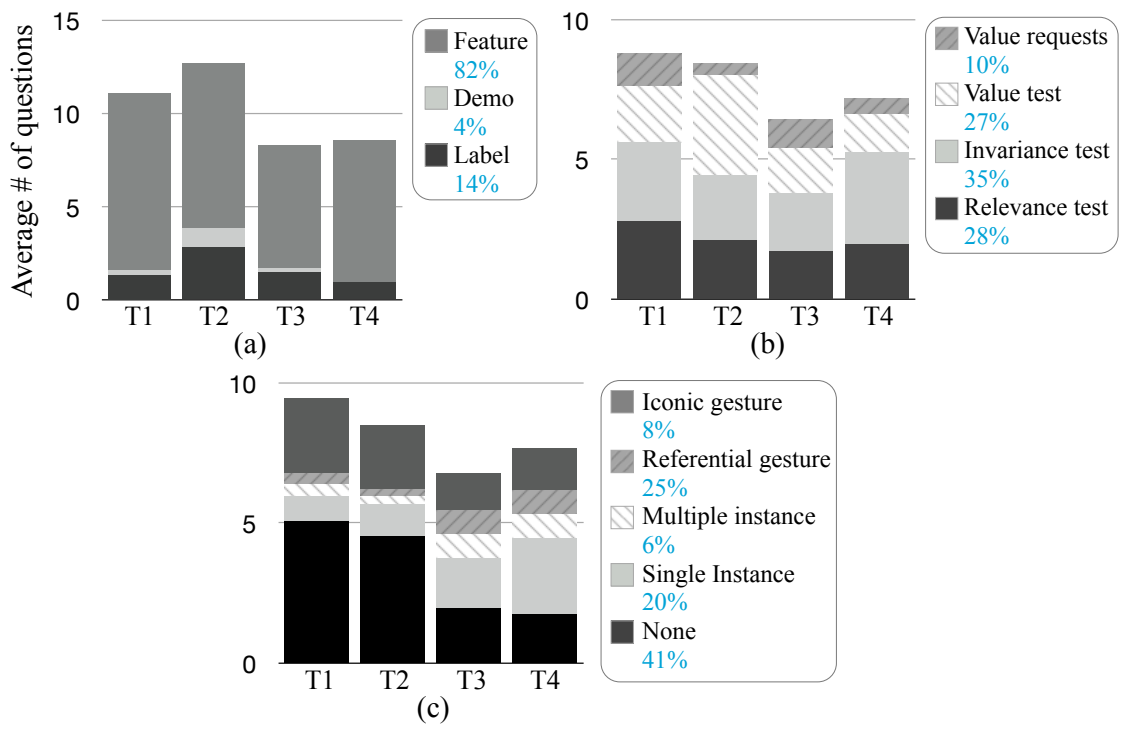


Figure 28: Distributions of human questions into categories in Experiment 7. (a) Distribution of questions into three types of queries for each task in the first experiment averaged across participants. (b) Distribution of feature queries into sub-categories. (c) Average distribution of common actions and gestures that physically ground questions across tasks.

7.3.1.3 Evaluation

All recordings were later transcribed by the experimenter using ELAN³. This includes annotations of the participant's questions, the answers given by the experimenter and the participant's interaction with the objects and gestures. In addition the demonstration views by the participant and the task executions were annotated.

The participants' questions are categorized into the three query types described in Sec. 7.1.1, Sec. 7.1.2 and Sec. 7.1.3 based on the following criteria. If the question asks the experimenter to evaluate an action or a complete task execution it is considered a *label query*. If the question requests an action, an action suggestion or a complete task execution from the experimenter, it is categorized as a *demonstration query*. Lastly, all questions that address a certain aspect of the task in general terms is considered a *feature query*.

Questions were also categorized in terms of their form based on the taxonomy established in [73]. This taxonomy is shown in Fig. 30 with examples from our experiment for each form of question.

7.3.2 Findings

12 participants (3 female, 9 male) completed this experiment. This resulted in 4h53m of interaction, involving 487 total questions asked by the participants, with an average of 40.58 (SD=13.09) questions per participant in about 25 minutes.

7.3.2.1 Query types

The distribution of questions asked by participants across the three query type categories is given in Fig. 28(a). A significant majority of questions asked by humans (82%) are feature queries. A large variety of query types within feature queries is observed, in terms of the information they attempt to gather from the teacher. The following subcategories of feature queries are established based on the data (a similar distinction was made in [55]).

³www.lat-mpi.eu/tools/elan/

- *Feature relevance tests* directly ask whether a feature is important for a task: *Does f matter?* (Fig. 31(b))
- *Feature invariance tests* ask whether a feature has to have a certain value: *Does f have to be f_1 ?* (Fig. 31(d))
- *Feature value tests* ask whether a feature is allowed to have a certain value: *Can f be f_1 ?* (Fig. 31(a))
- *Feature value requests* directly asks the value range that a feature is allowed to have: *What can f be?* (Fig. 30 (SIMPLE))

Note that f can refer to an attribute of the involved objects, relations between objects, a property of the involved movements, a parameter of the task description or the dependency between any two of these. The distribution of feature queries into these subcategories is shown in Fig. 28(b). A rather uniform distribution across the first three categories is observed, however the open-ended value requests are the least common.

In addition there is a distinction between *full* and *partial* demonstration and label queries. A full demonstration query requests a complete demonstration while a partial one requests a single step of the demonstration. Similarly, a full label query asks for an evaluation of a complete task execution while a partial label query asks whether a step was correct. Such partial label queries seem to be a useful mechanism for avoiding the credit assignment problem that arise when a negative label is encountered (Sec. 7.1.3). An example is shown in Fig. 29.

7.3.2.2 Question forms

The average usage percentage of each question form across tasks is shown in Fig. 30. There is a clear preference for yes/no questions (81%), more often posed in the simple form (60%). The choice between the three types of yes/no questions, seem to reflect whether



Figure 29: An example partial-label query from Experiment 7 where the participant elicits a label after each step of the skill execution.

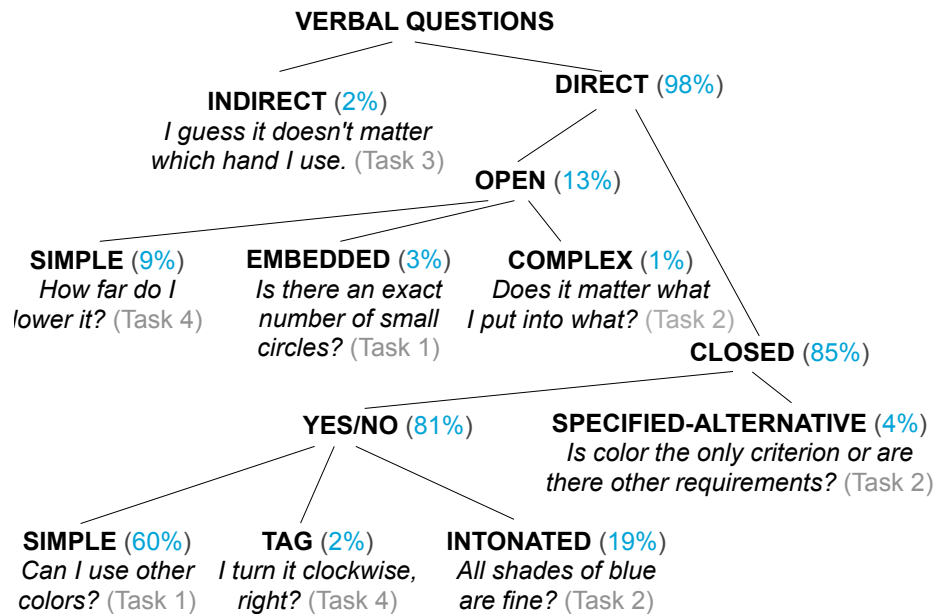


Figure 30: Question forms [73] and example instances from Experiment 7 (percentage of average occurrence shown in parentheses).

the person has an expectation about what the answer is, and indicates their confidence in this expectation. While *simple* yes/no questions are neutral about either answer, *tag* yes/no questions (e.g. *right?*, *correct?*) indicate strong expectation of the answer being “yes”. One example is the participant saying “I can use either hand, right?” when he has already started a movement with his alternate hand.

7.3.2.3 Embodiment of questions

Human questions during task learning are largely embodied and physically grounded. On average 62% of the questions asked by a participant involved interacting with the environment. Label queries are inherently embodied questions since they require the robot to perform the actions to be labelled. Similarly demo queries require the person to create the scenario in which a demonstration is requested. It is, however, possible for the person to verbally describe a sequence of actions (or a hypothetical scenario) and request a label (or a verbal demonstration).

Feature queries on the other hand are more amenable to purely verbal question asking, however they require the ability to talk about features and feature values. More specifically, they require tokens that have a common physical grounding for the two parties (person asking the question and person being asked) to replace f and f_1 in the questions exemplified earlier. Even though such tokens might exist, they are often replaced or accompanied by actions or gestures that communicate them. Four common types of embodiment usage were observed.

- A *single value instantiation* refers to actions that generate a certain feature value while asking a question about it. For instance in Fig. 31(a) the person asks if the yellow block can be lowered without being tilted while providing an instance of non-tilted orientations.
- *Multiple value instantiations* refer to actions that generate several different values of feature during a feature query. An example is the participant showing several

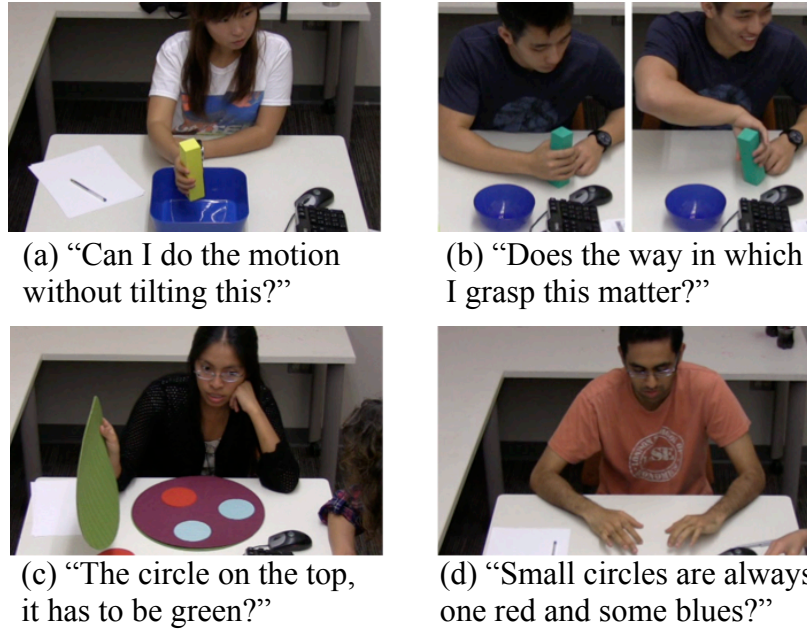


Figure 31: Examples from Experiment 7 of the four common ways in which humans use their embodiments to support the communication of their questions.

different grasps while asking whether the way in which the block is grasped matters (Fig. 31(b)).

- *Referential gestures* include pointing, touching or holding up a certain object in the environment while making a feature query that relates to a feature of the object. For instance, in Fig. 31(c) the participant holds up a green large circle while asking whether the top layer has to be green.
- *Iconic gestures* involve arm and hand movements that communicate a certain feature or feature value referenced in a question. An example, shown in Fig. 31(d), is placing two hands at a certain distance from each other to indicate a certain size.

The average distribution of these actions and gestures across participants is given in Fig. 28(c). 59% of feature queries made by participants involve some form of embodiment. Single value instantiations and referential gestures are more common than the other two types. Also interesting differences are observed in how they are used in different tasks,

which is discussed in the following.

7.3.2.4 *Learning Goals versus Skills*

There are no major differences in the distribution of query types or question forms between tasks that require learning goals (Tasks 1 and 2) versus tasks that involve learning skills (Task 3 and 4). The main difference in the questions asked during goal versus skill learning is seen in the physical grounding of the questions (Fig. 28(c)). Questions asked during skill learning are more likely to be physically grounded (74%) than during goal learning (46%). This is not surprising as movements are harder to describe verbally. Secondly, value instantiations are more common in skill learning (41%) as compared to goal learning (15%). Correspondingly, a larger portion of the questions during goal learning use referential and symbolic gestures. This is also an intuitive finding as the features involved in skill learning are about the movements, while the features involved in goal learning are about objects in the world. In a sense, referring to features of an object in the world, or using a gesture to indicate a feature value are two alternative ways to indicate a feature value.

Overall, this experiment contributes experimental findings on human question asking, that have important implications and insights for developing embodied queries. The experiment suggests that human questions are primarily *feature* queries which are focused on a certain feature in a temporally and spatially local portion or step of the skill or the task. It reveals a wide range of feature queries that would all have different implementations on a robot and would provide different types of information. It also gives inspiration for novel types of queries such as the *partial* label queries. These would involve the robot performing smaller portions of a skill one at a time and asking whether the performance is correct so far after each segment.

In terms of the question form, closed form questions are desirable as they are common in human learning, and they facilitate the interpretation of the response by limiting possible answers. In addition, question forms can be used as a transparency mechanism that reveals

the robot’s expectation and confidence about the anticipated answer.

Finally the experiment reveals some details about the use of embodiment in human question asking providing inspiration for embodied queries. In particular skills are often represented with features that are hard to verbalize, but easy to instantiate (*e.g.* positions and rotations). This idea was exploited in Experiment 6, where Simon was able to ask about the importance of “maintaining the end-effector orientation about the x-axis at about 90 degrees” without needing to explain these terms. This potential should be exploited whenever possible.

7.4 Discussion

In designing interactions for active robot learners, the first question to be addressed is the choice of query types. Results from the two experiments presented in this chapter suggest a priority on feature queries. These were the most common in human learning (82%) and were thought to be the smartest questions when used by the robot. On the other hand, these are the most challenging queries to produce automatically in a way that is interpretable by humans. Label queries are good alternatives that are easy to produce and are interpretable. They also have the advantage of very fast response and low effort requirements from the teacher. However, negative examples are likely to occur and are less informative in learning the skill. In addition, these negative examples can have undesirable consequences (*e.g.*, spilling cereal). One idea, inspired from the common human usage of label queries, is to do *partial* label queries. This would involve the robot performing smaller portions of a skill one at a time and asking whether the performance is correct so far after each segment. This idea is implemented in the next chapter.

In terms of the question form, closed form questions are desirable as they are common in human learning, and they facilitate the interpretation of the response by limiting possible answers. In addition, the fact that label and feature queries were yes/no questions was pointed out by participants as a positive property. However limiting the answers to yes/no

could result in inaccurate information or missed opportunities for additional information. Thus, it is worth trying to build in the capability to interpret answers like “sometimes” or “possibly”. In addition, question forms can be used as a transparency mechanism that reveals the robot’s expectation and confidence about the anticipated answer.

Another important consideration in designing question asking behaviors is the use of embodiment in the question. In particular skills are often represented with features that are hard to verbalize, but easy to instantiate (*e.g.* positions and rotations). For instance in Experiment 6, Simon was able to ask about the importance of “maintaining the end-effector orientation about the x-axis at about 90 degrees” without needing to explain these terms. This potential should be exploited whenever possible.

7.5 *Summary*

This chapter identifies three distinct types of embodied queries that can be used in skill learning.

- *Label queries* executes a varied version of the learned skill and asks whether the execution is valid or successful.
- *Demonstration queries* request a demonstration under some constraint, such as a specified start or goal configuration.
- *Feature queries* directly ask about relevance, invariance or values of a feature.

The query types were compared in a human robot interaction experiment with pre-scripted questions. The experiment shows that feature queries are seen as the smartest, but label queries are found to be easier to answer. In addition demonstration queries take considerably longer than the other two queries to get a response. A second experiment characterized the use of these queries in human learning, analyzing their use of embodiment while asking questions and the form of their questions. The experiment reveals that feature queries are the most common in humans and reveals several subtypes for the three types of queries.

CHAPTER VIII

ACTIVE KEYFRAME-BASED LFD

The previous chapter identified three types of embodied queries suitable for skill learning: label, demonstration and feature queries. The two experiments revealed the utility of these different types of queries from a human’s perspective, *e.g.* which query types are easier to understand, require less effort to respond to or result in more natural interactions. While these can guide the selection of query types, what is most important in comparing alternative candidate questions is their informativeness to the robot.

The queries used in the experiment that evaluated different question types were pre-scripted. Chapter 7 suggested possible implementations of the different query types. However there are no existing LfD frameworks that allow all of the different types of queries to be produced autonomously. To this end, this thesis contributes Active Keyframe-based LfD (A-KLfD); an extension of KLfD for generating embodied queries of different types.

An important motivation for using KLfD as the common framework for implementing different query types is the way it segments a skill. One observation about human questions during skill learning from the previous chapter, was that they are spatially and temporally local. Humans seem to automatically segment a skill and ask questions about particular segments. Existing representations for skill learning either do not segment the skill at all or have unintuitive segmentations. In KLfD the temporal segmentation of the skill is directly obtained from the human, in the form of keyframes. Questions about keyframes are therefore about segments that are meaningful for the human.

This chapter presents the implementation of three query types in the KLfD framework. It then provides an evaluation of these queries to demonstrate their utility over passive learning from humans and characterize the utility of different types of questions.

8.1 Embodied Queries in Keyframe-based LfD

The overall objective of questions in skill learning is to learn the most generalizable representation possible for a given skill. The learned model should expand to cover as much of the state space as possible, and not more. This will allow the robot to be able to execute the learned skill in more situations: different goal configurations, obstacles, or run-time constraints. Each question type tries to achieve this objective in a different way.

8.1.1 Label queries

Label queries involve exploring beyond what has been demonstrated with the goal of relaxing the skill, *i.e.* increasing its variance such that it has a larger region of applicability. In the KLfD framework, this involves trying to expand the keyframe distributions as much as possible. The first question to tackle is how to sample points from each keyframe distribution when making a label query.

8.1.1.1 Query variance

First consider a single keyframe of the skill model. The point in the query should be far enough from the mean such that, if it ends up being added to the cluster it increases the variance. According to this criterion the query point should be as far as possible from the mean. However as the query is moved away, the probability of it actually belonging to the cluster will decrease. Thus, the point should be selected carefully to take into account both factors. To this end, the utility of a point s in the state space \mathcal{S} as a query candidate for cluster k is defined as follows.

$$U_k(s) = P(s|\mathcal{D}_k)(\text{var}(\mathcal{D}_k \cup \{s\}) - \text{var}(\mathcal{D}_k)) \quad (1)$$

The first term indicates the probability of the point being generated by the same distribution as the observed points in cluster k , *i.e.* \mathcal{D}_k , and the second term indicates the increase

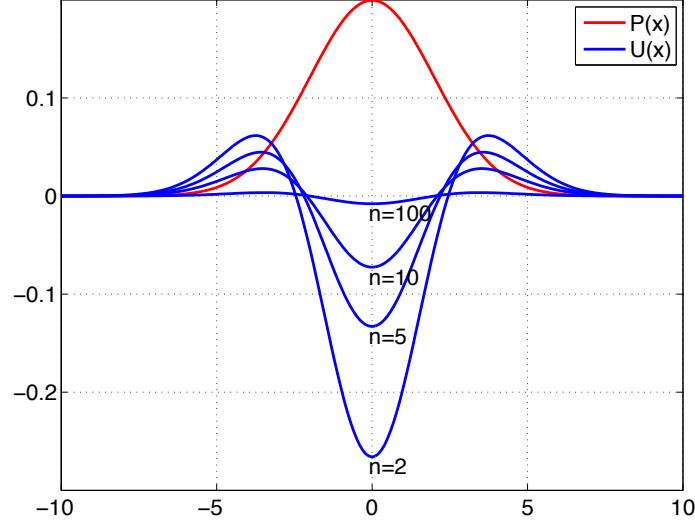


Figure 32: Plot of the utility function $U_k(s)$ for different values of n_k in 1D.

in the variance if the point were to be part of the cluster data. Since each keyframe is modeled with a multimodal normal distribution, the determinant of the covariance is used as a measure of variance; $\text{var}(\mathcal{D}_k) = \det(\Sigma_{\mathcal{D}_k})$ and $\text{var}(\mathcal{D}_k \cup \{s\}) = \det(\Sigma_{\mathcal{D}_k \cup \{s\}})$. Fig. 32 shows the value of $U_k(s)$ for a sample normal distribution in 1D for different values of n . It can be observed that $U_k(s)$ is maximized at a certain distance from the mean of the normal distribution. Next, it is shown that states that are at a certain Mahalanobis distance from a keyframe distribution maximize the defined utility, and the distance only depends on the number of points in the cluster.

Theorem 1 (Optimal query variance) $U_k(s)$ is maximized by states that satisfy the following property, where n_k is the number of points in cluster k , i.e. $n_k = |\mathcal{D}_k|$.

$$M_k(s) = (s - \mu_{\mathcal{D}_k})^T \Sigma_{\mathcal{D}_k}^{-1} (s - \mu_{\mathcal{D}_k}) = 3 + \frac{1}{n_k} \quad (2)$$

Proof. The criterion specified by Equation 2 is obtained analytically, by maximizing Equation 1. To this end, $U_k(s)$ is re-written in terms of the current estimations of the means and covariances based on the observed data set, and then differentiated and equated to zero. As a first step, the predicted future covariance $\Sigma_{\mathcal{D}_k \cup \{s\}}$ is written in terms of the current mean $\mu_{\mathcal{D}_k}$ and covariance $\Sigma_{\mathcal{D}_k}$ and the state s to be queried.

$$\Sigma_{\mathcal{D}_k \cup s} = \frac{n_k}{n_k + 1} \left(\Sigma_{\mathcal{D}_k} + \frac{1}{n_k + 1} (s - \mu_{\mathcal{D}_k})(s - \mu_{\mathcal{D}_k})^T \right) \quad (3)$$

For ease of calculation, we introduce the parameter $\bar{s} = s - \mu_{\mathcal{D}_k}$, rewrite $U_k(s)$ in terms of \bar{s} and maximize it with respect to \bar{s} , which is equivalent to maximizing with respect to s . Next the Matrix Determinant Lemma is used to rewrite the determinant of the predicted covariance, $\det(\Sigma_{\mathcal{D}_k \cup \{x\}})$. The lemma is as follows.

$$\det(A - uv^T) = (1 + v^T A^{-1} u) \det(A) \quad (4)$$

This allows us writing the second term of $U_k(s)$ (*i.e.* the difference of the two determinants), in terms of the known determinant of the current covariance matrix.

$$\det(\Sigma_{\mathcal{D}_k \cup \{x\}}) - \det(\Sigma_{\mathcal{D}_k}) = \left(\frac{n_k}{(n_k + 1)^2} \bar{s}^T \Sigma_{\mathcal{D}_k}^{-1} \bar{s} - \frac{1}{n_k + 1} \right) \det(\Sigma_{\mathcal{D}_k}) \quad (5)$$

This is useful as it takes all instances of s outside the determinant, making differentiation with respect to s much easier. Finally, $P(s|\mathcal{D}_k)$ in $U_k(s)$ is substituted with the Gaussian density function written in terms of \bar{s} .

$$P(s|\mathcal{D}_k) = \frac{1}{\sqrt{\det(\Sigma_{\mathcal{D}_k})}^d \sqrt{2\pi}} e^{-\frac{1}{2} \bar{s}^T \Sigma_{\mathcal{D}_k}^{-1} \bar{s}} \quad (6)$$

where d is the dimensionality. Using the substitution $M = \bar{s}^T \Sigma_{\mathcal{D}_k}^{-1} \bar{s}$ for the Mahalanobis distance, $U_k(s)$ is simplified to the following.

$$U_k(s) = K e^{\frac{1}{2} M} \left(\frac{n_k}{(n_k + 1)^2} M - \frac{1}{n_k + 1} \right) \quad (7)$$

where K constitutes of all multiplicative constants. $U_k(s)$ is maximized by equating its derivative to zero. Based on the chain rule, we can just solve for $\frac{\partial U_k(s)}{\partial M} = 0$, and using the product rule in this equation, the criterion in Equation 2 is obtained, *i.e.* $M_k(s) = 3 + \frac{1}{n_k}$. \square

Theorem 1 says that a query for keyframe k should be at the indicated Mahalanobis distance from the keyframe distribution. Fig. 33 shows how well a 2D normal distribution learned with different query strategies matches the true distribution over time, as new queries are made. Queries made with different constant Mahalanobis distances and uniform sampling of the space are compared. The match between the learned and the true multivariate Gaussian are measured with the Pillai-Bartlett trace, defined as:

$$\Lambda = \sum_{1..p} \frac{1}{1 + \lambda_p} \quad (8)$$

where λ_p are the Eigenvalues of the matrix $\Sigma_{\mathcal{D}_k} \times \Sigma_{true}^{-1}$. Fig. 33 shows how this match changes for the four query strategies, in three scenarios that differ in how well the distribution at the beginning matches the true distribution. It can be observed that optimal Mahalanobis distance determined in Theorem 1 is indeed the best choice, however when the initial match is low, querying with at larger distance at the beginning results in a steeper learning curve.

From this observation, the determined optimal variance of a keyframe query is used with a slight adaptation described in the following. For many goal-oriented skills, keyframes that are further away from the goal are likely to have a larger variance than the keyframes close to the goal. Therefore after very few demonstrations, it is likely that keyframes that are away from the goal will have a low match to the true distribution, and therefore could converge faster if the queries were larger than optimal. Thus, in the current implementation, the Mahalanobis distance for choosing queries is multiplied with a constant c_k varied between 1 and 3. This constant increases as the keyframe mean gets further away from the goal; $c_k = 1 + 2 \frac{|\mu_{\mathcal{D}_k}|}{|\mu_{max}|}$, where μ_{max} captures the limits of the state space imposed by the robot reachability.

There are infinite points that satisfy Theorem 1. In order to choose a single point query for a keyframe, a sampling approach is used. First a set of candidate directions E_k are

sampled uniformly. Each $d \in E_k$ is a unit vector in six dimensional space. Then a candidate query s that has Mahalanobis distance $c_k(3 + \frac{1}{n_k})$ is computed as follows.

$$s = d \sqrt{\frac{c_k(3 + \frac{1}{n_k})}{d \Sigma_{\mathcal{D}_k}^{-1} d^T}} \quad (9)$$

8.1.1.2 Query selection

So far the choice of variance for querying a single keyframe has been discussed. A query corresponding to a particular keyframe is referred to as a *keyframe query*. A label query involves a complete execution of the skill, hence multiple keyframe queries. The complete label query is denoted with $q = (s_1, \dots, s_K)$, where s_k corresponds to the query of a particular keyframe distribution \mathcal{N}_k . Making a keyframe query for all keyframes in the skill increases the chance of making the query q unsuccessful. Thus the number of keyframes that are queried is limited. Keyframes in q that are not being queried, are sampled conservatively from \mathcal{N}_k . Besides selecting q , the robot needs to select a goal configuration $g_i \in \mathcal{G}$ in which the query will be made. Thus the parameters that need to be instantiated to fully specify a label query are:

- the goal configuration g ,
- the subset of keyframes to be queried $\{k_1, k_2, \dots\}$, and
- direction of the keyframe queries (scale is chosen to be optimal).

The optimal Mahalanobis distance for a certain keyframe specifies a group of points that are on some kind of hyper-ellipse. To make a keyframe query, a single point needs to be selected for each keyframe. So the direction of the query needs to be determined. For this, a random sample of directions are used to obtain a set of candidate keyframe queries C_k . Each candidate in C_k has the optimal variance. Randomly selecting the direction was

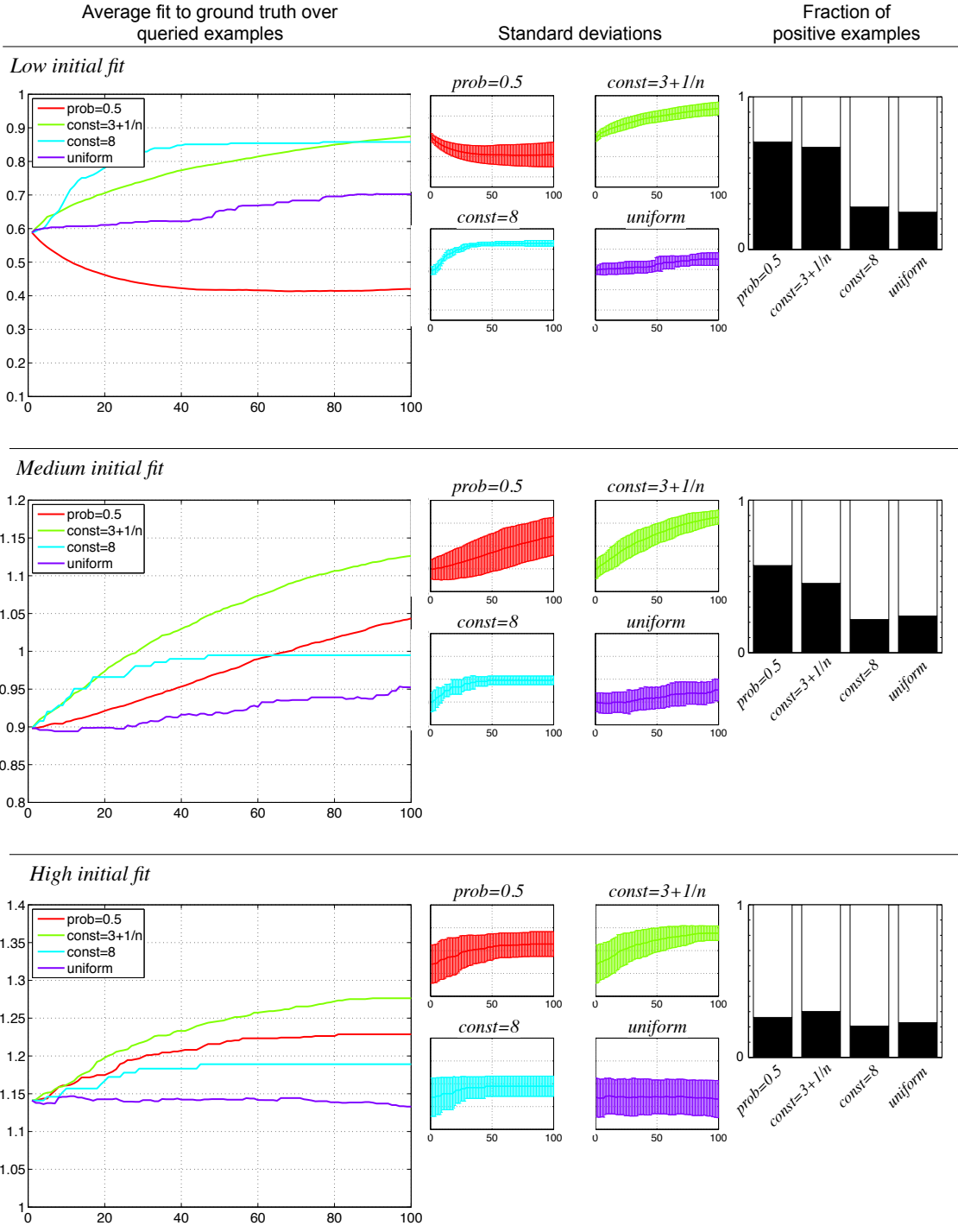


Figure 33: Progress of the fit between normal distributions learned through queries and the true distribution. Three different query variance constants, including the one proven to be optimal, are compared with uniform sampling.

empirically found to be a good strategy, when compared to different strategies that use the principle components of the keyframe distribution.

The utility of a candidate query point $s \in C_k$ for keyframe k , denoted by $V_k(s)$, is computed by iterating over goal configurations. For each possible g_i , the keyframe query is transformed into the robot coordinate frame. Then the configuration is checked for reachability, *i.e.* whether or not there exists an inverse kinematic (IK) solution that allows the end-effector to be in that coordinate frame, and that this configuration does not result in a collision with itself. If these conditions are met, the utility of the keyframe query candidate q_{ki} for the particular goal is measured by its rescaled Mahalanobis distance, otherwise it is zero. The overall utility of the keyframe query candidate, $V_k(s)$, is obtained by summing over all possible goal configurations. As the overall utility of querying a keyframe given a particular goal configuration, denoted by $V(k|g_i)$, the highest utility among the candidates for that keyframe is used, provided that the candidate is reachable.

$$V_k(s) = \sum_{g_i} c_k M_k(s) R(s|g_i) \quad (10)$$

$$V(k|g_i) = \max_{s \in C_k} (V_k(s) R(s|g_i)) \quad (11)$$

where $R(s|g_i)$ is an indicator of reachability; it is one if state s is reachable for the goal configuration g_i and zero otherwise. $V_k(s)$ captures the reachability of the keyframe query for various goal configurations. Having a high utility means that if the keyframe query was labelled positive, it will allow new ways of performing the skill in a variety of goal configurations. Note that $c_k M_k(s)$ has the same value for all candidates for a certain keyframe, thus it does not favor one candidate over another. Candidates of a certain keyframe are only selected based on their reachability for different goal configurations. However this multiplicative factor becomes important when comparing the candidates from different keyframes as explained next.

The robot first selects the goal configuration and then the two keyframes to be queried, as follows.

$$g = \arg \max_{g_i} (V(g_i)) \quad (12)$$

$$V(g_i) = \max_{k_1, k_2} (V(k_1|g_i) + V(k_2|g_i)), \text{ where } k_1, k_2 = 1, \dots, N_k, k_1 \neq k_2 \quad (13)$$

$$q = \arg \max_{k_1, k_2} V(g) \quad (14)$$

$V(g_i)$ indicates the utility making a query in the goal configuration g_i . It is computed by iterating over pairs of keyframes to be queried. For each pair, first it is checked that the two keyframes have non-zero utility and all the other keyframes have a reachable configuration, sampled from N_k , that is allowed by g_i . If these conditions are not satisfied, the pair is rejected (*i.e.* the query in which these two keyframes are varied has zero utility). Given that the conditions are satisfied, the utility of the query is determined by the sum of the utilities of the two keyframe queries (Equation 11). The goal configuration that allows the highest valued query is selected, along with the particular query that results in this highest utility.

The query selection process for label queries is summarized in Algorithm 2.

8.1.1.3 Query execution and response

Once a query $q = (s_1, \dots, s_K)$ has been specified, its execution is identical to the reproduction of the skill as described in Sec. 3.3.1.2. The execution is only different in that some of the points not directly sampled from the corresponding keyframe distribution, but they are queried based on the distribution.

The question associated with the execution asks whether the complete execution succeeds in achieving the goal of the skill, as in Sec. 7.1.1. In the implementation on the Simon platform, the robot first moves to the starting configuration of the query, and says “Can I do the following?”; then it executes the query and waits for an answer. The possible answers are *yes* and *no*.

Algorithm 2 Label query selection

Require: \mathcal{G} : Set of potential target configurations

Require: \mathcal{A}_k : List of negative labeled examples for each keyframe distribution

for all $k = 1..K$ **do**

 Randomly select a set of directions to explore E_k

 Compute candidates $s \in C_k$ such that $M(s) = c_k(3 + \frac{1}{n_k})$ and $\frac{s}{\|s\|} \in E_k$ (Eqn. 9)

for all $s \in C_k$ **do**

for all $s_a \in \mathcal{A}_k$ **do**

if $\|s_a - s\| < \|\mu_k - s\|$ **then**

 Remove s from C_k

else

 Find reachability $R(s|g_i)$ for each $g_i \in \mathcal{G}$

 Compute utility $V_k(s)$ (Equation 10)

end if

end for

end for

end for

Compute $V(g_i)$ for each $g_i \in \mathcal{G}$ (Equation 13)

Compute g and q (Equations 12 and 14)

return g and q

If the response to a label query is positive, the execution can be incorporated into the skill model. Only the queried keyframes of the execution affect the model. The two keyframe queries of the overall query q are added to the clusters corresponding to the respective keyframe distributions. Any keyframe in q that was simply sampled from the existing distribution is not added to the corresponding cluster.

If the response to the query is negative then the keyframe queries are added to the set of states to be avoided for corresponding keyframes, denoted by \mathcal{A}_k . This set is used to prune candidate queries for a keyframe. If a candidate is closer to a state in \mathcal{A}_k than the mean of the keyframe distribution, then it is removed from the candidate set. The label query execution process is summarized in Algorithm 3. Sample executions are shown in Fig. 35.

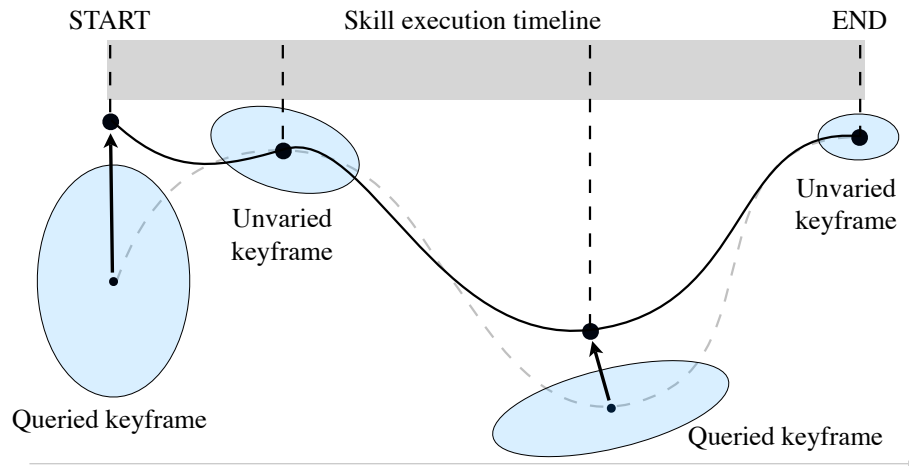


Figure 34: Illustration of the label and partial label query execution process.

Algorithm 3 Label query execution and dialog system.

Require: $q = (s_1, s_2, \dots, s_K)$ and g : Query generated through Algorithm 2.

Require: $\mathcal{I} = \{k_1, k_2\}$: Indexes of the queried keyframes

Move target to g

Move to s_1 and say “Can I do [this/the following]?”

for all $k = 2..K$ **do**

Move to s_k

end for

Get label ℓ from the teacher

for all $k \in \mathcal{I}$ **do**

if ℓ is positive **then**

Add s_k to \mathcal{D}_k

else

Add s_k to \mathcal{A}_k

end if

end for

Acknowledge receiving label by saying “Thank you”

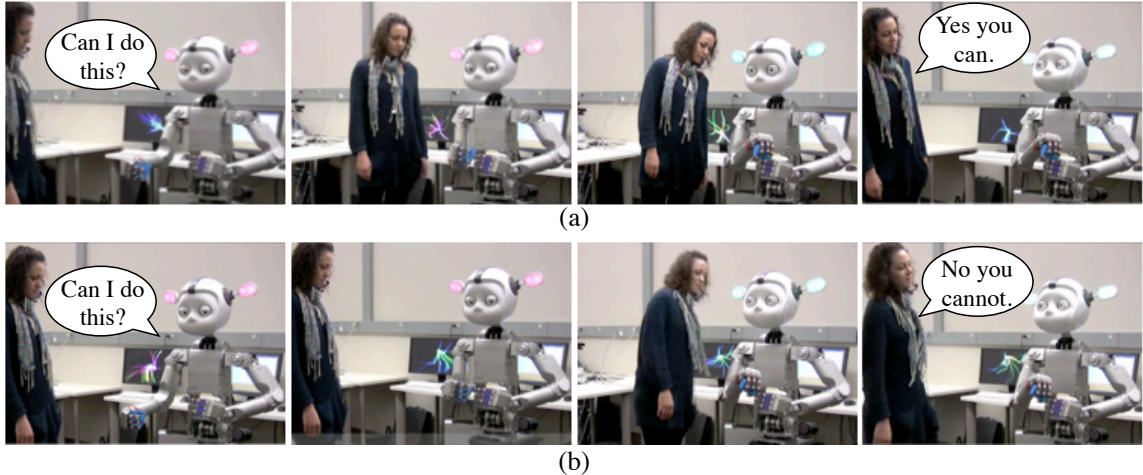


Figure 35: Snapshots from a sample execution of label queries that receive (a) positive and (b) negative labels while learning the *Pour* skill.

8.1.1.4 Partial-label queries

When a label query gets a negative response, both of the queried keyframes end up as a negative state to be avoided. In some cases, however, the unsuccessful execution is only due to one of the keyframe queries. For instance consider a label query for the pouring skill, in which the queried keyframes are the starting keyframe and the keyframe nearest to the target cup. Assume that the queried start configuration is valid, but the queried keyframe near the goal results in spilling — hence the label is negative. In this case, the queried state corresponding to the start keyframe is still added to the avoidance set \mathcal{A}_1 . As a result, in future label queries, the robot will try to avoid this start state, which, in fact, is a perfectly valid starting state. Therefore, label queries may result in inaccurate skill models when they receive negative responses.

This problem was pointed out earlier in Experiment 7, noting how humans deal with it. Instead of asking for a single label for a complete execution, humans asked multiple labels over the course of an execution. This strategy is implemented in A-KLFD, and referred to it as *partial label queries*. The selection of partial label queries is exactly the same as regular label queries, however the execution differs in several ways. The robot first says

Algorithm 4 Partial label query execution and dialog system.

Require: $q = (s_1, s_2, \dots, s_K)$ and g : Query generated through Algorithm 2.

Require: $\mathcal{I} = \{k_1, k_2\}$: Indexes of the queried keyframes

Move target to g

Move to s_1 and say “Let’s say I’m repeating the skill.”

for all $k = 1..K$ **do**

Move to s_k

if $k \in \mathcal{I}$ **then**

if $k = 1$ or $k = K$ **then**

Say “Can I [start/end] like this?”

else

Say “Then, can I [do this/move here]?”

end if

Get label ℓ_k from the teacher

if ℓ_k is positive **then**

Add s_k to \mathcal{D}_k

else

Add s_k to \mathcal{A}_k

end if

Acknowledge receiving label by saying “Okay”

else

if $k = 1$ or $k = K$ **then**

Say “I [start/end] like this.”

else

Say “Then, I [do this/move here].”

end if

end if

end for

Say “Thank you”

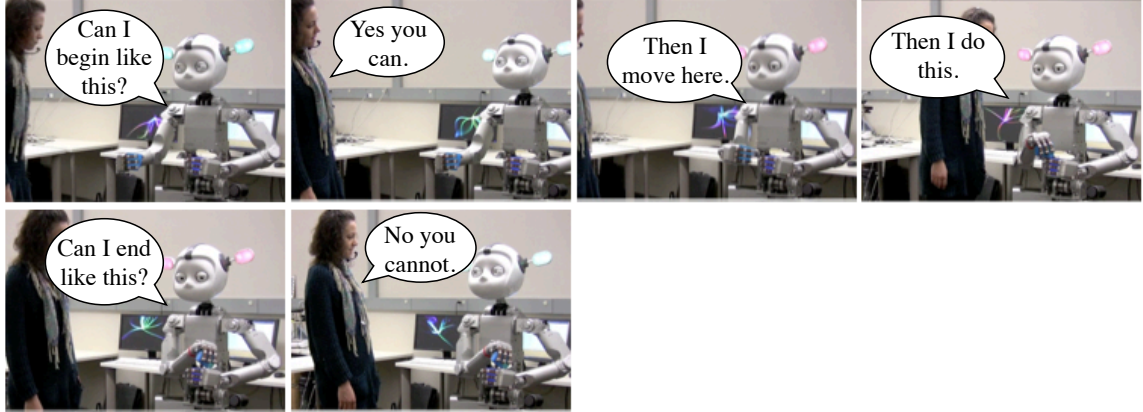


Figure 36: Snapshots from a sample execution of partial label queries for learning the *Pour* skill.

“Let me repeat the skill” to put the questions that follow in context, while it moves to the start configuration. Then it executes the skill stopping at each keyframe. If the keyframe involves a query, the robot asks whether the configuration is a valid point for the keyframe, by saying something like “At this point, can my hand be positioned like this?”. The robot stays at the queried configuration until a response is received. If a keyframe does not involve a query then the robot acknowledges passing through it by saying something like “Then I go here.” The partial label query execution process is summarized in Algorithm 4.

Through partial-label queries, the robot acquires a separate label for each of the queried keyframes. If the label is positive the queried state is added to the corresponding cluster and the model \mathcal{N}_k of the keyframe is updated. If the label is negative the queried state is added to the avoidance set \mathcal{A}_k of the keyframe. A sample execution is shown in Fig. 36.

8.1.2 Demonstration queries

Demonstration queries are requests for a new demonstration, given some constraints. The constraint type used in this thesis is the goal configuration, $g \in \mathcal{G}$. When the robot makes a demo query, it specifies a goal configuration and then it asks the teacher to provide another demonstration. The only parameter that needs to be specified for a demon query is the goal configuration g .

8.1.2.1 Query selection

The selection of the goal configuration to be queried, is based on the overall reachability of the skill in a goal configuration. The objective is to select the goal that makes the skill least reachable, such that the requested demonstration can provide ways to perform the skill in this configuration.

$$g = \arg \min_{g_i \in \mathcal{G}} \left(\sum_{k=1}^K \sum_{s \in C_k} R(s|g_i) \right) \quad (15)$$

The criterion that is minimized is the number of points that are reachable for a skill, summed over keyframes and candidates for each keyframe. The candidate set C_k involves both points directly sampled from \mathcal{N}_k and points that are label query candidates, *i.e.* a random sample of points that have a Mahalanobis distance of $c_k(3 + \frac{1}{n_k})$ as explained in Sec. 8.1.1.

8.1.2.2 Query execution and response

The execution of a demo query is as follows. The robot moves the goal to the selected configuration g . As mentioned earlier, it is assumed that the target is held by the robot in its secondary manipulator. Some goal configurations might not allow the skill to be executed. Therefore, there is an additional step before any query that involves a previously not visited goal configuration. The robot asks if it is allowed to move the goal to the new goal configuration, by saying “Can I move the target here?”. If the answer is negative, g is removed from the goal set \mathcal{G} , and the robot recomputes the demonstration query. If the goal configuration is approved, the robot says “Can you show me another demonstration in this target configuration?”. A sample execution is shown in Fig. 37.

8.1.3 Feature queries

In Experiment 7 we identified several types of feature queries. Three of these types are implemented in the A-KLfD framework.

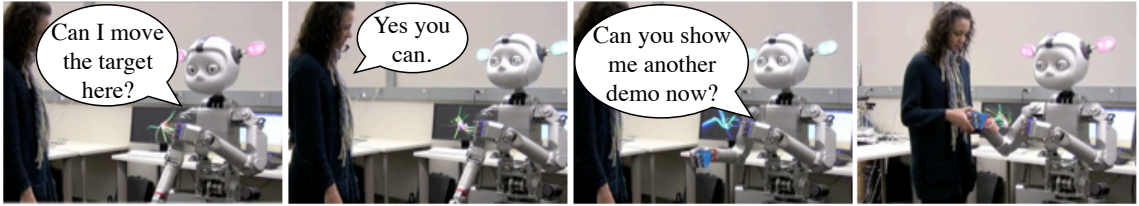


Figure 37: Snapshots from a sample execution of demonstration queries for learning the *Pour* skill.

- *Feature relevance queries* ask whether a feature is relevant for the skill.
- *Feature invariance queries* ask whether a feature has to be invariant.
- *Feature value queries* ask whether a feature can have a certain value.

The features used in the implementation are *position* and *orientation*. Note that these are 3-dimensional subspaces of the state space and not a single dimension. It is possible to consider each dimension of the chosen features, however these might be difficult to interpret for humans, particularly when considered relative to a changeable goal configuration.

8.1.3.1 Query selection

A feature query is always about a particular keyframe and feature, but it may involve a mixture of the feature query subtypes (relevance/invariance/value). A fixed sequencing of the three types of feature queries is used as illustrated in Fig. 38. The robot always starts with either a relevance or invariance query. Depending on the answer it might follow up with value queries. To fully specify a feature query the learner needs to specify the following.

- The keyframe that the query is about
- The feature that the query is about
- Whether to start with a feature invariance or relevance query

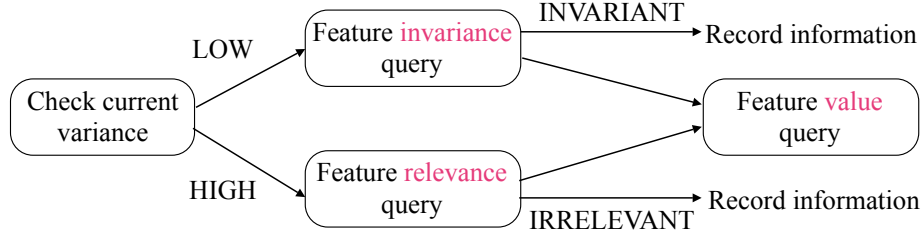


Figure 38: Fixed ordering logic for feature queries.

- Whether to follow up with value queries (depending on the answer of the first question)
- The goal configuration in which value queries are made
- The direction and scales of the value queries

For each keyframe-feature pair the learner first needs to decide whether it will ask a relevance or invariance question. These two queries provide meta information about the variance of a keyframe feature. These are referred to as the *meta-information queries*. Invariant means that variance of the feature is very small (ideally zero). Irrelevant means that the variance is very large (ideally infinite). Non-invariant or relevant means that the variance is non-zero but limited. As observed in humans, the robot should ask about invariance if something is unvaried across demonstrations. On the contrary if a keyframe feature is highly varied across demonstrations the robot should ask if it matters at all. To determine whether to ask about invariance versus relevance, we have a constant threshold on the current variance of the feature. The variance is measured by the scale of potential value queries as explained in the following.

For each keyframe and feature the robot first computes candidate value queries, $C_k = C_k^{pos} \cup C_k^{rot}$ with $k = 1..K$. These are equivalent to keyframe queries computed for label queries as described in Sec. 8.1.1, differing only in that either position or orientation is not varied in the keyframe query. This is achieved by randomly sampling directions that have non-zero values only in the position or orientation subspace. The Mahalanobis distance

of the query is still $c_k(3 + \frac{1}{n_k})$. The choice of the meta-information query is based on the scale of the value queries ($\|s\|$). If the average of this scale across candidates is below a fixed threshold (τ^{pos} or τ^{rot}) the robot chooses invariance queries, and otherwise it makes relevance queries. The threshold is determined empirically, such that a query with slightly below threshold scale results in a value query that has barely noticeable movement from the mean of the keyframe.

The selection of the particular keyframe and feature for the query is based on the computed scale of the value queries. The best outcome of a feature query is to discover that a feature is irrelevant, since it instantly increases the allowed configurations for a keyframe. Thus the learner has a preference for relevance queries. It prefers relevance queries that have larger variance, since these are more likely to be varied in the first place. Thus the keyframe and feature to be queried are selected as follows.

$$k_q, f_q = \arg \max_{k=1..N, f=\{pos, rot\}} \left(\frac{1}{\|C_k^f\|} \sum_{s \in C_k^f} \|s\| \right) \quad (16)$$

Note that a different approach could be to prefer ones with small variance since discovering irrelevance for these keyframes have more impact and upon discovering relevance the robot gets the opportunity to collect more samples for the keyframe. When all relevance queries are exhausted, the robot will make invariance queries. If all possible relevance/invariance queries have been exhausted (*i.e.* the robot already knows whether position and orientation of each keyframe is relevant or invariant), it will directly make value queries for keyframe-feature pairs that are known to be relevant and not invariant.

The query selection process for feature queries is summarized in Algorithm 5.

8.1.3.2 Query execution and response

There are two cases in which the robot does not ask any follow up feature value queries — if it gets a positive response to an invariance query, or if it gets a negative response to a

Algorithm 5 Feature query selection

Require: \mathcal{G} : Set of potential target configurations

Require: \mathcal{A}_k : List of negative labeled examples for each keyframe distribution

Sample directions to explore, non-zero only in position or rotation, E_k^{pos} and E_k^{rot}

Compute candidates C_k^{pos} and C_k^{rot} based on Equation 9

for all $s \in C_k^f$ **do**

for all $s_a \in \mathcal{A}_k$ **do**

if $\|s_a - s\| < \|\mu_k - s\|$ **then**

 Remove s from C_k^f

end if

end for

end for

Reduce $\|C_k^f\|$ to the maximum allowed number of value queries keeping the queries with higher scale $\|s\|$, for each k and f

Choose query keyframe and feature k_q and f_q based on Equation 16

if $k_q = 1$ or $k_q = K$ **then**

 Choose a target configuration $g \in \mathcal{G}$ such that $R(\mu_{k_q}|g) = 1$

else

 Choose a target configuration $g \in \mathcal{G}$ such that $R(\mu_i|g) = 1$ for $i = 1..k_q$

end if

if Average $\|s\|$ over $s \in C_{k_q}^{f_q}$ is smaller than τ_{f_q} **then**

$type_q = invariance$

else

$type_q = relevance$

end if

return $k_q, f_q, type_q, g$ and a set of follow-up value queries $C_{k_q}^{f_q}$

relevance query. These answers provide information that can directly be reflected in future queries or executions of the skill, without clarification or followup.

A feature invariance query consists of going to the chosen keyframe k_q , and then saying “At this point, does my hand have to be positioned like this?”. If the query is about the start or end keyframe, the robot directly goes to that keyframe and prefixes the question with “At the start/end” rather than “At this point.” Otherwise it sets up the context of the feature query by first saying “Let’s say I’m repeating the skill” and then going through the keyframes up to the keyframe of the feature query. When a feature is discovered to be invariant, the robot will not make any more queries about that feature. The information does not effect the skill execution explicitly. Since the keyframe distribution already has very low variance in the queried feature, sampling from the distribution results in points that have low variance.

A feature relevance query consists of going to the chosen keyframe k_q , and then saying “At this point, does the position/orientation matter?”. When a feature is discovered to be irrelevant, the robot will not make any queries about this feature. This information should allow more ways to execute a skill. This is accomplished by changing the sampling procedure for skill execution to gradually expand its search for a reachable sample point. Consider trying to sample a point from a keyframe distribution that has a feature that is irrelevant. The robot will first try to find a point sampled from the distribution that has been learned from the demonstrations provided by the teacher or previous queries. If it finds a sample that is reachable in the given goal configuration, then it does not need to use the information that position or orientation is irrelevant. Otherwise it gradually increases the scale of the samples in the irrelevant feature, until a reachable point is found or the scale reaches an upper limit. Note that this search is done by increasing the Mahalanobis distance in the subspace of the feature, thus it respects the shape of the keyframe distribution until the upper limits are reached.

When a feature is discovered to be relevant the robot will make a number of feature

Algorithm 6 Feature query execution and dialog system.

Require: $k_q, f_q, type_q, g, C_{k_q}^{f_q}$: Feature query generated through Algorithm 5.

Move target to g

if $k_q = 1$ or $k_q = K$ **then**

 Move to μ_{k_q}

if $type_q = relevance$ **then**

 Say “Does [position/orientation] matter at the [start/end]?”

else

 Say “Does [position/orientation] have to be this at the [start/end]?”

end if

 Get label ℓ_{meta} from the teacher

else

 Move to μ_1 and say “Let me repeat the skill.”

for all $k = 1..(k_q - 1)$ **do**

 Move to s_k

if $k = 1$ **then**

 Say “I start like this.”

else

 Say “Then, I [do this/move here].”

end if

end for

 Move to s_{k_q}

if $type_q = relevance$ **then**

 Say “At this point, does [position/orientation] matter?”

else

 Say “At this point, does [position/orientation] have to be this?”

end if

 Get label ℓ_{meta} from the teacher

end if

if ($type_q = relevance \ \& \ \ell_{meta} = 1$) OR ($type_q = invariance \ \& \ \ell_{meta} = 0$) **then**

for all $s_i \in C_{k_q}^{f_q}$ **do**

 Move to s_i

 Say “Can it be this?”

 Get label ℓ_i from the teacher

if ℓ_i is positive **then**

 Add s_i to \mathcal{D}_{k_q}

else

 Add s_i to \mathcal{A}_{k_q}

end if

end for

end if

Say “Thank you”

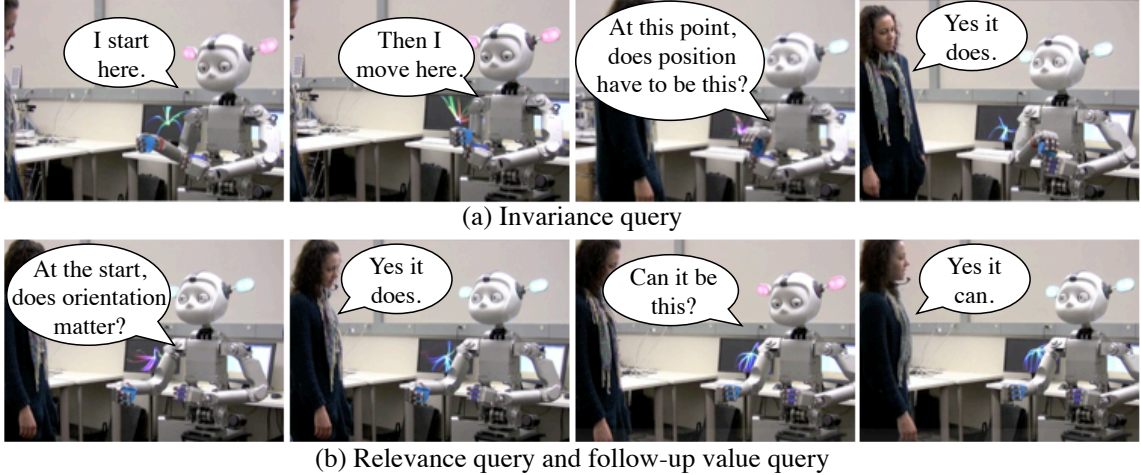


Figure 39: Snapshots from sample executions of feature queries.

value queries. First it will move the goal to a configuration that allows the most value queries to be reachable, *i.e.* $g = \arg \max_{g_i \in \mathcal{G}} (\sum_{s \in \mathcal{C}_{k_q}^f} R(s|g_i))$. Then the robot makes a number of feature value queries sequentially, by changing the end-effector configuration to a randomly chosen reachable query from $\mathcal{C}_{k_q}^{pos}$ or $\mathcal{C}_{k_q}^{rot}$. When the query point is reached, the robot says “Can it be like this?” and waits for a response. Note that this question follows an invariance/relevance query that mentions the feature and therefore the reference to “it” in the question is assumed to be clear. If the response is positive the point is added to the cluster and the keyframe distribution is recomputed. If the answer is negative, the point is added to the avoidance set \mathcal{A}_{k_q} . Feature query execution is summarized in Algorithm 6. Example executions are shown in Fig. 39.

8.2 Experiment 8: Evaluation of Active Keyframe-based LfD

This experiment evaluates the effectiveness of the queries implemented in the A-KLfD framework. It involves the comparison of the different types of queries with a baseline of passive learning from naive users.

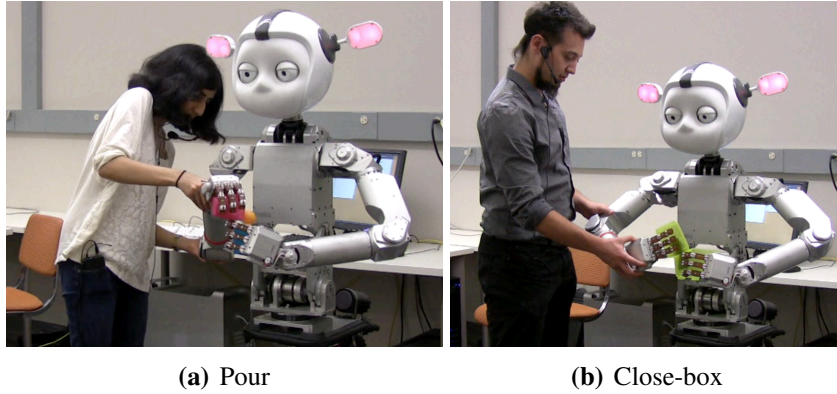


Figure 40: Snapshots from Experiment 8 while participants in the baseline data collection demonstrate the two benchmark skills to Simon.

8.2.1 Experimental design

The evaluation is done on the Simon robot platform. Two skills from the 2-handed goal-oriented manipulation skills are considered. These are *pouring* and *closing* the box (Fig. 7(a-b)). Snapshots from the experiment while participants demonstrate these skills kinesthetically are shown in Fig. 40. In both skills the robot holds the target object in its left hand and the skill is demonstrated on the right arm. Pouring consists of transferring the content of one cup to the other. For easy assessment of success, a ping-pong ball is used as the content to be transferred. Closing the box involves a box whose lid opens around a hinge and stays horizontal. So the robot needs to push it up and then towards the closing direction, while avoiding any collisions with the box.

The interaction with the robot is regulated with a number of speech commands recognized using Microsoft Speech API with a fixed grammar. The commands are described in the following.

- *Release your arm* makes the robot’s right arm go into gravity compensation where it can be manipulated by the teacher.
- *Hold your arm* makes the robot’s right arm go into position control and maintain its

current position.

- *Open/close your [right/left] hand* changes the state of the hand. If the teacher does not explicitly indicate right or left, the command is applied to the right hand.
- *Move the target up/down/to your left/to your right/forward/backward* changes the position of the goal in the indicated direction. The target is not allowed to move beyond one step from the starting center position in each direction — hence there are only 7 possible positions of the target.
- *Rotate the target out/in* changes the orientation of the target around the vertical axis. Rotation around other axes is not allowed since they would result in undesirable goal configurations. Rotation was only allowed at the center location. Thus the total number of allowed goal configurations were nine, *i.e.* $|\mathcal{G}| = 9$.
- *Start like this* indicates the beginning of a new demonstration and marks the current end-effector configuration as the first keyframe.
- *Then move here* indicates a keyframe in the demonstration. It adds the current end-effector configuration into the demonstration as a new keyframe.
- *Finish like this* indicates the end of the ongoing demonstration and marks the last keyframe.
- *Can you try it yourself?* triggers an execution of the skill.
- *Do you have any questions?* triggers the next query.

The movements of the goal are controlled with speech commands rather than gravity compensation to make controlling two arms manageable and imposing some constraints on the possible goal configurations. The same set of goal configurations \mathcal{G} mentioned above were used in the robot's queries.

All demonstrations in this experiment are keyframe demonstrations and the skill learning method is described in Sec. 3.3.1.

8.2.1.1 Procedure

This experiment compares each query type used individually, with the baseline of unguided teaching. The data collection for the baseline starts with video instructions given to the participant. The video goes over the speech commands showing their usage and their effects. Then it presents an example of teaching with keyframes. The person in the video provides two keyframe demonstration of a skill other than the ones used in the experiment. The skill is dropping a ping-pong ball that is in the robot's hand into a cup. Then an execution of the skill is shown. Finally the video shows the robot executing the two skills that the participants are asked to teach in the experiment. After the participant watches the video, the experimenter answers any questions that the participant might have and describes the idea of keyframe demonstrations, stating that none of the movements between the keyframes are recorded.

Then the practice session begins. The participant tries all the commands that are about releasing/holding the arm, opening/closing the hands and changing the configuration of the target. Then the participant provides two demonstrations of the skill that was given as an example in the video and views the outcome. They can choose to change the target configuration before their practice demonstrations or the execution.

After the practice participants are told that there will be a contest after collecting data from all participants. They are told that the taught skills will be tested in different goal configurations and the skill that succeeds in most tests will be the winner and receive an award of \$15. This is done to motivate the participants to teach successful and generalizable skills.

Next the data collection starts. The participant is given five minutes to provide as many demonstrations as they can. Executions of the skill are not allowed during this period. The

order of the two skills is counterbalanced across participants. At the end of the five minutes the robot indicates that the time is up. If the five minutes end during a demonstration, it waits until the end of the demonstration. If a larger fraction of the demonstration was given before the end of five minutes, the demonstration is included in the data set, otherwise it is thrown away.

The participant is allowed to see the execution of the two skills after providing demonstrations for both skills. They do so by requesting an execution of the particular skill, *e.g.* “Can you try to close the box?”.

Data collection for the different A-KLFD queries did not involve any naive users. For a total of five minutes, the experimenter provides two demonstrations of the skill and then responded to the robot’s queries. The two initial demonstrations are referred to as the *seed demonstrations*. The same seed demonstrations are used for all types of queries for learning each skill. The number of keyframes used for teaching the skills were chosen to be the median of the numbers used by participants. A separate data set was collected for four different types of queries described in Sec. 8.1.1, Sec. 8.1.2 and Sec. 8.1.3. Thus data evaluation involves five conditions:

- *No Queries* with *non-expert* teachers (NQ)
- *Label Queries* (LQ)
- *Partial-label Queries* (PQ)
- *Demonstration Queries* (DQ)
- *Feature Queries* (FQ)

8.2.1.2 Evaluation

Skills are evaluated on two main metrics: successful execution and generality. To measure success, each of the taught pouring and closing skills were executed in five representative

goal configurations from \mathcal{G} and they were deemed successful or unsuccessful by two experimenters. Successful execution for pouring required the ping-pong ball to go into the cup held in the left hand, and successful execution for closing the box required the box to be closed and not displaced from its initial configuration due to a collision. When an execution was a near miss, it was repeated two more times and if it succeeded twice it was deemed successful. The testing involved ten individual skills each for pouring and closing — six taught by participants and four learned by the robot using each query type separately.

Generality is measured with the skill coverage of possible goal configurations. Fifty different goal configurations were sampled. All of these were within the convex hull of the goal configurations \mathcal{G} that were allowed in the data collection for the baseline and that were also used as the potential goal configurations in the robot’s queries. For each goal configuration, it was checked whether the learned skill allowed a reachable execution of the skill. The fraction of goal configurations out of the fifty for which the learned skill can be executed is used as the measure of coverage.

8.2.2 Findings

For the baseline, demonstrations were collected from six participants recruited from the campus community. Participants either received \$5 or extra credit in an Artificial Intelligence course, for their participation. The data collection procedure took around 30 minutes in total, 10 minutes of which were devoted to providing the demonstrations of the two skills.

The coverage of learned skills and the number of demonstrations provided within 5 minutes are shown in Table 24. The success of the learned skills in the tests are shown in Table 25. The following observations are made.

All queries improve skill coverage. The results demonstrate that skills learned with any type of embodied query has much larger coverage (around 85 to 100%) than skills taught by human teachers (around 65%) for both skills.

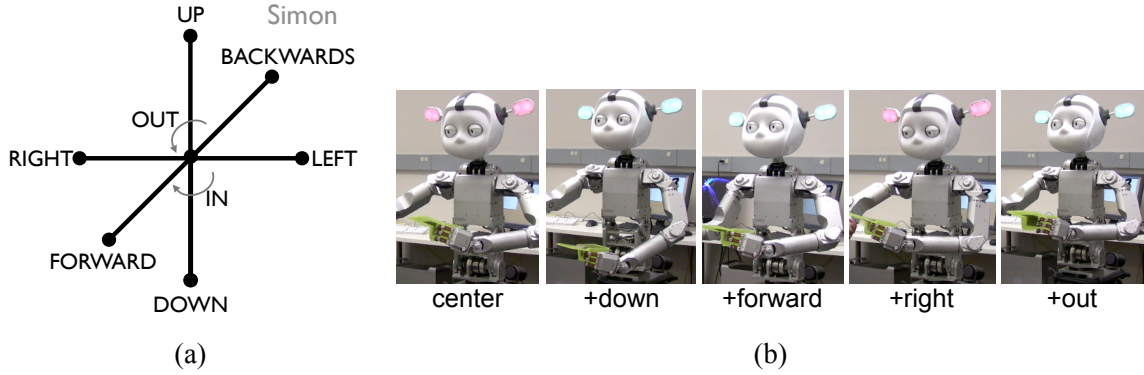


Figure 41: Layout of possible goal configurations \mathcal{G} relative to Simon and the five canonical goal configurations in which the robot is tested to assess the success of the skills.

It was observed that two out of the six participants did not change the configuration of the goal across their demonstrations, even though they had explicitly practiced all speech commands for changing the goal configuration and were given a detailed description how the goal can be changed showing Fig. 41. Two other participants used the commands to change the goal configuration only once at the beginning and then provided all their demonstrations in that configuration. One of them mentioned trying to find the goal configuration in which the skill was easiest to demonstrate,

Demo queries provide best coverage. For both skills, demo queries achieve the highest coverage as compared to the other query types. This is not surprising as demo queries are chosen explicitly to increase reachability of the skills in possible goal configurations \mathcal{G} (see Equation 15). The difference is particularly notable in the closing skill. All other query types have very similar coverage.

Similar pace in all conditions. The number of demonstrations/queries completed within five minutes does not differ much across conditions. The number of label queries is one more than the number of partial label queries for both skills, which is also expected since a single partial label query involves two questions while a label query involves a single question. Given that PQs obtain twice the label as LQs, this difference is very small and

Table 24: Coverage of the taught skills and the number of demonstrations provided within 5 minutes in the six conditions of Experiment 8. For conditions that involve queries, the number of demonstrations indicate the sum of the two initial demonstrations and the number of queries made in the rest of the time.

Condition	Pouring		Closing	
	Coverage (%)	# demos	Coverage (%)	# demos
NQ	64 (SD=29.69)	8.17 (SD=3.06)	63.67 (SD=20.14)	8.67 (SD=3.20)
LQ	94	9	88	8
PQ	96	8	86	7
DQ	100	8	96	7
FQ	94	8	88	7

PQs can be considered as highly efficient.

All queries improve skill success. Skill success is much higher in skills learned with queries as compared to the skills taught by non-expert teachers. Percentage of successful executions is particularly low for the pouring skill taught by non-expert teachers (around 13%), as compared to an 60 to 80% success of skills learned from queries. All of the five goal configurations are covered by the skills learned through queries, while an average of 30 to 37 percent of skills taught by non-experts do not cover the tested goal configurations (*i.e.* they cannot even try to execute the skill).

Note that the success of skills learned through queries are dependent on the first two demonstrations, which were provided by the expert teacher. It is expected that the skill is already successful after the first two demonstrations in goals that are covered, however has low coverage. Therefore queries expand coverage while maintaining a successful skill.

It should be noted that success of skills also depend on how well the KLfD framework captures the particular skill. Overall, the pouring skill is more challenging than the closing skill, since the locations that allow successful pouring are quite restricted, while closing has more flexibility in how the goal is achieved.

Table 25: Success of learned skills taught in the six conditions of Experiment 8, tested in five different goal configurations. The numbers indicate the percentage of tests out of five that were successful, failed, or N/A. N/A means that the learned skill did not cover the goal configuration in which it was being tested.

	Pouring			Closing		
	Success	Fail	N/A	Success	Fail	N/A
NQ	13.33 SD=32.66	56.67 SD=29.44	30.00 SD=27.57	53.33 SD=24.22	10.00 SD=10.95	36.67 SD=29.44
LQ	60	40	0	80	20	0
PQ	60	40	0	100	0	0
DQ	80	20	0	100	0	0
FQ	80	20	0	100	0	0

Another factor affecting the success of the skills is the choice of keyframes and consistency in the number of keyframes. Although participants were told to be consistent, they sometimes gave demonstrations with different number of keyframes. For example, most participants demonstrated the tilting of the cup for the pouring skill in several keyframes. If they had a smaller number of demonstrations with the largest number of keyframes, then some consecutive keyframes in these demonstrations were added into the same cluster and resulted in larger than intended clusters. As a result the skill execution ended up not tilting the cup sufficiently to pour the content. The second issue observed was the failure to provide obstacle avoidance keyframes, resulting in skills that had two arms collide and fail at achieving the goal. This is the same issue as the one observed earlier in the earlier experiment that involved teaching with keyframe demonstrations (Experiment 2).

Expert is on par with queries. Skills taught by the expert has very similar coverage and success rate as the skills learned through different queries. This demonstrates that a good teacher can follow certain strategies which allow them to teach skills that are as successful as skills learned through queries that are specifically designed to succeed in the objective metrics for evaluating the skill.

8.3 Summary

This chapter describes the Active Keyframe-based LfD framework. It develops methods for generating different types of queries from skills that are represented as a sequence of keyframe distributions.

- *Label queries* involve replaying the skill with some of the keyframes sampled explore outside of what has been demonstrated. The variance of the explored keyframes are selected to be at a theoretical optima that balances maximizing the probability of the point being valid (*i.e.* yielding a positive label), and maximizing how much the variance will be increased by the explored point.
- *Partial-label queries* are label queries whose execution involves stopping at explored keyframes and getting separate label for each of them, rather than getting a single label for the whole execution.
- *Demo queries* involve requesting a demonstration in a specified goal configuration. The queried goal configuration is the one that allows the least possible executions of the learned skill.
- *Feature queries* involve a first step that tries to identify whether a feature (position or orientation of a keyframe) is irrelevant (unimportant for the skill) or invariant (cannot be varied from the demonstrated value). If it is either, a second step queries several values of the feature to obtain more positive points for the particular keyframe distribution.

Different queries are evaluated with an experiment comparing them to skills taught by six naive human teachers and one expert teacher. The experiment demonstrates that queries can improve both the success of the skill and its coverage (the number of goal configurations that it is applicable in). While the differences are minor in the provided evaluation, the most successful and general skills are learned through demonstration queries.

CHAPTER IX

TEACHING HEURISTICS FOR CLASSIFICATION

In this chapter we switch gears to the second mechanism for guiding humans in teaching interactions with robots — *teaching heuristics*. Teaching heuristics are instructions given to the teacher on how to teach. There are several motivations for this mechanism. The first is the differences observed between naive and expert or optimal teachers pointed out in the previous chapter. Having a good understanding of the underlying learning mechanism, and having a lot of practice teaching the robot allows experts to provide more informative examples, which, as we have seen in Chapter 4, does not happen naturally with naive teachers. However, it is not feasible to require that users understand how the robot learns and practices teaching extensively. Just being told what to do can have a great impact in making humans do the right thing.

Consider dog training. It is difficult for humans to train dogs to do something they want. Part of the reason is that dogs learn very differently from humans. Expert dog trainers go through a curriculum on dog learning behavior, such as habituation or operant conditioning, and practice methods like clicker training with many different dogs [64, 120]. On the other hand, people who are not expert dog trainers, can still find ways train their dogs. They can watch online video tutorials, or sign up for a crash course. These do not go into how a dog learns and do not provide practice. Instead they provide the person with a recipe of what to do — step by step strategies and things to generally keep in mind. We employ the same idea of giving humans instructions on how to teach a robot in order to elicit the behavior that is suited for the learner.

A second motivation is that humans will need instructions using robots anyways. Making it possible to teach robots without getting any instructions at all is extremely challenging. For instance detecting when a human is intending to teach, and determining the start and end of a demonstration are difficult problems that are hardly solved even for restricted contexts. It is a much more practical solution to have simple commands that trigger learning, and mark the boundaries of a demonstration. The small cost to pay is that users need to be instructed about these commands in order to be able to teach. We suggest to expand such instructions to include instructions on the types of demonstrations that they should provide.

The idea of providing teaching instructions is rather simple and may seem obvious. The reason why it has not been explored in the literature is perhaps that learning from human generated data is a relatively young area of research. Traditional ML algorithms are designed to learn from data generated by a process that can be modeled. In these applications of ML, humans may take part in generating the data, however, the content of the data is mostly dictated by the environment. In more recent applications of ML, such as training a gesture recognition system or recommendation systems, humans have much more direct influence on the content of the input data. Humans, unlike physical phenomenon, are difficult to model as a data generating process. There can be a large variance in the data provided by humans. At the same time, humans are very flexible and their behaviors can be easily influenced. The core idea of teaching instructions from a ML perspective, is to exploit the flexibility of the data generating process, and to influence it towards what is useful for the learner.

There are several challenges related to creating teaching instructions for machine learners. First, it is not clear what constitutes a good teacher for any given learner. To this end, the field of Algorithmic Teaching provides great intuition and principled ways of studying teaching. However, for many complex domains, teaching is an intractable algorithmic

problem and there are no optimal or even approximate teaching algorithms that are efficient. For other problems, such as sequential decision tasks, algorithmic teaching has not been explored at all in the literature. Thus there are challenges in defining the behavior that teaching instructions should try to elicit. Secondly, even if the desired behavior was known, it is not clear whether instructions extracted from this information will succeed in eliciting the desired behavior. Finally, converting an algorithm to natural language instructions that can be understood by common users is in itself a challenge.

In this chapter, these issues are addressed in four classifier learning problems that involve humans teaching virtual agents. An experiment in each of these problems compare unguided natural teaching, with *guided* teaching where the participants are given teaching heuristics.

9.1 *Optimal Teaching in Classification Problems*

A helpful teacher can significantly improve the learning rate of a ML algorithm in classification problems, as shown in the field of Algorithmic Teaching [13, 92, 57]. As discussed in Sec. 2.4, this field studies the *teaching problem*, that is, producing a set of labeled examples based on a *known* target concept. The goal is to find efficient algorithms that can teach with as few examples as possible. Often times this is much smaller than the number of randomly, or even actively, chosen examples needed to learn a concept [5, 57]. In other cases, a concept that is not PAC learnable under arbitrary distributions, may become learnable with a helpful teacher [48].

To illustrate the potential of good teaching, consider the following canonical example from [47] (illustrated in Fig. 42). The learning problem is to find a linear separator, in a 1-D space, from observed examples. A simple consistent learner achieves this by placing the threshold between the rightmost negative example and the leftmost positive example. In this setting an active learner achieves logarithmic advantage over random sampling, by always querying the unlabeled sample closest to the estimated boundary. However, a good

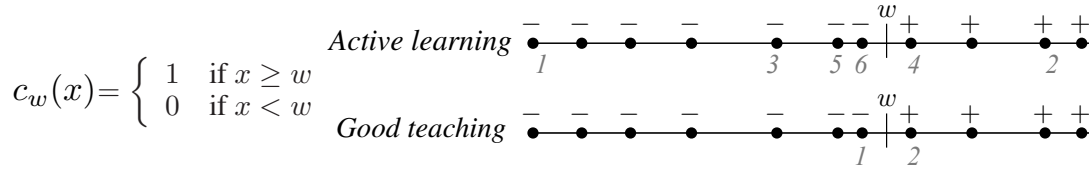


Figure 42: Motivating example from [47] which illustrates the power of optimal teaching as compared to active learning.

teacher could directly provide the two examples closest to the true decision boundary, to achieve the smallest possible error rate.

As elaborated in Chapter 4, humans are not naturally optimal teachers, they have the capacity to adapt for the needs of the learner. With the appropriate mental model of the learning process, their teaching can be much closer to optimal.

The source of information for creating teaching instructions is the learner. The concept class used by the learner as well as the particular learning algorithm both affect what it means to be a good teacher for the learner. Different classifier learning problems, lead to different ways of generating teaching heuristics. Three in particular are discussed:

- heuristics based on known concept-specific optimal teaching algorithms,
- heuristics based on algorithms that give the best performance on a finite dataset, and
- heuristics based on an abstract understanding of what constitutes a useful example.

The subsequent sections explain the teaching problem for these three settings and illustrate them with an example learning problem.

9.1.1 Provably Optimal Teaching

Characterizing the teachability of concepts is a central problem in Algorithmic Teaching, and many teachability metrics have been proposed. The most popular metric is the *Teaching Dimension* [57], which quantifies the success of a teacher with the number of examples required to teach a concept.

Let c be a target concept from the concept class C , and T_c the set of example sequences that uniquely identify c . The shortest sequence of examples that uniquely identify a target concept ($\tau_c^* = \operatorname{argmin}_{\tau \in T_c} |\tau|$) is referred to as an *optimal teaching sequence*.

Definition 1 (Teaching Dimension) *The teaching dimension of a concept c with respect to C is the size of its optimal teaching sequence, denoted by $TD(c, C)$. The teaching dimension of the concept class C is the teaching dimension of the hardest-to-teach concept in C , denoted by $TD(C)$.*

$$TD(c, C) = \min_{\tau \in T_c} |\tau|; \quad TD(C) = \max_{c \in C} (TD(c, C))$$

Other teachability metrics have been proposed. [99] uses the number of examples needed until the target concept is the most specific hypothesis consistent with the data. This measure, like $TD(C)$, assesses the difficulty of teaching a concept class by the concept that is *hardest* to teach. In contrast, the *average teaching dimension* [6] captures the overall difficulty of teaching a concept class. More recently, two variants of the teaching dimension were proposed where (i) the learner is biased towards less complex solutions or (ii) the learner assumes that the teacher is optimal [11].

Teachability metrics reflect the sample efficiency of learning concepts when they are taught by the best possible teacher. In addition to computing these lower bounds, it is important to show that this lower bound is achievable by an *efficient* algorithm. In general, finding an optimal teaching sequence for an arbitrary concept is NP-hard (by reduction to the minimum cover set problem [57]). However polynomial time algorithms have been proposed for certain concept classes: conjunctions, monotone decision lists, orthogonal rectangles, monotone K-term DNFs among others [57]. These algorithms exploit the compact (polynomial-size) representation of concepts and instances, to avoid enumerating all possible example sequences. In some cases, information exchange prior to the teaching session [67], or feedback from the learner during teaching [12] can make certain classes

teachable in polynomial time or reduce their teaching dimension.

This thesis proposes using optimal teaching algorithms as a reference for generating teaching instructions for human teachers. The idea is to explain these algorithms in natural human language so as to elicit closer to optimal teaching sequences from humans. Conjunctions are used as a case study.

9.1.1.1 Teaching Conjunctions

The problem of teaching conjunctions was discussed earlier as part of the task learning problem in Sec. 3.2.1. Experiment 1 compared unguided human teaching of conjunctions with an optimal teaching algorithm. The following briefly recaptures this problem more formally and discusses teaching instructions for conjunctions.

A conjunction is a list of feature values that must all be true in a sample, for it to be labelled as positive. *Non-monotone* conjunctions allow more than one value of a feature to be in the conjunction. Features that appear in the conjunction are referred to as *relevant*.

Consider the concept class C_n of non-monotone conjunctions over n binary variables. With respect to C_n , the teaching dimension of a target concept c_r with r relevant variables is identified as $TD(c_r, C_n) = \min(n + 1, r + 2)$ [57]. The algorithm that produces an optimal teaching sequence of this size is as follows.

- *Step 1:* Show one positive example.
- *Step 2:* Show another positive example in which all irrelevant features are reversed.
- *Steps 3- $|\tau^*$:* Show r negative examples in which one of the r relevant features is reversed.

The first two steps in this algorithm prove to the learner that all changed features are irrelevant since they have different values in two positive examples. The rest of the steps show the relevant features one by one, since in each negative example only one feature is different from an instance that is known to be positive.

The teaching heuristics provided to humans for teaching a conjunction is a step-by-step strategy based on this algorithm. Some terms such as *positive/negative example* or *relevant/irrelevant features* need to be replaced or explained in the instructions given to the teacher. A specific example of such instructions for teaching conjunctions are given in Experiment 9 and Experiment 10.

9.1.2 Empirically Optimal Teaching

As discussed in Sec. 9.1.1, when an optimal and time-efficient teaching algorithm exists, it can be directly used to devise teaching instructions for the concept. However for a large number of more expressive concept classes no such algorithm exists. One reason for this is the requirement that the target concept is uniquely identified for optimal teaching. For an uncountably infinite concept class, such as linear classifiers in \mathbb{R}^N , converging to the exact target hypothesis may require an exponentially large number of carefully chosen examples, or may even be impossible. The problem is further complicated in the presence of noise (e.g. wrong labels provided by the teacher).

In many practical settings, teaching involves picking examples from a large pool of potential examples, rather than instantiating examples feature by feature. Teaching then consists of choosing a sequence of examples τ to be provided to the learner from the example set S_c consistent with a target concept c^* . To quantify teachability in this setting, the following metric is defined.

Definition 2 (Empirical Teaching Dimension) *The empirical teaching dimension of a concept c , with respect to an example set S_c , is the size of the smallest sequence of examples from S_c , such that the concept learned from this set results in a desired empirical error over S_c . It is denoted by $ETD(c, S_c)$.*

$$ETD(c, S_c) = \min_{\tau \in T_\epsilon} |\tau|, \quad T_\epsilon = \{\tau : err_S(\hat{c}_\tau) < \epsilon\}$$

where $err_S(\hat{c}_\tau)$ denotes the empirical error over the example set S of the concept \hat{c}_τ learned from the example sequence τ . The desired error rate ϵ can be chosen as the smallest possible error rate that can be achieved with teaching sequences from S_c .

As a result of the input set being finite, there are a finite number of possible empirical error rates that can be achieved. Although this makes the problem more tractable, finding an algorithm that produces a teaching sequence that achieves the best performance with fewest examples possible might still not be trivial. As in the optimal teaching setting, the known structure of the concept class, and the relationship between examples in S_c can be exploited to create algorithms that produce empirically optimal teaching sets efficiently. These algorithms can then be used for devising teaching instructions as was done in Sec. 9.1.1. As an illustrating example, the concept class of linear separators in \mathbb{R}^1 is discussed.

9.1.2.1 Teaching linear separators in \mathbb{R}^1

Consider the problem shown in Fig. 42, which involves choosing examples to be shown to the learner from a set $S_c = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in [0, 1]$ and $y_i \in \{0, 1\}$. The concept class of linear separators in \mathbb{R}^1 is $C = \{c_w : w \in \mathbb{R}\}$, $c_w(x) = 1$ if $x \geq w$, 0 if $x < w$. Assume a simple consistent learner that places the threshold at the midpoint of the rightmost negative example and the leftmost positive example. The smallest error rate on S_c , which is zero, can be achieved by providing two examples closest to the decision boundary, $\tau = \{x_w^+, x_w^-\}$, thus $ETD(C) = 2$.

The teaching heuristics provided to humans for teaching a linear separator in \mathbb{R}^1 consists of describing the two examples to be shown to the learner. This requires an intuitive way to refer to the arrangement of the examples in 1-D, and describing the notion of decision boundary. An example for such heuristics are given in Experiment 11.

9.1.3 Approximately Optimal Teaching

For teaching linear separators on a line (Fig. 42), there exists a straight forward strategy that lets the teacher directly provide the empirically optimal teaching sequence. This is made

possible by the human's ability to visually browse the set of examples in S_c in a structured manner and assess their distance to the true decision boundary. In some cases this is not possible. Two characteristically different scenarios are discussed in the following.

Online Teaching. One case is when the sample space is high dimensional and the example set has no intuitive structure to allow browsing by humans. For instance if a human teacher needed to browse through a set of images to select a teaching set, it would be difficult for them to remember and compare all images to pick the optimal ones. Instead they would need heuristics that let them decide whether to label an image or not as they browse. This can be considered an *incremental* or *online* version of the teaching problem, analogous to the online learning paradigm [85].

The online teaching problem can be characterized as finding the right *ordering* in which to present the examples to the learner. For instance in the case of the linear separator in \mathbb{R}^1 (Sec. 9.1.2.1), providing the two examples closest to the border first, is an ordering choice. The ordering problem was studied by MacGregor [89], who states that some orders provide relevant information more rapidly than others and proposes two heuristics for choosing good orderings. The *fast-match* heuristic evaluates each example in terms of the degree that it improves the goodness-of-fit between current and true models and orders them from best to worst. The *close-all* heuristic is derived from observing the examples chosen by the fast-match heuristic. While trying to rapidly match the final model the fast-match heuristic tends to select examples which create a variety of values on each dimension, rather than placing prototypical, pure-case types of examples at the start of the sequence. Therefore examples that are similar to both categories (*i.e.* borderline examples) are included early in the sequence. While the fast-match heuristic is impractical for human teachers, the close-all heuristic gives an intuitive description of the type of example that is most useful to the learner.

Infinite Example Sets. In other cases there is no existing set of examples (S_c) to browse

and pick teaching examples from. For instance, teaching skills to a robot or programming a gesture recognition system by demonstrating gestures, involve composing examples from scratch as opposed to choosing examples from a dataset. This gives flexibility to the teacher, but makes the optimal teaching problem intractable. In this case teaching heuristics should guide the teacher on how they should compose their positive and negative examples and how they should weigh different category.

Heuristics for example evaluation versus generation. The two cases point to two different types of teaching heuristics. Online teaching needs heuristics that allow the teacher to *evaluate* examples in terms of their informativeness for the learner. This will help the teacher decide whether or not to label an example. The objective for these heuristics is to be as close to empirically optimal as possible.

The infinite example set case needs heuristics that let the teacher *generate* examples that will be most informative to the learner. The objective for this type of heuristics is harder to identify. Considering computational heuristics, empirical error over a test set can be used to assess and compare heuristics. The goal of an algorithmic teaching heuristic is to provide the best possible learning curve in comparison to other alternative algorithmic teachers, for instance sampling the input space uniformly. Since the purpose of devising computational teaching heuristics is for them to be used by humans, a better way of assessing the quality of the heuristic is how well it performs in comparison to baseline human teaching. Again, this requires a benchmark test to evaluate the performance of the learned classifiers.

Devising teaching heuristics for humans. In both the online teaching case and the infinite example set case it is difficult or impossible to devise optimal or empirically optimal teaching algorithms. As a result the approaches for creating teaching instructions discussed in Sec. 9.1.1 and Sec. 9.1.2 are not applicable. But the fact remains that some teaching examples are better than others. In these cases, the teaching heuristics need to take inspiration from algorithms that are *approximately* optimal or rely on intuitions about the types of

examples that are informative to the learner (*e.g.* MacGregor’s heuristics).

The first approach is to first develop an approximately optimal teaching algorithm and then extract teaching heuristics for humans – either based on the intuition of the approximate algorithm or based on an inspection of example sets produced by the algorithm. Approximate algorithms are sub-optimal algorithms whose solutions differ from optimal ones within provable bounds (*e.g.* optimal up to a small constant factor). They are often associated with NP-hard problems. As mentioned earlier, optimal teaching can be reduced to the well-known *set covering* problem, for which several approximate algorithms exist.

The second approach is to directly construct teaching heuristics based on an understanding of the concept class as well as the particular learning algorithm. Either approach is much less structured than the ones considered in Sec. 9.1.1 and Sec. 9.1.2 and require more iterative improvements through human subject testing. To illustrate the potential of teaching heuristics in the approximate settings, the class of binary Nearest Neighbor (NN) classifiers are discussed in the following.

9.1.3.1 Teaching Nearest Neighbor Classifiers

A NN classifier stores all labeled examples $S_\tau = \{(x_i, y_i)\}_{i=1}^{|\tau|}$ provided by the teacher. It assigns a label to a new sample $x \in \mathcal{X}$ as the label of the most similar example in S_τ , using a similarity function d over \mathcal{X} , *i.e.* the label of x is predicted as y_j , where $j = \operatorname{argmin}_{i=1, \dots, n} d(x_i, x)$.

Intuitively, for any discriminative classifier, a good teacher needs to provide examples that allows differentiating between the classes. For a binary NN classifier, all examples in one category should be closer to one of the examples provided for that category, than they are to any of the examples provided for the other category. For instance, the two examples closest to the border in Fig. 42 satisfy this property. Thus choosing borderline examples as suggested by [89] can be an intuitive heuristic that can be used as teaching instructions. Note that the optimal teaching algorithm for conjunctions explained in Sec. 9.1.1 is also

in line with this heuristic, since it provides the negative examples that are closest to being positive (*i.e.* different by one property).

As mentioned, one approach for devising teaching heuristics is to first consider an approximately optimal teaching algorithm. To illustrate this with nearest neighbor classifiers, consider the teaching problem which consists of choosing a sequence of examples from a finite set $S_c : \mathcal{X} \times \{0, 1\}$ that will be revealed to the learner. An exhaustive approach to achieve this is to evaluate all possible ordered subsets of S_c as potential teaching sequences and choosing the best one, which results in an exponential algorithm. This is equivalent to the set cover problem. Consider the following polynomial time greedy approximation for solving this problem. The teacher first considers all possible S_τ of size 2, that involve one positive and one negative example. It chooses the example pair S_τ that gives the smallest empirical error on $S_c \setminus S_\tau$. Then, one by one the teacher chooses the example $(x_i, y_i) \in S_c$ that reduces the empirical error the most, until a desired empirical error rate is achieved. Although this algorithm is sub-optimal, it has been shown to be the best possible polynomial time approximate algorithm for the set cover problem [41].

Note that the intention is not to provide this algorithm as a teaching heuristic for humans, as in Sec. 9.1.1. Although this is a polynomial time algorithm, it is intractable for a human and the teaching task for the human does not involve a finite set to choose examples from. Instead, this algorithm can be used to produce exemplar approximately optimal teaching sequences. These sequences can be inspected as to gain insights into what constitutes a good example for the learner from a human teacher's perspective. Teaching heuristics that will let human teachers produce sets that are also reasonable approximations of an optimal teaching set can be devised based on these observations. As will be demonstrated within a specific domain in Sec. 9.6.1.1, good teaching examples produced by this algorithm are in line with the *borderline* heuristic that seems intuitive for a nearest neighbor learner.

Sec. 9.1 described three ways of formulating teaching heuristics and exemplified three

problem domains suitable for each type. This section describes three experiments conducted to evaluate the effectiveness of teaching instructions in these teaching problems on specific domains.

9.2 *Experiment 9: Instructed teaching of tasks*

Our first experiment in Chapter 4 compared unguided human teaching of toy object categories represented as conjunctions to optimal teaching. This experiment explores using the optimal teaching algorithm described in Sec. 9.1.1.1 as teaching heuristics for humans¹.

9.2.1 Experimental Design

This experiment involves the concept learning with discrete features and uses the algorithm described earlier in Sec. 3.2.1. The domain in this experiment is the six-feature toy objects domain. The concepts are taught to a virtual version of the Simon robot through a Graphical User Interface shown in Fig. 43. The setup is similar to its physical counterpart; it involves one of each object part in Simon’s workspace. The object parts are dragged to the demonstration area using the mouse. The constructed object in the demonstration area is labeled or tested using the buttons at the top which correspond to the three speech commands used in Experiment 1. Participants are told to use the “I’m done” button when they think Simon has learned the concept. A “Clean up” button helps the participants place all the object parts back in their original locations.

Participants taught two of the concepts used in Experiment 5: HOUSE and ICE-CREAM, described in Table 1.

9.2.1.1 Procedure

Participants first watched a video that showed a screen capture video of the GUI, involving dragging object to the demonstration area and pressing on all buttons once to demonstrate

¹This experiment appears in [25].

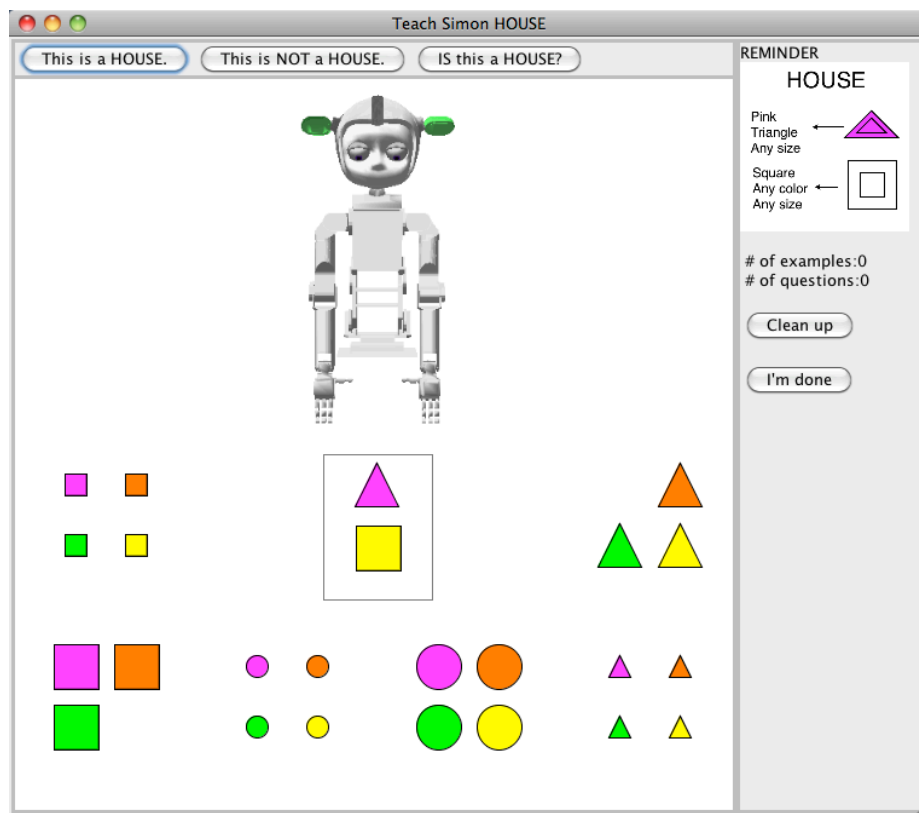


Figure 43: Graphical interface used for teaching Simon in Experiment 9.

their effect. The description of the concepts were given to the participant before starting to teach and a reminder was shown on the right of the interface. All participants taught the first concept without any specific teaching instructions. These are referred to as the *Baseline*. Then participants were prompted with specific instructions before teaching the second symbol. The two conditions had different instructions.

- *Motivated*: This time try to teach Simon with as few examples and questions as possible.
- *Guided*: This time we want you to use a specific teaching strategy: (1) First, show Simon an example of ice-cream. (2) Then, change everything that does not matter for ice-cream, and show another ice-cream. (3) Then show examples of objects that are not ice-creams, but that differ from ice-cream by only one property. Change the properties that matter one-by-one.

The experiment has a between-groups design. The instructions were repeated to the participants by the experimenter and their questions were answered. A reminder of the instructions was also shown on the right part of the interface.

9.2.2 Findings

This experiment was completed by 20 participants, 10 in each condition. The same metrics of evaluation as Experiment 1 are used. These involve (i) quality of teaching, (ii) speed of teaching and (iii) similarity to optimal teaching strategy which are described in detail in Sec. 4.1.1.3. The following observations are made.

Optimal strategy not seen without teaching heuristics. The distribution of good and bad examples does not change considerably in the *Motivated* condition (Fig. 44(b)) as compared to the *Baseline* condition (Fig. 44(a)). This shows that teachers cannot easily come up with a more optimal strategy on their own even when motivated to be more efficient.

Table 26: Summary of all teaching performance metrics in Experiment 9.

Metric	Baseline	Motivated	Guided	Comparison	Optimal
Quality of teaching					
$\%(V =1)$	40% (8/20)	20% (2/10)	30% (3/10)	N/A	N/A
$\%(F=1)$	90% (18/20)	70% (7/10)	90% (9/10)	N/A	N/A
$ V $	2.05 <i>SD=1.23</i>	8.10 <i>SD=11.68</i>	2.00 <i>SD=0.94</i>	$t(18)=1.65, p>.05$	1
F	0.97 <i>SD=0.10</i>	0.82 <i>SD=0.32</i>	0.94 <i>SD=0.19</i>	$t(18)=-1.03, p>.05$	1
Speed of teaching (excluding tests)					
$ T $	9.88 <i>SD=4.02</i>	6.00 <i>SD=0.00</i>	7.00 <i>SD=1.73</i>	$t(3)=-0.77, p>.05$	5
$ T _F$	6.00 <i>SD=2.22</i>	4.86 <i>SD=1.95</i>	3.00 <i>SD=2.12</i>	$t(14)=1.80, p>.05$	2
$avg(V)$	41.27 <i>SD=32.80</i>	26.22 <i>SD=9.43</i>	18.36 <i>SD=3.52</i>	$t(18)=2.47, p<.05^*$	15.80
$avg(F)$	0.42 <i>SD=0.20</i>	0.52 <i>SD=0.20</i>	0.72 <i>SD=0.17</i>	$t(18)=-2.41, p<.05^*$	0.82
Speed of teaching (including tests)					
$ T_t $	16.38 <i>SD=8.14</i>	19.50 <i>SD=19.09</i>	7.67 <i>SD=2.52</i>	$t(3)=1.16, p>.05$	5
$ T_t _F$	8.28 <i>SD=6.29</i>	7.29 <i>SD=4.57</i>	3.11 <i>SD=2.42</i>	$t(14)=2.36, p<.05^*$	2
$avg_t(V)$	43.84 <i>SD=32.96</i>	29.72 <i>SD=11.41</i>	18.44 <i>SD=3.51</i>	$t(18)=2.99, p<.01^*$	15.80
$avg_t(F)$	0.37 <i>SD=0.25</i>	0.45 <i>SD=0.26</i>	0.72 <i>SD=0.17</i>	$t(18)=-2.79, p<.05^*$	0.82

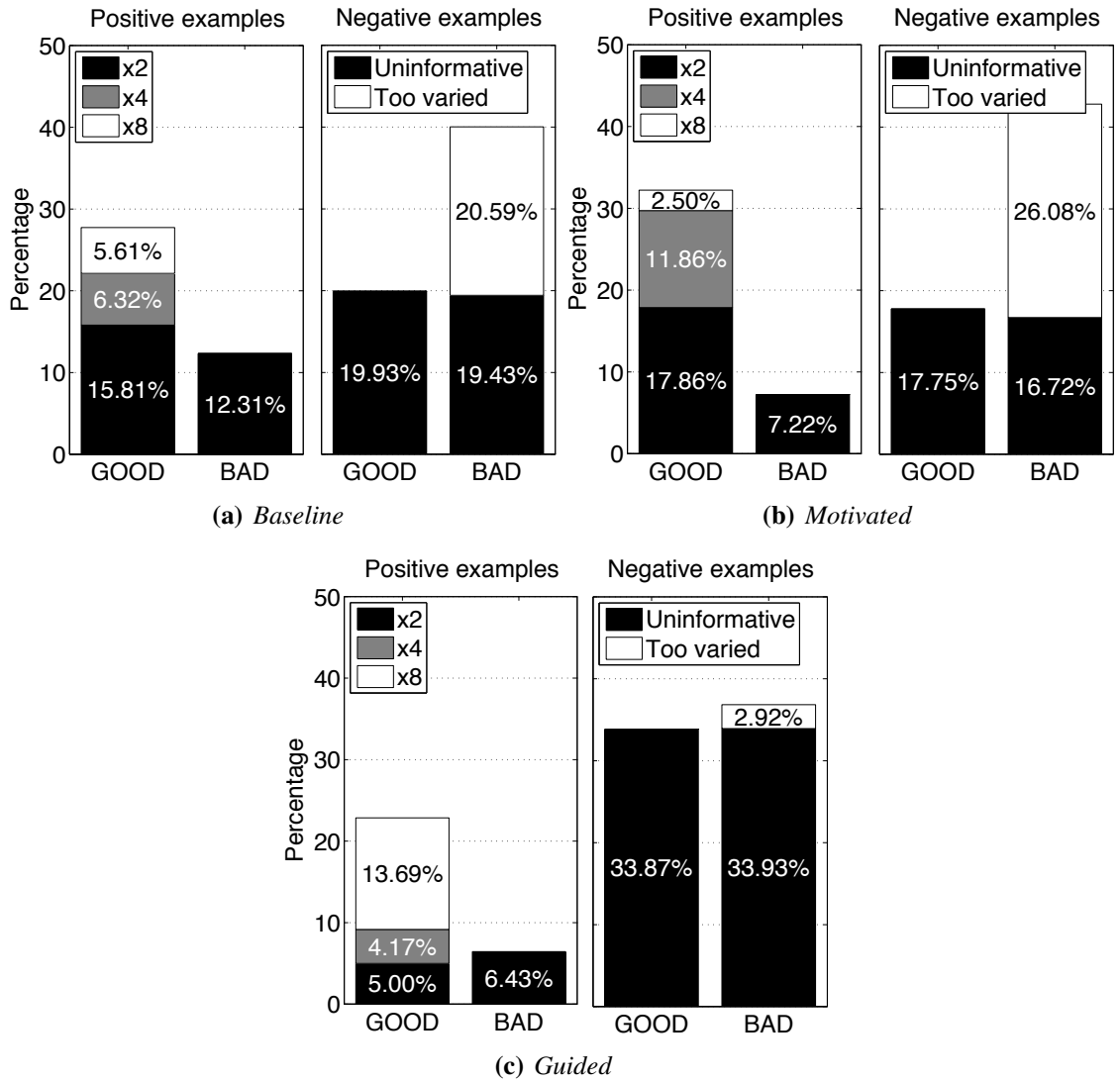


Figure 44: Distribution of *good* and *bad* positive/negative examples given by participants in Experiment 9. For good examples, breakdowns of how much of the version space was pruned is shown (*i.e.* by a factor of 2, 4, or 8). Bad negative examples are broken down as uninformative or too varied.

Table 27: Wall clock time (in seconds) spent on an example or test (t_{avg}), expected total time to get to $F = 1$, $t_{avg} * |T_t|_F$, and number of tests used by participants in Experiment 9.

Metic	Baseline	Motivated	Strategy
t_{avg}	10.43 (SD=4.34)	11.01 (SD=5.47)	18.06 (SD=4.75)
$t_{avg} * T_t _F$	91.04 sec	97.70 sec	55.94 sec
# of tests	9.10 (SD=6.70)	7.90 (SD=8.62)	6.10 (SD=5.92)

Optimal strategy is difficult to grasp. When successfully adopted, the optimal strategy lets the teacher exactly identify the target concept with five examples. But in the *Guided* condition only one person out of ten could teach with five examples. Most participants were not able to fully adopt the strategy. People usually failed in giving the correct set of negative examples when following the strategy. While their negative examples were correctly chosen to have only one relevant feature different from a positive example, they failed at varying the feature that was different. Instead they varied the value of the feature that was different resulting in uninformative examples. As a result, the *Guided* condition has a high number of uninformative negative examples (Fig. 44(c)).

Final performance not improved by teaching heuristics. Compared to the baseline, fewer people achieve exact identification ($|V|=1$) after the instructions in both groups. The final $|V|$ and F are worse than the baseline after the additional instructions in both conditions. The observed reason is that in the *Motivated* condition participants stop teaching too early, trying to be efficient and in the *Guided* condition most fail to correctly follow the strategy. On the other hand, in the baseline condition participants strongly rely on testing the learner extensively to fill any gaps in its knowledge.

Convergence rate improved by teaching heuristics. While exact identification is not achieved, the average performance in the first five steps of teaching ($avg(|V|)$, $avg(F)$, $avg_t(|V|)$, $avg_t(F)$) is better for both groups after the additional instructions. Teaching in the *Guided* condition is significantly faster than the *Motivated* condition.

The average time taken for providing examples or tests (t_{avg}) is larger when participants receive additional instruction (Table 27). In the *Guided* condition teachers were slower because they had to repeatedly go over the strategy and produce examples accordingly. In the *Motivated* condition they spent more time considering the informativeness of each example to be more efficient. In addition, for both conditions there is the extra mental load of trying to remember what they have already shown.

Even though participants were slower in providing examples in the *Guided* condition, they needed less examples and they needed to ask less questions. As a result, the expected total time ($t_{avg} * |T_i|_F$) is much smaller in the *Guided* condition (Baseline: 91.04sec, Motivated: 97.70sec, Guided: 55.94sec).

Example distribution more like optimal. The percentage of highly informative positive examples (x8) is much higher in the *Guided* condition (Fig. 44(c)). Most participants were able to successfully adopt the teaching heuristic for positive examples (7 of 10) and were able to teach the irrelevant features with fewer positive examples. As a result, the number of examples and tests required to achieve $F=1$ is significantly smaller in the *Guided* condition than the *Motivated* condition. This is because the optimal strategy reaches $F=1$ after the first 2 positive examples and most participants were correctly following this part of the strategy. Additionally, the *Guided* condition involved less tests on average (Table 27).

The number of uninformative positive examples went down in both groups (Fig. 44(b) and 44(c)). The percentage of negative examples with little information is smaller in the *Guided* condition (Fig. 44(c)). Participants understood that negative examples should have small variance from positive examples to be maximally informative.

Overall, the experiment demonstrates that teaching heuristics can have an important positive impact on human teaching. In this experiment, participant got much faster when they were asked to follow the optimal teaching algorithm. However, parts of the instructions were less intuitive. In particular, people intuitively gave optimal positive examples, but had trouble with optimal negative examples. This points to an ideal combination of

teaching heuristics and queries. As demonstrated in Chapters 5, 6 and 8, queries are particularly good in guiding humans to complete teaching. Having both mechanisms has the potential to combine the speed gains of teaching heuristics, and the coverage gains of queries. This idea is further explored in Chapter 12 in a follow-up to this experiment.

9.3 *Web-based experiments*

Experiment 9 produced promising results on the benefits of teaching heuristics. However, the results are limited – the experiment involved one particular concept class, conjunctions, and particular concepts that involved a fix number of relevant features. To further support the utility of teaching heuristics, a number of larger scale experiments are needed to explore the effects of certain parameters (*e.g.* size of the state space, specificity of the concept) and confirm results in different concept classes that were discussed earlier in Sec. 9.1. For this, a series of web-based experiments were conducted. Before moving on to the individual experiments, the experimental design that is common to all of them is presented in the following.

Three web-based experiments were conducted; one in each of the domains discussed in Sec. 9.1. These experiments involve a human teaching a particular concept in the concept classes from each domain to a virtual agent. The target concepts are pre-specified and explained to the teacher prior to the teaching session. This is done to ensure comparability of results, but is analogous to a scenario where a human teacher decides to teach their own concepts to an agent. A teaching session involves sequentially generating instances in the sample space and providing category labels for them. All classification tasks are binary. The teacher is allowed to *test* the learner at any time during the interaction.

All experiments have a between-groups design, *i.e.* each participant teaches only one concept in one of the following conditions:

- *Natural*: The teacher is motivated to teach efficiently but not given instructions about how to teach.

- *Guided*: The teacher is given a set of additional instructions that guides their teaching.

Note that the *Natural* condition in this experiment is equivalent to the *Motivated* condition of Experiment 9. The experiments share a common webpage layout with three parts separated by horizontal lines. The top part has general instructions to the participant. It motivates the teaching task and describes the concept that the participant will be teaching. The middle part is a Java applet that lets the teacher configure examples, and submit them to the learner or test the learner with the configured example.

The bottom part of the webpage has questions to be answered by the teacher after completing the teaching task. In the *Natural* condition two open-ended questions were given. The first asks the participants to characterize their teaching strategy and the second asks their criteria for ending teaching. In the *Guided* condition, the question asks if the teaching instructions were easy to follow and which parts they thought were most challenging.

The participants were solicited and compensated through Amazon Mechanical Turk²; a popular crowd-sourcing platform. Mechanical Turk has been successfully used for user studies in the past [77] and there have been some efforts towards identifying the demographics and other characteristics of its workers [115]. Mechanical Turk was found to be an appropriate platform based on free-form responses elicited from the workers in a pilot study. The responses to the open ended questions were also used for filtering the data obtained in experiments. Data from participants who did not provide any open-ended response, or whose responses were not comprehensible were removed and a new participant was recruited to replace it. An individual was allowed to take part only once in any of our experiments. Participants were paid \$0.50 in Experiments 10 and 12, and \$0.40 in Experiment 11, for their participation.

²www.requester.mturk.com

Table 28: Description of the conjunction concepts in the Chernoff faces domain used in Experiment 10.

Name	n	r	$ \tau^* $	Description
Angry Face	16	2	4	$f_8 \bar{f}_{12}$
	8	2	4	$f_8 \bar{f}_{12}$
Humpty Dumpty	16	8	10	$\bar{f}_0 \bar{f}_1 \bar{f}_2 \bar{f}_4 \bar{f}_5 \bar{f}_9 \bar{f}_{11} \bar{f}_{14}$
	8	4	5	$\bar{f}_0 \bar{f}_2 \bar{f}_4 \bar{f}_{11}$

9.4 Experiment 10: Instructed teaching of face categories

This experiment investigates the effect of the teaching heuristics that are based on a provably optimal teaching algorithm (Sec. 9.1.1.1)³.

9.4.1 Experimental Design

9.4.1.1 Domain

The experiment involves conjunctions in the *Chernoff faces* domain [38]. This is a visual representation of multi-dimensional data, where each dimension is a facial feature, and a face instance is a point in this multi-dimensional space. Faces with 16 binary features f_i are used. Fig. 45(a) shows two instances with opposite values for all features $x_0 = \bar{f}_0 \bar{f}_1 \dots \bar{f}_{15}$ and $x_1 = f_0 f_1 \dots f_{15}$. A face concept is a conjunction of feature values for a subset of the features. For example an angry face has two relevant feature values: *brow shape* with outer ends raised, and *mouth shape* turned down.

In this experiment, in addition to the teaching mode (*Natural* versus *Guided*), the *number of features* (n) of the concept class and the *number of relevant features* (r) of the concept are varied between groups. Two concepts from the class C_{16} with $r=2$ and $r=8$ and two from C_8 with $r=2$ and $r=4$ are used. Eight features of the face are kept constant for C_8 . Table 28 summarizes the four concepts used in the experiments and gives their teaching dimension.

As a learner for conjunctions, the halving algorithm based on version spaces is used [85].

³This experiment appears in [28].

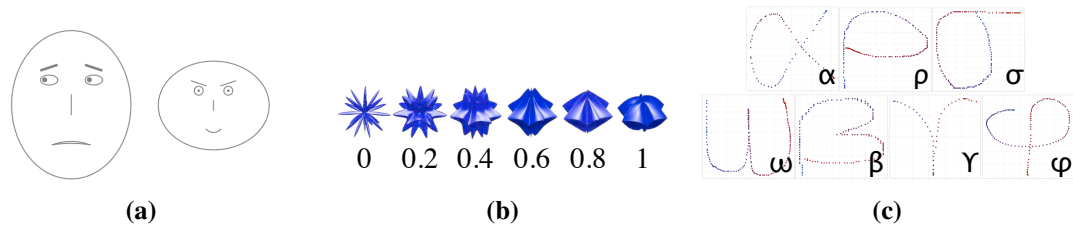


Figure 45: Illustrations of the three domains used in Experiments 10, 11 and 12. (a) Two extreme faces in the Chernoff face domain with 16 binary features. (b) Instances from the shape domain. (c) Examples from the mouse gesture domain.

Unique identification corresponds to being left with a single hypothesis in the version space. In order to have an accuracy that is consistent with unique identification, the learner responds to a test only when the whole version space agrees on its label. Otherwise the learner will say “Not sure” even if the majority of the version space predicts one label.

9.4.1.2 Interface

The applet has all facial feature names listed on the left, with the two possible values of each feature shown as icons. At any time one of the two feature values is *selected*, and the selected value is indicated by a blue square around the corresponding icon. The teacher can change the selected value of a feature by clicking on the icon of the opposite value. The selection of features is randomized when the applet is loaded. The face that corresponds to the currently selected list of feature values is depicted at the center of the interface. Relevant features of the concept that is being taught are highlighted as light blue to help the teachers.

There are two buttons below the face. One lets the teacher submit the current face as an example to the learner. The label of the example is automatically changed to the correct label. This avoids teachers submitting wrong examples and assists them about what the currently configured example is as they alter the feature values. The second button allows the teacher to test the learner with the currently configured face. The learner is depicted as a robot, and it responds to questions with a speech bubble (e.g. , “Not an angry face”).

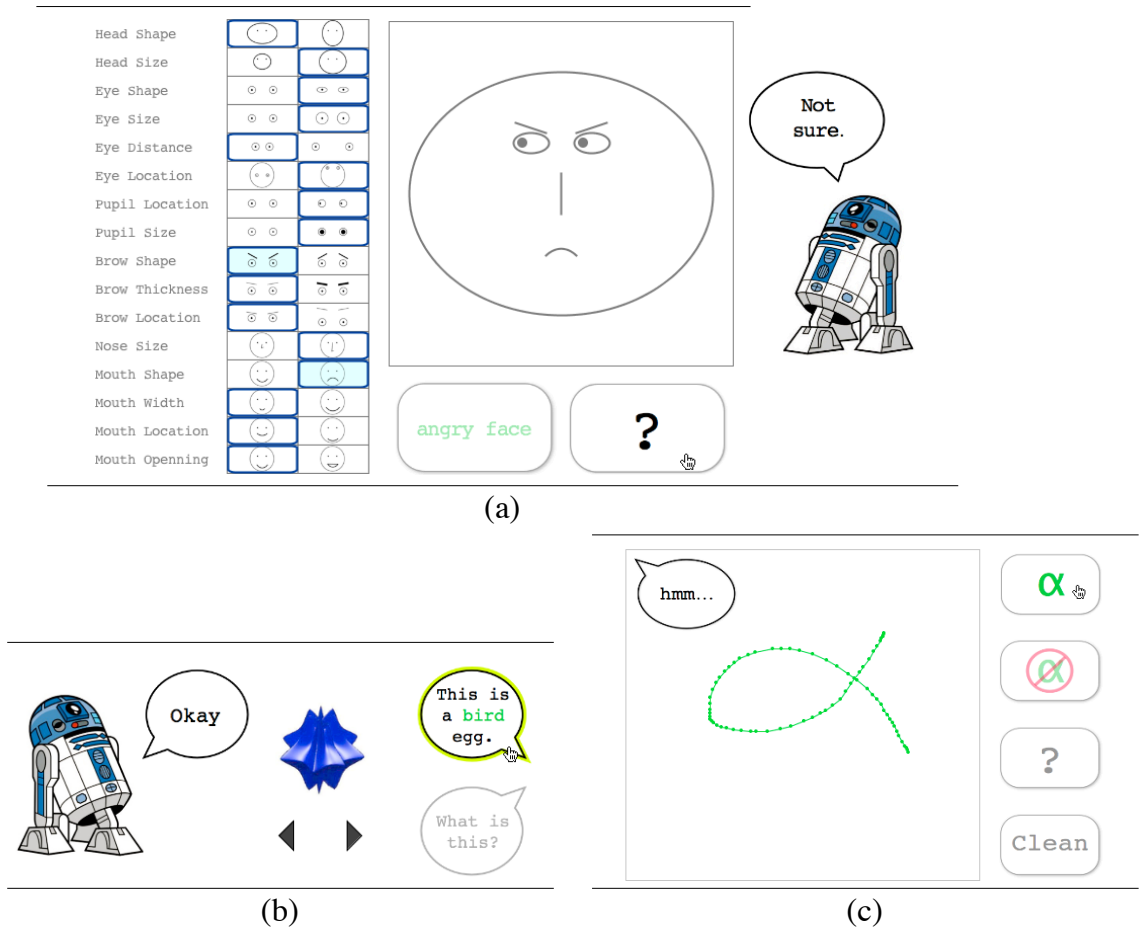


Figure 46: Screenshots of the teaching interfaces used in Experiments 10, 11 and 12. (a) Interface for teaching conjunctions in the Chernoff faces domain (16 binary features, *Angry face* concept) (b) Interface for teaching a linear classifier in 1-D in the *Shapes* domain. (c) Interface for training a nearest-neighbor classifier to classify mouse gestures.

The learner also responds to provided examples by saying “hmm...”, as a feedback to the teacher that the example is successfully received. A screenshot of the teaching interface is shown in Fig. 46(a).

9.4.1.3 Instructions to participants

The instructions motivate the teaching task with a scenario: Participants are told that a robot, R2D2, will go to a planet to deliver medication to certain humans to be recognized by their faces. Then, the n facial features and the particular face category (c_r) that the robot needs to learn are described.

In the *Natural* condition, teachers are motivated to teach optimally. They are asked to not give redundant examples and do redundant tests, and are told that an award will be given to the teacher who trains the best learner with the fewest examples and tests. They are told to stop teaching when they think the concept has been learned.

In the *Guided* condition, teachers are asked to follow the provided teaching instructions based on the algorithm given in Sec. 9.1.1.1. The description of the algorithm to human teachers was iterated to improve understandability based on Experiment 9. As an example, the teaching instructions for teaching *angry face* is as follows:

- Step 1: Configure one angry face example and show this to R2D2.
- Step 2: Starting from the example from Step 1, change all properties that don't matter for angry face to the opposite value. This results in another angry face maximally different from the previous one. Show this example to R2D2.
- Step 3: Starting from the example from Step 2, change one of the properties that matter for angry face to the opposite value. This results in a non-angry face that is minimally different from an angry face (by just one property). Show this example to R2D2.
- Change the property from Step 3 back to the right value. Then repeat Step 3 for other properties that matter for angry face. So, show different examples of NOT angry face which differ from an angry face by just one property.

Note that the instructions are not specific to *angry face*, the guidance for *Humpty Dumpty* are exactly the same other than the category name. Fig. 47 shows example optimal teaching sequences produced with the algorithm described in Sec. 9.1.1.1 for two of the concepts used in this experiment.

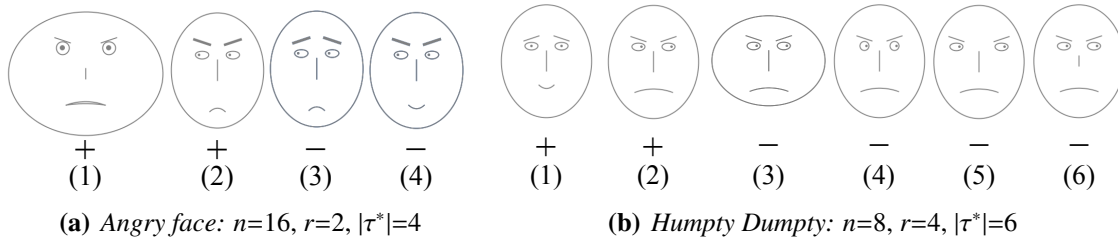


Figure 47: Example optimal teaching sequences for two conjunction concepts in the *Chernoff faces* domain used in Experiment 10.

9.4.2 Findings

Findings from the experiment are reported in terms of the performance of the taught learner and the participants' self assessment of their teaching. The responses to the open ended questions were coded independently by the two experimenters, based on several criteria shown on Tables 29 and 30.

The experiment was completed by 160 participants, 20 in each condition. Fig. 48 compares the average accuracy achieved with the example sequences provided in the *Natural* versus *Guided* teaching conditions across the four different target concepts⁴. For comparison the learning curve of the optimal teaching sequence described in Sec. 9.1.1.1 is also shown. The following observations are made.

Natural teaching is rarely optimal. The first observation is that the results from Experiment 1 are supported by this experiment. In the *Natural* condition there was a single instance of optimal teaching. Most teachers in this condition were largely sub-optimal. Only 17 out of the 80 participants were eventually able to provide sets that uniquely identify the target concept.

From the participant's description of his teaching strategy, it can be seen that the single occurrence of optimal teaching was not accidental; the description accurately matches the

⁴The reason that accuracy starts at zero is that the learner does not respond to tests when it is not 100% certain (see Sec. 9.4.1.1). Zero accuracy does not mean that the learner classifies all instances wrongly; instead it means that it classifies nothing correctly. Due to this behavior, the learner never makes a mistake when it responds to a test.

optimal teaching algorithm:

“Showed him examples of everything changing but the key elements that create the angry face, then kept all of the other variables constant and just changed the parts that make the face not angry or angry to show the difference between the two.”

As shown in Table 29 one other participant’s described strategy matches the guidance, though his teaching sequence was not optimal. In addition, 18 participants partially described the optimal teaching algorithm, where *partially* is defined as giving two or more steps of the algorithm. An example is:

“I showed him the correct combination first, then changed each correct feature one by one, until I had shown him all the incorrect combinations.”

This shows that even though the optimal teaching sequence is unlikely to occur spontaneously without teaching heuristics, humans have an intuition for what examples would be informative to the learner. A different strategy that was observed in this experiment is presenting examples only when the learner responds “Not sure” when tested with it. 16 participants in the *Natural* condition mentioned using tests as part of their strategy. Such tests can allow the teacher to build a better mental model of what the learner knows and how it learns. Consequently, the teacher presents more informative examples.

Teaching heuristics improves teaching. In the *Guided* condition, 41 participants out of 80 uniquely identified the target concept with their examples, 17 of them being optimal. The average accuracy after $|\tau^*|$ examples is 0.82 (SD=0.31) as compared to a 0.39 (SD=0.32) in the *Natural* condition ($t(133.9)=-7.98, p<.001$, in a student’s *t*-test).

Parts of guidance are less intuitive. Even though guidance improves learning as compared to the *Natural* condition in general, it could be followed perfectly by only 17 participants out of the 40. The rest of the participants had some problem adhering to the teaching

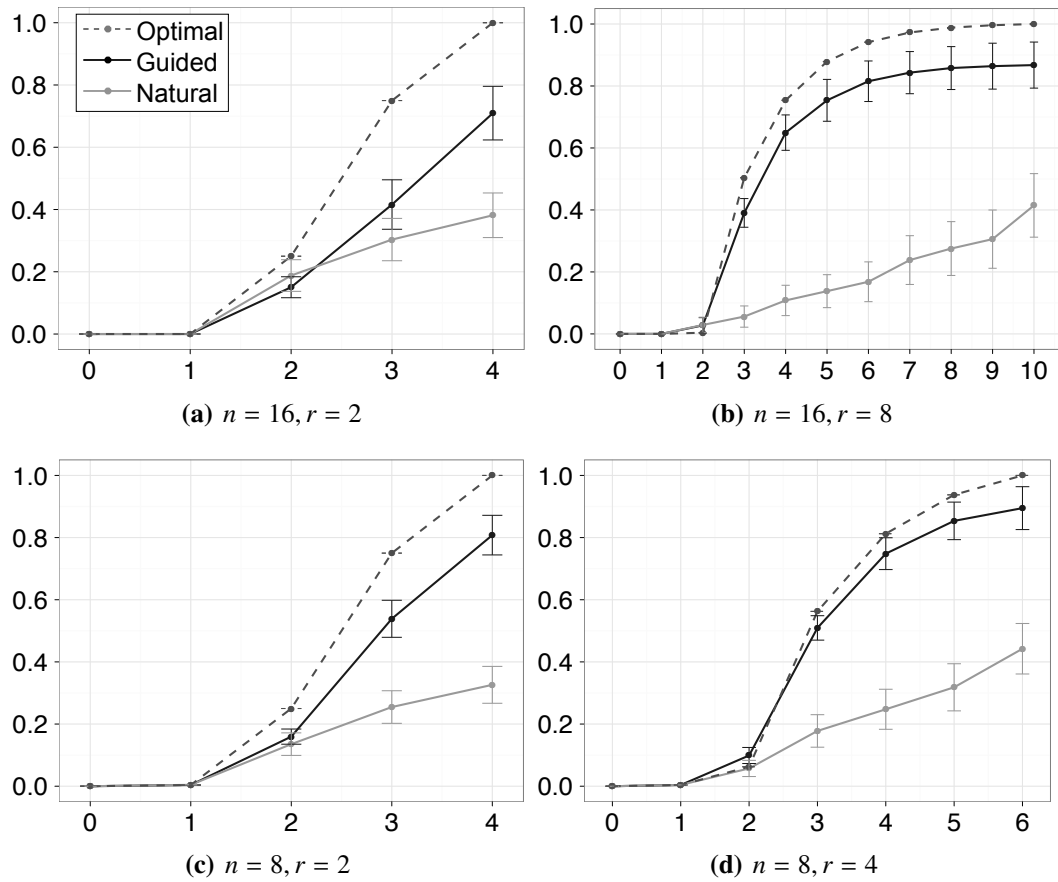


Figure 48: Results for teaching conjunctions in Experiment 10. The graphs show the progression of the average accuracy on the whole sample space over the first $|\tau^*|$ examples for human teachers teaching the four different target concepts. Each graph shows the accuracy curves for *Natural* and *Guided* human teaching as well as the optimal teaching sequence. The error bars denote standard error.

	Experiment 10	Experiment 11	Experiment 12
<i>Fully explained guidance</i>	2	4	4
<i>Partially explained guidance</i>	18	5	2
κ	0.94	0.97	0.97

Table 29: Responses to the open-ended question in the *Natural* condition for Experiments 10, 11 and 12. The question asks the participants to characterize their teaching strategies. The responses are coded in terms of their overlap with the guidance given in the *Guided* condition. The responses are coded independently by the two authors as one or none of the given rows; the reported numbers are for responses where both coders agreed. Inter-coder agreement is reported with Cohen’s Kappa (κ).

heuristics. In the open-ended question asking participants whether they had any problems following the provided guidance, a majority of the participants (51 out of 80) said that they had no problems. This means that some participants made mistakes while applying the guidance without realizing it. Nine participants expressed difficulties in following parts of the guidance, all of which pointed to the part of the guidance related to giving negative examples (*i.e.* starting at Step 3 of the guidance given in Sec. 9.4.1.3). Some examples are “While repeating step 3 readjusting the face to the original state was difficult” or “It was a little hard to remember which ones had I switched already”.

It is more difficult for humans to be optimal in larger state spaces. Fig. 48 shows that the learning curves for the *Guided* condition are closer to optimal for $n = 8$, for both values of r .

Teaching instructions have more impact when the optimal teaching sequence is longer. When r has a larger value with respect to n we see a larger difference between the learning curves of the *Natural* and *Guided* conditions (*e.g.* Fig.48(b) and 48(d)).

	Experiment 10	Experiment 11	Experiment 12
<i>No problems</i>	51	14	15
<i>Problem understanding guidance</i>	6	2	1
<i>Problem following guidance</i>	9	0	0
κ	0.87	0.98	0.97

Table 30: Responses to the open ended question in the *Guided* condition for Experiments 10, 11 and 12. The question asks the participants whether they had any problems following the guidance, and if so, what they found most challenging. The responses are coded independently by the two authors as one or none of the given rows. Inter-coder agreement is reported with Cohen’s Kappa (κ).

9.5 Experiment 11: Instructed teaching of linear separators

This experiment investigates natural and guided teaching based on the empirically optimal teaching scenario discussed in Sec. 9.1.2.1⁵.

9.5.1 Experimental Design

9.5.1.1 Domain

The teaching problem is explored using the *Shapes* domain from [35, 142], which consists of so-called super-shapes parametrized by a single real number in the range [0,1]. A few example shapes across the range are shown in Fig. 45(b). 100 shapes were uniformly sampled to generate the example set S_c used in the teaching experiment.

Target concepts are counter-balanced across participants; each person teaches a linear separator with either $w = 0.2$ or $w = 0.8$. The two categories for the shapes are *snake eggs* (negative) versus *bird eggs* (positive).

The learner places the threshold at the midpoint of the rightmost negative example and the leftmost positive example from the set of examples that it has been presented. Unlike the learner in the first experiment (Sec. 9.4.1.1), this learner never says “Not sure”; even before it has received any examples, it assumes that the threshold is at $w = 0.5$. This is

⁵This experiment appears in [28].

analogous to starting with one positive example at the positive extreme, and one negative example at the negative extreme.

Human teaching of a threshold in 1D was also studied by [75]. The domain used in their experiment was the graspability of a set of objects varying from small (*e.g.* toothbrush) to large (*e.g.* furniture).

9.5.1.2 Interface

A screenshot of the interaction interface is shown in Fig. 46(b). The interface has the image of a sample shape at the center. This sample can be changed using increase or decrease arrows displayed below the sample shape. A single click on the arrows changes the displayed shape to the next sample. A continued click allows a faster browsing through the range of shapes. The initial shape that is displayed when the applet loads is randomized. A button to the right of the shape allows the participant to submit the shape as an example to the learner. This button is a clickable speech bubble that has the text “This is a snake egg” or “This is a bird egg.” The label is automatically assigned based on the currently configured shape. A second button, in the form of a speech bubble that says “What is this?” lets the teacher test the learner with the currently configured shape. The learner is again depicted as a robot and responds to new examples and questions with a speech bubble.

9.5.1.3 Instructions

As in Experiment 10, the teaching task is motivated with a scenario. Participants are told that the shapes correspond to bird and snake eggs and that they will teach that bird eggs are smooth while snake eggs are spiky. The target concept is given to the participants as a picture of the spikiest bird egg. They are asked to teach this concept to R2D2 whose mission is collect a bird eggs on a foreign planet.

Participants in the *Natural* condition are motivated to teach well. In the *Guided* teaching condition, participants receive the following teaching instructions:

- Step 1: Change the sample egg to the bird egg that is closest to being a snake egg and show this example to R2D2.
- Step 2: Change the sample egg to the snake egg that is closest to being a bird egg and show this example to R2D2.

9.5.2 Findings

This experiment was completed by 40 participants, 20 in each condition (*Guided* versus *Natural*). Fig. 49(a) shows the average accuracy achieved by the examples provided by the teachers over the first two examples in both conditions, as well as the empirically optimal teaching sequence of two examples. The following observations are made.

Natural teaching is rarely optimal. Although the teaching strategy is very simple in this experiment, without teaching instructions, participants provided examples across the 1-D range. 16 out of the 20 participants eventually reached an empirical error of zero, resulting in an average of 0.98 (SD=0.03) final accuracy. However, they did so with an average of 5.44 (SD=2.85) examples. Only 2 people out of 20 gave the empirically optimal teaching sequence in the *Natural* condition. Note that this is a higher rate (10%) as compared to the one in Experiment 10 (1.25%) reported in Sec. 9.4.2.

Even though they are not optimal, the examples given by human teachers are not completely random. A common pattern observed in the examples provided by participants is a balance of positive and negative examples. Due to the target concepts used in this experiment, a uniform sampling of the sample space would result in a 1:4 ratio of positive and negative examples. In the example sets provided by the participants we see an average ratio of 1:1.1. This bias in the example set results in a learned threshold that is closer to the true threshold.

Similar to [75] different teaching patterns are observed; some start from the extremes of the state space while others start closer to the boundary. The two strategies are also evidenced by the participants' descriptions of their own strategy: "Show the robot the

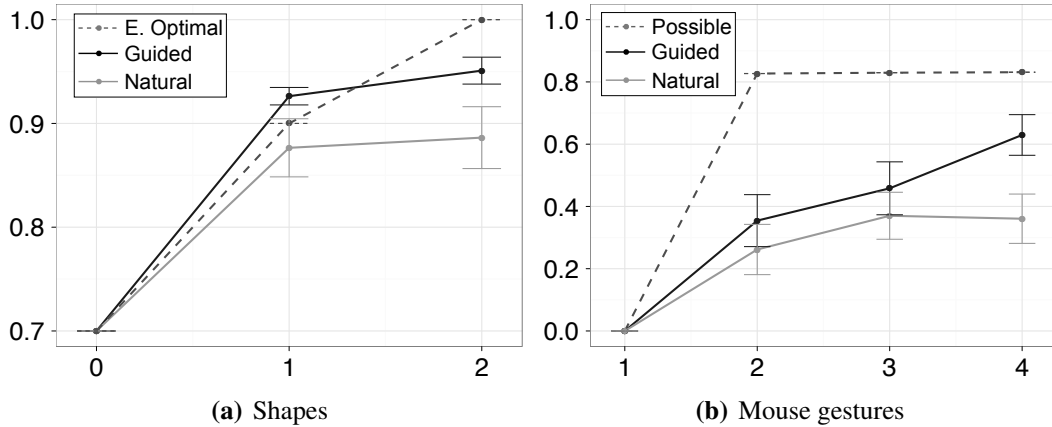


Figure 49: Accuracy of the (a) linear separators in \mathbb{R}^1 (Shapes) in Experiment 11 and (b) nearest-neighbor classifiers (Mouse gestures) in Experiment 12 over the first few examples of the teaching sequences provided by the human teachers in the *Natural* and *Guided* conditions. (a) Compared to the empirically optimal teaching sequence. (b) Compared to the average of teaching sequences produced by the greedy approximate algorithm (indicated as *Possible*). Error bars indicate standard error.

extremes of the spectrum and the most similar snake and bird eggs” versus “I used the reference points right around the point when it changed from bird to snake”.

Teaching heuristics improve teaching. The error rate is marginally smaller in the *Guided* condition ($t(24.47)=-1.98, p=.058$) after the first two examples. 5 participants out of the 20 provided the empirically optimal teaching sequence, which is not a large improvement over the *Natural* condition. While this means that the participants did not provide the exact two border examples as the first and second example in their sequence, the improvement observed in Fig. 49(a) confirm that they provided examples closer to the boundary⁶.

9.6 Experiment 12: Instructed teaching of mouse gestures

This experiment investigates natural and guided teaching for Nearest Neighbor (NN)

⁶In this figure we observe that, after a single example, the learners taught in the *Guided* condition have a better average accuracy than the optimal teaching sequence. The reason for this is that, when the learner has no examples from one of the two classes, it places the threshold between the given example closest to the opposite class and the extreme of the range to the side of the opposite class. As a result, in the case of the true threshold being 0.8, if the first example given by the teacher is around 0.6 the learned threshold will be close to the true one after a single example.

classifiers based on the teaching heuristics developed from an approximately optimal teaching algorithm as discussed in Sec. 9.1.3.1⁷.

9.6.1 Experimental Design

9.6.1.1 Domain

The scenario considered for this problem is training a computer mouse gesture recognition system. A mouse gesture is represented as a time series of 2D points. Gestures can be of varying length. As a dissimilarity measure between two gestures, the warping distance obtained from applying dynamic time warping between two gestures is used. For simplicity, only the binary classification of whether a gesture belongs to a category or not is considered. This involves gestures for drawing Greek letters (Fig. 45(c)), where alpha is used as the positive category.

To evaluate this teaching problem, a test corpus for seven different gestures were collected from 20 participants. All participants provided 5 examples for each gesture. Sample gestures are shown on Fig. 45(c).

In order to get a better idea about the teachability of gestures, and verify the teaching heuristics for the domain, the greedy approximate teaching algorithm described in Sec. 9.1.3.1 was ran and the produced teaching sets were inspected. Based on this algorithm, the smallest error rate that can be achieved with a teaching set of length 2 is around 0.13 and there are 13 alpha-non-alpha pairs that achieve this error rate. Three example non-alpha gestures from these pairs are shown in Fig. 50(a). The examples have characteristics similar to an alpha gesture and fit well to the description of a *borderline* example. The performance of the approximate teaching algorithm is used as a benchmark for human teaching.

⁷This experiment appears in [28].

9.6.1.2 Interface

The interaction interface is shown in Fig. 46(c). It has an area for drawing the gesture, and the trace of the gesture is visualized as it is being drawn. Two buttons let the teacher submit the drawn gesture as an example of α or not- α . A third button lets them test the learner with the current gesture. A fourth button allows for cleaning up the drawing area. The gesture is automatically cleaned up when submitted as an example, or when another gesture is performed on the drawing area. Therefore, each gesture is a single movement.

The interface for collecting the test set is similar. There is only one button for submitting the drawn gesture. The button indicates the gesture that is to be drawn, which is one of the seven Greek letters. After five examples of each letter the button automatically changes to the next letter. This interface also has the clean up button.

9.6.1.3 Instructions

The instructions tell the participant that there are seven possible gestures corresponding to drawing Greek letters with the mouse. They are told to teach the system to recognize whether a gesture is alpha or not.

To motivate good teaching in the *Natural* condition participants are told that the system that they train will be tested on a separate set of gestures and that the teacher of the best performing system will receive an award. In the *Guided* condition the borderline teaching heuristic is explained to the participant in the form of guidelines for choosing examples. The teaching heuristics are worded as follows:

- When you show examples of alpha gestures, vary them as much as possible. Try to show alphas that are borderline examples (*i.e.* almost not-alpha).
- When you show examples of not alpha gestures, try to make them quite similar to alpha. Focus on the gestures that look most like alpha and have a similar structure with alpha. In other words, make your not alpha examples borderline examples which are near-misses.

9.6.2 Findings

This experiment was completed by 40 participants (20 in each condition). Learning curves corresponding to the first few examples provided by the human teachers in the two conditions are given in Fig. 49(b). The learning curves are in terms of the accuracy on the collected test set described in Sec. 9.6.1.1. Both conditions are also compared to the performance achieved by the approximate teaching algorithm described in Sec. 9.1.3.1. The following observations are made.

Teaching heuristics improve teaching. The teaching heuristics help humans pick better examples. By visually inspecting the examples given by human teachers in the *Guided* condition, it can be observed that the notion of *borderline* is employed by the participants in different ways. Fig. 50(b) shows a few examples of non-alpha gestures given in this condition. Humans interpret the notion of *borderline-negative* examples as examples that differ from a positive example by a *few* and *relevant* features. The shown non-alpha gestures give some examples of features that are important for a gesture to be considered *alpha* from the human’s perspective. Although the features they modify to make an alpha gesture into a negative one are not directly captured in the learner’s representation of the gesture, these examples still provide improvement over natural teaching.

Nevertheless, some participants in the *Natural* condition realized the importance of *borderline* examples. For instance, one participant describes his strategy as follows:

“Give the machine a few examples of alpha, then a few examples of obviously not alpha, then test close to but not quite cases of alpha in order to fill the gap.”

This demonstrates that the heuristic is intuitive for human teachers, however it is less likely to be used without explicit instructions.

Both Natural and Guided teaching are sub-optimal. The performance of the learner is low in both conditions when compared to the performance of the approximate teaching algorithm. Note that algorithm is not optimal and it is restricted to examples from the test

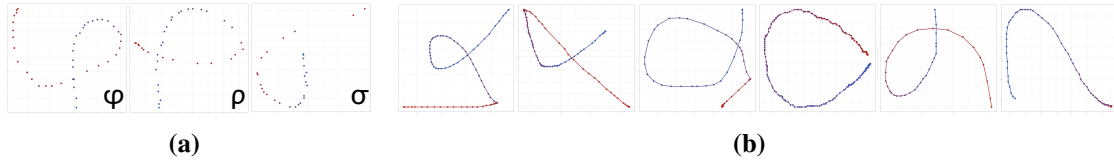


Figure 50: Additional results from the mouse gesture teaching domain used in Experiment 12. (a) Sample non-alpha gestures chosen by the greedy algorithm as part of the first two examples. (b) Non-alpha gesture examples given by participants using the *borderline* teaching heuristic.

set. It only produces an approximately optimal teaching sequence and it indicates a lower bound on the optimal performance (*i.e.* there might be better teaching sets). This shows that there is room for improvement in the human teaching heuristics for this class of learning problems.

9.7 Discussion

The scenarios considered in this section involve a human training an agent to perform a classification task by showing examples. While the particular concepts that humans might want to teach an agent can vary arbitrarily, what an *informative example* means for a particular learner remains constant. The experiments demonstrate that teaching heuristics given to a teacher about what informative examples are, positively impacts the examples that they present to the learner. These heuristics are not specific to a particular concept, but to a particular concept class and learning algorithm. Hence, teaching heuristics only need to be specified once for a particular learner.

The experiments supported earlier results from Chapter 4 that humans are usually not spontaneously optimal. This is partly due to not knowing how the learner learns. It cannot be expected that explaining how the learner learns (*e.g.* version spaces and the halving algorithm) would have resulted in more occurrences of optimal teaching, since inferring the optimal teaching algorithm for a concept class is difficult even for a theoretician. Another reason could be anthropomorphization, *i.e.* assuming that the learner learns like a human

would. For instance, one common strategy in the first experiment was changing features one-by-one rather than multiple irrelevant ones all at once. This makes sense since it might be difficult for a human learner to pay attention to *everything* that changes between two examples if multiple things changed. Similarly, in the second and third experiments giving *typical, obvious* or *easy* examples first rather than *borderline* or *difficult* ones was common, since this would better support human learning. This is consistent with the curriculum teaching approach which was also found to be more prominent in human teaching in previous studies [75, 93, 15]. One approach is to make learning agents that better accommodate these common teaching patterns in humans [132, 124], whereas this thesis looks at the extent to which human teaching can be influenced to accommodate how an agent learns.

The Experiments 10, 11, and 12 focus on teaching problems where the source of the teaching heuristics vary, from crisp to vague. The first leverages the existence of an efficient algorithm guaranteed to be optimal, and the second leverages the existence of an easy to browse dataset to teach from. The third case is when neither of these is available. In all settings, it is possible to devise teaching heuristics that provided an improvement over natural teaching. Notice that each is a special case of the next. For instance, the borderline heuristic is applicable to teaching conjunctions as well as teaching a threshold in 1-D. The guidance given in these experiments are in fact consistent with the borderline heuristic, only more specific. One could even argue that the borderline heuristic is a fallback for *any* discriminative learner, when no other helpful guidance specific to the learner can be extracted.

Even though teaching heuristics were useful in all three cases, it did not guarantee eliciting optimal teaching from humans. In some cases parts of the heuristics were easier to follow than others (Experiment 10), while in other cases they made the teacher closer to optimal but not perfect (Experiments 11 and 12). For instance in the case of conjunctions, the steps in which the teacher needs to provide multiple negative examples by changing one feature at a time was hardest follow for human teachers (Sec. 9.4.2). As discussed

in Experiment 9, this points towards a *mixed-initiative* learning approach that combines teaching instructions with active learning. In this problem, the *query by committee* strategy for active learning, results in queries that change one property at a time, once given a positive seeding example. This allows the learner to test the relevance of each dimension independently. Therefore an ideal mixed-initiative approach would be to allow the teacher to provide the first two positive examples that differ in all irrelevant dimensions and then allow the learner to make queries with this strategy. This combination would still be optimal, since the teacher would be implementing the first two steps of the optimal teaching algorithm, and the queries made by the learner would be equivalent to the rest of the optimal teaching algorithm. This would also reduce the amount of guidance given to the teacher to the parts that were most intuitive for them. Similarly, in the other domains, by giving better examples than they normally would, the guided teacher would be seeding an active learning algorithm such that the space that the learner needs explore is much smaller than it would be if seeded with random examples.

9.8 Summary

This chapter introduces teaching heuristics — instructions given to humans on how to teach. The chapter focused on classification problems. Teaching heuristics are based on a theoretical understanding of what constitutes an optimal teacher for a given learner. Three increasingly complex scenarios are considered.

- When available, teaching heuristics should explain the strategy of an algorithm that is proven to be *optimal*. This was illustrated with conjunctions.
- When there are no generic optimal teaching algorithms for the considered concept class, one can still construct an algorithm that chooses teaching examples optimally from a finite set of possible examples. These rely on a structured organization of the examples. Teaching heuristics for humans should explain the strategy of such

empirically optimal algorithms. This scenario was illustrated with linear classifiers in 1D.

- When there are no efficient algorithms that guarantee optimality, teaching heuristics can be based on *approximately optimal* algorithms. Often times, the strategy followed by such algorithms are not procedurally intuitive for humans. Instead, teaching heuristics should be based on observations of examples produced by these algorithms. This scenario was illustrated with nearest neighbor classifiers.

Teaching guidance based on these different scenarios were evaluated in four experiments. All experiments demonstrated the utility of teaching heuristics over unguided teaching, but were not adopted perfectly by all participants.

CHAPTER X

TEACHING HEURISTICS FOR SEQUENTIAL DECISION TASKS

The previous chapter demonstrated the benefits of teaching heuristics for increasingly complex teaching problems and could be extended to other similar teaching problems. In particular, the borderline heuristic based on approximate teaching considered in Experiment 12 is highly generic and can be useful for any discriminative learner. However, findings from these experiments might not directly extrapolate to characteristically different teaching tasks, such as training an agent who tries to maximize its reward in a Markov Decision Process. Such problems are common in robotics and a number of methods have been proposed for teaching new skills to robots in this formulation. Optimal teaching of such sequential decision tasks has not been previously addressed in the Algorithmic Teaching literature.

This section develops a framework that addresses the optimal teaching of sequential decision tasks, focusing on an Inverse Reinforcement Learning agent. An approximately optimal teaching algorithm is proposed in this framework. Then, the teaching examples produced by this algorithm are inspected. Based on observations from these teaching examples, teaching heuristics for training an IRL agent are developed and their utility is verified through another web-based experiment.

10.1 Optimal Teaching in MDPs

This section first presents preliminaries on MDPs and *Inverse Reinforcement Learning* (IRL) and then formulates approximately optimal teaching in this framework.

10.1.1 Inverse Reinforcement Learning

A standard MDP is defined as a five element tuple (S, A, P, R, γ) [129]. S and A are the state and action spaces, R is a reward function, P is a state transition model and γ is the discount factor. The goal of Reinforcement Learning (RL) is to find a policy $\pi : S \rightarrow p(A)$ that tells the learning agent what action to take in a given state such that its total reward is maximized. Both *deterministic* policies that map a state to an action and *stochastic* policies that associate a certain probability to each available action are considered. The stochastic policies are assumed to give equal probability to all optimal actions, and zero probability to sub-optimal actions.

A *value function* corresponds to the expected return for a state when following policy π : $V^\pi(s_0) = E_{\pi, s_0}[\sum_{t=0}^{\infty} \gamma^t R(s_t)]$, where s_t is the state reached at step t when the agent starts at state s and follows policy π . A *Q-function* is the value associated with taking an action in a state, and can be written in terms of the value function as $Q^\pi(s, a) = R(s) + \gamma E_y[V^\pi(y)]$, where y_a is the state reached by taking action a in state s .

In the RL framework, Learning from Demonstration involves an agent learning a policy by observing demonstrations given by a teacher, rather than interacting with the environment [8]. Two main approaches for achieving this are direct policy learning and IRL [103]. The former models the policy directly from the observed state-action pairs. The latter assumes that the teacher is trying to maximize a reward and models this reward based on the demonstrations. In IRL the reward function R is unknown and it cannot be sampled from the process.

In many situations the reward function is more accurately described as a combination of features. It can be assumed that the reward can be written as a linear combinations of features $R(s) = \sum_i w_i f_i(s)$ as in [2], where $f_i(s)$ is a function on the i th feature of state s . With this substitution the value function can be re-written as $V^\pi(s) = \sum_i w_i E_{s, \pi}[\sum_t \gamma^t f_i(s_t)]$.

The inner term in the second summation is known as the *feature counts* [2]. These are denoted with $\mu_i^{\pi, s} = E_{s, \pi}(\sum_t \gamma^t f_i(s_t))$. When the action on the initial state is a it is

represented by μ^{π, s_a} . Note that this represents the value of a state if the system follows policy π and the reward is $R(s) = f_i(s)$. Thus the value function is:

$$V^\pi(s) = \sum_i w_i \mu_{\pi, s, i} = \mathbf{w}^\top \bar{\mu}_{\pi, s}$$

In order to learn the reward function, an IRL agent uses the following intuition. If the teacher chooses a particular action a in state s , then action a must be at least as good as all the other available actions in s : $\forall b, Q^*(s, a) \geq Q^*(s, b)$. This can be re-written in terms of the value functions, and thus in terms of the feature counts, as follows:

$$\forall b, \mathbf{w}^\top \bar{\mu}_{\pi, s_a} \geq \mathbf{w}^\top \bar{\mu}_{\pi, s_b}. \quad (17)$$

This presents a constraint directly on the weight vector. With additional prior information, these constraints can be used to estimate the reward function. Several methods proposed in the literature, allow estimation of the reward and the policy from such constraints obtained from demonstrations [101].

10.1.2 Teaching IRL agents optimally

Based on the formulation of the IRL agent, an informative set of demonstrations is one that allows the learner to compute the relevant feature counts and infer the weights as accurately as possible. Therefore, the most informative demonstrations are the ones that reduce the uncertainty in the reward estimation.

A demonstration given by the teacher is a trajectory of state-action pairs. Assume that all state-action pairs from all trajectories provided by the teacher are pooled together in a demonstration set $D = \{(s_t, a_t)\}_{t=1}^M$. Based on Equation 17 the constraints placed by this demonstration set on available reward functions can be summarized as:

$$\forall (s, a) \in D, \forall b, \mathbf{w}^\top (\bar{\mu}_{\pi, s_a} - \bar{\mu}_{\pi, s_b}) \geq 0 \quad (18)$$

Note that these inequalities give a set of half-spaces defined by hyperplanes going through the origin in the space of weight vectors. The true weight vector lies in the intersection of these half-spaces. It is assumed that the weights are bounded within a hypercube

described by $(-M_w < w_i < M_w)$. These bounds will depend on the task and will provide a meaningful scale for the reward function.

The subspace described by the combined constraints is denoted as $C(D)$. Given the bounded space of hypotheses $C(D)$, a measure of uncertainty is needed for comparing alternative demonstrations. While a number of measures are possible (see [79] for an analysis), the one used here is the volume of the space of possible weights. This volume is estimated using a sampling approach. Weight vectors from the hypercube $(-M_w < w < M_w)$ are sampled and the ones that are within $C(D)$ are counted. This gives an indicator of the uncertainty in the reward estimation, denoted as $G(D)$.

$$G(D) = -\frac{1}{N} \sum_j^N \delta(w_j \in C(D))$$

The $\delta()$ function is an indicator function that is 1 if the argument is true and 0 otherwise. $G(D)$ corresponds to the negative of a Monte Carlo estimate of the ratio of the valid space with respect to D , to the space of all possible reward functions. Weight vectors are sampled uniformly.

Finding the set of optimal demonstrations that maximize G is a hard-to-solve combinatorial problem. Thus, a greedy approximation is used. This involves sequentially choosing demonstrations that increase $G(D)$ as much as possible. A set of potential start states are evaluated in terms of the reduction in uncertainty provided by all possible trajectories that starts at that state.

If the teacher’s policy is *deterministic*, then this evaluation can be done directly with the complete trajectory that starts at s and follows the teacher’s optimal policy. Thus, $s_0 = \arg \max_s (G(\{D \cup \tau_\pi(s)\}))$ where $\tau_\pi(s)$ is a trajectory starting at s and following π for a certain horizon. Then the demonstration provided to the learner is $\tau_\pi(s_0)$.

If the teacher’s policy is *stochastic*, then the teacher can sample a set of trajectories that start at s_0 and demonstrate the one that results in the largest $G(\{D \cup \tau_\pi(s_0)\})$. This results in a two step algorithm that first selects the start state for the demonstration, and after committing to the best start state, performs a second step that selects a particular trajectory

allowed by the stochastic policy. Algorithm 7 outlines the algorithm that assumes a stochastic policy. At each step this algorithm greedily selects the demonstration that reduces the uncertainty in the feature weights \mathbf{w} as much as possible, maximizing $G(\cdot)$. Taking into account the particular structure of the uncertainty measure, the following optimality bounds are obtained for this greedy algorithm.

Algorithm 7 Optimal Teaching for IRL

Require: Set of possible initial states S_0
Require: Feature weights \mathbf{w} of the optimal reward function
Initialize $D \leftarrow \emptyset$
Compute optimal policy π^* based on \mathbf{w}
while $G(D) < \epsilon$ **do**
 $s_0 \leftarrow \arg \max_{s \in S_0} E_{\pi}(G(\{D \cup \tau_{\pi}(s)\}))$
 Generate K trajectories $\tau_{\pi}(s_0)$
 for all $\tau_j, j = 1 \dots K$ **do**
 Compute $G(D \cup \tau_j)$
 end for
 $\tau \leftarrow \arg \max_j G(D \cup \tau_j)$
 if $G(D \cup \tau) > G(D)$ **then**
 $D \leftarrow D \cup \{\tau\}$
 end if
end while
return Demonstration set D

Theorem 2 For a given number of demonstrations, Algorithm 7 selects a set of demonstrations D_g which satisfies the inequality $G(D_g) \geq (1 - 1/e)G(D_{OPT})$ where e is the base of the natural logarithm.

Proof (Sketch for the case of deterministic policies). The theorem follows from the fact that $G(\cdot)$ is a monotonous sub-modular function. Every new sample τ only *removes* hypotheses from w , never adding new hypotheses. Thus, $G(w)$ is a non-decreasing monotonous function. $G(\cdot)$ is submodular iff for $A \subseteq B$, $s \notin B$, $G(A \cup \{s\}) - G(A) \geq G(B \cup \{s\}) - G(B)$. This can be verified by observing that a sample added in later stages cannot remove more hypotheses than if it was added earlier. The same demonstration presented at a later stage

can increase G at most as much as if it was presented in the beginning. From [100] it is known that for monotonous sub-modular functions, the value of the function for the set obtained with the greedy algorithm $G(D_g)$ is lower-bounded by the value corresponding to the optimal set $G(D_{OPT})$ with a factor of $(1 - 1/e)$.

This result shows that the heuristic solution for the maximization of the combinational problem is a good approximation to the optimal set. For the case of stochastic policies, the maximization might not be explicit and thus it cannot be ensured that the best example is chosen at each step. Nevertheless, if the approximation error in the maximization is small, then the incurred loss will also be small [80].

10.1.3 Evaluation of optimal teaching algorithm

An evaluation of the proposed algorithm on a navigation task is presented next. Two maps shown in Fig. 51 are considered. Both domains have three features. Each square on the map (other than the obstacles) includes one of the three features. This means that the observed feature vector when the agent is on a particular square, is a 3-dimensional vector where only one of the components is 1 and the others are 0. This simple domain was chosen to allow teaching tasks that are understandable by humans and easy to visualize.

For each map, we consider two teaching tasks. A task corresponds to a particular reward function, *i.e.* a weight vector. For the first map (Fig. 51(a)), both tasks have one terminal state – the corner that has the star shaped feature. For Task 1, the weights given to the other two features are equal (both negative). Thus the resulting optimal policy always chooses the shortest path to the terminal state. For Task 2, the negative weight is larger for the dark gray feature. Depending on the ratio of the weights, this results in policies that prefer light gray paths that are longer.

For the second map (Fig. 51(b)), tasks that have two terminal states are considered. The two terminal states are at the two corners that have the star and diamond shaped features. Both of these features are given a positive weight in the reward function. The rest of the

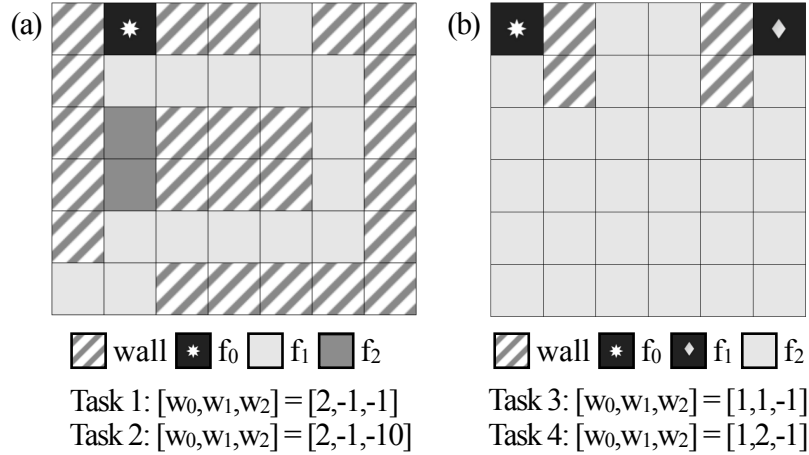


Figure 51: The two maps and the associated reward weights for the four tasks used in evaluating the optimal MDP teaching algorithm and used in Experiment 13.

map has the light gray feature that has a certain negative weight associated with it. For Task 3, the positive weights for the two terminal states are equal. This results in a policy that goes to the nearest terminal state. For Task 4, the diamond shaped feature has a larger weight. Depending on the ratio of the weights this results in policies that prefer to go towards the diamond even though it is further away. The actual weight vectors for each task are shown in Fig. 51.

Examples of Optimal Demonstrations. The start states and trajectories produced by the proposed algorithm for Tasks 1 and 2 are shown in Fig. 52. For both tasks the algorithm decides that a single demonstration is sufficient. The significance of the chosen start states is that they are at the mid point between the two possible ways that have similar reward returns. In Task 1, the two terrains have equal cost. Hence the start state of the chosen demonstration is at the furthest location on the map where both paths have equal length. In Task 2, the chosen start state is at the critical point that balances the length of the path and the different costs of the two terrains. Intuitively this communicates that the shortest path is so costly that it is better to take the longest road. The start state selected in for Task1 would not be as informative here because it could also be explained by having equal costs.

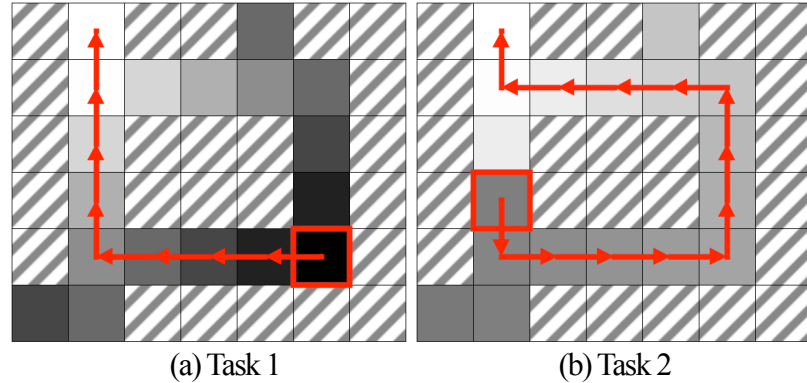


Figure 52: The start states and trajectories chosen by the MDP teaching algorithm for Tasks 1 and 2. The gray-scale color of the squares on the map indicate the value function according to the optimal policy (light colors have high value).

Fig. 53 shows the demonstrations chosen by our algorithm in Tasks 3 and 4. In Task 3, each goal attracts the trajectories that start closer to them. The best places to start a demonstration, are the states around the mid point between the two goals. Presenting a single trajectory is not sufficient, since this could be explained by weights that try to avoid the other goal. In Task 4, the mid point is shifted towards the goal that has a higher reward.

From observing example outcomes of the optimal teaching algorithm a better intuition about what constitutes an informative demonstration for the learner is obtained. A good teacher must show the range of important decision points that are relevant for the task. The most informative trajectories are the ones where the demonstrator makes rational choices among different alternatives, as opposed to those where all possible choices would result in the same behavior. The teaching heuristics provided to human teachers in Experiment 13 are based on this intuition.

Learning Gains. The gains achieved with the proposed teaching algorithm is measured by comparing it with learning from demonstrations whose start states are selected randomly. The two are compared in terms of (i) the uncertainty in the reward function $G(D)$ achieved by the produced demonstration sets and (ii) the performance of an IRL agent trained with

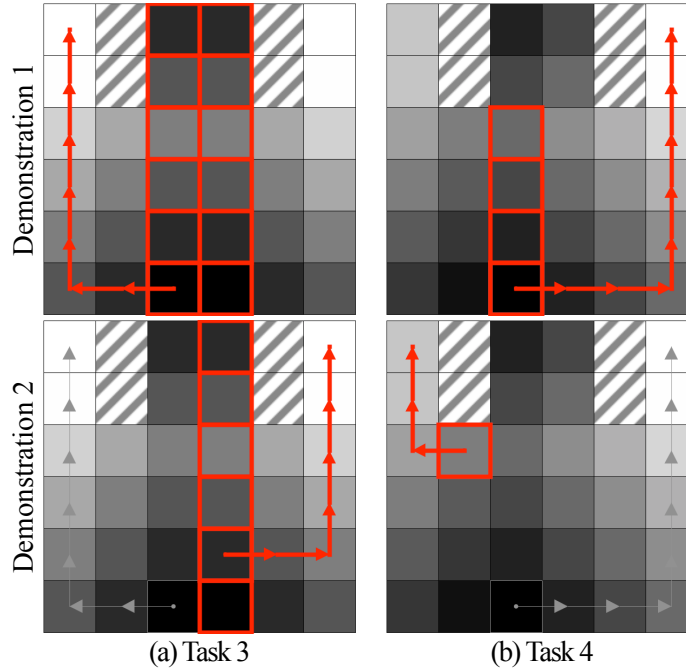


Figure 53: The start states and trajectories chosen by the MDP teaching algorithm for Tasks 3 and 4. The gray-scale color of the squares on the map indicate the value function according to the optimal policy for corresponding reward functions.

the produced demonstrations. The learning agent uses a gradient IRL approach [102]. The performance of the IRL agent is measured by the overlap between the learned policy and the true policy. This is the percentage of actions chosen by the agent that are consistent with the true policy.

The learning curves for all four tasks are shown in Fig. 54. It can be observed that the teaching algorithm allows much faster convergence in all four tasks. On average, for the first two tasks 16 times more demonstrations are required to reach a similar performance. For the last two tasks this fraction is 1:8. From these graphs, it is also observed that the difficulty of the problem is increased when the weights are less balanced (Task 2 and 4).

10.2 Experiment 13: Instructed teaching of navigation tasks

This experiment explores how humans teach sequential decision tasks and how their

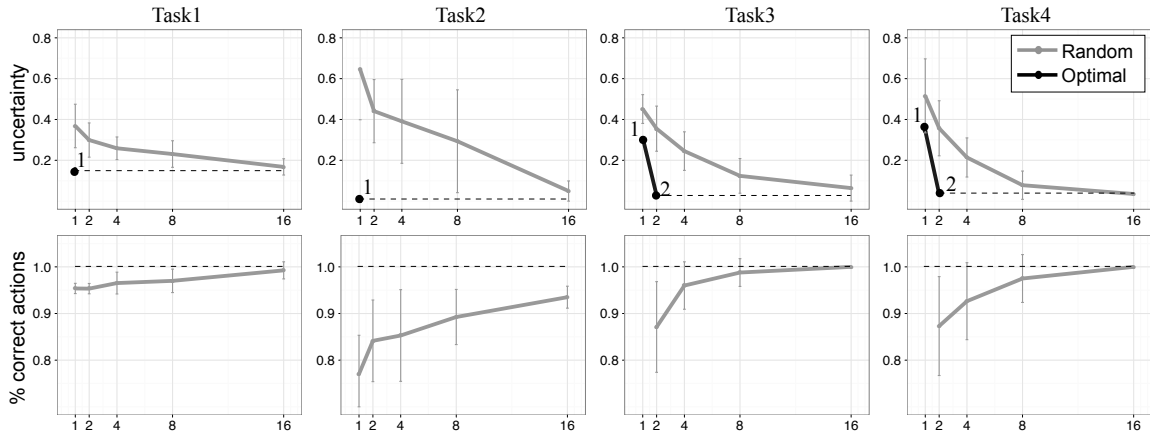


Figure 54: Comparison of learning from demonstrations selected by the MDP teaching algorithm and demonstrations whose start states are randomly selected. The top row shows the decrease in the uncertainty on the rewards, and the bottom row shows the change in the percentage of correctly chosen actions with the policy obtained from the estimated rewards. The x-axes are the increasing number of demonstrations.

teaching can be improved with teaching heuristics that are based on the approximate teaching algorithm developed in Sec. 10.1.2¹.

10.2.1 Experimental Design

10.2.1.1 Domain

The navigation tasks used for evaluating the approximate teaching algorithm in Sec. 10.1.3 are used in this experiment. The experiment has a between-groups design with two factors. The first factor is whether or not the participant is given teaching heuristics (*Natural* versus *Guided* conditions). The second factor is the teaching task, which can be one of the four tasks illustrated in Fig. 51. Thus each participant performs a single task, with or without teaching heuristics, and resulting in a total of eight unique conditions.

10.2.1.2 Procedure

The experiments are web-based. The webpage layout consists of three parts separated by horizontal lines. The top part has general instructions to the participant. The middle part

¹This experiment appears in [23].

is a Java applet that lets the teacher interact with the map, and present demonstrations to the learning agent. The bottom part has questions to the teacher answered after completing teaching. The participants were solicited and compensated through Amazon Mechanical Turk. Each individual was allowed to take part only once in any of our experiments and was compensated with \$0.25.

The applet allows interaction with the map to create a demonstration and submit it to the learner. When a square on the map is clicked, the trajectory that starts in this square and follows the optimal policy is plotted on the map. This avoids mistakes by participants in following the optimal policy. In addition, it clarifies the optimal policy that is verbally described to the teacher. A “demonstrate” button on the applet submits the currently displayed demonstration to the learner.

The instructions motivate the teaching task with a scenario: Participants are told that a robot will be sent to a foreign planet for a mission and that their goal is to teach the robot how to navigate the terrain on this planet. For the tasks that involve the first map (Fig. 51(a)) they are told that the robot needs to get to the square that has the star. For the tasks that involve the second map (Fig. 51(b)) they are told that the robot needs to get to the square that has the star or the one with the diamond. The individual task descriptions have the following details:

- *Task 1:* Navigating through dark or light gray areas has no difference in terms of consumed battery energy.
- *Task 2:* Navigating through dark gray areas consumes ten times more battery energy.
- *Task 3:* Each of the targets have equal treasures.
- *Task 4:* Diamond has two times more treasures as Star.

For Task 1 and 2 participants are told to choose a single demonstration that will be most informative to the learner. For Task 3 and 4 they are told that they can give more than one demonstration, but that they should try to teach the task with as few examples as possible.

For the two teaching conditions the following instructions are given:

- *Natural*: Try to choose the most informative paths for the robot.
- *Guided*: The most informative paths for the robot are those that allow him to see all types of terrains and demonstrate what path to choose when there are multiple viable alternatives. A path is more informative if the alternative paths that start around that location are also good options. If the alternatives are obviously bad, then demonstrating that path is not as informative.

Note that these instructions are exactly the same for all four tasks. To further motivate good teaching in both conditions, participants are told that an award of \$4.0 will be given to the teacher who trains the best learner with the fewest examples.

10.2.1.3 Evaluation

The main measure for comparison is the performance of a learner that is trained by the demonstrations chosen by the participants. For this, the uncertainty in the learned reward models is used ($G(D)$). In addition one open-ended question was asked after completing the teaching task. In the *Natural* condition participants were asked to describe how they chose the demonstrations that they provided to the learner. In the *Guided* condition participants were asked if the teaching instructions was intuitive and understandable. If the participant reports that they did not understand it, their data is deleted and a new participant is recruited to replace it.

10.2.2 Findings

This experiment was completed by a total of 80 participants (ages between 22-56), 10 in each of the 8 conditions. Table 31 summarizes the results of this experiment. Observations from this experiment are summarized as follows.

	<i># optimal</i>		<i>Uncertainty</i>			<i>t</i> -test
	Natural	Guided	Natural	Guided	Optimal	
T1	3/10	4/10	0.34 (0.25)	0.31 (0.04)	0.16	t(10.51)=0.41, p=0.69
T2	0/10	4/10	0.54 (0.16)	0.3 (0.25)	0.0006	t(15.23)=2.59, p<0.05
T3	2/10	6/10	0.23 (0.14)	0.11 (0.11)	0.035	t(13.14)=1.96, p=0.07
T4	0/10	3/10	0.39 (0.15)	0.24 (0.17)	0.027	t(16.19)=1.97, p=0.06

Table 31: Results from Experiment 13. First two columns show the number of participants who taught with an optimal sequence. The next two columns show the average uncertainty in the reward estimation achieved by the demonstrations given by the participants. The optimal value is given for comparison. The last column reports *t*-test results between the two conditions.

Natural teaching is sub-optimal but spontaneous optimality is possible. The first observation is that natural human teaching is likely to be sub-optimal in sequential decision tasks, adding to the results from Chapter 4. Only five out of the 40 people in this condition spontaneously produced optimal demonstrations. The participants’ descriptions of how the demonstration was chosen reveals that these were not chosen by chance, but were indeed insightful. For instance one participant in the *Natural* group describes his thought process as follows:

“I would assume that there is no limitation to where R2D2 should start his path so the best choice would be the square directly next the red square, but in order to teach a robot to navigate different terrain in varying situations it would be necessary to demonstrate a route that contains several different elements the robot may encounter. In other words, if there is only one chance to teach a robot to navigate versatile environments the best choice is to provide a route with as many obstacles as possible so that the robot would be ready.”

Two other participants have similar intuitions:

“I wanted a path that showed all possible obstructions and how to deal with them, such as the walls and the lack of difference between blue and green sidewalks”,

“I tried to involve as many crossroads as possible and to use both green and blue tiles in the path.”

While such intuitions may allow human to give optimal demonstrations, that they occur rarely. As a result of such intuitions being rare, the performance of the trained learners averaged across participants is far from the optimal values.

Teaching heuristics improve performance. From Table 31 it is observed that the number of optimal teachers is increased in all four tasks. The uncertainty in the estimation of the rewards is reduced for all four tasks. This shows that the teaching instructions was mostly effective in eliciting better teaching behavior. In addition, this shows that the teaching heuristics was generic enough, such that it resulted in a positive effect in four different tasks. The size of the effect varies across the tasks; a statistically significant effect is seen only in Task 3, and a positive trend is seen in Tasks 2 and 4, however the difference is insignificant in Task 1. One observed trend is that the task difficulty impacts the usefulness of teaching instructions. If the task is more difficult then the guidance is more helpful for human teachers. This might simply be due to the fact that in easier tasks, a randomly chosen example is not much worse than an optimally chosen demonstration. As a result, a human who is not following a particular strategy has more chance of providing relatively informative examples in the Natural group.

The number of participants who provide the optimal demonstration set determined by the algorithm is increased in all tasks, however a large portion of the participants are still not optimal. Nonetheless, we see that the demonstrations provided are more informative. For instance in Task 2, only four participants provided the optimal trajectory that starts at the lower dark gray square. However three more participants provided a path that starts in one of the squares at the bottom left corner, which all go through the junction where the agent has the choice between the long light gray path and the short dark gray path. Such demonstrations are also relatively informative.

Although there is an improvement due to teaching instructions, the average uncertainties are still far from the optimal values. Some of the common mistakes causing the sub-optimality were, assuming that the “longest” path would be the most informative (Task 2, 4), giving demonstrations very close to the goal in (Task 1,2), trying to involve obstacles in the path (Task 4), or not providing sufficient demonstrations (Task 3,4). This points towards the challenges in creating intuitive and understandable teaching heuristics for users that have no prior knowledge in RL.

10.3 Summary

This chapter extends Algorithmic Teaching to sequential decision tasks. An approximately optimal teaching algorithm for training an Inverse Reinforcement Learning agent is developed and evaluated in a navigation task domain. Teaching heuristics for humans are devised based on the intuition of the proposed teaching algorithm and observations about the types of demonstrations produced by the algorithm. A human subject experiment confirms the utility of these teaching heuristics on four different navigation tasks.

CHAPTER XI

TEACHING HEURISTICS FOR KEYFRAME-BASED LFD

This chapter investigates the use of teaching heuristics in the Keyframe-based LfD framework. The utility of teaching heuristics have been demonstrated in a number of domains in Chapters 9 and 10. While all of these findings provide important insights into how humans can be guided with teaching heuristics in their teaching interaction with robots, none of the experiments so far involved an actual human-robot interaction. All experiments involved teaching a virtual agent.

In both classification and sequential decision tasks the ground truth of what is being taught was available to the teacher. This was either an exact description (*e.g.* a known conjunction or a set of weights that define the reward function) or was specified with a data set (*e.g.* test corpus for alpha, non-alpha mouse gestures). These ground truths are crucial in being able to formally study the teaching problem. For skill learning problems considered in this thesis, there are no ground truth skill descriptions. Collecting a data set is also not satisfactory, because even skills taught by experts are not guaranteed to succeed. Even if they were known to succeed, they cannot be used to evaluate the success of taught skills. A taught skill can only be judged by executing it and observing the outcome.

For these reasons, teaching heuristics for skills can not guide the teacher to provide examples that lead to a known golden skill. Instead, they try to *(i)* elicit examples that result in skills that have certain desired properties, and *(ii)* avoid common mistakes or inefficiencies observed in unguided naive human teachers when compared to an expert teacher. Obviously, the first desired property is that the skill succeeds. For this, teaching guidance can identify common reasons for failure and provide suggestions on how to avoid them. Secondly, good skills are applicable in more scenarios, *i.e.* they are general. Teaching

heuristics that try to elicit skills with these properties are discussed in the following.

Teaching heuristics for more successful skills. In both of the earlier experiments that involved teaching skills with keyframe demonstrations (Experiment 2 and 8), one of the main reasons for unsuccessful skills was that humans were not able to provide the right set of keyframes, and often forgot keyframes that play a role in avoiding obstacles. This happened even though participants were clearly instructed that the robot goes on straight line between any two consecutive frames. Thus, the teaching heuristics should explicitly mention the issue of collisions and ask teachers to spend effort on remembering their last keyframe and provide the next keyframe as to avoid any collisions.

Teaching heuristics for more general skills. Three earlier experiments demonstrate the need for guiding humans to teach skills that are more general, *i.e.* applicable in a larger range of scenarios. The experiment that observed unguided teaching of skills (Experiment 2) supported that humans do not cover the state space available for performing a skill. Similarly in the experiment that compares different query types (Experiment 6) the seeding demonstrations provided by humans were accumulated in a certain region of the state space, and even pre-scripted label and demonstration queries could cover a much larger portion of the space. The baseline of the experiment that evaluates A-KLFD queries (Experiment 8) also confirmed that demonstrations from non-expert teachers are not good at covering the space as compared to an expert teacher or embodied queries produced by the robot. Thus the first focus of the teaching heuristics is addressing this problem.

For the two-handed goal-oriented skills considered within the KLFD framework, the full range of applicability of the skill is determined by the reachable state space for the two arms. It is important for the skill to be applicable for as many configurations of the left end-effector as possible. Therefore the first component of the teaching instructions should motivate providing demonstrations in varying goal configurations. In particular changing

the orientation of the goal configuration, results in large displacements in the absolute positions of the keyframes of a skill — especially for keyframes that are further away from the goal. Thus the instructions should emphasize providing varied demonstrations by changing the orientation of the goal configuration. This issue also emphasizes the importance of keeping the skill as close to the goal as possible. Points that are closer to the goal are less affected by changes in the goal configuration, and are therefore less likely to move outside the reachable region. This is another strategy that can be elicited with teaching heuristics. Finally the instructions should ask the teacher to provide as much variance as the skill allows, by changing the configurations of the keyframes across demonstrations.

11.1 Experiment 14: Instructed teaching of skills

This experiment evaluates teaching heuristics devised for the KLfD framework based on the intuitions of good teaching discussed above. It compares unguided and instructed teaching of skills.

11.1.1 Experimental Design

The experiment involves teaching two-handed goal-oriented skills to Simon through keyframe demonstrations. Skills are learned in the KLfD framework. Two skills that are taught in the experiment are *pouring* from one cup to another, and *closing* a box (Fig. 7(a-b)). Therefore the baseline of the experiment in which Active KLfD was evaluated against unguided human teaching of these skills (Experiment 8) also serves as the baseline for this experiment. In addition, one data set for each skill is collected from one expert demonstrator (*i.e.* the author) for an additional comparison. Here we use the term *expert* to refer to a teacher who understands how the robot learns. All teachers (non-expert or expert) are considered *domain experts*, as their demonstrations are taken as successful executions of the skill.

11.1.1.1 Procedure

This experiment, like other experiments on teaching heuristics, compare the unguided *Natural* teaching of skills, with *Guided* teaching. The procedure for collecting the data in the *Natural* condition was described earlier in Experiment 8. The procedure for collecting data in the *Guided* condition is very similar. The behaviors of the robot, used command, and order of events are identical. This procedure is recaptured in the following, pointing out the difference in the *Guided* condition.

Participants first watch an instructional video that introduced all speech commands used for interacting with the robot. These commands allow changing the state of the right arm between gravity compensation and holding a position, open and close both hands, and move the position of the left arm holding the target object. The video also goes over how a demonstration is presented, with the commands that indicate the start and end of demonstrations, as well as keyframes in between. Then the participant practices the commands and provides two demonstrations of a practice skill and views the execution of the learned skill.

Next participants are told about the incentive for teaching successful skills. They are told that the skill that they teach will be tested in several different scenarios and the teacher of the most successful skill will receive \$15. Then in the *Guided* condition, participants are told that they will be shown another video that will give them tips on how to teach successful skills. The video involves a slide show that goes over these tips supplemented with images shown in Fig. 55.

- Tip #1: Show as much variance as possible across your demonstrations.
 - Vary the goal configuration, especially by changing rotation
 - Vary the position and orientation of keyframes when possible
 - Prefer configurations that are closer to the goal

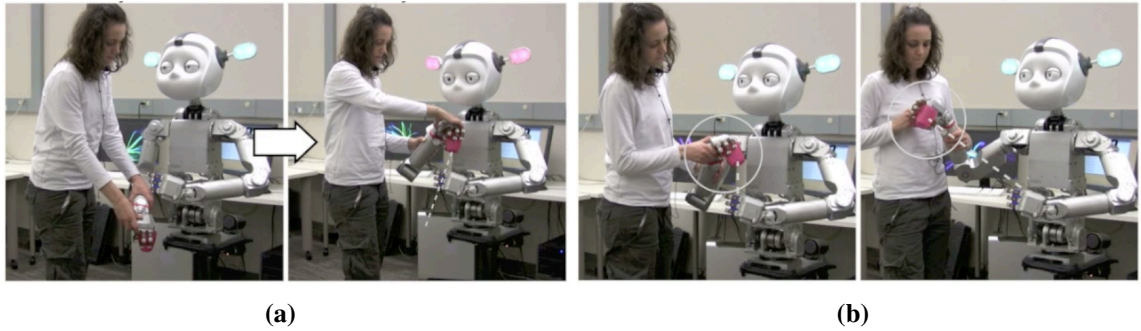


Figure 55: Images used to supplement the teaching heuristics for the instructed teaching of skills in Experiment 14. (a) Explanation of the collision problem due to missing obstacle avoidance keyframes. (b) Guidance for preferring keyframes that are closed to the goal.

- Tip #2: Avoid collisions between keyframes

Next the participant provides demonstrations for each skill, for five minutes each. The order of skills is counterbalanced. When time is up for both skills, participants get a chance to see the learned skills before completing an exit survey.

11.1.2 Findings

The experiment was completed by six participants in the *Natural* condition and six others in the *Guided* condition. The total duration of the experiment (including watching the initial instructions, practice, teaching, viewing learned skills and survey) took around 30 minutes in the *Natural* condition and an additional 5 minutes in the *Guided* condition for watching the second video that gives the teaching instructions.

Taught skills are evaluated with the same metrics as the ones in Experiment 8. The coverage of learned skills and the number of demonstrations provided within 5 minutes are shown in Table 32. The success of the learned skills in the tests are shown in Table 33. The following observations are made.

Teaching heuristics improve skill coverage. The coverage of skills are about 15 to 20% higher in the *Guided* condition. This shows that Tip #1, which gives several suggestions on

Table 32: Coverage of the taught skills and the number of demonstrations provided within 5 minutes in the two conditions of Experiment 14.

Condition	Pouring		Closing	
	Coverage (%)	# demos	Coverage (%)	# demos
Natural	64 (SD=29.69)	8.17 (SD=3.06)	63.67 (SD=20.14)	8.67 (SD=3.20)
Guided	84.33 (SD=22.92)	6.5 (SD=1.22)	78.67(SD=21.53)	6.5 (SD=1.64)
Expert	94	9	88	9

Table 33: Success of learned skills taught in the two conditions of Experiment 14, tested in five different goal configurations. The numbers indicate the percentage of tests out of five that were successful, failed, or N/A. N/A means that the learned skill did not cover the goal configuration in which it was being tested.

	Pouring			Closing		
	Success	Fail	N/A	Success	Fail	N/A
Natural	13.33 SD=32.66	56.67 SD=29.44	30.00 SD=27.57	53.33 SD=24.22	10.00 SD=10.95	36.67 SD=29.44
Guided	53.33 SD=16.73	43.20 SD=36.67	10.00 SD=29.44	66.67 SD=25.30	41.31 SD=13.33	20.00 SD=24.22
Expert	80	20	0	100	0	0

how to induce variance across demonstrations, is successfully employed by the participants in order to cover the state space better. However, the average coverage is still smaller than what the expert teacher can achieve.

There is a large variance in the coverage achieved by participants in each condition. While skills taught in the *Guided* condition have higher overall coverage, some skills taught in this condition have low coverage. Similarly, some skills taught in the *Natural* condition have high coverage.

The teaching heuristic that tried to elicit better coverage from the demonstrations elicited from human has three different tips. In demonstrations provided in this condition, it is observed that all participants utilized the tip about changing the goal configuration. Their demonstrations involved an average of 5.08 (SD=0.99) different goal configurations per skill, compared to a 3.00 (SD=2.29) in the *Natural* condition. Two out of the six participants in the *Natural* condition did not change the goal configuration at all, even though they had used the commands for changing the goal configuration during their practice session. Two other participants used the commands before providing any demonstrations and kept the goal configuration constant across their demonstrations.

The other two tips seemed to be used mutually exclusively by five participants. Four out of the five tried to induce as much variance as possible in their demonstrations, particularly exaggerating the variance in the start states. Examples of such starting keyframes are shown in Fig. 56. These participants did not have any preference towards keyframes that are close to the goal. The other participant did not provide a very large variance and seemed to closely follow the teaching heuristic that advised having preference towards keyframes that are close to the goal.

The last participant explicitly tried to combine the two teaching heuristics. After watching the video that present the teaching guidance, he deliberated on how he could both prefer keyframes that are close to the goal and at the same time provide large variance. The strategy that he came up with was to go on concentric spheres around the goal; providing

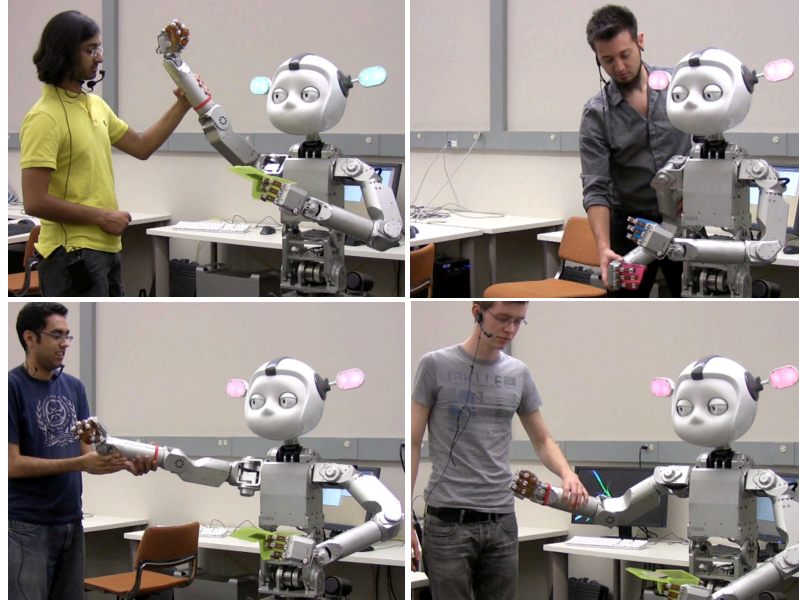


Figure 56: Example starting keyframes provided by participants in Experiment 14, as a result of the teaching heuristic that advises the teachers to vary the position and orientation of keyframes when possible.

variance within the sphere then gradually increasing the radius. This is also the strategy followed by the expert teacher.

The lack of induced variance in the keyframes apparent in the demonstrations provided by participants in the *Natural* condition. Two participants explicitly mentioned this being part of their teaching strategy in the survey, even though they had been told that the robot would be tested in various scenarios.

“I tried to keep the same starting point for the left hand and move the right hand with minimum variance so that it learns the trajectory accurately. I believed that multiple starting points with multiple paths would not lead to accurate reproduction of the action.”

“I was trying to keep track of the number of positions that I recorded. Other than that, the positions I recorded were usually in the same spatial configuration and at the same distance to the target object.”

The teaching strategies explained by participants in the *Natural* condition reveal various mental models of the robot learns. One participant had a two step teaching strategy described as follows.

“The first few demonstrations, I focused on teaching Simon the movement no matter how awkward his hand movements were, and once I felt I had that down, I tried to find more natural methods of placing his hand.”

This is a strategy that can be expected from humans, however it will not result in skills that are expected by the teacher, since the robot’s learning algorithm does not make any distinctions between demonstrations that focus on movements versus hand placements. One other participant mentioned focusing on trying to give as few keyframes as possible while changing as few joints as possible between keyframes. Again, this indicates a flawed mental model of how skills are learned and what demonstrations lead to successful skills. Only one participant in the *Natural* condition mentioned trying to demonstrate the skills from various positions and orientations as part of their teaching strategy.

Teaching heuristics improve skill success. The skills taught in the *Guided* condition are more successful in the tests. Both tips in the teaching heuristics have an impact on the success. In the unguided condition, four participants had issues related to collisions in one or both of the skills, while in the guided condition only one did with one of the skills. This can be credited to the explicit warning against collisions in the teaching instructions. Secondly, as observed in Table 33, the better coverage in both skills (*i.e.* lower percentage of goal configurations with N/A) also impacts the success. Although five out of the six participants in the *Guided* condition indicated in the survey that they found the teaching heuristics about inducing variance more useful than the teaching heuristic for avoiding collisions, the collision problem was rare in the Guided condition. This is perhaps due to the heuristic being obvious but easily overlooked if not brought into attention.

Slower teaching with heuristics. The average number of demonstrations provided within

five minutes in the *Natural* condition is about three more than that of the *Guided* condition. Teachers spend time between demonstrations to deliberate about the teaching heuristics and deciding on their next demonstration. In addition, they spend time on changing the goal configuration (saying the command and waiting the left arm to move).

Expert performance cannot be reached with heuristics. While better performance is achieved with teaching heuristics, the average performance is not as good as the expert teacher in terms of both metrics. Note that the variance is rather high. In fact one participant in the *Guided* condition achieved coverage and success equal to that of the expert in both skills. This shows that guidance has the potential to elicit performance by the teacher that is equivalent to that of a teacher who knows exactly how the robot learns and understands what demonstrations are useful for learning (*i.e.* the person who devised the teaching heuristics). However the data demonstrates that this cannot be expected in the average. Longitudinal experience teaching the robot and a deeper understanding of the robot's learning mechanism might still be necessary for reaching a higher average performance while learning from users.

Expertise is also reflected in the speed of teaching. The expert in general provides slightly more demonstrations than average non-expert users in the *Natural* condition, which is on average three more demonstrations than the what participants provide in the *Guided* condition. In other words, the expert teacher uses the teaching heuristics without spending time thinking about it. Thus, practicing the usage of teaching heuristics might have a positive impact on the speed of guided teaching.

11.2 Summary

This chapter explored the use of teaching guidance in robot skill learning, where no ground truth skills or test data sets are available to inform the process of devising teaching heuristics. Instead teaching heuristics for skills were based on desired properties of taught skills (accuracy in achieving goal and applicability in as many scenarios as possible), common

mistakes observed in unguided teaching leading to undesirable skills, and teaching strategies iteratively developed by an expert teacher. The devised heuristics were compiled into a set of *tips* for teaching successful skills. These were tested in a human subject experiment comparing natural and guided teaching of two-handed goal oriented skills. The experiment showed the utility of the heuristics in achieving both more accurate and general skills.

CHAPTER XII

COMBINING TEACHING HEURISTICS AND QUERIES

Theoretically, teaching heuristics are more powerful than queries. An active learner can never learn faster than when it is taught optimally [57]. While teaching heuristics had positive impacts across different problems and domains in the experiments presented in this thesis, they were often not perfectly followed. This is particularly problematic when the teachers think they followed the heuristics accurately, while they actually failed. As exemplified in the experiment that involved instructed teaching of toy object categories (Experiment 9), when humans fail to follow the teaching heuristics, the resulting models are incomplete and inaccurate. The teaching guidance creates a false sense of trust in the teachers, and as a result they do not test the robot as extensively. Therefore, in practice teaching heuristics are still far from eliciting optimal demonstration sets.

In terms of achieving complete and accurate final models queries are more reliable. As observed in Chapter 6, even when the learner makes queries only intermittently and the control of query timings are given to the teacher, the resulting concepts are much more accurate than ones taught without any queries. However, queries are strongly dependent on the initial seeding set provided by the human. The query mechanism used for learning conjunctions can make steady progress to uniquely identify the target, only once it has seen a positive example. If the teacher does not provide the first positive example, it can take many queries until the learner stumbles upon a positive example. In addition, even the steady progress made by queries is slower than what an optimal teacher can do.

Therefore, teaching heuristics and queries present two complementary strengths that strongly suggests their combination to be used for guiding human teachers in their teaching

interactions with robots. This idea is explored in this chapter and its potential is demonstrated through a follow up experiment in the task learning problem.

12.1 Teaching Heuristics for Intermittently-Active Learning

The intermittently-active learning paradigm lends itself well for combining the two mechanisms. In this paradigm the responsibility of choosing examples to learn from is shared between the learner and the teacher. The timing of the queries can be chosen by the teacher (teacher-triggered queries), or decided by the learner by comparing the utilities of making a query versus accepting another example from the teacher (conditional queries). In this framework, teaching heuristics should instruct the teacher for the examples that the teacher is responsible for. This potentially reduces the content of the teaching heuristics. In the case of teacher triggered queries, the teaching heuristics can also guide the teacher about when to trigger queries. In this way teaching heuristics have the potential guide how the responsibility of choosing examples should be divided between the teacher and the learner. This idea is further explored in the following.

There are two important factors when dividing the responsibility of choosing examples.

- Human strengths should be leveraged. As observed in experiments of instructed teaching, humans do not always succeed in fully using the teaching heuristics, but parts of the teaching heuristics are intuitive for a majority of the teachers. Removing parts of teaching heuristics that are not intuitive might also be beneficial since following a misunderstood teaching strategy might be harmful or at best waste time.
- The power of teaching heuristics should be leveraged. As mentioned, optimal teaching is theoretically more powerful than active learning. The trick of the optimal teaching algorithm that achieves this advantage should be identified and kept as part of the teaching heuristics. Other parts of the teaching heuristics that are naturally performed by humans, or could be achieved with queries, can be removed from the original heuristics.

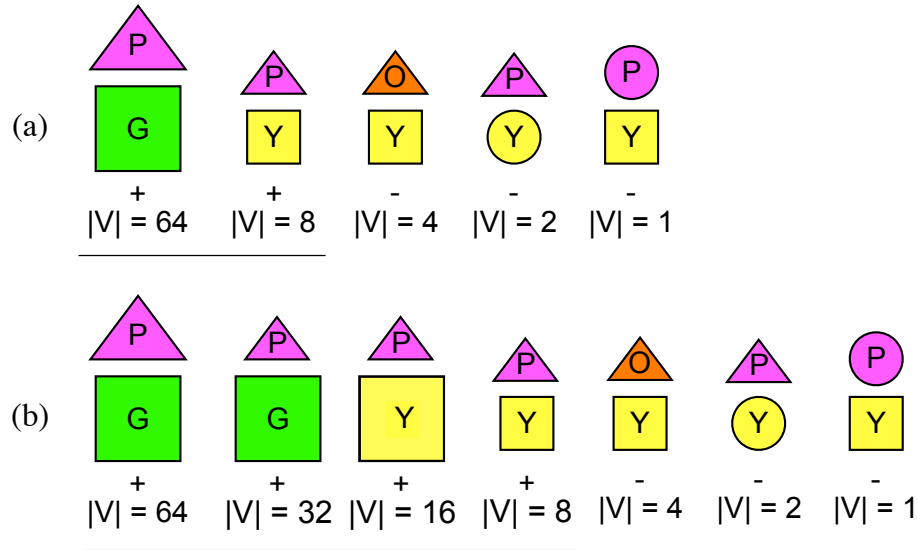


Figure 57: Teaching sequences for the HOUSE concept produced by (a) the optimal teaching algorithm and (b) by the learner through queries once the initial positive example was given.

Consider the optimal teaching sequence for the HOUSE concept in the six-feature toy objects domain and the sequence produced by the query mechanism once a positive example has been given (Fig. 57). The underlined parts of the sequences are equivalent. They both reduce the version space by the same factor. The optimal teaching sequence achieves this with only two examples by changing all irrelevant features at once. The query mechanism on the other hand cannot risk to change more than one feature at a time. When more than one feature is changed and the new sample ends up having a negative label, the learner can only prune the hypotheses that involve all of the changed features, which is a much smaller portion of the version space. Thus the active learner always explores one feature at a time, to guarantee pruning half of the version space.

In this example, the power of the optimal teaching algorithm over the active learner is in the first two positive examples. After those two examples, the rest of the teaching sequence is identical to the sequence created by an active learner. Therefore the teaching heuristics should focus on eliciting the parts of the optimal teaching strategy that is about those first

two positive examples. Incidentally, this was the part that was more intuitive for participants in the experiment that involved instructed teaching of this concept (Experiment 9). Participants had trouble following the part about negative examples, particularly keeping track of which relevant features had already been varied. Thus the ideal scenario is for the human teacher to provide the first two positive examples and the learner to query the rest of the examples.

Assuming teacher triggered queries, the teaching guidance for this scenario is as follows:

- First show one positive example
- Then show another positive example in which all irrelevant features are changed to a different value
- Then trigger queries and label them

Note that this results in an optimal teaching sequence, without requiring the teacher to have full responsibility of following all steps of an optimal teaching algorithm. The following experiment evaluates the use of such teaching guidance.

12.2 Experiment 15: Instructed Teaching in I-AL

This experiment is a follow up on Experiment 9, exploring the use teaching heuristics in the intermittently-active learning paradigm¹.

12.2.1 Experimental Design

This experiment involves teaching a virtual Simon toy object categories represented as conjunctions over six discrete features using the concept learning algorithm described earlier in Sec. 3.2.1. The domain in this experiment is the six-feature toy objects domain. The

¹This experiment appears in [27]

concepts are taught through a Graphical User Interface similar to the one from Experiment 9 shown in Fig. 43. In this interface, object parts are dragged to the demonstration area using the mouse. The constructed object in the demonstration area is labeled as positive or negative and tested using the buttons at the top. In addition there is a button for triggering queries. When this button is pressed, a query is generated and the queried instance is placed onto the demonstration area. A speech bubble next to Simon asks for a label for the queried instance, *e.g.* “Is this a HOUSE?” An “I’m done” button allows participants to end the experiment when they think they are finished teaching. A “Clean up” button places all the object parts back in their original locations.

Participants taught two of the concepts used in Experiment 5: HOUSE and ICE-CREAM, described in Table 1.

12.2.1.1 Procedure

Participants first watched an instructional video that demonstrated how the GUI works. The video involves dragging object parts and using different buttons on the interface. The description of the concepts were given to the participant before starting to teach and a reminder was shown on the right of the interface. Participants taught the first concept without any specific teaching instructions. The query trigger button is not displayed in this baseline condition. Then participants are prompted with specific instructions before teaching the second symbol. The two conditions had different instructions.

- *Teacher-triggered queries (TTQ)*: This time you will have a new button that says “Any questions?” When you press this button, Simon will configure an example himself and will ask you tell him what it is.
- *Guided TTQ*: [Same as TTQ] We want you to use a specific teaching strategy: (1) First, show Simon an example of ice-cream. (2) Then, change everything that does not matter for ice-cream, and show another ice-cream. (3) Use the “Any questions?” button and answer Simon’s questions until he finishes asking questions.

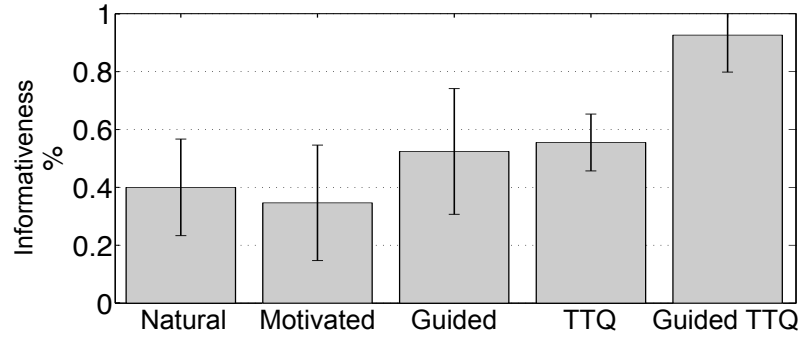


Figure 58: Average informativeness of examples provided by human teachers in five conditions: *Natural*, *Motivated* and *Guided* conditions from Experiment 9 and *TTQ* and *Guided TTQ* from Experiment 15.

The instructions were repeated to the participants by the experimenter and their questions were answered. The position of the “Any questions?” button was also pointed out. A reminder of the instructions was also shown on the right part of the interface.

12.2.2 Findings

This experiment was completed by 20 participants, 10 in each condition. The two conditions, in addition to the ones from Experiment 9 are compared in terms of average informativeness of examples provided by the participants. Informativeness of an example is defined as the ratio of the accuracy gain provided by the example to the maximum accuracy gain provided by any example in the given state of the learner. Thus, the informativeness of all examples in the optimal teaching sequence is 1 and the informativeness of a redundant example is 0. This is a more compact metric for measuring similarity to optimal teaching, unlike the good and bad example distributions used earlier (Sec. 4.1.1.3).

The average informativeness in all five conditions are shown in Fig. 58. We see that the average informativeness in the mixed-initiative condition is close to 1 and is significantly higher than the other conditions. TTQ alone does not spontaneously result in close to optimal teaching. This result confirms the intuition that queries and teaching heuristics can be used together to outperform the use of individual mechanisms alone.

12.3 Summary

This chapter explores the idea of combining the two mechanisms proposed in this thesis. The strengths and weaknesses of each mechanism are characterized and the complementary nature of the two mechanisms is highlighted. Then, teaching heuristics for the Intermittently-Active Learning scenario, particularly teacher-triggered queries (TTQ), are discussed. These heuristics have two components. The first is a subset of the heuristics developed for optimal teaching. The subset is selected based on the heuristics' impact in achieving performance better than an active learner, and based on empirical evidence of intuitiveness for humans. The second part guides the triggering of queries. In one of the common domains that were used in earlier chapters, these TTQ teaching heuristics are evaluated, comparing them to the conditions in the earlier experiment as well as unguided use TTQ. The experiment shows that Guided TTQ results in close to optimal teaching by humans, with significantly better performance than any other condition.

CHAPTER XIII

CONCLUSIONS AND FUTURE WORK

This thesis is motivated by the vision of personal assistive robots that can be reprogrammed by their end users. Learning from Demonstration is a method that allows users to program new capabilities on a robot, by providing demonstrations of what is required from it. This thesis characterizes the issues that are encountered when this method is used with naive users. To address the most important issues, *i.e.* sub-optimal and incomplete teaching, two mechanisms to augmenting LfD interactions are proposed. These mechanisms guide the user in their interactions with the robot during which they provide demonstrations to the robot. The first mechanism involves questions asked by the robot to elicit more informative demonstrations from the teacher. The second mechanism involves instructions given to the teacher prior to the teaching interaction, to influence them towards providing more informative demonstrations. Several technical contributions are made in developing these mechanisms for task and skill learning problems in robotics. In addition, a diverse set of human subject experiments characterize important factors that need to be taken into account in designing teaching interactions for users, and demonstrate the utility of the proposed mechanisms in learning better tasks and skills faster.

13.1 Contributions

The scientific contributions made by this thesis are as follows.

13.1.1 Experimental findings about teaching-learning interactions

Through three initial observational experiments and the baseline conditions of all other experiments presented in this thesis, issues that occur in learning from naive users are characterized. These provide motivation and suggestions for building mechanisms that

guide the users to provide more informative demonstrations such that they can teach more accurate and general skills and tasks. In addition these experiments reveal useful patterns common in human teaching, which can be exploited by robot learners.

This thesis also presents observations from several experiments that involve interactions in which the robot asks questions to its human teacher. These highlight three unexplored issues that are encountered in these interactions: (i) balance of control, (ii) question clarity, and (iii) compliance. Experimental findings include evidence for human preferences towards particular interaction modes, and query types which will guide the design of interactive learning agents.

13.1.2 Intermittently-Active Learning

This thesis contributes a novel active learning paradigm, Intermittently-Active Learning (I-AL) in which the robot selectively makes queries during the teaching process and the learner keeps providing demonstrations when the learner does not make a query. Two approaches for deciding *when* to make a query are presented. I-AL provides the same benefits as fully-active learning, while creating more desirable, balanced interactions between the teacher and the robot.

13.1.3 Active Keyframe-based Learning from Demonstration

This thesis introduces a novel demonstration type, *keyframe demonstrations* and an associated skill representation and learning algorithm for modeling skills in continuous action spaces. This framework allows for human-segmented representations of skills that are intuitive to teach. In addition, it allows autonomously generating different types of queries that improve the skill. Theoretical methods and algorithms for generating four different types of queries are contributed. All queries result in skills that are more accurate and general than skills learned just from demonstrations provided by naive humans. This includes three novel types of queries, *partial-label*, *demonstration* and *feature* queries.

13.1.4 Experimental findings on human question asking

An empirical characterization of human question asking in terms of question types, forms, and use of embodiment is presented. These provide evidence for preference towards certain ways of asking questions which can guide the design of more natural robot questions. They also provide inspiration for novel query types (*e.g.* partial-label, partial-demo or feature-invariance queries), for ways to use embodiment to better communicate the question (*e.g.* multiple feature value instantiations) and for additional transparency mechanisms (*e.g.* communicating predictions of the answer with difference certainty by altering the form of the question).

13.1.5 Teaching heuristics

This thesis introduces the idea of instructing human teachers on how to teach and proposes methods for deriving such instructions based on a theoretical analysis of the teaching problem. A characterization of teaching problem types with increasing complexity is established for classification problems (provably, empirically or approximately optimal teaching) and methods for deriving teaching heuristics for humans from these types of teaching problems are presented. The optimal teaching problem and teaching heuristics are extended to sequential decision tasks involving Inverse Reinforcement Learning agents. Methods for generating teaching heuristics for skills, where no ground truth skill representation is available, are also presented.

Empirical findings comparing unguided human teaching and teaching instructed with heuristics are presented for six different domains in different learning problems are contributed. All experiments demonstrate the utility of teaching heuristics.

13.2 Open questions and future work

More query types. Some of the query types observed in humans were not studied in this thesis. These include partial demo queries that involve executing part of the skill

and requesting a demonstration for the other portion; or feature value requests which asks the teacher to show possible values of a feature in a certain keyframe rather than asking whether a certain value is valid. These might have benefits over the other query types in certain scenarios and would increase the repertoire of available queries. Another query type not explored in this thesis is *comparison queries* [20]. While assigning value to an execution might be difficult for humans, comparing two executions of a skill (*i.e.* saying which one is better) is intuitive.

Mixed-query Learning. This thesis introduced a framework that allows several types of queries to be autonomously generated, however the use of multiple query types within a single interaction has not been explored. This involves several open problems related to the comparison of the utilities of different query types as well as grouping and ordering of questions.

Augmented keyframe demonstrations. Certain query types allow the user to input information which is not normally taken as input outside of queries. When an invariance query is made the teacher can say that a feature has to have a certain value, however the teacher cannot communicate this information directly to the robot unless the query is made. There is no reason for not allowing the teacher to provide such information, since the learner has a way to process and make use of it. Teacher can indicate that a feature of a keyframe is invariant as they provide the keyframe, or indicate that a keyframe has a lot of flexibility to quickly build skills that are accurate and general. Other meta-information about keyframes indicating its purpose could also be useful (*e.g.* start, end, avoidance or contact keyframes).

Skill adaptation and transfer. Active-KLFD is an ideal framework for skill adaptation and transfer. For example a pouring skill that has been learned with a certain bottle can be a good starting point for learning to pour with a different bottle. The source skill can be

adapted through different types of queries. Such queries would require different selection mechanisms from the ones designed with the objective of increasing skill variance. In contrast, skill adaptation might require reducing variance in certain parts of the skill. Similarly, certain keyframe distributions in a skill have generic purposes that could be transferred across skills. For example all skills that require the manipulator to start at a position lower than the target could share starting keyframe distributions.

Adapting to teaching style. Conditional queries in Intermittently-Active Learning, involved a mechanism that altered the timing of queries by the learner based on the teacher’s performance. A similar mechanism can alter the choice of query types to optimize information gain based on the teacher’s performance. For example, demonstration queries can be preferred for fast teachers while label queries could be more efficient with slow teachers. In general, patterns observed in human subject experiments provide insights into how certain parameters of the learning interaction should be chosen to maximize information gain and enhance usability. However, these could be further improved through personalization.

Interactive Teaching Heuristics. The form of teaching heuristics studied in this thesis were written or video instructions given to the teacher prior to their teaching interaction. Instead one can imagine incorporating such guidance into the interaction, where the teaching heuristics are provided by the learner. This would be similar to demonstrations queries that request new demonstrations under certain constraints. The difference would be that rather than constraints, the robot would provide a meta-level description of the type of demonstration that the person should provide, *e.g.* “Can you give me another demonstration that varies maximally from the previous one?” This could potentially facilitate following teaching heuristics, and allow associating meta-information with the acquired data. For example a movie recommendation system could ask for examples of movies that the user really likes or dislikes, and then ask for borderline movies that they might like but would not watch.

APPENDIX A

PROTOCOL FOR EXPERIMENT 2

General Instructions

- * **Thank you** very much for your participation in this study.
- * Here is our humanoid robot, **Simon**.
- * **In this experiment**, you will teach Simon various skills by physically guiding its right arm.
- * We will ask you to teach several skills in **four different interaction modes**.
- * We refer to these four interaction modes by color.
- * After every interaction we will ask you to rate or compare your interactions with some **questions**.
- * You will teach Simon **2 skills** in every interaction mode. So you will teach him a **total of 8 skills**.
- * Before teaching the two skills in a new mode, you will also teach him a **different skill just for practice**, but that will not count.
- * Here is the list of skills that you will teach Simon today. [Illustrate by performing skill yourself]
 - * **Insert**: Simon will be holding a block in his hand. You should make Simon insert the block through the hole without touching other blocks.
 - * **Touch**: Simon's hand will be in a pointing pose, and you should make Simon's finger touch the red circle on the foam.
 - * **Close the Box**: You should make Simon's close the lid of the box without moving it.
 - * **Stack**: Simon will be holding this block in its hand. You should make Simon stack the block on top of the other block on the table. Simon will not release the object, but you should try to make it such that if he released it, it would be stacked.
 - * **Raise hand**: Make Simon perform raise his hand as if he was asking for permission. Assume the person who is watching Simon is straight in front of him.
 - * **Beckon**: Make Simon perform a beckoning gesture as if asking someone to come closer. Don't make him repeat the gesture, so make him do the gesture only once. Assume the person is in front of him.
 - * **Salute**: Make Simon perform a salute gesture. Assume the person he is saluting is in front of him.
 - * **Throw**: Make Simon preform a throwing gesture with this ball. He will not actually release the ball. Make him throw in the forward direction. Don't make him throw from below.
- * For practicing the interaction modes you will teach simon how to **point** to this plus sign on the floor.
- * Simon's **hand will remain in a fixed** configuration throughout a skill; either pointing, holding something, or open. We will prepare Simon's hand configuration for you, before you start teaching a new skill.
- * The way that you teach a skill is by giving **demonstrations** of the skill. A demonstration is one example of performing the skill. You can give many demonstrations to make the skill better and better. Simon learns from all the demonstrations.
- * To give a demonstration you will need to tell Simon **when you start** and **when you end** a demonstration. For this you will interact with Simon with some speech commands.
- * Simon will acknowledge each speech command by giving you a verbal feedback. If he doesn't, try repeating the command.

Microphone test and familiarization

- * Please put on this microphone and try reading the sentence that I point to. *[Equip the participant with the **microphone**, and **test** to see if the participant's pronunciation works. If not tell them the **alternative** commands.]*
- * Please take a moment to touch and **move around Simon's arm** until you get a good feel of how it moves.
- * As an **exercise** try to make Simon touch my hand. *[Repeat for 4 different positions]*
- * By the way, you **don't have to stand** in any particular location, so feel free to move around.

RED and Blue Modes: Trajectory demonstrations (BT) and Keyframe demonstrations (BK)

- * The first two modes are RED and BLUE.
- * RED mode is what we call teaching by trajectories.
- * In this mode you say **NEW DEMONSTRATION** to start giving a demonstration to Simon. From this point on, you will be able to move Simon's arm. Simon will **record all the movements** you make with his arm.
- * To end the demonstration say **END OF DEMONSTRATION**.
- * *[Experimenter demonstration of trajectory mode with the arm.]*
- * After giving a demonstration you can make Simon show you what it has learned so far by saying **CAN YOU PERFORM THE SKILL**. If you are not pleased with the performance you can keep giving new demonstrations in the same way.
- * When you feel that Simon has learned the skill you can move on to the next skill by saying **LET'S LEARN A NEW SKILL**.
- * Then **Simon will tell you** which skill he wants to learn. He will always first ask for pointing, for practice, and then he will choose skills from the other 8.
- * Make sure to leave Simon's arm **close to its starting position** before you end a demonstration.
- * After teaching both skills you can say **THAT'S IT SIMON YOU HAVE LEARNED BOTH SKILLS**.
- * There is a **reminder** of these speech commands on the **board**. There will be other speech commands in **different modes**, but we will tell you about those as we go.
- * The BLUE mode is what we call keyframe demonstrations. In this mode Simon will not be recording all the movements. It will only record the arm configuration when you say **RECORD FRAME**.
- * You can give **as many keyframes** as you wish within a demonstration.
- * *[Experimenter demonstration of keyframe mode with the arm.]*
- * Everything else is the same.
- * Reminder: In each mode you will first practice the interaction by teaching pointing and then you will teach 2 skills.
- * Then we will ask you to fill out a **survey** about your interaction. The survey has some questions that refer to the skills that you taught Simon as the **first skill** and the **second skill**; so try to remember things that were different between the two. Note that the questions are **not about the practice skill**, pointing, so by first skill we mean the one right after pointing.

Mode #1

- * First mode is RED/BLUE.
- * Reminder: Make sure to leave Simon's arm **close to its starting position** before you end a demonstration.
- * Survey -- [Page1: General info, Page 2: MODE1 rating]

Mode #2

- * Second mode is RED/BLUE.
- * [Point out the difference from previous mode].
 - * BLUE: This time you will only show keyframes by saying RECORD FRAME and the arm movements between the frames will be ignored.
 - * RED: This time you don't have the command for recording frames, Simon will record the whole movement.
- * Survey -- [Page3: MODE2 rating, Page 4: RED vs BLUE]

GREEN Mode: Keyframe demonstrations with Iterations (IK)

- * In the **BLUE mode**, when you were not satisfied with what Simon had learned, you could **give new demonstrations**. In the next interaction you will be able to **start** your new demonstration **from what Simon has learned** so far. So you can make changes in the middle without giving it a full demonstration. Using this method, you will teach Simon 2 new skills.
- * The **first demonstration** will be given exactly like in the BLUE mode. You say **NEW DEMONSTRATION** to start giving a demonstration to Simon. You say **RECORD FRAME** to save the keyframes and **END OF DEMONSTRATION** to end the demonstration.
- * After this you will be able to **navigate and modify** the learned skill to produce new demonstration. You won't give any more demonstrations from scratch.
- * To navigate through the current skill you can say **NEXT FRAME** or **PREVIOUS FRAME**. If you try to go to the next frame while you are on the last frame it will **loop** to the first frame.
- * You can edit the navigated skill to create a new demonstration for Simon. You can edit it in three ways:
 - * You can modify an existing frame by saying **MODIFY THIS FRAME** and changing the arm configuration for the frame and then saying **RECORD FRAME** to save it.
 - * You can add a new frame by saying **ADD NEW FRAME**, moving the arm around and saying **RECORD FRAME** to save it. This will add a frame that comes right after the frame in which you say ADD NEW FRAME.
 - * You can delete a frame by saying **DELETE FRAME**. This will make Simon move to the previous frame.
- * Make sure to **hold Simon's arm** before you say ADD or MODIFY because Simon will release its arm so that you can move it around.
- * You can ask Simon to perform the modified skill as a whole by saying **PLAY CURRENT DEMONSTRATION**.
- * Once you are pleased with the modified demonstration, you will need to submit it by saying **RECORD THIS DEMONSTRATION**.

- * After giving a demonstration you can make Simon show you what it has learned so far by saying **CAN YOU PERFORM THE SKILL**. Note that if you say this before submitting a modified demonstration, Simon will forget the modifications.
- * Survey -- [Page 5: GREEN mode rating, Page 6: BLUE vs GREEN]

YELLOW Mode: Adapting existing skills (IKA)

- * You will teach Simon 2 more skills using the same method as in the GREEN MODE. However, this time Simon already has a skill to start with. So you will not give a first demonstration. Everything else is the same.
- * You can begin by navigating through the skill with **NEXT FRAME** or **PREVIOUS FRAME**, or performing the skill with **CAN YOU PERFORM THE SKILL**.
- * You can edit the skill to create new demonstrations using **ADD NEW FRAME**, **MODIFY THIS FRAME**, **RECORD FRAME** and **DELETE FRAME**. You can see the modified demonstration using **PLAY CURRENT DEMONSTRATION**. If you are pleased with it use **RECORD THIS DEMONSTRATION** to submit it.
- * Survey -- [Page 7: YELLOW mode rating, Page 8: GREEN vs YELLOW, Page 9: Exit survey]

APPENDIX B

PROTOCOL FOR EXPERIMENT 5

BEFORE STARTING

Be sure to have the robot off whenever talking about the tasks so people don't think he "hears" you.

- * Your task is to teach Simon to recognize 4 types of objects in different interaction modes.
- * An object is composed of two tangram pieces, a top and bottom part.
 - * Top/bottom are from Simon's perspective.
 - * Make sure the pieces aren't touching each other.
 - * Orientation does not matter.
 - * Make sure you can tell sizes apart.
- * For example, a HOUSE is an object with a pink triangle top and a square bottom. The color of the bottom and the sizes doesn't matter.
- * You'll teach Simon about one type of object at a time in 4 teaching sessions. Before every session:
 - * Simon's memory is wiped out so he will be learning from scratch.
 - * You'll be given specific instructions about that specific session.
- * After teaching each object, you will fill out survey questions about that teaching session. After teaching all four objects, you will fill out survey questions asking you to compare all four interaction modes.
- * To teach an object to Simon, you need to arrange any object in the demo area from Simon's perspective and say one of two types of sentences:
 - * Simon, this is a {house}.
 - * Simon, this is NOT a {house}.
- * At any time, you can also test if Simon can recognize the object by saying the following sentence:
 - * Simon, is this a {house}?
- * Simon will not understand any other sentences.
- * Pieces that you aren't currently using in the demo area should be put back to their original location.
- * After saying a sentence to Simon, watch his response, and listen to what he says.
 - * Wait for his ears to flash before trying to say another sentence.
- * You will decide when to stop teaching. When you think Simon has learned to recognize the object, or if you think Simon is not making any progress, let us know and we will move on to the next object.

NO QUERIES

- * First you will teach Simon the HOUSE. Again, a HOUSE is an object with a pink triangle top and a square bottom. In this session Simon will talk to you only to acknowledge that he heard you. He will not be asking any questions. You can ask questions to Simon at any time.
- * Wait for me to tell you when you can begin teaching.

OTHER

- * In the rest of the experiment, Simon can request specific examples from you. He knows this workspace and can point at different object categories. For example when you show an example of a house, he might ask you to place the large pink circle on the top and point at it. If you respond to his question with speech he will not understand you. You can respond to his question by doing what he ask for.
- * [Reminder] Remember that you don't have to do what he asks you to do, can always respond using ANY of the sentences he understands. You are the teacher, and you can decide to show him a different example, or you can decide to ask him a question. It's okay to respond to his question with a question.

[Randomized order]

UNCONDITIONAL QUERIES

- * Now you will teach Simon the SNOWMAN. A SNOWMAN is an object with a small circular top and a circular bottom. The size of the bottom and colors don't matter.
- * In this session Simon might request specific examples from you after you show him examples.
- * [Reminder]

CONDITIONAL QUERIES

- * Now you will teach Simon the ALIEN. An ALIEN has a circular top and both the top and the bottom are green. The shape of the bottom and sizes don't matter.
- * In this session Simon might request specific examples from you after you show him examples.
- * [Reminder]

TEACHER TRIGGERED QUERIES

- * Now you will teach Simon ICE CREAM. ICE CREAM has a yellow triangular bottom and a round top. The color of the top and sizes don't matter.
- * Only for this session Simon can understand another sentence:
 - * Simon, do you have any questions?
 - * In response to this, Simon might ask for a specific example.
 - * [If this is the first after supervised: Just described the type of questions Simon might ask]
 - * [Otherwise: refer to the questions Simon was asking in the previous session]
- * [Reminder]

APPENDIX C

LEARNING CURVES FOR INDIVIDUALS IN EXP. 5

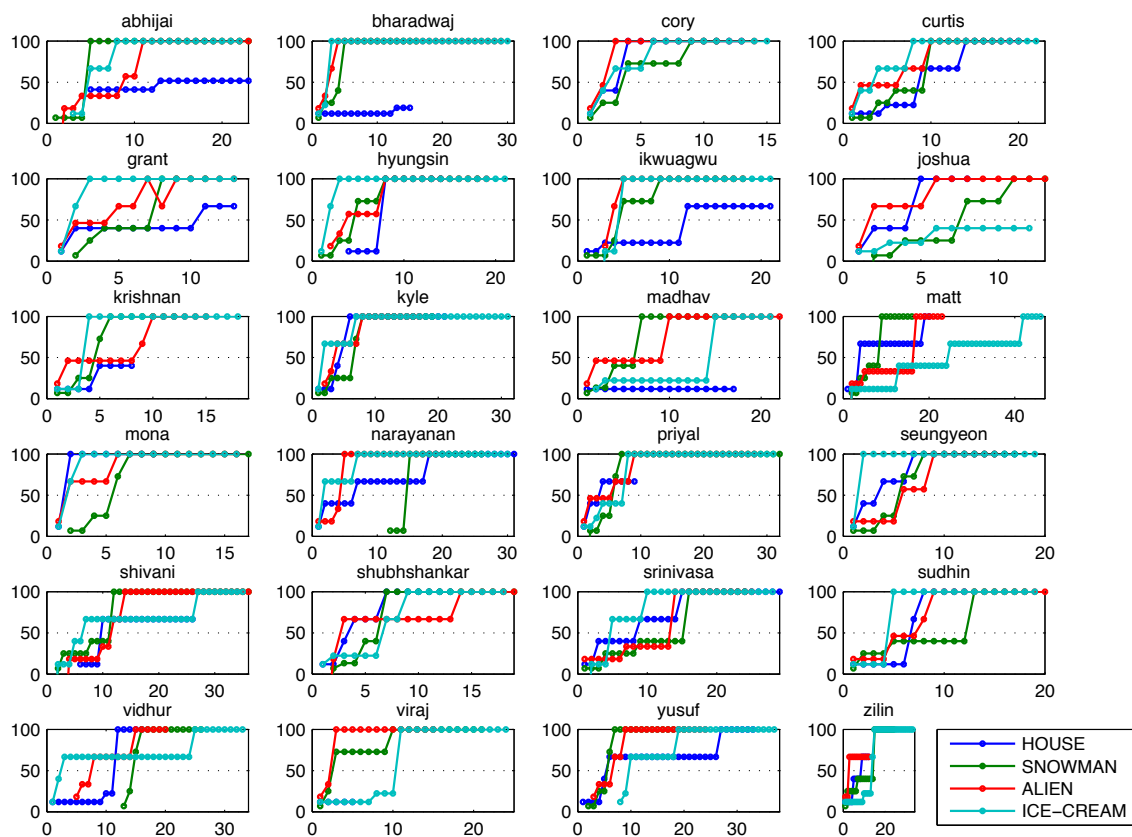


Figure 59: F-score over time for individuals in Experiment 5 including tests made by the participant.

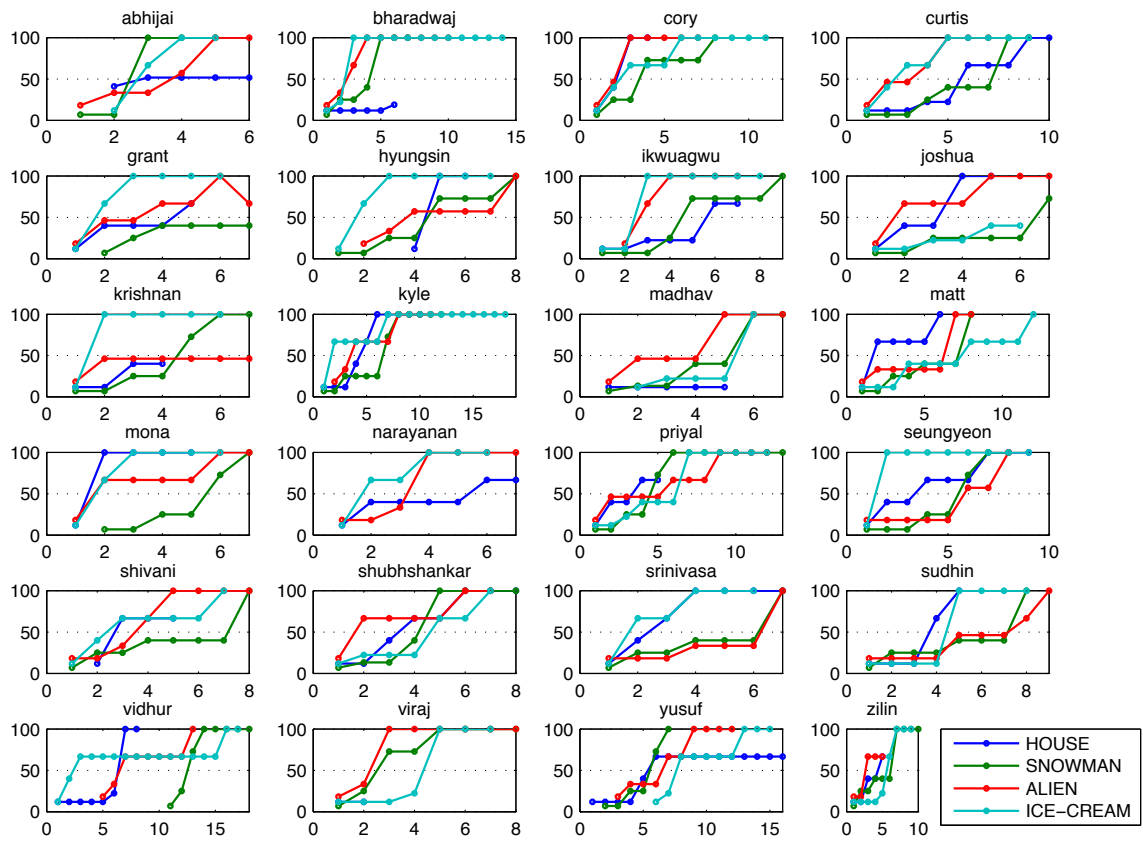


Figure 60: F-score over time for individuals in Experiment 5 excluding tests made by the participant.

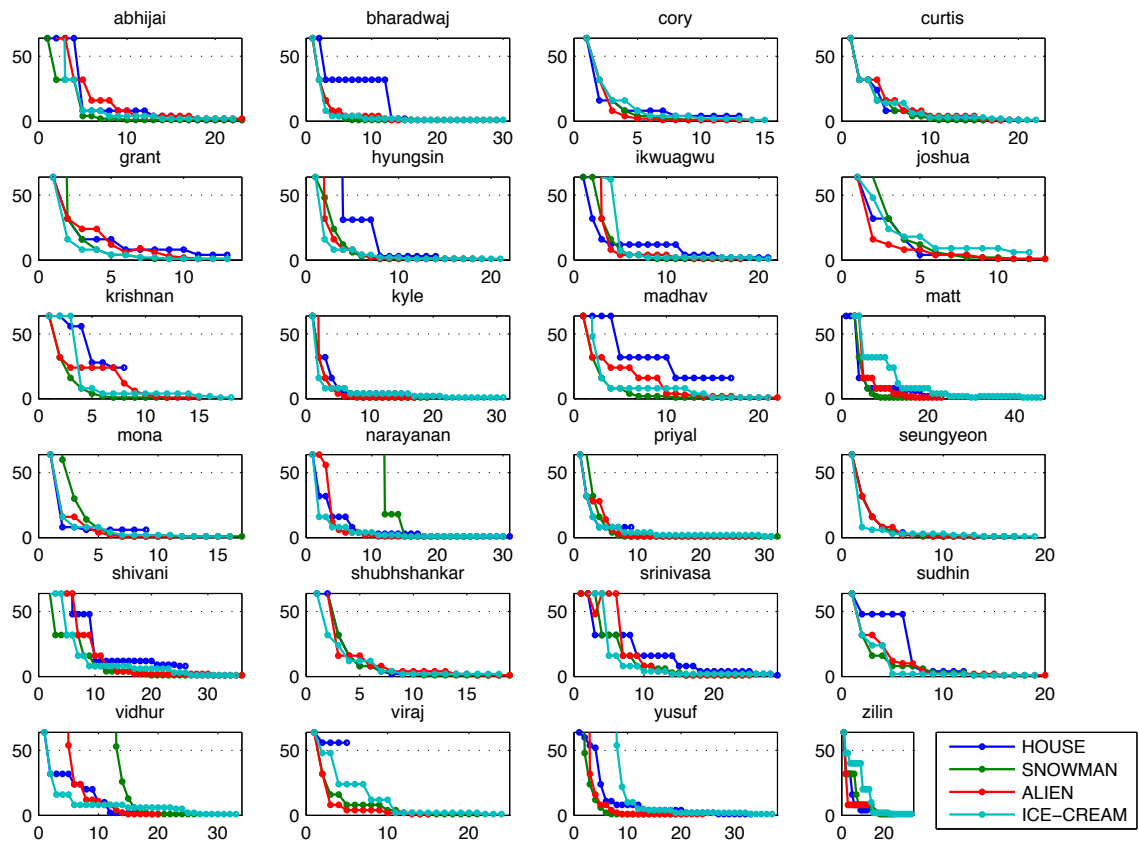


Figure 61: Size of the version space over time for individuals in Experiment 5 including tests made by the participant.

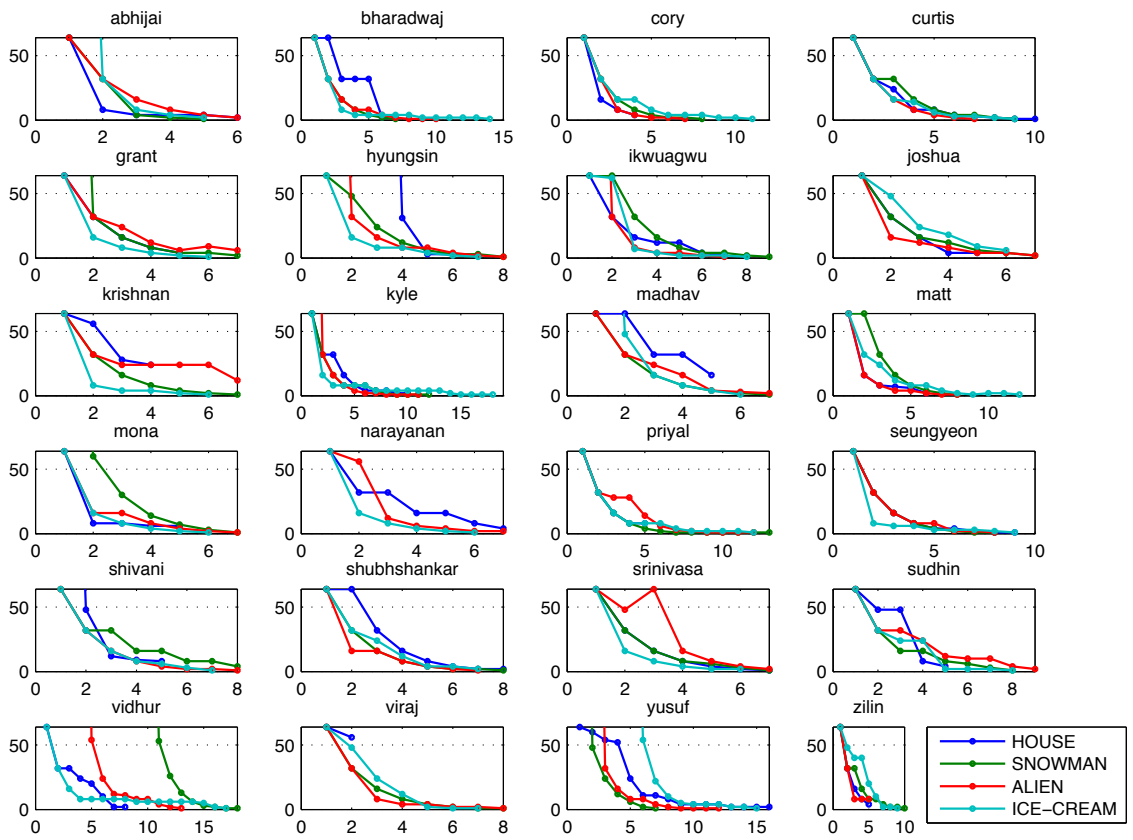


Figure 62: Size of the version space over time for individuals in Experiment 5 excluding tests made by the participant.

APPENDIX D

SURVEY IN EXPERIMENT 6

		Please rate Simon's questions in terms of informativeness for the robot.								
			1	2	3	4	5	6	7	
Q1	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
Q2	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
		Please rate Simon's questions in terms of unexpectedness .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
Q2	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
		Please rate Simon's questions in terms of ease of answering .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer
Q2	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer
		Please rate Simon's questions in terms of informativeness for the robot.							Please explain.	
			1	2	3	4	5	6	7	
Q1	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
Q2	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
		Please rate Simon's questions in terms of unexpectedness .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
Q2	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
		Please rate Simon's questions in terms of ease of answering .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer
Q2	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer
		Please rate Simon's questions in terms of informativeness for the robot.							Please explain.	
			1	2	3	4	5	6	7	
Q1	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
Q2	Not informative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very informative
		Please rate Simon's questions in terms of unexpectedness .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
Q2	Predictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Unexpected
		Please rate Simon's questions in terms of ease of answering .							Please explain.	
			1	2	3	4	5	6	7	
Q1	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer
Q2	Difficult to answer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Easy to answer

Simon asked you three types of questions. Please rank the three types in terms of the following criteria.

	<i>Can I pour cereal like this?</i>	<i>How do I add salt starting like this?</i>	<i>Does this matter for pouring coke?</i>	Please explain.
Which one is the smartest?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	
Which one is the least smart?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	

	<i>Can I pour cereal like this?</i>	<i>How do I add salt starting like this?</i>	<i>Does this matter for pouring coke?</i>	Please explain.
Which one is the easiest to answer?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	
Which one is the hardest to answer?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	

	<i>Can I pour cereal like this?</i>	<i>How do I add salt starting like this?</i>	<i>Does this matter for pouring coke?</i>	Please explain.
Which is the most informative for the robot?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	
Which is the least informative for the robot?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	

APPENDIX E

PROTOCOL FOR EXPERIMENT 7

Instructions

- * Purpose: The goal of our research is to build robots that can learn new things from humans efficiently by asking good questions. In this study we want to investigate how humans ask questions while learning a task or a skill.
- * Formalities: Please read this consent form and sign at the last page. We will be recording the study for analysis, is it okay to use parts of the recording in our conference presentations/etc.?
- * Summary: We will teach you some new tasks and skills with these objects by giving you demonstrations, and we will ask you to perform what you learn from these demonstrations. Then we will let you ask questions about the task and we will ask you to perform the task once more.
- * Procedure:
 - (1) There are four tasks, all independent from one another.
 - (2) For each task, first I will show you what objects are involved.
 - (3) Then you will watch a video on this screen, that involves me demonstrating the task twice.
 - (4) You can watch the video as many times as you want.
 - (5) Then I will ask you if you have any questions about the task and answer them.
 - (6) Then I will place the objects in a certain way and ask you to perform the task once. Don't worry about doing the task wrongly, this is not a test, it is aimed at making you think about the task and more questions to ask.
 - (7) Then I will let you ask more questions, if you have any. You can ask as many questions as you want, but we need at least three questions. Feel free to use the objects in your questions, for example you can say 'Can I do this?' using the object.
 - (8) Once I have answered all your questions I will set up the objects again and ask you to do the task once more.
- * Incentive: Try to ask questions that will help you make sure you perform the task correctly. The participant who performs all tasks most accurately after the questions will receive an extra compensation of 10\$.

Procedure

- * Give instructions above
- * Start recording
- * For each task/skill:
 - * Place all objects on the table
 - * Let participant watch the video, note the number of times they watch it
 - * Let participant ask questions (possibly referring to objects), make sure you understand the questions, answer the question. Note down question/answer in the experiment sheet.
 - * Setup test scenario #1, and ask the participant to perform the task/skill
 - * Place all objects on the table
 - * Let participant ask questions and answer, take notes.
 - * Setup test scenario #2, and ask the participant to perform the task/skill

APPENDIX F

MECHANICAL TURK HIT FOR EXPERIMENT 10

Instructions

Hello Captain! We are sending our robot R2D2 on a mission to deliver medication to humans on planet Hathrox III. For this mission R2D2 will need to be able to recognize angry human faces since the medicine is intended for them. So you need to teach him how an **angry face** looks like on any person. We don't want R2D2 to deliver the medicine to the wrong humans.

R2D2 represents faces with 16 properties (see all properties on the left in the java applet below). An angry face is defined as a face which

- has "Brow shape" with outer ends higher than the inner ends, and
- has "Mouth shape" with the ends lower than the middle.

These properties are highlighted as light blue in the applet.

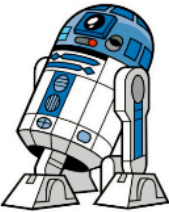
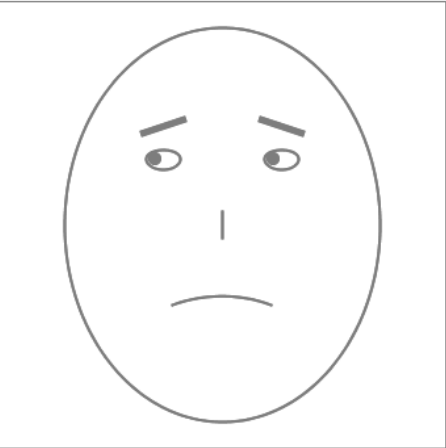
You will teach R2D2 by showing him examples of angry and non-angry faces. Click on the properties on the left to change their values, and use the left button to tell R2D2 what the configured face is. To see what R2D2 has learned, use the "?" button to ask him if the configured face is an angry face.

Notes:

- The examples and questions you give R2D2 will be evaluated by a researcher. Please make sure you understand your task and do not provide random examples and questions.
- After you train R2D2, he will be tested on the classification of all possible faces. A bonus of 4\$ will be awarded to the teacher who trains the best performing robot with the least number of examples and questions.
- When you are done teaching please answer the three questions given below the applet.
- Allow about 2 minutes for the java applet to load. If you get errors or you are unable to use the applet as expected, try reloading the page or restarting your browser. If the problem persists you can describe the problem in the box at the bottom of the page to get credit for this HIT.
- After showing an example, allow 2-10 seconds for R2D2 to process your example. The speech bubble will disappear when R2D2 is ready for the next example.

Interaction

Head Shape		
Head Size		
Eye Shape		
Eye Size		
Eye Distance		
Eye Location		
Pupil Location		
Pupil Size		
Brow Shape		
Brow Thickness		
Brow Location		
Nose Size		
Mouth Shape		
Mouth Width		
Mouth Location		
Mouth Opening		



NOT angry face

?

Questions

1) Your age

2) How did you decide that you were done teaching?

3) Briefly describe your teaching strategy.

Problems? If you were unable to complete the HIT due to a persisting problem, please give a description of the problem. Make sure to include your operating system and browser information.

APPENDIX G

MECHANICAL TURK HIT FOR EXPERIMENT 11

Instructions

Hello Captain! We are sending our robot R2D2 on a mission to collect bird eggs on planet Hathrox III. But first you need to teach him what the eggs look like. We don't want him coming back with snake eggs. Bird eggs are smooth, whereas the snake eggs are spiky. The spikiest bird egg looks like:

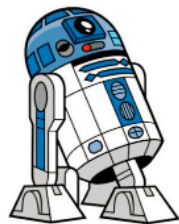


You will teach R2D2 by showing examples of **bird** and **snake** eggs (through the java applet below). Use the arrows to change the sample egg and click on the speech bubble to tell him what kind of egg it is. To see what R2D2 has learned, ask him to tell you what the sample egg is. Try not to show redundant examples or ask redundant questions. You can stop teaching when you think he will not make mistakes during his mission.

Notes:

- The examples and questions you give R2D2 will be evaluated by a researcher. Please make sure you understand your task and do not provide random examples and questions.
- After you train R2D2, he will be tested on the classification of a large set of eggs. A bonus of 4\$ will be awarded to the teacher who trains the best performing robot with the least number of examples and questions.
- When you are done teaching please answer the three questions given below the applet.
- Allow about 2 minutes for the java applet to load. If you get errors or you are unable to use the applet as expected, try reloading the page or restarting your browser. If the problem persists you can describe the problem in the box at the bottom of the page to get credit for this HIT.

Interaction



APPENDIX H

MECHANICAL TURK HIT FOR EXPERIMENT 12

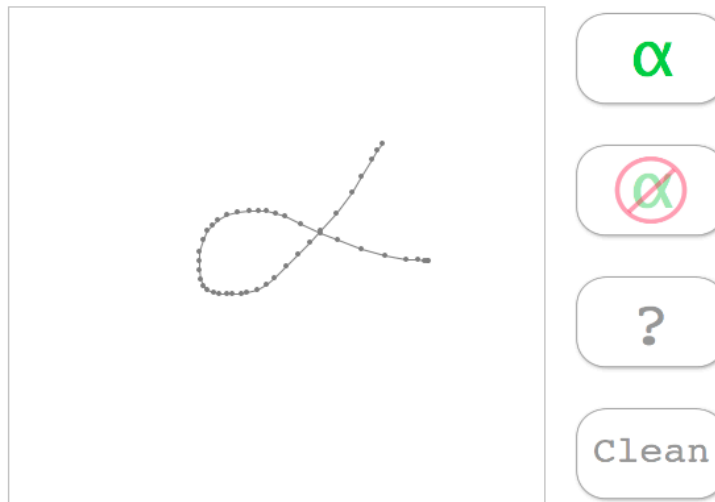
Instructions

Hello! In this task you will teach your computer a new mouse gesture using the interface below. The mouse gesture that you will teach is alpha, which looks like α . You will teach by showing examples of alpha gestures as well as gestures that are not alpha. These include the six gestures $\rho, \sigma, \omega, \beta, \gamma, \varphi$. Use the mouse to make the gesture in the square area and use the buttons on the left to submit them as examples of alpha or non-alpha. In order to observe what has been learned, you can make a gesture and press the "?" button to see what it is classified as. Try not to show redundant examples or ask redundant questions. You can stop teaching when you think the computer can reliably recognize the gestures.

Notes:

- The examples and questions you provide will be evaluated by a researcher. Please make sure you understand your task and do not provide random examples and questions.
- After you train your gesture recognizer, it will be tested on the classification of a large set of gestures. A bonus of 4\$ will be awarded to the teacher who trains the best performing recognizer with the least number of examples and questions.
- When you are done teaching please answer the three questions given below the applet.
- Allow about 1 minute for the java applet to load. If you get errors or you are unable to use the applet as expected, try reloading the page or restarting your browser. If the problem persists you can describe the problem in the box at the bottom of the page to get credit for this HIT.
- After showing an example, allow 2-10 seconds for the recognizer to process your example. The speech bubble will disappear when it is ready for the next example.

Interaction



APPENDIX I

MECHANICAL TURK HIT FOR EXPERIMENT 13

Instructions

Hello Captain! We are sending our robot R2D2 on a mission on planet Hathrox III. But first you need to teach him how to navigate the terrain on this planet. There is a sample map of the planet below.

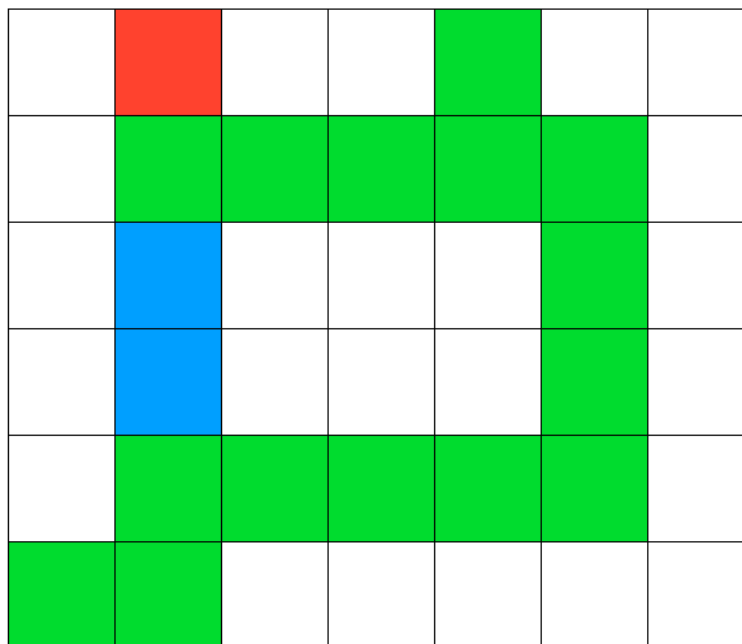
- Gray colored parts of the map are the walls. R2D2 cannot go through these.
- The red square shows the target location R2D2 needs to get to.
- The green squares are pavements which allow easy navigation for R2D2.
- The blue squares are gravel which are more difficult to navigate.
- Navigating through one blue square uses 10 times more battery energy than going through a green square.

You will train R2D2 by showing him one example path to the target. When you click on a square in the map the best path that goes to the target will be automatically drawn on the map. Clicking on a different square will show paths that starts from that square. When you have decided which path you want to show to R2D2 click on submit. Try to choose the most informative path.

Notes:

- The examples you give R2D2 will be evaluated by a researcher. Please make sure you understand your task and do not provide random examples.
- After you train R2D2, he will be tested different maps and starting positions. A bonus of 4\$ will be awarded to the teacher who trains the best performing robot with the least number of examples and questions.
- When you are done teaching please answer the questions given below the applet.
- Each worker is allowed to complete this HIT once. If a worker completes this HIT more than once, only one of them will be approved.

Interaction



demonstrate

APPENDIX J

PROTOCOL FOR EXPERIMENT 14

- 1) **DESCRIBE PURPOSE:** "The goal of our research is to build robots that can learn new things from humans efficiently. In this study we ask you to teach three skills to our robot Simon."
- 2) **GET TWO SIGNATURES:** "Please read this consent form and sign at the last page. We will be recording the study for analysis, is it okay to use parts of the recording in our conference presentations/etc.?"
- 3) **SUMMARIZE STUDY:** "First I'll ask you to watch a video that demonstrates the different speech commands you can use. Then we will do some practice. Then you will provide N demonstrations for three different skills. At the end you will be able to see what Simon has learned. And finally we have a short survey."
- 4) **SHOW VIDEO:** "Watch this video and let me know if you have any questions."
- 5) **START SERVER and C6 SCRIPT ON RTPC**
- 6) **START C6 with CORRECT PARTICIPANT ID, CONNECT TO ROBOT, PULL KILL SWITCH**
- 7) **EQUIPE MICROPHONE and TEST:** "Put on this microphone. There is also reminders of the commands here. I will show you flash cards for any other commands."
-- [FLASH CARD] Simon can you hear me?
- 8) **PRACTICE:**
Now let's try to repeat everything you saw in the video.
-- [FLASH CARD] Let's do some practice.
-- Release arm, hold arm, close your hand, open your hand.
-- Same for LEFT.
-- [FLASH CARD] Let's learn a new skill.
-- Start like this. Then go here. Then open your hand. Finish like this.
-- [FLASH CARD] Can you try it yourself?
- 9) **INCENTIVE/GUIDANCE:** "The participant that teaches the most successful skills will receive an extra reward of 15\$. By successful skills we mean skills that successfully achieve the goal and are applicable in different scenarios.

[GUIDED CONDITION] **SHOW SECOND VIDEO:** "We have another video with some tips on how you can achieve most successful skills. Now watch this video and let me know if you have any questions."
"In summary try to show demonstrations that vary as much as possible."
- 10) **START VIDEO RECORDING**
- 11) **START DEMONSTRATION INTERACTION, TAKE NOTES**
-- [FLASH CARD] Let's begin the experiment.
- 12) **START EXECUTION INTERACTION:**
-- [FLASH CARD] Can you try to remove the lid of the coke bottle?
-- [FLASH CARD] Can you try to close the box?
-- [FLASH CARD] Can you try to pour the coffee beans?
- 13) **STOP VIDEO RECORDING, PRESS KILL SWITCH, STOP C6**
- 14) **GET SURVEY RESPONSE:** "Please fill out this survey."
- 15) **GIVE COMPENSATION AND GET SIGNATURE:** "Please sign this form to indicate that you got the 5\$."
- 16) **BACK UP INTERACTION DATA**

REFERENCES

- [1] ABBEEL, P. and NG, A., “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2004. [2.1](#)
- [2] ABBEEL, P. and NG, A. Y., “Apprenticeship learning via inverse reinforcement learning,” in *In Proceedings of the Twenty-first International Conference on Machine Learning*, ACM Press, 2004. [10.1.1](#)
- [3] AKGUN, B., CAKMAK, M., JIANG, K., and THOMAZ, A., “Keyframe-based learning from demonstration,” (*In press*) *Journal of Social Robotics, Special issue on Learning from Demonstration*, 2012. [3.3](#)
- [4] AKGUN, B., CAKMAK, M., WOOK YOO, J., and THOMAZ, L. A., “Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective,” in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2012. [3](#)
- [5] ANGLUIN, D., “Queries and concept learning,” *Machine Learning*, vol. 2, pp. 319–342, 1988. [1.2.1](#), [2.3](#), [2.4](#), [3.2.2](#), [9.1](#)
- [6] ANTHONY, M., BRIGHTWELL, G., and SHAWE-TAYLOR, J., “On specifying boolean functions by labelled examples,” *Discrete Applied Mathematics*, vol. 61, no. 1, pp. 1–25, 1995. [2.4](#), [9.1.1](#)
- [7] ARGALL, B., SAUSER, E., and BILLARD, A., “Tactile guidance for policy refinement and reuse,” in *IEEE International Conference on Development and Learning*, pp. 7–12, 2010. [2.1](#)
- [8] ARGALL, B., CHERNOVA, S., VELOSO, M. M., and BROWNING, B., “A survey of robot learning from demonstration,” *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009. [2.1](#), [2.1](#), [3.3](#), [4.4.1.4](#), [4.4.2.4](#), [10.1.1](#)
- [9] ASHBY, F. and MADDOX, W., “Human category learning,” *Annual Review of Psychology*, vol. 56, pp. 149–178, 2004. [2.5](#)
- [10] ATKESON, C. G. and SCHAAL, S., “Robot learning from demonstration,” in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 12–20, Morgan Kaufmann, 1997. [2.1](#), [3.3](#)
- [11] BALBACH, F., “Measuring teachability using variants of the teaching dimension,” *Theoretical Computer Science*, vol. 397, no. 1–3, pp. 94–113, 2008. [2.4](#), [9.1.1](#)

- [12] BALBACH, F. and ZEUGMANN, T., “Teaching randomized learners,” in *Proceedings of the Annual Conference on Learning Theory (COLT)*, pp. 229–243, 2006. [2.4](#), [9.1.1](#)
- [13] BALBACH, F. and ZEUGMANN, T., “Recent developments in algorithmic teaching,” in *Proceedings of the International Conference on Language and Automata Theory and Applications*, pp. 1–18, 2009. [1.2.2](#), [2.4](#), [9.1](#)
- [14] BALCAN, M., HANNEKE, S., and VAUGHAN, J., “The true sample complexity of active learning,” *Machine Learning*, vol. 80, pp. 111–139, 2010. [2.3](#)
- [15] BENGIO, Y., LOURADOUR, J., COLLOBERT, R., and WESTON, J., “Curriculum learning,” in *Proceedings of the International Conference on Machine Learning (ICML)* (BORTOU, L. and LITTMAN, M., eds.), (Montreal), pp. 41–48, June 2009. [2.5](#), [9.7](#)
- [16] BILLARD, A., CALINON, S., DILLMANN, R., and SCHAAL, S., “Robot programming by demonstration,” in *Handbook of Robotics* (SICILIANO, B. and KHATIB, O., eds.), Springer, 2007. [2.1](#), [2.1](#)
- [17] BILLARD, A., CALINON, S., and GUENTER, F., “Discriminative and adaptive imitation in uni-manual and bi-manual tasks,” *Robotics and Autonomous System*, vol. 54, no. 5, pp. 370–384, 2006. [2.1](#)
- [18] BLUMBERG, B., DOWNIE, M., IVANOV, Y., BERLIN, M., JOHNSON, M., and TOMLINSON, B., “Integrated learning for interactive synthetic characters,” in *Proceedings of the ACM SIGGRAPH*, 2002. [3.1.3](#)
- [19] BREAZEAL, C. and THOMAZ, A. L., “Learning from human teachers with socially guided exploration,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008. [2.1](#)
- [20] BROCHU, E., DEFREITAS, N., and GHOSH, A., “Active preference learning with discrete choice data,” in *Advances in Neural Information Processing Systems (NIPS)*, 2007. [13.2](#)
- [21] BUCK, S., BEETZ, M., and SCHMITT, T., “Issues in human/robot task structuring and teaching,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1062–1067, 2002. [2.1](#)
- [22] CAKMAK, M., CHAO, C., and THOMAZ, A., “Designing interactions for robot active learners,” *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 2, pp. 108–118, 2010. [1](#)
- [23] CAKMAK, M. and LOPES, M., “Algorithmic and human teaching of sequential decision tasks,” in *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2006. [1](#)
- [24] CAKMAK, M. and THOMAZ, A. L., “Designing robot learners that ask good questions,” in *Proceedings of the International Conference on Human-Robot Interaction (HRI)*, 2012. [1](#), [2](#)

- [25] ÇAKMAK, M. and THOMAZ, A., “Optimality of human teachers for robot learners,” in *Proceedings of the IEEE International Conference on Development and Learning (ICDL)*, pp. 64 – 69, 2010. [1](#), [1](#)
- [26] ÇAKMAK, M. and THOMAZ, A., “Active learning with mixed query types in learning from demonstration,” in *Proceedings of the ICML Workshop on New Developments in Imitation Learning*, 2011. [1](#)
- [27] ÇAKMAK, M. and THOMAZ, A., “Mixed-initiative active learning,” in *ICML Workshop on Combining Learning Strategies to Reduce Label Cost*, 2011. [1](#)
- [28] ÇAKMAK, M. and THOMAZ, A., “Eliciting good teaching from human teachers for machine learners,” (*In revision*) *Artificial Intelligence*, 2012. [3](#), [5](#), [7](#)
- [29] CALINON, S. and BILLARD, A., “Incremental learning of gestures by imitation in a humanoid robot,” in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 255–262, 2007. [2.1](#)
- [30] CALINON, S. and BILLARD, A., “What is the teacher’s role in robot programming by demonstration? - Toward benchmarks for improved learning,” *Interaction Studies. Special Issue on Psychological Benchmarks in Human-Robot Interaction*, vol. 8, no. 3, 2007. [3.3](#)
- [31] CALINON, S. and BILLARD, A., “Statistical learning by imitation of competing constraints in joint and task space,” *Advanced Robotics*, vol. 23, no. 15, pp. 2059–2076, 2009. [2.1](#), [7.1.2](#)
- [32] CALINON, S., GUENTER, F., and BILLARD, A., “On learning, representing and generalizing a task in a humanoid robot,” *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, vol. 37, no. 2, pp. 286–298, 2007. [3.3](#)
- [33] CANTRELL, R., SCHERMERHORN, P., and SCHEUTZ, M., “Learning actions from human-robot dialogues,” in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2011. [2.3.1](#)
- [34] CARO, T. and HAUSER, M., “Is there teaching in nonhuman animals?,” *Quarterly Review of Biology*, vol. 67, pp. 151–174, 1992. [2.5](#)
- [35] CASTRO, R., KALISH, C., NOWAK, R., QIAN, R., ROGERS, T., and ZHU, X., “Human active learning,” in *Advances in Neural Information Processing Systems (NIPS)*, pp. 241–248, 2008. [2.5](#), [9.5.1.1](#)
- [36] CHANG, C.-C. and LIN, C.-J., *LIBSVM: a library for support vector machines*, 2001. [4.3.1.2](#)
- [37] CHAO, C., ÇAKMAK, M., and THOMAZ, A., “Transparent active learning for robots,” in *Proceedings of the ACM International Conference on Human-Robot Interaction (HRI)*, 2010. [1](#)

- [38] CHERNOFF, H., “The use of faces to represent points in k-dimensional space graphically,” *Journal of the American Statistical Association*, vol. 68, no. 342, pp. 361–368, 1973. [9.4.1.1](#)
- [39] CHERNOVA, S. and VELOSO, M., “Confidence-based policy learning from demonstration using gaussian mixture models,” in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2007. [2.1](#), [3.3](#)
- [40] CHERNOVA, S. and VELOSO, M., “Multi-thresholded approach to demonstration selection for interactive robot learning,” in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2008. [1.2.1](#), [2.1](#), [2.3.1](#), [7.1](#)
- [41] CHVATAL, V., “A greedy heuristic for the set-covering problem,” *Mathematics of Operations Research*, vol. 4, pp. 233–235, August 1979. [9.1.3.1](#)
- [42] COHN, D., GHAHRAMANI, Z., and JORDAN, M., “Active learning with statistical models,” in *Advances in Neural Information Processing Systems (NIPS)* (TESAURO, G., TOURETZKY, D., and ALSPECTOR, J., eds.), vol. 7, Morgan Kaufmann, 1995. [1.2.1](#), [2.3](#)
- [43] COHN, D. A., CARUANA, R., and McCALLUM, A., “Semi-supervised clustering with user feedback,” Tech. Rep. TR2003-1892, Cornell University, 2003. [2.3](#)
- [44] CSIBRA, G., “Teachers in the wild,” *Trends in Cognitive Sciences*, vol. 11, no. 3, pp. 95–95, 2006. [2.5](#), [4](#)
- [45] CULOTTA, A., KRISTJANSSON, T., McCALLUM, A., and VIOLA, P., “Corrective feedback and persistent learning for information extraction,” *Artificial Intelligence*, 2006. [2.2](#), [2.3](#)
- [46] CULOTTA, A. and McCALLUM, A., “Reducing labeling effort for structured prediction tasks,” in *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*, pp. 746–751, 2005. [2.3](#)
- [47] DASGUPTA, S., “Coarse sample complexity bounds for active learning,” in *Advances in Neural Information Processing Systems (NIPS)*, pp. 235–242, 2006. [\(document\)](#), [2.3](#), [9.1](#), [42](#)
- [48] DENIS, F. and GILLERON, R., “Pac learning under helpful distributions,” in *Algorithmic Learning Theory (ALT)*, vol. 1316, pp. 132–145, Springer-Verlag, 1997. [2.4](#), [9.1](#)
- [49] DENIZ, O., LORENZO, J., CASTRILLON, M., MENDEZ, J., and FALCON, A., “Learning to recognize faces incrementally,” in *Proceedings of the DAGM Conference on Pattern Recognition*, (Berlin, Heidelberg), pp. 365–374, Springer-Verlag, 2007. [2.5](#), [4.3.2](#)
- [50] DIANA, C. and THOMAZ, A. L., “The shape of simon: creative design of a humanoid robot shell,” in *Proceedings of the 2011 Annual Conference Extended Abstracts on Human factors in Computing Systems*, (New York, NY, USA), pp. 283–298, ACM, 2011. [3.1.1](#)

- [51] DRUCK, G., SETTLES, B., and McCALLUM, A., “Active learning by labeling features,” in *Proceedings of the Empirical Methods in Natural Language Processing*, pp. 81–90, 2009. [2.3](#), [7.1.3](#)
- [52] ELMAN, J., “Learning and development in neural networks: the importance of starting small,” *Cognition*, vol. 48, pp. 71–99, 1993. [2.5](#), [4.3.2](#)
- [53] FAILS, J. and OLSEN, D., “A design tool for camera-based interaction,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 449–456, 2003. [2.2](#)
- [54] FOGARTY, J., TAN, D., KAPOOR, A., and WINDER, S., “Cueflik: interactive concept learning in image search,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 29–38, 2008. [2.2](#)
- [55] GERVASIO, M. and MYERS, K., “Question asking to inform procedure learning,” in *Proceedings of the AAAI Spring Symposium on Agents that Learn from Human Teachers*, 2009. [2.3](#), [7.3.2.1](#)
- [56] GERVASIO, M. and MYERS, K., “Learning to ask the right questions to help a learner learn,” in *Proceedings of the International Conference on Intelligent User Interfaces (IUI)*, 2011. [2.3](#)
- [57] GOLDMAN, S. and KEARNS, M., “On the complexity of teaching,” *Computer and System Sciences*, vol. 50, pp. 20–31, February 1995. [1.2.2](#), [2.4](#), [2.5](#), [9.1](#), [9.1.1](#), [9.1.1.1](#), [9.1.1.1](#), [12](#)
- [58] GRIBOVSKAYA, E., d’HALLUIN, F., and BILLARD, A., “An active learning interface for bootstrapping robot’s generalization abilities in learning from demonstration,” in *RSS Workshop Towards Closing the Loop: Active Learning for Robotics*, 2010. [1.2.1](#), [2.3.1](#)
- [59] GROLLMAN, D. H. and JENKINS, O. C., “Sparse incremental learning for interactive robot control policy estimation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008. [2.1](#)
- [60] GROLLMAN, D. and BILLARD, A., “Donut as I do: Learning from failed demonstrations,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011. [7.1.1](#)
- [61] GROLLMAN, D. and JENKINS, O., “Dogged learning for robots,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007. [1.2.1](#), [2.3.1](#), [3.3](#), [7.1](#)
- [62] GUILLORY, A. and BILMES, J., “Simultaneous learning and covering with adversarial noise,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2011. [5.2.1](#)

- [63] HALL, M., FRANK, E., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., and WITTEN, I., “The weka data mining software: An update,” *SIGKDD Explorations*, vol. 11, no. 1, 2009. [4.3.1.2](#)
- [64] HIBY, E., ROONEY, N., and BRADSHAW, J., “Dog training methods: their use, effectiveness and interaction with behavior and welfare,” *Animal Welfare*, vol. 13, no. 1, pp. 63–69, 2004. [9](#)
- [65] HUANG, Y. and MITCHELL, T. M., “Text clustering with extended user feedback,” in *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 413–420, 2006. [2.2](#)
- [66] ISBELL, C., SHELTON, C., KEARNS, M., SINGH, S., and STONE, P., “Cobot: A social reinforcement learning agent,” in *Proceedings of the International Conference on Autonomous Agents*, pp. 377–384, 2001. [2.2](#)
- [67] JACKSON, J. and TOMKINS, A., “A computational model of teaching,” in *Proceedings of the Annual Conference on Learning Theory (COLT)*, pp. 20–31, 1992. [2.4](#), [9.1.1](#)
- [68] JETCHEV, N. and TOUSSAINT, M., “Task space retrieval using inverse feedback control,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2011. [7.1.3](#)
- [69] JUDAH, K., FERN, A., and DIETTERICH, T., “Active imitation learning via state queries,” in *ICML Workshop on Combining Learning Strategies to Reduce Label Cost*, 2011. [2.1](#)
- [70] KAEHLING, L. P., LITTMAN, M. L., and MOORE, A. P., “Reinforcement learning: A survey,” *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996. [4.2.2](#)
- [71] KAISER, M., FRIEDRICH, H., and DILLMANN, R., “Obtaining good performance from a bad teacher,” in *International Conference on Machine Learning, Workshop on Programming by Demonstration*, 1995. [4.4.1.4](#)
- [72] KAPOOR, A., LEE, B., TAN, D., and HORVITZ, E., “Interactive optimization for steering machine classification,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1343–1352, 2010. [2.2](#)
- [73] KEARSLEY, G. P., “Questions and question asking in verbal discourse: A cross-disciplinary review,” *Journal of Psycholinguistic Research*, vol. 5, pp. 355–375, 1976. ([document](#)), [7.2.2.2](#), [7.3.1.3](#), [30](#)
- [74] KEMP, C., GOODMAN, N., and TENENBAUM, J., “Learning and using relational theories,” in *Advances in Neural Information Processing Systems (NIPS)*, pp. 753–760, 2008. [2.5](#)
- [75] KHAN, F., ZHU, X., and MUTLU, B., “How do humans teach: On curriculum learning and teaching dimension,” in *Advances in Neural Information Processing Systems (NIPS)*, 2011. [2.5](#), [9.5.1.1](#), [9.5.2](#), [9.7](#)

- [76] KHANSARI-ZADEH, S. M. and BILLARD, A., “Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models,” *IEEE Transaction on Robotics*, 2011. [3.3](#)
- [77] KITTUR, A., CHI, E. H., and SUH, B., “Crowdsourcing user studies with mechanical turk,” in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 453–456, 2008. [9.3](#)
- [78] KNOX, W. and STONE, P., “Training a tetris agent via interactive shaping: a demonstration of the tamer framework,” in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1767–1768, 2010. [2.2](#)
- [79] KRAUSE, A. and GUESTRIN, C., “Near-optimal nonmyopic value of information in graphical models,” in *Uncertainty in AI*, 2005. [10.1.2](#)
- [80] KRAUSE, A. and GUESTRIN, C., “A note on the budgeted maximization of submodular functions.” 2005. [10.1.2](#)
- [81] KRISTJANSSON, T., CULOTTA, A., VIOLA, P., and MCCALLUM, A., “Interactive information extraction with constrained conditional random fields,” in *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 412–418, 2004. [2.2](#)
- [82] KROEMER, O., DETRY, R., PIATER, J., and PETERS, J., “Active learning using mean shift optimization for robot grasping,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2610–2615, 2009. [2.3.1](#)
- [83] LAVE, J. and WENGER, E., *Situated learning: legitimate peripheral participation*. Cambridge: Cambridge University Press, 1991. [4](#)
- [84] LEE, Y. and GRAUMAN., K., “Learning the easy things first: Self-paced visual category discovery,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1721–1728, 2011. [2.5](#), [4.3.2](#)
- [85] LITTLESTONE, N., “Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm,” *Machine Learning*, vol. 2, pp. 285–318, 1988. [3.2.1.2](#), [9.1.3](#), [9.4.1.1](#)
- [86] LOMASKY, R., BRODLEY, C., AERNECKE, M., WALT, D., and FRIEDL, M., “Active class selection,” in *Proceedings of the European Conference on Machine Learning (ECML)*, pp. 640–647, Springer-Verlag, 2007. [7.1.2](#)
- [87] LOPES, M., MELO, F., and MONTESANO, L., “Active learning for reward estimation in inverse reinforcement learning,” in *Proceedings of the European Conference on Machine Learning (ECML)*, 2009. [2.3.1](#)
- [88] LOPES, M. and OUDEYER., P., “Active learning and intrinsically motivated exploration in robots: Advances and challenges,” *IEEE Trans. on Autonomous Mental Development*, vol. 2, June 2010. [2.3.1](#)

- [89] MACGREGOR, J., “The effects of order on learning classifications by example: Heuristics for finding the optimal order,” *Artificial Intelligence*, vol. 34, no. 3, pp. 361–370, 1988. [9.1.3](#), [9.1.3.1](#)
- [90] MARTINEZ-CANTIN, R., PETERS, J., and KRAUSE, A., eds., *RSS Workshop Towards Closing the Loop: Active Learning for Robotics*. 2010. [2.3.1](#)
- [91] MASCOLO, M., “Change processes in development: The concept of coactive scaffolding,” *New Ideas in Psychology*, vol. 23, pp. 185–196, 2005. [2.5](#), [4](#), [4.3.2](#)
- [92] MATHIAS, H. D., “A model of interactive teaching,” *Computer and System Sciences*, vol. 54, no. 3, pp. 487–501, 1997. [2.4](#), [9.1](#)
- [93] McCANDLISS, B., FIEZ, J., PROTOPAPAS, A., CONWAY, M., and McCLELLAND, J., “Success and failure in teaching the [r]-[l] contrast to Japanese adults: Tests of a Hebbian model of plasticity and stabilization in spoken language perception,” *Cognitive, Affective and Behavioral Neuroscience*, vol. 2, no. 2, pp. 89–108, 2002. [9.7](#)
- [94] MELO, F., LOPES, M., SANTOS-VICTOR, J., and RIBEIRO, M., “A unified framework for imitation-like behaviors,” in *Proceedings of the International Symposium on Imitation in Animals and Artifacts*, 2007. [2.1](#)
- [95] MERICLI, C., VELOSO, M., and AKIN, H. L., “Task refinement for autonomous robots using complementary corrective human feedback,” *International Journal of Advanced Robotic Systems*, 2011. [2.1](#)
- [96] MONTESANO, L., LOPES, M., BERNARDINO, A., and SANTOS-VICTOR, J., “Learning object affordances: From sensory-motor coordination to imitation,” *IEEE Transactions on Robotics*, vol. 24, pp. 15–26, 2008. [4.2.2](#)
- [97] MUHLIG, M., GIENGER, M., HELLBACH, S., STEIL, J., and GOERICK, C., “Automatic selection of task spaces for imitation learning,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009. [2.1](#), [7.1.3](#)
- [98] MUHLIG, M., GIENGER, M., HELLBACH, S., STEIL, J., and GOERICK, C., “Task-level imitation learning using variance-based movement optimization,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1177–1184, 2009. [2.1](#)
- [99] NATARAJAN, B., “On learning boolean functions,” in *Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 296–304, 1989. [2.4](#), [9.1.1](#)
- [100] NEMHAUSER, G., WOLSEY, L., and FISHER, M., “An analysis of approximations for maximizing submodular set functions,” *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978. [10.1.2](#)
- [101] NEU, G. and SZEPESVÁRI, C., “Training parsers by inverse reinforcement learning,” *Machine Learning*, vol. 77, no. 2, pp. 303–337, 2009. [10.1.1](#)

- [102] NEU, G. and SZEPESVÁRI, C., “Apprenticeship learning using inverse reinforcement learning and gradient methods,” in *Uncertainty in Artificial Intelligence (UAI)*, pp. 295–302, 2007. [10.1.3](#)
- [103] NG, A. Y. and RUSSELL, S. J., “Algorithms for inverse reinforcement learning,” in *Proceedings of the International Conference Machine Learning*, 2000. [10.1.1](#)
- [104] OTERO, J. and GRAESSER, A., “Preg: Elements of a model of question asking,” *Cognition and Instruction*, vol. 19, no. 2, pp. 143–175, 2001. [7.2.2.2](#)
- [105] OUDEYER, P.-Y. and KAPLAN, F., “Intelligent adaptive curiosity: a source of self-development,” in *Proceedings of the International Workshop on Epigenetic Robotics*, vol. 117, pp. 127–130, 2004. [2.3.1](#)
- [106] OUDEYER, P.-Y., BARANES, A., and KAPLAN, F., “Intrinsically motivated exploration for developmental and active sensorimotor learning,” in *From Motor Learning to Interaction Learning in Robots* (SIGAUD, O. and PETERS, J., eds.), vol. 264 of *Studies in Computational Intelligence*, pp. 107–146, Springer Berlin / Heidelberg, 2010. [2.3.1](#)
- [107] PASTOR, P., HOFFMANN, H., ASFOUR, T., and SCHAAAL, S., “Learning and generalization of motor skills by learning from demonstration,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009. [2.1](#)
- [108] PETERS, R. A. and CAMPBELL, C. L., “Robonaut task learning through teleoperation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (Taipei, Taiwan), 2003. [2.1](#)
- [109] PRATT, G. and WILLIAMSON, M., “Series elastic actuators,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 399–406, 1995. [3.1.1](#)
- [110] RAGHAVAN, H., MADANI, O., and JONES, R., “Active learning with feedback on features and instances,” *J. of Machine Learning Research*, vol. 7, pp. 1655–1686, 2006. [7.1.3](#)
- [111] ROBBEL, P., TOUSSAINT, M., and VIJAYAKUMAR, S., “Active learning in motor control,” in *NIPS Workshop on Robotics Challenges for Machine Learning*, 2007. [2.3.1](#)
- [112] ROSENTHAL, S., DEY, A., and VELOSO, M., “How robots’ questions affect the accuracy of the human responses,” in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2009. [2.3.1](#), [5.2.4](#)
- [113] ROSENTHAL, S. and DEY, A., “Towards maximizing the accuracy of human-labeled sensor data,” in *Proceedings of the International Conference on Intelligent User Interfaces (IUI)*, pp. 259–268, 2010. [2.2](#), [2.3.1](#), [5.2.4](#)
- [114] ROSENTHAL, S. and VELOSO, M., “Modeling humans as observation providers using pomdps,” in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2011. [2.3.1](#)

- [115] ROSS, J., IRANI, L., SILBERMAN, M. S., ZALDIVAR, A., and TOMLINSON, B., “Who are the crowdworkers?: shifting demographics in mechanical turk,” in *Proceedings of the International Conference on Human factors in computing systems (Extended abstract)*, pp. 2863–2872, 2010. [9.3](#)
- [116] SAHIN, E., CAKMAK, M., DOGAR, M., UGUR, E., and UCOLUK, G., “To afford or not to afford: A new formalization of affordances toward affordance-based robot control,” *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007. [4.2.2](#)
- [117] SAUNDERS, J., NEHANIV, C., and DAUTENHAHN, K., “Teaching robots by moulding behavior and scaffolding the environment,” in *Proceedings of the ACM International Conference on Human-Robot Interaction (HRI)*, pp. 118–125, 2006. [2.1](#)
- [118] SAUNDERS, J., OTERO, N., and NEHANIV, C., “Issues in human/robot task structuring and teaching,” in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 708 – 713, 2007. [2.1](#)
- [119] SCHAAL, S., PETERS, J., NAKANISHI, J., and IJSPEERT., A., “Learning movement primitives,” in *Proceedings of the International Symposium on Robotics Research (ISRR)*, 2003. [2.1](#)
- [120] SELINGMAN, M., “On the generality of the laws of learning,” *Psychological Review*, vol. 77, no. 5, pp. 406–418, 1970. [9](#)
- [121] SETTLES, B., CRAVEN, M., and FRIEDLAND, L., “Active learning with real annotation costs,” in *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, pp. 1069–1078, ACL Press, 2008. [2.3](#)
- [122] SETTLES, B., “Active learning literature survey,” Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2010. [1.2.1](#), [2.3](#), [6.3.2.1](#)
- [123] SEUNG, H., OPPER, M., and SOMPOLINSKY, H., “Query by committee,” in *Proceedings of the ACM Workshop on Computational Learning Theory (COLT)*, pp. pages 287–294, 1992. [3.2.2](#)
- [124] SHAFTO, P. and GOODMAN, N., “Teaching games: Statistical sampling assumptions for learning in pedagogical situations,” in *Proceedings of the Annual Conference of the Cognitive Science Society*, pp. 1–6, 2008. [2.5](#), [9.7](#)
- [125] SHILMAN, M., TAN, D., and SIMARD, P., “Cuetip: a mixed-initiative interface for correcting handwriting errors,” in *Proceedings of the Annual ACM Symposium on User interface software and technology*, pp. 323–332, 2006. [2.2](#)
- [126] STONE, P. and VELOSO, M., “Layered learning,” in *Proceedings of the European Conference on Machine Learning (ECML)* (DE MANTARAS, R. L. and PLAZA, E., eds.), (Barcelona, Catalonia, Spain), pp. 369–381, Springer Verlag, May/June 2000. [2.5](#)

- [127] STUMPF, S., RAJARAM, V., LI, L., WONG, W., BURNETT, M., DIETTERICH, T., SULLIVAN, E., and HERLOCKER, J., “Interacting meaningfully with machine learning systems: Three experiments,” *International Journal of Human-Computer Studies*, vol. 67, no. 8, pp. 639 – 662, 2009. [2.2](#)
- [128] STUMPF, S., SULLIVAN, E., FITZHENRY, E., OBERST, I., WONG, W., and BURNETT, M., “Integrating rich user feedback into intelligent user interfaces,” in *Proceedings of the International Conference on Intelligent User Interfaces (IUI)*, pp. 50–59, 2008. [2.2](#)
- [129] SUTTON, R. S. and BARTO, A. G., *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 1998. [10.1.1](#)
- [130] TAYLOR, M., “Assisting transfer-enabled machine learning algorithms: Leveraging human knowledge for curriculum design,” in *Proceedings of the AAAI Spring Symposium on Agents that Learn from Human Teachers*, 2009. [2.5](#)
- [131] THOMAZ, A. L., *Socially Guided Machine Learning*. PhD thesis, Massachusetts Institute of Technology, 2006. [2.2](#)
- [132] THOMAZ, A. L. and BREAZEAL, C., “Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance,” in *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 1000–1005, 2006. [2.2](#), [9.7](#)
- [133] THOMAZ, A. L. and BREAZEAL, C., “Experiments in socially guided exploration: Lessons learned in building robots that learn with and without human teachers,” *Connection Science, Special Issue on Social Learning in Embodied Agents*, 2008. [2.2](#)
- [134] THOMAZ, A. and CAKMAK, M., “Learning about objects with human teachers,” in *HRI '09: Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, 2009. [4](#)
- [135] TOMASELLO, M., *The Cultural Origins of Human Cognition*. Harvard University Press, March 2001. [1.1.2](#), [4](#)
- [136] TONG, S. and KOLLER, D., “Support vector machine active learning with applications to text classification,” in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 999–1006, 2000. [2.3](#)
- [137] UDE, A., ATKESON, C. G., and RILEY, M., “Programming full-body movements for humanoid robots by observation,” *Robotics and Autonomous Systems*, vol. 47, pp. 93–108, 2004. [2.1](#)
- [138] VIJAYANARASIMHAN, S. and GRAUMAN, K., “What’s it going to cost you? predicting effort vs. informativeness for multi-label image annotations,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. [2.3](#)

- [139] VONAHN, L. and DABBISH, L., “Labeling images with a computer game,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 319–326, 2004. [2.2](#)
- [140] WANG, L., “A combined optimization method for solving the inverse kinematics problem of mechanical manipulators,” *IEEE Transactions on Robotics and Automation*, 1991. [3.1.3](#)
- [141] WARNER, R., STOESS, T., and SHAFTO, P., “Reasoning in teaching and misleading situations,” in *Proceedings of the Annual Conference of the Cognitive Science Society*, pp. 1430–1435, 2011. [2.5](#)
- [142] ZHU, X., ROGERS, T., and GIBSON, B., “Human rademacher complexity,” in *Advances in Neural Information Processing Systems (NIPS)*, pp. 2322–2330, 2009. [2.5](#), [9.5.1.1](#)
- [143] ZILLES, S., LANGE, S., HOLTE, R., and ZINKEVICH, M., “Teaching dimensions based on cooperative learning,” in *Proceedings of the Annual Conference on Learning Theory (COLT)*, 2008. [2.4](#)

Guided Teaching Interactions with Robots:
Embodied Queries and Teaching Heuristics

Maya Cakmak

277 Pages

Directed by Professor Andrea L. Thomaz

The vision of personal robot assistants continues to become more realistic with technological advances in robotics. The increase in the capabilities of robots, presents boundless opportunities for them to perform useful tasks for humans. However, it is not feasible for engineers to program robots for all possible uses. Instead, we envision general-purpose robots that can be programmed by their end-users.

Learning from Demonstration (LfD), is an approach that allows users to program new capabilities on a robot by demonstrating what is required from the robot. Although LfD has become an established area of Robotics, many challenges remain in making it effective and intuitive for naive users. This thesis contributes to addressing these challenges in several ways. First, the problems that occur in teaching-learning interactions between humans and robots are characterized through human-subject experiments in three different domains. To address these problems, two mechanisms for guiding human teachers in their interactions are developed: *embodied queries* and *teaching heuristics*.

Embodied queries, inspired from Active Learning queries, are questions asked by the robot so as to steer the teacher towards providing more informative demonstrations. They leverage the robot's embodiment to physically manipulate the environment and to communicate the question. Two technical contributions are made in developing embodied queries. The first is *Active Keyframe-based LfD* — a framework for learning human-segmented skills in continuous action spaces and producing four different types of embodied queries to improve learned skills. The second is *Intermittently-Active Learning* in which a learner

makes queries selectively, so as to create balanced interactions with the benefits of fully-active learning. Empirical findings from five experiments with human subjects are presented. These identify interaction-related issues in generating embodied queries, characterize human question asking, and evaluate implementations of Intermittently-Active Learning and Active Keyframe-based LfD on the humanoid robot Simon.

The second mechanism, teaching heuristics, is a set of instructions given to human teachers in order to elicit more informative demonstrations from them. Such instructions are devised based on an understanding of what constitutes an optimal teacher for a given learner, with techniques grounded in Algorithmic Teaching. The utility of teaching heuristics is empirically demonstrated through six human-subject experiments, that involve teaching different concepts or tasks to a virtual agent, or teaching skills to Simon.

With a diverse set of human subject experiments, this thesis demonstrates the necessity for guiding humans in teaching interactions with robots, and verifies the utility of two proposed mechanisms in improving sample efficiency and final performance, while enhancing the user interaction.