Title
A stochastic optimal control perspective on affect-sensitive teaching

Permalink
https://escholarship.org/uc/item/74p1t69f

Author
Whitehill, Jacob Richard

Publication Date
2012

Peer reviewed|Thesis/dissertation
UNIVERSITY OF CALIFORNIA, SAN DIEGO

A Stochastic Optimal Control Perspective on Affect-Sensitive Teaching

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

in

Computer Science

by

Jacob Richard Whitehill

Committee in charge:

Garrison Cottrell, Chair
Serge Belongie
Andrea Chiba
Javier Movellan
Harold Pashler
Lawrence Saul
David Weber

2012
The dissertation of Jacob Richard Whitehill is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

Chair

University of California, San Diego

2012
DEDICATION

To Ms. Kenia Milloy and her 2011 Preuss School advisory students, for whom I served as a math tutor for four years, and who lifted my spirits during graduate school.
## TABLE OF CONTENTS

Signature Page .......................................................... iii
Dedication ................................................................. iv
Table of Contents .......................................................... v
List of Figures ............................................................... ix
List of Tables ............................................................... xii
Acknowledgements ........................................................... xiii
Vita ........................................................................ xv
Abstract of the Dissertation .................................................. xvii

### Chapter 1
Introduction ................................................................. 1
1.1 Historical perspective .................................................. 5
1.2 Notation .................................................................. 6
1.3 Bayesian probabilistic reasoning ..................................... 6
  1.3.1 Bayesian belief updates ......................................... 7
1.4 Teaching as an Optimal Control Problem ......................... 11
  1.4.1 Immediate reward/costs ....................................... 13
  1.4.2 Belief ............................................................ 14
  1.4.3 Graphical model ............................................... 14
  1.4.4 Belief updates .................................................. 15
  1.4.5 Control policy ................................................. 16
  1.4.6 Partially Observable Markov Decision Processes ....... 17
1.5 Affect-sensitive teaching ............................................... 19
  1.5.1 Affective and cognitive transition dynamics ............... 20
  1.5.2 Affective and cognitive observation likelihood ............ 21
  1.5.3 Belief updates using affective observations ............... 22
  1.5.4 Designing affect-sensitive teachers .......................... 22
1.6 Dissertation outline and contributions .............................. 23

### Chapter 2
Measuring the Benefit of Affective Sensors ............................... 26
2.1 Introduction ............................................................ 26
2.2 Related work .......................................................... 28
2.3 Cognitive skills training .............................................. 28
  2.3.1 Human training versus computer training ................. 29
2.4 Experiment ............................................................. 30
  2.4.1 Conditions ..................................................... 32
2.4.2 Subjects ........................................... 34
2.5 Experimental results ............................... 35
  2.5.1 Learning conditions ......................... 35
  2.5.2 Facial expression analysis ................. 35
2.6 Towards an automated affect-sensitive teaching system .. 37
2.7 Summary and further research ................... 38
2.8 Acknowledgement .................................. 39

Chapter 3  Automatic Recognition of Student Engagement .... 40
  3.1 Introduction ...................................... 41
    3.1.1 Recognizing student “engagement” .......... 42
  3.2 Related work ..................................... 43
  3.3 Dataset collection and annotation for an automatic engagement classifier .......... 45
    3.3.1 Data annotation .............................. 47
    3.3.2 Engagement categories and instructions .... 49
    3.3.3 Timescale .................................... 51
    3.3.4 Static pixels versus dynamics ............... 52
  3.4 Automatic recognition architectures .......... 54
    3.4.1 Binary classification ....................... 55
    3.4.2 Data selection ............................... 57
    3.4.3 Cross-validation ............................. 58
    3.4.4 Accuracy metric .............................. 58
    3.4.5 Hyperparameter selection ................... 59
    3.4.6 Results: binary classification ............. 59
    3.4.7 Generalization to a different dataset ...... 61
    3.4.8 Regression ................................. 61
    3.4.9 Results: regression ......................... 61
  3.5 Reverse-engineering the human labelers .......... 62
  3.6 Comparison to objective measurements .......... 64
    3.6.1 Test performance ............................ 64
    3.6.2 Learning ..................................... 66
  3.7 Error analysis .................................. 66
  3.8 Conclusion ..................................... 68
  3.9 Appendix: calculating inter-human accuracy ..... 68
  3.10 Acknowledgments .............................. 69

Chapter 4  Measuring the Perceived Difficulty of a Lecture Using Automatic Facial Expression Recognition .... 70
  4.1 Introduction ..................................... 71
  4.2 Facial Expression Recognition .................. 72
    4.2.1 FACS ......................................... 72
    4.2.2 Automatic Facial Expression Recognition ... 73
4.3 Experiment .................................................. 73
  4.3.1 Procedure ............................................. 74
  4.3.2 Human Subjects ....................................... 76
  4.3.3 Data Collection and Processing ....................... 76
4.4 Results .................................................... 77
  4.4.1 Predicting Difficulty from Expression Data ......... 79
  4.4.2 Learning to Predict ................................... 79
4.5 Conclusions ............................................... 81
4.6 Acknowledgement .......................................... 82

Chapter 5  Teaching Word Meanings from Visual Examples ......... 83
  5.1 Introduction ............................................... 84
    5.1.1 Contributions ....................................... 85
    5.1.2 Notation ............................................. 87
  5.2 Related work ............................................. 87
  5.3 Learning setting ......................................... 89
  5.4 POMDPs .................................................... 92
  5.5 Modeling the student as a Bayesian learner ............ 93
    5.5.1 Inference ............................................ 95
    5.5.2 Adding noise ....................................... 98
    5.5.3 Dynamics model .................................... 100
    5.5.4 Observation model ................................ 101
  5.6 Teacher model ........................................... 104
    5.6.1 Representing and updating $B_t$ ....................... 106
  5.7 Computing a policy ...................................... 107
    5.7.1 Macro-controller .................................... 108
    5.7.2 Micro-controller .................................... 110
  5.8 Procedure to train automatic teacher ................. 110
  5.9 Experiment ............................................... 111
    5.9.1 How the OptimizedTeacher behaves ................. 113
    5.9.2 RandomWordTeacher ................................ 115
    5.9.3 HandCraftedTeacher ................................ 116
    5.9.4 Experimental conditions .......................... 117
    5.9.5 Results ............................................. 117
    5.9.6 Correlation between time and information gain .... 118
  5.10 Incorporating affect .................................... 119
    5.10.1 Simulation .......................................... 120
  5.11 Summary ................................................ 122
  5.12 Appendix: Conditional independence proofs ............ 123
    5.12.1 d-Separation of $W_j$ from $W_{j'}$ for $j' \neq j$ .. 123
    5.12.2 d-Separation of $W_j$ from $A_{\nu \nu'}$ for all $\nu$ such that $\nu \neq j$ .......... 124
  5.13 Acknowledgement ....................................... 124
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Simple probabilistic graphical model representing whether Frank has mastered the Pythagorean Theorem ($S \in {\text{unlearned}, \text{learned}}$) and whether he answers a question about it correctly ($O \in {\text{incorrect}, \text{correct}}$).</td>
</tr>
<tr>
<td>1.2</td>
<td>Graphical model of a POMDP.</td>
</tr>
<tr>
<td>1.3</td>
<td>Optimal policy for the simple teaching example in Section 1.4.</td>
</tr>
<tr>
<td>1.4</td>
<td>POMDP to model affect-sensitive teaching. $S^K_t$ is the student’s “knowledge state”, $S^A_t$ is the student’s “affective state”.</td>
</tr>
<tr>
<td>1.5</td>
<td>Dissertation roadmap.</td>
</tr>
<tr>
<td>2.1</td>
<td>A screenshot of the “Set” game implemented on the Apple iPad for the cognitive skill learning experiment.</td>
</tr>
<tr>
<td>2.2</td>
<td>Three experimental conditions. <strong>Top</strong>: Human teacher sits with the student in a 1-on-1 training setting. <strong>Middle</strong>: An “automated” teacher is simulated using a Wizard-of-Oz (WOZ) technique. The iPad-based game software is controlled by a human teacher behind a wall. The teacher can see live video of the student. <strong>Bottom</strong>: Same as middle condition, except the teacher cannot see the live video of the student – the teacher sees only the student’s explicit game actions.</td>
</tr>
<tr>
<td>2.3</td>
<td>Average PostTest-minus-PreTest scores versus experimental condition on the “Set” spatial reasoning game. Error bars represent the standard error of the mean. In the two highest-scoring conditions (WOZ and 1-on-1) the teacher was able to observe the student’s affect.</td>
</tr>
<tr>
<td>2.4</td>
<td><strong>Left</strong>: A student who smiles as a result of receiving and acting upon a “giveaway” hint after having not scored any points for approximately 20 seconds. <strong>Right</strong>: A student who smiles after making a mistake, which resulted in a “buzzer” sound.</td>
</tr>
<tr>
<td>2.5</td>
<td>A student who is in the midst of scoring multiple points.</td>
</tr>
<tr>
<td>2.6</td>
<td>An example of the “traces” collected of the student’s actions, the student’s video, and the teacher’s actions, all recorded in a synchronized manner.</td>
</tr>
<tr>
<td>3.1</td>
<td><strong>Left</strong>: Experimental setup in which the subject plays cognitive games software on an iPad. Behind the iPad is a web camera that records the session. <strong>Right</strong>: Webcam view of one subject playing the game software.</td>
</tr>
<tr>
<td>3.2</td>
<td>Mean faces images across either the HBCU (top) or UC (bottom) dataset for each of the four engagement levels.</td>
</tr>
</tbody>
</table>
Figure 3.3: Sample faces for each engagement level from the HBCU subjects. All subjects gave written consent to publication of their face images. ................................. 50

Figure 3.4: Automatic engagement recognition pipeline. ......................... 54

Figure 3.5: Weights associated with different Action Units (AUs) to discriminate Engagement = 4 from Engagement $\neq 4$, along with examples of AUs 1 and 10. Pictures courtesy of Carnegie Mellon University’s Automatic Face Analysis group webpage. ............ 63

Figure 3.6: Representative images that the automated engagement detector misclassified. Left: inaccuracy face or facial feature detection. Middle: Thick-rimmed eyeglasses that was incorrectly interpreted as eye closure. Right: Subtle distinction between looking down and eye closure. ................................. 67

Figure 4.1: Example of comprehensive FACS coding of a facial expression. The numbers identify the action unit, which approximately corresponds to one facial muscle; the letter (A-E) identifies the level of activation. ................................. 73

Figure 4.2: Representative video frames from each of the 7 video clips contained in our “lecture” movie. ................................. 74

Figure 4.3: The self-reported difficulty values, and the predicted difficulty values computed using linear regression over all AUs, for Subj. 6. 80

Figure 5.1: The robot RUBI (Robot Using Bayesian Inference) interacting with children at the UCSD Early Childhood Education Center, who are playing visual word learning games with the robot. ... 86

Figure 5.2: Student model. ........................................ 93

Figure 5.3: Top: two example images that could be used to teach the Hungarian word férfi. (Bottom-left): prior belief about the meaning of the word férfi. (Bottom-middle): posterior belief about the meaning of word férfi after seeing the left image in the figure. (Bottom-right): posterior belief after seeing both images. 97

Figure 5.4: Teacher model. ........................................ 104

Figure 5.5: Image set used to teach foreign words that mean “man”, “cat”, “eat”, “drink”, etc. All images were found using Google Image Search. ............................................................................. 112

Figure 5.6: Policy of the OptimizedTeacher. Each row corresponds to the policy weight vector $w_a$ for the action specified on the left, e.g., “Teach j” means teach the word indexed by j. Dark colors correspond to low values of the associated weight vector; light colors represent high values. ........................................ 114
Figure 5.7: Average time to completion of learning task versus experimental condition. Error bars show standard error of the mean of each group.

Figure 5.8: Simulation results comparing a teacher with “affective sensors” to one without them. The affect-sensitive teacher is able to teach the student more quickly (left), allowing the student to pass the test, on average, more quickly (right).


### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.1:</td>
<td>Transition dynamics for the simple teaching example of Section 1.4.</td>
<td>12</td>
</tr>
<tr>
<td>Table 1.2:</td>
<td>Observation likelihoods for the simple teaching example of Section 1.4.</td>
<td>12</td>
</tr>
<tr>
<td>Table 1.3:</td>
<td>Reward function $r$ for the simple teaching example of Section 1.4.</td>
<td>13</td>
</tr>
<tr>
<td>Table 3.1:</td>
<td><strong>Top</strong>: Subject-independent, within-dataset (HBCU), image-based engagement recognition accuracy for each engagement level $e \in {1, 2, 3, 4}$ using each of the three classification architectures, along with inter-human classification accuracy. <strong>Bottom</strong>: Engagement recognition accuracy on a different dataset (UC) not used for training.</td>
<td>60</td>
</tr>
<tr>
<td>Table 3.2:</td>
<td>Correlation of student engagement with test scores. Correlations with a * are statistically significant ($p &lt; 0.05$).</td>
<td>64</td>
</tr>
<tr>
<td>Table 4.1:</td>
<td>List of FACS Action Units (AUs) employed in this study.</td>
<td>77</td>
</tr>
<tr>
<td>Table 4.2:</td>
<td><strong>Middle column</strong>: The three significant correlations with the highest magnitude between difficulty and AU value for each subject. <strong>Right column</strong>: The overall correlation between predicted and self-reported Difficulty value, when using linear regression over the whole set of AUs for prediction.</td>
<td>78</td>
</tr>
<tr>
<td>Table 4.3:</td>
<td>The three significant correlations with highest magnitude between preferred viewing speed and AU value for each subject.</td>
<td>79</td>
</tr>
<tr>
<td>Table 4.4:</td>
<td>Accuracy (Pearson $r$) of predicting the perceived Difficulty, as well as the preferred viewing Speed, of a lecture video from automatic facial expression recognition channels. All results were computed on a validation set not used for training.</td>
<td>82</td>
</tr>
<tr>
<td>Table 5.1:</td>
<td>List of words and associated meanings taught during a word learning experiment.</td>
<td>111</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENTS

Foremost, I thank my thesis advisor, Javier Movellan, for advising me during the last 7 years. Javier’s clarity of insight never fails to bedazzle me during our research discussions, and his humanity as an advisor resulted in a very rich and enjoyable graduate school career.

I thank Marian Stewart Bartlett for many valuable research and career discussions, and Gwen Littlewort-Ford, who brought me to the laboratory in the first place.

I thank Gary Cottrell for serving as my “in-house” advisor within the Department of Computer Science & Engineering, and for facilitating my interdisciplinary research at UCSD.

I thank Zewelanji Serpell at Virginia State University (VSU) for an enriching research collaboration over the past two years and for facilitating several visits to her lab at VSU.

I thank my friends and co-authors both at the MPLab and the Serpell Lab: Paul Ruvolo, Tingfan Wu, Nicholas Butko, Ian Fasel, Aysha Foster, Yi-Ching (Gloria) Li, and Brittney Pearson.

Finally, I thank my other committee members, Serge Belongie, Andrea Chiba, Harold Pashler, Lawrence Saul, and David Weber, for their advice and for taking the time to examine my thesis.

Chapter 2, in full, is a reprint of the material as it appears in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2011. Jacob Whitehill, Zewelanji Serpell, Aysha Foster, Yi-Ching Lin, Brittney Pearson, Marian Bartlett, and Javier Movellan. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is currently being prepared for submission for publication of the material. Jacob Whitehill, Zewelanji Serpell, Yi-Ching Lin, Aysha Foster, and Javier Movellan. The dissertation author was the primary investigator and author of this material.

Chapter 4, in full, is a reprint of the material as it appears in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*
(CVPRW), 2008. Jacob Whitehill, Marian Bartlett, and Javier Movellan. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in full, is currently being prepared for submission for publication of the material. Jacob Whitehill and Javier Movellan. The dissertation author was the primary investigator and author of this material.
VITA

2012 Ph.D. in Computer Science, University of California, San Diego – La Jolla, California

2007 M.Sc. in Computer Science, cum laude, University of the Western Cape – Bellville, South Africa

2001 B.S. in Computer Science, with departmental honors, Stanford University – Palo Alto, California

PUBLICATIONS


A Stochastic Optimal Control Perspective on Affect-Sensitive Teaching

by

Jacob Richard Whitehill

Doctor of Philosophy in Computer Science

University of California, San Diego, 2012

Garrison Cottrell, Chair

For over half a century, computer scientists and psychologists have strived to build machines that teach humans automatically, sometimes dubbed intelligent tutoring systems (ITS). The earliest such systems focused on “flashcard”-style vocabulary learning, while more modern ITS can tutor students in diverse subjects such as high school geometry, physics, algebra, and computer programming. Compared to human tutors, however, most contemporary ITS still use a rather impoverished set of low-bandwidth sensors consisting of mouse clicks, keyboard strokes, and touch events. In contrast, human teachers utilize not only students’ explicit responses to practice problems and test questions, but also auditory and visual information about the students’ affective, or emotional, states, to make decisions. It is possible that, if automated teaching systems were affect-sensitive
and could reliably detect and respond to their students’ emotions, then they could teach even more effectively. In this dissertation we examine the affect-sensitive teaching problem from a stochastic optimal control (SOC) perspective. Stochastic optimal control theory provides a rigorous computational framework for describing the challenges and possible benefits of affect-sensitive teaching systems, and also provides computational tools that may help in building them. After framing the problem of affect-sensitive teaching using the language of SOC, we (1) present an experimental technique for measuring the importance to teaching of affect-sensitivity within a given learning domain. Next, we develop machine learning and computer vision tools to recognize automatically certain aspects of the student’s affective state in real-time, including (2) student “engagement” and (3) the student’s perception of curriculum difficulty. Finally, (4) we propose and evaluate an automated procedure, based on SOC, for creating an automated teacher that teaches foreign language by image association (à la Rosetta Stone [73]). In a language learning experiment on 90 human subjects, the controller developed using SOC showed higher learning gains compared to two heuristic controllers, and also allows for affective observations to be easily integrated into the decision-making process.
Chapter 1

Introduction

This dissertation is about learning, teaching, and emotion. Specifically, it is about automated teaching and whether a machine can learn to teach a human student more effectively by modeling and sensing the student’s emotions.

For over half a century now, since the 1950s when B.F. Skinner conceived of a “teaching machine” that could overcome the vast inefficiencies of standard classroom instruction [79], psychologists and computer scientists have strived to create automated teaching systems that teach as effectively, if not even more effectively, than expert human tutors working with students in a 1-on-1 setting. The benefit of such technology is obvious – there are not nearly enough human tutors in the world to teach every pupil on the planet with the attention and dedication he/she deserves. Computers, on the other hand, especially when defined to include cellular phones, are nearly ubiquitous, even in developing countries, and computer tutors could go a long way towards increasing access to high-quality education on a global scale. The research field of automated teaching, which is sometimes called the intelligent tutoring systems (ITS) community, has progressed significantly since its inception about 50 years ago. Whereas the earliest computer teaching systems focused on “flashcard”-style vocabulary learning (e.g., [61, 6]), modern ITS can tutors students in complex skills such as high school algebra [49], physics [90], geometry and computer programming [5]. Some systems such as Carnegie Learning’s Algebra Tutor [16] have been deployed in thousands of schools and reached hundreds of thousands of students across the United States. And yet, despite
such success stories, automated teaching machines have not really revolutionized education to the extent that some had hoped.

One striking feature about contemporary ITS which may partially account for their relatively mild effect on education is that ITS typically consist of rigidly-structured *practice environments* in which students solve a series of practice problems, rather than an *interactive teacher* that explains new concepts dynamically and adjusts itself in real-time to the particular student. In popular ITS such as Carnegie Learning’s Algebra Tutor [16] or the ALEKS math tutor [22], for example, the emphasis is on selecting practice problems at the appropriate difficulty level, and on providing users with a reasonably convenient user interface (keyboard + mouse) in which to enter their responses. When the system needs to explain a new concept to the student for the first time, it simply asks the student to read a webpage. While this may be effective for some students, it is unlikely to engage many students as well as a skilled human teacher.

Another noteworthy feature of most modern ITS (with a few exceptions, e.g., [26, 99]) is that they employ a rather impoverished set of sensors, consisting only of a computer mouse, keyboard, and perhaps a touchscreen. In contrast, human tutors consider not only their students’ explicit responses to practice problems and test questions but also continuously process a vibrant stream of input signals such as facial expression, body posture, speech, and prosody. These signals give the tutor a moment-by-moment sense of the student’s *affective state*, such as whether the student is confused, interested, challenged, bored, attentive, etc. Some of these signals may not yet be realistically decipherable by a computer. It is unlikely, for example, that a computer teacher could, at least in the foreseeable future, understand the nuances of colloquially expressed stuttered speech as effortlessly as a human can. Other signals, such as body posture and facial expression, on the other hand, may well provide useful information to a computer tutor, especially considering the tremendous progress that has been made over the last decade in the fields of machine learning and computer vision. Automatic face analysis systems, for example, have reached the point that they can recognize human facial expressions with reasonable accuracy, in real-time, and from a variety of realis-
tic lighting conditions. It is possible that such technology could help automated teaching systems to become not only more emotionally intelligent practice environments, but also perhaps to start becoming interactive teachers that explain a concept or algorithm to a student for the first time, repeating itself when necessary, pausing for emphasis, waiting for the student’s full attention, and congratulating him/her appropriately once it has ascertained that the student has indeed grasped the new concept.

Over the past few years, the ITS community has witnessed the first few efforts to make automated teaching systems affect-sensitive [26, 99], i.e., to endow ITS with the ability to sense and respond to aspects of the student’s affective state. These efforts consist of both recognizing key emotional states in the student automatically, as well as very nascent attempts toward integrating these emotion estimates into the decision-making process. To date, however, these early affect-sensitive tutors have employed only simple sets of rules to decide how to respond to certain recognized emotions. If a student appears frustrated, for example, then the tutor might switch to a different topic. While intuitively this seems useful, there is little empirical evidence that existing affect-sensitive tutors actually teach more effectively than similar “affect-blind” systems. In fact, in a comparison between an affect-sensitive AutoTutor (for computer literacy skills) with an affect-blind AutoTutor [25], the affect-sensitive tutor was 37% less effective at teaching than the affect-blind tutor during the first day of the study, and only 8% more effective than the affect-blind system during the second day. Moreover, even if a set of rules about how to process affective sensor inputs is useful in some teaching situations, it is unlikely that such an approach could scale up to scenarios in which multiple high-bandwidth sensor readings – e.g., from web cameras, skin conductance sensors, etc. – arrive simultaneously and must somehow be used to achieve a teaching advantage. Here, we draw an analogy with the progress made in computer vision due to the application of machine learning: when developing an automatic face detector, for example, it would be simply infeasible to enumerate all the “rules”, in terms of pixel values, necessary to define whether a particular region of an image contains a human face. It was not until researchers applied supervised learning to
the face detection problem that face detectors became practical. Similarly, it is possible that affect-sensitive tutoring systems will require principled computational decision-making frameworks that can seamlessly integrate high-volume sensor inputs in order to succeed.

One such candidate framework is stochastic optimal control theory; in fact, optimal control was the basis of some of the earliest computer-aided instructional systems (e.g., [61, 80, 6]). Stochastic optimal control is concerned with making intelligent decisions in uncertain and changing environments in order to minimize some cost, or equivalently to maximize some value, over the long-term. In teaching, the uncertain and changing “environment” is the student, and the cost could be expressed in terms of time, e.g., how long it takes the student to learn a certain lesson. Stochastic optimal control provides a principled method of updating the teacher’s uncertain belief about the student’s affective and cognitive states with real-time sensor inputs the teacher reads from the student. It also allows the task of teaching to be posed as an optimization problem, and once the optimization problem is defined, a variety of algorithms, including dynamic programming, policy gradient techniques, and even supervised learning methods can be used to solve, or at least approximately solve, the optimal teaching problem. In addition to bringing concrete computational tools, stochastic optimal control theory provides a language, including such terms as state, action, observation, and belief, for explaining the challenges and possible benefits of making affect-sensitive teaching systems.

In this chapter, we frame the problem of automatic affect-sensitive teaching using one particular mathematical framework from stochastic optimal control, namely the Partially Observable Markov Decision Process (POMDP). The theory of POMDPs is rooted in probabilistic inference, and hence we provide an introduction to both Bayesian inference and POMDPs in the following sections of this chapter. After doing so, we then define an affect-sensitive POMDP which both illustrates the necessary components of an affect-sensitive teaching system and also serves as a roadmap of the contributions we make to the field of automated teaching in the remainder of this dissertation.
1.1 Historical perspective

Much of the early work on automated teaching systems took place at Stanford University in the 1960s and 1970s [84, 81, 61, 53, 7, 6]. This early research posed automated teaching as an optimal control problem, and some of the techniques that were developed then still play a role in many successful teaching systems (e.g., [21]). Much of this work focused on teaching a list of “paired-associate” items, e.g., vocabulary words and basic facts. The decisions of which “item” to teach next were based on optimal control theory. Research at this time focused on either deriving analytical solutions to determine control policies, which are possible only in a few cases, or on computing exact solutions numerically using dynamic programming, which becomes computationally intractable ($O(2^{2\tau})$, where $\tau$ is the length of the teaching session) except for very small teaching problems. Possibly due to this overemphasis on exact solutions, the optimal control approach to automated teaching languished.

In the 1980s, the field of automated teaching was revived with John Anderson’s “cognitive tutor” movement at Carnegie Mellon University. Cognitive tutors are based loosely on his ACT* and ACT-R theories of cognition [3, 4]. Notable examples of cognitive tutors include the LISP Tutor and Geometry Tutor [5]. Instead of teaching simple facts, these tutors provide students with structured practice environments in which to hone their proficiency in cognitive skills such as solving algebra problems and proving geometry theorems. Teaching decisions, such as which problem to present to the student next, are made mostly using heuristic methods.

Since the mid 2000s, there has emerged a small renaissance of the optimality approach to teaching, both for inference and for decision-making: In [20], for example, Bayesian networks were used to optimally infer the student’s problem-solving plan, and in [13] they were employed to assess whether students benefit from receiving hints during tutoring. A few researchers have designed teaching systems that make decisions by maximizing some form of immediate reward [66, 38], i.e., one-step greedy look-ahead search over all possible actions. [8] and [35] employed fully observable Markov Decision Processes (MDPs) to select hints so as
to maximize the probability of the student reaching a solution. Finally, [19] showed that using MDPs to make “micro-level” tutorial decisions yielded a measurable benefit in learning. In all of these recent works, however, optimal control theory was used for only limited aspects of the total control policy, and in none of them is it used to utilize modern sensors that could benefit automated teaching.

1.2 Notation

In this document we use upper-case letters to represent random variables and lower-case letters to represent particular values they take on. $P(X = x)$ is the probability that random variable $X$ takes on value $x$. For brevity, we may sometimes write $P(X = x)$ simply as $P(x)$. $P(X = x \mid Y = y)$ is the conditional probability that random variable $X$ takes on value $x$ given that random variable $Y$ takes on value $y$. For brevity, we may sometimes write this simply as $P(x \mid y)$.

1.3 Bayesian probabilistic reasoning

Probability theory is concerned with modeling and predicting events whose outcomes are uncertain, either because they haven’t happened yet – e.g., a student will receive a score on a test he takes tomorrow – or because they are “obscured” from us for some reason – e.g., a student read a chapter from her algebra textbook today, but whether or not she understood the quadratic formula is unclear because we cannot directly “peer” inside her mind. Bayesian probability theory in particular allows us to assign a probability – a real number between 0 and 1 – to either of these events to express how certain we are that they are true. Larger probabilities are associated with greater certainty. For instance, we might assign a probability of 0.95 to the event that a particular student will pass a test tomorrow because, say, he has never failed an exam in the class before. We can even assign a probability to an event whose outcome is, in some sense, already certain but unknown to us. For instance, if a student took a multiple-choice exam yesterday that is graded by a computer, but we have not yet seen her exam paper, then whether
or not the student answered $\geq 50\%$ of the questions correctly is, in some sense, already determined (assuming that student cannot change her answers). From a Bayesian perspective, however, it is still perfectly valid for us to assign a probability to the event that the student passed the exam to convey our certainty in that belief. Note that this contrasts with other views of probability theory such as the frequentist view that probabilities can only represent the proportion of times that some random experiment, repeated an infinite number of times, would have a certain outcome.

### 1.3.1 Bayesian belief updates

One core aspect of Bayesian reasoning is the procedure for updating one’s *prior belief* about the outcome of some event based on observed *evidence* to obtain a *posterior belief* about that event. As an example, let us suppose we are interested in whether or not a math student, Frank, has mastered the Pythagorean Theorem. We can represent this “knowledge state” in Frank’s brain using a random variable $S$ (“state”) that takes value “learned” if Frank has mastered the theorem and value “unlearned” if he has not. Let us assume that we can never directly examine the value of $S$ because it is hidden inside of Frank’s brain. In this case, $S$ is called a *latent variable*. We may, however, have some “prior belief” over the value of $S$, based perhaps on Frank’s past performance in the class. Even though we can never observe $S$, we can easily ask Frank to answer a math problem that applies his knowledge (if any) of the Pythagorean Theorem. For instance, we might ask him to answer a 2-answer multiple choice question: “If the two shorter sides of a right triangle have length 3 and 4, then the length of the third side must be: (a) 5 or (b) 6.” We can then represent the correctness of Frank’s response with a random variable $O$ that equals “correct” if Frank responds with (a) (the correct answer) and “incorrect” otherwise. The “$O$” stands for “observation” because, from the teacher’s perspective, we *observe* Frank’s answer to the test question (in contrast to the latent state $S$). After Frank tells us his answer $O$, we can use this information to compute a “posterior belief” about $S$.

Random variables are sometimes displayed graphically in what are variously
Figure 1.1: Simple probabilistic graphical model representing whether Frank has mastered the Pythagorean Theorem ($S \in \{\text{unlearned, learned}\}$) and whether he answers a question about it correctly ($O \in \{\text{incorrect, correct}\}$).

called Bayesian belief networks or probabilistic graphical models. An example graphical model for our example scenario is shown in Figure 1.1. It contains two “nodes” – one to represent $S$, and one to represent $O$. Node $O$ is shaded to indicate that it is observed; $S$ is not shaded because it is latent – we can estimate its value using probabilistic inference, but we never observe it directly. Nodes $S$ and $O$ are also connected by an arrow, or directed edge. In this model, the direction is from $S$ to $O$ to indicate that the value of $S$, i.e., whether Frank knows the Pythagorean Theorem, has an influence on $O$, i.e., the correctness of his response to a quiz question. We will assume that, if Frank knows the Pythagorean Theorem, then he is almost certain to answer a question about it correctly. Hence, we might set $P(O = \text{correct} \mid S = \text{learned}) = 0.9$. Note that we chose not to set this probability to 1 because we want to allow for the unlikely possibility that Frank makes a careless mistake. On the other hand, if Frank does not know the Pythagorean Theorem, then all he can do is guess the correct answer. Since there are two possible answers, it is reasonable to assume that he would pick either of the two randomly; hence, we set $P(O = \text{correct} \mid S = \text{unlearned}) = 0.5$. Based on these probabilities and the fact that the sum over a probability distribution must equal
1, we can also compute the conditional probability of an incorrect response:

\[
P(O = \text{incorrect} \mid S = \text{learned}) = 1 - P(O = \text{correct} \mid S = \text{learned}) \\
= 1 - 0.9 \\
= 0.1
\]

\[
P(O = \text{incorrect} \mid S = \text{unlearned}) = 1 - P(O = \text{correct} \mid S = \text{unlearned}) \\
= 1 - 0.5 \\
= 0.5
\]

Finally, let us suppose we have some prior belief about the value of \( S \); in particular, suppose that we have no idea whether Frank already knows the Pythagorean Theorem. We might then set \( P(S = \text{learned}) = 0.5 \). If we had evidence that Frank already knew this theorem, e.g., if we observed him reading about it during class, then we might set this probability to a larger value.

Given that we have defined both a prior distribution over \( S \) as well as a conditional probability distribution for \( O \) given \( S \), we are now ready to conduct Bayesian inference of the value of \( S \). Let us suppose that Frank answers the math problem incorrectly, i.e., \( O = \text{incorrect} \). How much do we now believe that Frank knows the Pythagorean Theorem, i.e., with what probability does \( S = \text{learned} \) given that \( O = \text{incorrect} \)? To compute this probability \( P(S = \text{correct} \mid O = \text{incorrect}) \) we will make use of Bayes’ rule, from which Bayesian statistics gets its name:

\[
P(S = \text{learned} \mid O = \text{incorrect}) = \frac{P(O = \text{incorrect} \mid S = \text{learned})P(S = \text{learned})}{P(O = \text{incorrect})} \quad (1.1)
\]

Bayes rule tells us that the probability of \( S = \text{learned} \) given \( O = \text{incorrect} \) can be computed (in part) by “flipping the conditionality around”, i.e., from the probability of \( O = \text{incorrect} \) given \( S = \text{learned} \). This probability is already known to us – in our example, we supposed it to be 0.1. The other term in the numerator on the right hand side is \( P(S = \text{learned}) \) – our prior belief that \( S = \text{learned} \) – which we supposed was 0.5.

To compute the denominator, we use the law of total probability, which tells us that \( P(O = \text{incorrect}) \) can be computed as the sum of probabilities of
giving an incorrect answer given either of the two knowledge states ("learned" or "unlearned"), weighted by the probability of each knowledge state:

\[ P(O = \text{incorrect}) = P(O = \text{incorrect} \mid S = \text{unlearned})P(S = \text{unlearned}) + P(O = \text{incorrect} \mid S = \text{learned})P(S = \text{learned}) \]

\[ = 0.5 \times 0.5 + 0.1 \times 0.5 \]

\[ = 0.3 \]

Plugging this into Equation 1.1 we get:

\[ P(S = \text{learned} \mid O = \text{incorrect}) = \frac{0.1 \times 0.5}{0.3} \approx 0.17 \]

Hence, the posterior probability, or our posterior belief, that Frank knew the Pythagorean Theorem, given his incorrect answer, is about 0.17. Compared to our prior belief about Frank’s knowledge state, this is a rather large change.

It is instructive to also consider the case that Frank had given the correct answer, and then to compute the posterior probability that he is in the "learned" state:

\[ P(S = \text{learned} \mid O = \text{correct}) = \frac{P(O = \text{correct} \mid S = \text{learned})P(S = \text{learned})}{P(O = \text{correct})} \]

\[ = \frac{0.9 \times 0.5}{0.9 \times 0.5 + 0.5 \times 0.5} \]

\[ = \frac{0.45}{0.45 + 0.25} \]

\[ \approx 0.64 \]

Notice the asymmetry here in how much we revise our belief that \( S = \text{learned} \) when Frank gives a correct answer compared to when he gives an incorrect answer – when Frank gives a correct answer, then we only slightly increase our belief. This is because it is quite possible that he simply guessed correctly.

Finally, it is worth mentioning that Equation 1.1 is often written as a proportionality, without the denominator, in order to avoid notational clutter:

\[ P(S = \text{learned} \mid O = \text{incorrect}) \propto P(O = \text{incorrect} \mid S = \text{learned})P(S = \text{learned}) \]

(1.2)
1.4 Teaching as an Optimal Control Problem

Having given a whirlwind introduction to Bayesian inference, we now attempt to motivate and illustrate the main concepts of the Partially Observable Markov Decision Process (POMDP) using a hypothetical teaching scenario. The example is highly simplistic and not intended to accurately model real students; however, it is sufficiently rich to illustrate some of the fundamental challenges in teaching and how algorithms for optimal control can be used to derive reasonable teaching behavior.

Suppose that a teacher wishes to teach a student some skill. The student’s knowledge of the skill is assumed to be binary, i.e., it is either “learned” or “unlearned,” as in [14]. Hence, the state space $S = \{\text{unlearned}, \text{learned}\}$, and the student’s state at time $t$ is represented as $S_t \in S$. Although the teacher does not observe the state $S_t$, it has a prior belief $P(s_1)$ over the student’s initial state $S_1$. Here, we assume that $P(S_1 = \text{unlearned}) = 1$.

At each timestep, the teacher can perform one of three actions: it can teach, meaning that the teacher attempts to transmit knowledge of the skill to the student without eliciting any feedback from the learner; it can query the student’s knowledge by asking him/her to demonstrate the skill; and the teacher can stop the teaching session, after which no further teach or query actions can be performed. Hence, the action space $U = \{\text{teach}, \text{query}, \text{stop}\}$, and the teacher’s action at time $t$ is represented by $U_t \in U$. When the teacher teaches, the student’s state may transition with probability 0.2 from the unlearned to the learned state. If the skill is already learned, then it will always stay learned, i.e., there is no “forgetting.” When the teacher queries or stops, the student’s state does not change. The effects of the teacher’s action on the student’s state constitute the transition dynamics $P(s_{t+1} \mid s_t, u_t)$ of the student; they are represented in Table 1.1.

During “query” actions, the student will attempt to demonstrate the skill to the teacher, and the demonstration will either be correct or incorrect. Hence, the observation space $O = \{\text{incorrect, correct}\}$. The student’s observation, i.e., his response to the teacher’s action at time $t$, is $O_t$. If the skill is learned, then the student demonstrates the skill correctly with probability 1. If the skill is un-
Table 1.1: Transition dynamics for the simple teaching example of Section 1.4.

<table>
<thead>
<tr>
<th>$s_{t+1}$</th>
<th>unlearned</th>
<th>learned</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlearned</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>learned</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

$P(s_{t+1} \mid s_t, U_t = \text{teach})$

<table>
<thead>
<tr>
<th>$s_{t+1}$</th>
<th>unlearned</th>
<th>learned</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlearned</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>learned</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

$P(s_{t+1} \mid s_t, U_t \in \{\text{query, stop}\})$

Table 1.2: Observation likelihoods for the simple teaching example of Section 1.4.

<table>
<thead>
<tr>
<th>$o_t$</th>
<th>incorrect</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlearned</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>learned</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

$P(o_t \mid s_t, U_t = \text{query})$

<table>
<thead>
<tr>
<th>$o_t$</th>
<th>incorrect</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlearned</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>learned</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$P(o_t \mid s_t, U_t \in \{\text{teach, stop}\})$

learned, then the demonstration is correct with probability 0.1 – the student must essentially “guess” the right answer or “blindly” try to execute the skill correctly. When the teacher executes the “teach” or “stop” actions, the student’s observation is not meaningful; hence, we set the probability of a “correct” observation under the “teach” and “stop” actions to be uninformative – the probability of each observation is independent of the student’s state. The probabilities $P(o_t \mid s_t, u_t)$ of which observation (“correct” or “incorrect”) the student emits as a function of the teacher’s action and the student’s state constitute the observation likelihoods of the teaching setting; they are shown in Table 1.2.

Given these three actions to choose from, how should the teacher act at each timestep? How should the history of actions, along with the history of observations received from the student, influence the teacher’s next action? Why should the teacher bother to “query” the student at all, considering that such actions do not directly influence the student’s state? These are the fundamental questions that face all teachers, whether human or automatic, no matter what the exact teaching setting is. Optimal control theory provides tools to tackle this problem. The next step in solving our particular teaching problem using POMDPs is to define the rewards and costs of teaching.
Table 1.3: Reward function $r$ for the simple teaching example of Section 1.4.

<table>
<thead>
<tr>
<th>$u_t$</th>
<th>$s_t$</th>
<th>$r(u_t, s_t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>teach</td>
<td>unlearned</td>
<td>$-1$</td>
</tr>
<tr>
<td>teach</td>
<td>learned</td>
<td>$-1$</td>
</tr>
<tr>
<td>query</td>
<td>unlearned</td>
<td>$-0.5$</td>
</tr>
<tr>
<td>query</td>
<td>learned</td>
<td>$-0.5$</td>
</tr>
<tr>
<td>stop</td>
<td>unlearned</td>
<td>$0$</td>
</tr>
<tr>
<td>stop</td>
<td>learned</td>
<td>$10$</td>
</tr>
</tbody>
</table>

### 1.4.1 Immediate reward/costs

In optimal teaching problems, the teacher has a preference structure for which actions it prefers to execute and in which states it prefers the student to be. For instance, the teacher may prefer the student to be in the “learned” state rather than in the “unlearned” state. This preference structure is expressed as an immediate reward function $r(u_t, s_t)$, where $u_t \in U$ and $s_t \in S$. Costs of executing certain actions can be formulated as negative-valued rewards. Note that these rewards are not amounts of “money” that the teacher receives (or has to pay) from someone while teaching – rewards are only used during planning to express the teacher’s preferences for how the learning session should evolve.

In our example, the teaching and querying have associated costs; hence, we set the corresponding immediate “rewards” to be $-1$ and $-0.5$, respectively, regardless of the student’s state. If the teacher stops and the student’s knowledge state was learned, then, in our example, there is a reward of 10. Together, the immediate rewards are shown in Table 1.3. For an example from daily life, the costs of teaching might consist of both the time that the teacher and student must invest, as well as any monetary costs, e.g., salary, teaching supplies, etc. Note the form that “querying” can play in modern education systems: standardized tests, for example, may retrieve valuable information about students’ knowledge, but they also interrupt normal school instruction and hence incur a cost.

The goal of the teacher in optimal teaching scenarios is to execute actions so as to maximize the expected long-term reward (or minimize the expected long-term cost). The teacher chooses its actions based on a control policy. It turns
out that this control policy can be formulated as a function of the teacher’s *belief* about the student’s state; in fact, the belief is a sufficient statistic of the teaching history in order to maximize the expected long-term reward. We define *belief* and the *belief update* process below.

### 1.4.2 Belief

In our setting, the teacher never directly observes the student’s state $S_t$ because the state is *hidden*. Instead, the teacher maintains a probability distribution, known as the *belief*, over the student’s state, given the history of actions it has executed and the observations it has received from the student. We represent the teacher’s belief by random vector $B_t$. The $j$th component of $B_t$ represents the teacher’s belief that the student is in the $j$th state (in our example, $j$ is either 1 or 2 because $|S| = 2$) at time $t$. If the teacher previously executed actions $u_1, \ldots, u_{t-1}$ and received observations $o_1, \ldots, o_{t-1}$ from the student, then

$$b_{tj} = P(S_t = j \mid u_1, \ldots, u_{t-1}, o_1, \ldots, o_{t-1})$$

Note that, in contrast to the state space $S$ which is finite in our example, the size of the belief space is uncountably infinite because it is a probability distribution. This unfortunately makes the control problem much harder compared to scenarios in which $S_t$ is observed.

### 1.4.3 Graphical model

The student’s state $S_t$, the teacher’s action $U_t$, the observation of the student $O_t$, and the teacher’s belief $B_t$ about the student’s state are represented together in the graphical model shown in Figure 1.2. The particular configuration of which nodes are connected to which other nodes via directed edges encodes a dependence structure, or rather a *conditional independence structure*, among the random variables. Conditional independence is used to simplify the inference process, such as the teacher’s *belief update* described below.
1.4.4 Belief updates

Whenever the teacher executes a new action $u_t$ and receives another observation $o_t$, it must compute the posterior belief of the student’s state given the new “history” of actions and observations. In particular, the teacher performs a belief update to obtain $b_{t+1,j}$ for each $j$; the derivation of the belief update equation is based on the conditional independence structure encoded in the graphical model:

$$b_{t+1,j} \doteq P(S_{t+1} = j \mid u_1, \ldots, u_t, o_1, \ldots, o_t)$$
$$\propto P(S_{t+1} = j, o_t \mid u_1, \ldots, u_t, o_1, \ldots, o_{t-1})$$
$$= \sum_k P(S_{t+1} = j, o_t \mid S_t = k, u_1, \ldots, u_t, o_1, \ldots, o_{t-1})$$
$$= \sum_k P(S_{t+1} = j \mid o_t, S_t = k, u_1, \ldots, u_t, o_1, \ldots, o_{t-1})$$
$$= \sum_k P(S_{t+1} = j \mid S_t = k, u_t) P(o_t \mid S_t = k, u_t)b_{tk}$$
The first term in the summation of the last line comes from the transition dynamics; the second term is from the observation likelihood; and the last term is the teacher’s prior belief of the student’s state at time $t$. This equation allows the teacher to efficiently compute the belief update by summing over its prior beliefs for each possible state and weighting them by the product of the transition probability and observation likelihood.

### 1.4.5 Control policy

At each timestep $t$, the teacher must select an action $U_t$ to execute. The teacher chooses this action according to a control policy $\pi$. A deterministic policy maps from $b_t$ into an action $u_t \in U$; a stochastic policy maps from $b_t$ into a probability distribution over $U$. In the case of a stochastic policy, to choose the next action, the teacher would first compute $\pi(b_t)$, and then sample an action from the resultant probability distribution. Different policies tend to choose certain actions more often than others, and they may cause the student to enter certain states more frequently. Since the teacher associates rewards with these state+action combinations, some policies may be better than others in terms of the teacher’s preferences; this is quantified by the value function $V(\pi)$ which computes the expected sum of discounted rewards over the time horizon $\tau$:

$$V(\pi) = E\left[\sum_{t=1}^{\tau} \gamma^t r(S_t, U_t) \mid \pi, P(s_1)\right]$$

The time horizon $\tau$ specifies the length of the teaching session, and the discount factor $\gamma \in [0, 1]$ specifies how much rewards in the near future are to be weighted compared to rewards in the distant future. If $\gamma$ is small, then rewards in the distant future have little effect on the value of a particular policy. Choosing $\tau = \infty$, which is admissible as long as $\gamma < 1$, is a mathematical convenience that allows for the formulation of a stationary policy that does not change as a function of time. Given the formula for $V$ above, an optimal (stationary) policy is a function $\pi^*$ that maximizes $V$:

$$\pi^* \doteq \arg\max_{\pi} V(\pi)$$

In our example, we choose $\gamma = 0.99$ and $\tau = \infty$. 
1.4.6 Partially Observable Markov Decision Processes

We have now defined all the variables that constitute a Partially Observable Markov Decision Process (POMDP). Formally, a POMDP is defined by a tuple

\[(S, U, O, P(s_{t+1} \mid s_t, u_t), P(o_t \mid s_t, u_t), r(u, s), \tau, \gamma, P(s_1))\]

whose components are the state space, action space, observation space, transition dynamics, observation likelihood, reward function, time horizon, discount factor, and prior belief of the student’s state, respectively.

The challenge when working with POMDPs to solve teaching problems is to compute the policy \(\pi\) according to which the teacher will act. In some very restricted teaching scenarios, the optimal policy \(\pi^*\) can be found analytically (e.g., [81]). Otherwise, numerical methods must be used. The method of value iteration [17], which is based on dynamic programming, can be used to estimate the optimal policy of a POMDP with arbitrary precision, but the computational costs are formidable: for a time horizon \(\tau\), the worst-case time cost is \(O(2^{2\tau})\). The reason for the enormous time complexity for exact solutions is that the state \(S_t\) is hidden, and hence all teaching decisions must be made in terms of a real-valued belief \(B_t\) instead of the (typically) discrete-valued \(S_t\). Due to the computational costs, a variety of approximation methods are typically used to solve POMDPs for real-world control applications. Approximate methods based on belief compression (e.g., [74]) seek to reduce the size of the state space to speed up computation. Point-based value function approximation methods (e.g., [83]) give up on trying to find a policy that works well for all beliefs \(b_t\) and instead focus on the most likely beliefs that the agent (teacher) would encounter. Finally, policy gradient methods (e.g., [98, 15]) formulate the policy \(\pi\) using a parameter vector and use gradient descent to maximize \(V\) with respect to the policy’s parameters. In Chapter 5 of this thesis, we use a policy gradient approach to optimize our word teacher.

For now, however, let us return to the simple teaching example we sketched above, and let us examine how the optimal policy behaves. We used the ZMDP software [82], which internally uses point-based value iteration, to compute the optimal policy, shown in Figure 1.3. The horizontal axis of the figure shows the
Figure 1.3: Optimal policy for the simple teaching example in Section 1.4.

teacher’s belief $b_t$ that the student is in the “learned” state, and the vertical axis shows the value, i.e., the expected sum of discounted future rewards given the optimal policy $\pi^*$ and the teacher’s current belief, associated with $b$. In addition, the vertical bars in the graph separate the “regions” of the belief space within which a certain action is optimal. According to the computed policy, the “teach” action is optimal when the teacher has a small belief that the student is in the “learned” state, and the “stop” action is optimal when the teacher has a large belief that the student has learned the skill. Interestingly, it turns out that, when the teacher’s belief is uncertain, i.e., between about 0.3 and 0.82, then the optimal action is to “query” the student. Note that querying does not immediately help the student to learn the skill; rather, it helps the teacher to become more certain about the student’s state and thereby to make more intelligent actions in the future. In the framework of stochastic optimal control, such “information foraging” actions can emerge naturally as the optimal action because they implicitly help the teacher to minimize the expected long-term cost.

Let us now recapitulate what it means to pose “teaching” as an optimal control problem using the language of POMDPs: the teacher executes actions so as
to alter the student’s state in ways that the teacher finds desirable according to its reward function. However, the teacher cannot directly observe the student’s state because it is “hidden” inside the student’s mind; instead, it can only maintain and update a belief about the student’s state given the history of actions and observations it has received from the student during the learning session. The teacher’s action at each timestep is chosen according to a control policy which maps the teacher’s current belief into an action (or a probability distribution over actions). An optimal control policy chooses actions, based on the teacher’s current belief, so as to maximize the expected long-term reward, or equivalently, to minimize the expected long-term cost, of teaching.

In practice, successful application of control theoretic methods to optimal teaching problems requires effective use of approximate methods to finding good policies. Although the simple example we presented in this section modeled the state as a binary variable, in Chapter 5 of this dissertation we consider a much more complex learning model: the student’s state is a set of probability distributions over the meanings of a set of words, and the teacher’s belief is therefore a probability distribution over a set of probability distributions. Despite the enormity of the belief space, a good policy was still found using a policy gradient optimization approach.

1.5 Affect-sensitive teaching

So far in our modeling of teaching as an optimal control problem we have only considered the “cognitive” elements of a student’s state, e.g., whether the student knows some skill, or whether his answer to a question is right or wrong. Let us now come to the crux of the dissertation and consider the importance of the non-cognitive, or what we call the affective, components of a student’s state, e.g., how the student feels, whether the student is attentive, whether he/she is trying to learn, etc. In addition, let us also examine the affective components of the observations the teacher receives of the student – does the student appear attentive based on visual and auditory information we receive? Does the student look like
Figure 1.4: POMDP to model affect-sensitive teaching. \( S^K_t \) is the student’s “knowledge state”, \( S^A_t \) is the student’s “affective state”.

He’s trying to succeed? To illustrate this shift from a purely “cognitive” view of teaching to an “affect-sensitive” perspective, we have created a revised graphical model to include both affective and cognitive state and both affective and cognitive observations, shown in Figure 1.4. In the figure, the student’s state is split into the knowledge state \( S^K_t \) and the affective state \( S^A_t \). Similarly, the observation \( O_t \) is split into the knowledge observation \( O^K_t \) and the affective observation \( O^A_t \).

1.5.1 Affective and cognitive transition dynamics

Given the new “affect-sensitive” POMDP, we can examine more concretely how modeling of affect – whether in the state or the observation – could impact both learning and teaching. Let us first consider the influence of the knowledge and affective state components both on themselves and each other. These influences are shown in Figure 1.5 (a). Arrow 1, from \( S^K_t \) to \( S^K_{t+1} \) represents the fact that the student’s knowledge at time \( t \) influences her knowledge at time \( t + 1 \). This is natural – if a student has mastered differential calculus at time \( t \), then it is very unlikely that she would suddenly lack the knowledge to solve a linear equation at
time \( t + 1 \). Similarly, arrow 4 represents the fact that a student’s emotional state at \( t \) will likely have some effect on his emotional state at \( t + 1 \) – emotions, though possibly erratic, will likely show some temporal consistency.

More interesting and subtle are the other two arrows. Arrow 2 represents the influence of a student’s knowledge on her affect. As an arbitrary example, it is possible that, if a student cannot manage to learn a certain topic for a long time, i.e., her knowledge state stagnates, then this lack of progress could impact the student’s affective state, e.g., cause her to become frustrated. Arrow 3 represents the opposite effect – if a student is frustrated, then she may have difficulty learning the curriculum, causing her knowledge state to stagnate.

Note that, as a teacher, we may associate value with both the knowledge and affective components of the student’s state. We may care that the student be in a positive affective state so that she can learn better (i.e., because we value \( S^K_t \)), but we may also wish for the student to be in a positive affective state just for the sake of the student’s happiness (i.e., because we value \( S^A_t \)). Both kinds of value that we associate with the student’s state can influence our teaching decisions.

1.5.2 Affective and cognitive observation likelihood

Next, Figure 1.5 (c) portrays how the state components are reflected in the observations. Arrow 1 is natural and represents how the student’s knowledge influences his test answers, factual responses to the teacher’s questions, etc. Similarly, arrow 3 represents how the student’s affective state is reflected in affective observations made of the student – for instance, though a student may try to mask his emotional state, it is likely that an expert teacher (or a highly accurate automatic emotion classifier) can at least partly perceive that emotion. Arrow 2 is more subtle and represents how the student’s affective state can impact how he performs – if he is not trying, then his test scores may be low, regardless of what he knows or how skilled he is. If the teacher, whether human or computer, attributes poor test performance to the wrong cause, then this can lead to bad teaching.
1.5.3 Belief updates using affective observations

Finally, Figure 1.5 (b) represents how the observations of the student — both knowledge and affective — are used to update the teacher’s belief about the student’s state. Node $O_t^A$ is half-shaded to indicate that, depending on the particular learning setting, it may or may not be observed by the teacher. In Chapter 2 of this dissertation, we examine the impact on teaching effectiveness of the teacher having access to the affective observations, specifically being able to view the student’s face while teaching.

1.5.4 Designing affect-sensitive teachers

Good human teachers are adept at knowing which aspects of a student’s cognitive and affective state and which kinds of observations he/she receives from the student are important and which can be ignored. Similarly good judgment is required when designing effective automated teachers. In particular, as the designer of an ITS, we need to decide which nodes and which directed edges of the affect-sensitive POMDP in Figures 1.4 and 1.5 we will include in our teaching model. For instance, perhaps in a given learning domain we, as the teacher, can completely ignore the student’s “affective observations” because all of the useful information is already contained in the student’s test performance. In this case, making the teacher “affect-sensitive” might well be a waste of time. The importance to teaching effectiveness of the teacher being able to see some of the student’s affective observations is a question we tackle in Chapter 2. If, on the other hand, we decide that using affective observations is important, then we need a mechanism of regressing from those observations to the student’s state. In Chapters 3 and 4 we thus develop automatic classifiers of student “engagement” and student perception of curriculum difficulty that map from the pixels of the student’s face to these affective states. Finally, and most crucially, creating automated teachers requires that we devise a control policy for how the teacher should act. This control policy should deliver good learning gains and should be tractably computable given a reasonable model of the learner. In Chapter 5, we propose an automated procedure for computing a control policy for the particular domain of language
1.6 Dissertation outline and contributions

The rest of this dissertation is structured as follows: In Chapter 2 we propose and demonstrate an experimental protocol for measuring the importance to teaching of having access to the “affective observations” of the student. This is a useful procedure to execute prior to investing the effort to develop an affect-sensitive automated teacher for a given learning domain – if affective sensors make no difference, then using them is probably a waste of time. We apply this experimental framework to cognitive skills training, which is a training regimen designed to boost students’ performance in academic subjects by first improving basic memory, attention, and logical reasoning abilities. This project emerged out of a collaboration with the Serpell lab at Virginia State University. Zewe Serpell visited our lab as a visiting scholar during Summer 2010, and I visited her lab in Virginia.
twice during the ensuing two years.

Chapters 3 and 4 consider the problem of recognizing important aspects of students’ affective state from affective sensor measurements. While the computer vision and automatic face analysis communities have studied extensively the problems of basic emotion (e.g., [51, 103, 55]) and facial action [32] (e.g., [50, 102, 58, 9, 88]) recognition, much less work has been done recognizing more nebulously defined affective states related to learning. It is thus unclear how well existing methods for both labeling video and image data, and for estimating these labels automatically, would work in practice in automated teaching settings. In Chapter 3 we attempt to detect students’ degree of “engagement” with their learning task; here, we make use of the video data we collected as part of the Cognitive Games study from Chapter 2. As the “ground truth” for students’ engagement scores, we use the perceptual judgments of human observers who viewed either video clips or images of students interacting with the game software. This may not be a perfect label of student’s engagement state, but we would already be very happy if a machine could predict what human observers would say about a student’s level of engagement. In contrast, in Chapter 4, we ask the students themselves how hard or easy they perceive the curriculum of a lecture video to be at each moment in time, and then attempt to estimate these perceived difficulty scores automatically. In both these chapters we make use of the Computer Expression Recognition Toolbox (CERT), which is a tool for automated real-time face processing developed at our laboratory (and to which this dissertation author contributed several components including the smile detector [95] and head pose estimator [96]), and attempt to regress from CERT’s output channels to more subjective categories such as “engagement” and “perceived difficulty”.

Finally, in Chapter 5, we present and evaluate a prototype automatic teaching system whose controller was computed so as to minimize the expected time needed by the student to learn the material and pass the test. The teaching problem is modeled as a POMDP, and the student is modeled as a Bayesian learner who conducts probabilistic inference in the same manner as described in Section 1.3; hence, our system is an example of applying model-based control techniques
to develop an automated teacher. The target learning domain is foreign language learning by image association. For instance, to teach the meaning of the German word *trinken*, the program might show an image of a girl drinking a cup of milk, followed by an image showing a man drinking tea. From these word+image pairs, a rational student could reasonably infer that *trinken* probably means “drink”. This is the same teaching approach used Rosetta Stone language software [73] and the Web-based DuoLingo learning system [30]. While an automated, optimized teacher for this domain is useful in its own right, the greater goal of this chapter is to propose and demonstrate methods for constructing automated teachers using a principled decision framework such as POMDPs. After showing that the learned teaching engine performs favorably compared to two baseline controllers, we describe a plausible architecture for how it might be extended to incorporate affective observations and demonstrate in simulation the potential benefits of doing so.
Chapter 2

Measuring the Benefit of Affective Sensors

Abstract: While affect-sensitive automated teaching systems are becoming an active topic in the ITS community, there is yet no consensus whether responsiveness to students’ affect will result in more effective teaching systems. Even if the benefits of affect recognition were well established, there is yet no obvious path for creating an affect-sensitive automated tutor. In this chapter we present an experimental protocol for measuring the effect on teaching of being able to see the student’s face during teaching. The learning setting we focus on is cognitive skills training. In addition, while conducting the experiment, we simultaneously collect training data with ecological validity that could later be used to develop an automated teacher on cognitive skills. Experimental results suggest that affect-sensitivity in the cognitive games setting is associated with higher learning gains. Behavioral analysis using automatic facial expression coding of recorded videos also suggests that smile may reveal embarrassment rather than achievement in learning scenarios.

2.1 Introduction

Until recently, ITS typically employed only a relatively impoverished set of sensors consisting of a keyboard and mouse, which amounts to only a few bits per
second that they process from the student. While various researchers in the field of ITS have been migrating towards modeling affect in their instructional systems [99, 26, 94], there is, surprisingly, no firm consensus yet on whether affect sensitivity actually makes better automated teachers: In his keynote address [89] to the ITS’2008 conference in Montreal, Kurt VanLehn, a prominent ITS researcher who pioneered the Andes Physics Tutor [90], asserted that affective sensors such as automatic facial expression recognition systems were not useful in ITS, and efforts to utilize them for automated teaching were misguided. Indeed, it is conceivable that the explicit feedback given by the student to the teacher in the form of keystrokes, mouse clicks, and screen touches might constitute all that is needed for the teacher to teach well. On the other hand, we posit two reasons why modeling of affect may be important: (1) **State preference**: Certain affective states in the student may be more desirable than others. For example, a teacher might wish to avoid a situation in which the student becomes extremely upset while attempting to solve a problem. (2) **State disambiguation**: Consider a student who has been asked a question and who has not responded for several seconds. Is the student confused? Is he/she still thinking of the answer and just about to respond? Or has the student disengaged completely and perhaps even left the room? Without some form of affective sensors, these very different states may not be easily distinguished.

In this chapter we tackle two problems: (1) For one particular domain of learning – cognitive skill training (described in Section 2.3) – we investigate whether affective state information is useful for human teachers to teach effectively. We analyze the utility of affective state in terms of learning gains as assessed by a pre-test and post-test on a spatial reasoning task. We use a Wizard-of-Oz (WOZ) paradigm to simulate the environment a student would face when interacting with an automated system. While conducting this experiment, we also (2) Collect data that could be used to train an automated cognitive skills teacher. These data consist of timestamped records of the student’s actions (e.g., move cards on the screen), the teacher’s commands (e.g., change task difficulty), and the student’s face video. The ultimate goal of our research is to identify learning domains in which affect sensitivity is useful to human tutors, and to develop automated sys-
tems that utilize affective information the way human tutors do.

2.2 Related work

Although a number of affect-aware ITS have emerged in recent years, such as affect-sensitive versions of AutoTutor [26] and Wayang Outpost [99], it is still unclear how beneficial the affect sensitivity in these systems actually is. Some research has been conducted on the impact of the use of pedagogical agents on student’s engagement and interest level [99], but studies on the impact of actual learning gains are scarce. The only study to our knowledge that specifically addresses this point is by Aist, et. al [1]: They augmented an automated Reading Tutor, designed to boost reading and speech skills by asking students to read various vocabulary words out loud, with emotional scaffolding using a WOZ framework. In their experiment, a human teacher (in a separate room) watching the student interact with the tutor could provide supplementary motivational audio prompts to the student, e.g., “You’re doing fine.” Compared with students in a control condition who received no emotional scaffolding, students in the affect-enhanced condition chose to persist in the learning task for a longer time. However, no statistically significant increase in learning gains was found. In their study, the only action the human teachers could execute was to issue a prompt – teachers could not, for instance, also change the task difficulty. Moreover, the study did not assess whether the tutors could have been as effective if they did not have access to the video of the student, i.e., if their prompts had been based solely on the student’s accuracy on the task.

2.3 Cognitive skills training

In recent years there has emerged growing interest in “cognitive training” programs that are designed to hone basic skills such as working memory, attention, auditory processing, and logical reasoning. The motivation behind cognitive skills training is that if basic cognitive skills can be improved, performance in academic
subjects such as mathematics and reading may also increase. In recent years cognitive training has been shown to correlate both with increased cognitive skills themselves [60] as well as increased performance in mathematics in minority students [42]. Certain cognitive training regimes have also been shown to boost fluid intelligence (Gf), with larger doses of training associated with larger increases in Gf [45].

In some cognitive skill training programs such as Learning Rx [54], cognitive training sessions are conducted 1-on-1 by a human trainer. Since employing a skilled human trainer for every pupil is expensive, it would be useful to automate the cognitive training process, while maintaining the benefits of having a human teacher.

2.3.1 Human training versus computer training

Learning Rx prescribes a dose of both 1-on-1 human-facilitated training, along with “homework” consisting of computer-based training of the same skills using the same games. In a study comparing the effectiveness of human-based versus computer-based learning with Learning Rx cognitive skill-building games, Hill, et. al found that human-based 1-on-1 training was more effective in terms of learning gains both on the cognitive skills tasks themselves as well as in associated mathematics performance [42]. When trying to develop an automated teaching system of cognitive skills, it is important to understand the causes of this result. We suggest three different hypotheses:

1. **Skill level hypothesis**: Human teachers are very adept at adapting their teaching to the the student’s apparent skill level and explicit game actions.

2. **Affect-sensitivity hypothesis**: Human teachers can adapt to the affective state of the student and thereby teach more effectively.

3. **Mere presence hypothesis**: The mere presence of a human observer can positively influence the student’s performance [41].

Suppose that, in a given learning domain, the reason why human tutors are more effective is because of the skill level hypothesis. Then the benefit of making an
automated teaching system affect sensitive may be minimal and not worthwhile. Similarly, if the mere presence of a human or perhaps a human teacher’s ability to converse freely with the student using perfect speech recognition is the deciding factor in effectiveness, then there is little hope that an automated system can match a human. If, however, affect sensitivity is important for the human teacher, then it may also prove useful for automated systems. In the experiment we describe in the next section, we evaluate the three hypotheses above.

2.4 Experiment

We conducted an experiment to assess the importance of affect in cognitive skills training by an automated teacher. Since we have not yet built such a system, we simulate it using a WOZ paradigm. In WOZ experiments, a human operates behind a “curtain” (a wall, in our case), unbeknownst to the student, and controls the teaching software. For our experiment we developed a battery of three cognitive games that we developed for the Apple iPad:

- **Set**: Similar to the classic card game, Set consists of cards that contain shapes with multiple attributes including size, shape, color, and (for higher-difficulty levels) orientation. The goal is to make as many valid “sets” of 3 cards each during the time allotted. A set is valid if and only if, for each dimension, the values of the three cards are either all the same or all different.

- **Remember**: A series of randomly generated patterns appear for a brief moment (the duration depends on the current difficulty level) on the screen. If the current pattern is the same as the previous pattern, then the student presses the left button on the screen. If the pattern is different, he/she presses the right button. At each time step the student must both act (press a button) and remember the current card.

- **Sum**: Similar to Remember, a series of small integers is presented to the user at a variable rate dependent on the current difficulty level. If the sum
During piloting, students typically found the Set game the most challenging, and the other two tasks were perceived as more recreational and diverting. Hence, we used Set as the primary task that teachers should focus on. The other two tasks were provided as options to the teachers with which to give “breaks” to the students. However these breaks were to be taken only to the extent that they would help with the long term performance on Set. Before each training session, each student performed a 2-minute pre-test on Set, and after the training session (30 minutes) each student performed a 2-minute post-test on the same task. The performance metric during tests was the number of valid sets the student could make in the time allotted. A screenshot of Set (recorded on the iPad simulator) is shown in Figure 2.1. Students control the game by touching an iPad-1. Student actions consist of dragging cards in the Set task, and pressing a Left or Right button during the Remember and Sum tasks. The students’ game inputs, along with videos of their face and upper body, were timestamped and recorded. In addition, we also recorded the teacher’s actions, which consisted of increasing/decreasing task difficulty, switching tasks, giving a hint, and providing motivation in the form of the current and previous numbers is even, then the user presses the left button; if it is odd, he/she presses the right button.
of pre-recorded audio prompts. The teachers were instructed to execute whatever commands they deemed necessary in order to maximize the student’s learning gains on Set. These data was collected with an eye towards analyzing the teaching policies used by teachers and porting them into an automated teacher (see Section 2.6).

2.4.1 Conditions

We compared learning gains on Set across three experimental conditions:

1. **1-on-1**: The student works 1-on-1 with a human trainer who sits beside the student and makes all teaching decisions. The student is free to converse with the teacher. All of the student’s and teacher’s actions on the iPad, as well as a video of the student, are recorded automatically and synchronously.

2. **WOZ (full)**: The student works by him/herself on the iPad. The student is told that the iPad-based game software is controlled by an automatic teacher. In reality, it is controlled by a human trainer in another room who sees both the student’s actions on the iPad as well as the student’s face and upper body behavior over a videoconference. The student does not see or hear the teacher. The teacher’s actions, student’s actions, and student’s video are all recorded automatically and synchronously.

3. **WOZ (blind)**: This condition is identical to the WOZ (full) except that the teacher cannot see or hear the student – the video camera records the student’s behavior but does not transmit it to the teacher. In other words, the teacher is forced to teach without seeing the affective information provided by the student’s face, gestures, and body posture.

Of all the students we interviewed afterwards who had participated in a WOZ condition, none suspected that the “automated teacher” was actually human.

The three conditions were designed to help distinguish which of the three hypotheses given in Section 2.3.1 is most valid. Consider the following possible outcomes, where performance is measured in learning gains (PostTest minus PreTest):
Figure 2.2: Three experimental conditions. **Top:** Human teacher sits with the student in a 1-on-1 training setting. **Middle:** An “automated” teacher is simulated using a Wizard-of-Oz (WOZ) technique. The iPad-based game software is controlled by a human teacher behind a wall. The teacher can see live video of the student. **Bottom:** Same as middle condition, except the teacher cannot see the live video of the student – the teacher sees only the student’s explicit game actions.
Figure 2.3: Average PostTest-minus-PreTest scores versus experimental condition on the “Set” spatial reasoning game. Error bars represent the standard error of the mean. In the two highest-scoring conditions (WOZ and 1-on-1) the teacher was able to observe the student’s affect.

1. 1-on-1 human training is better than WOZ (full): This supports the hypothesis that merely a human’s presence influences learning.

2. All three conditions are approximately equal: This supports the skill level hypothesis that affect is irrelevant to good teaching in this domain.

3. WOZ (full) is better than WOZ (blind): This supports the hypothesis that affect-sensitivity is important to effective teaching.

4. 1-on-1 is worse than the two WOZ conditions: This would suggest that a human’s presence could actually detract from learning, possibly because the student felt intimidated by the human trainer’s presence.

2.4.2 Subjects

The subject pool for this experiment consisted of 66 undergraduate students (51 female), all of whom were African-American, who were recruited from Virginia State University. Each subject was randomly assigned to one of the three conditions described above.
2.5 Experimental results

2.5.1 Learning conditions

Performance was measured as the average PostTest minus PreTest score across each condition; results are shown in Figure 2.3. Although the differences (assessed by 1-way ANOVA) were not statistically significant, the two higher-performance conditions were WOZ (full) and 1-on-1. These were the two conditions in which the student’s affect was visible to the teacher, thus suggesting that affect sensitivity may indeed be important for this learning domain. Interestingly, the WOZ (full) was also higher than 1-on-1 – it is possible that the human teacher’s presence was intimidating for some students and thus led to smaller learning gains.

2.5.2 Facial expression analysis

In addition to assessing differences in learning gains, we also examined how learning performance relates to students’ facial expressions. One particular question of note is the role of smile in learning: Does occurrence of smile perhaps indicate mastery? To investigate this question we performed automatic smile detection across the videos collected of the students during the game play. We employed the Computer Expression Recognition Toolbox (CERT) [57], which is a tool for fully automatic real-time facial expression analysis from video.

To our surprise, the correlation between the average smile intensity (as estimated by CERT) over each video with PostTest-minus-PreTest performance was $-0.34$ ($p < 0.05$). In other words, students who learned more tended to smile less. This suggests that the smiles that do occur may be due more to embarrassment than to a sense of achievement. This also dovetails with findings by Hoque and Picard [44], who found that smiles frequently occur during natural episodes of frustration. Broken down by gender, the correlations were $r = -0.24$ for male ($p > 0.05, N = 15$), and $r = -0.35$ for female ($p < 0.05, N = 51$), suggesting that the effect may be more pronounced for females. We caution, however, that the reliability of CERT’s smile detector on the cognitive training data has yet to be thoroughly validated against manual expression codes. Accuracy of contemporary
Figure 2.4: **Left:** A student who smiles as a result of receiving and acting upon a “giveaway” hint after having not scored any points for approximately 20 seconds. **Right:** A student who smiles after making a mistake, which resulted in a “buzzer” sound.

Figure 2.5: A student who is in the midst of scoring multiple points.

Face detection and smile detection systems on dark-skinned people in particular is known to be less reliable than for other ethnicities [95].

Examples of smiles that occurred during the experiment are shown in Figure 2.4. In Figure 2.4 (right), the subject had just made a mistake (formed an invalid set from 3 cards) which resulted in the game making a “buzzer” sound. Similarly, in Figure 2.4 (left), the teacher had just given a “give-away” hint consisting of all 3 cards necessary to form a valid set. The student “took” the hint (made the hinted set) and then produced the expression shown, which suggests that she may have been embarrassed at needing the assistance. In contrast, the subject in Figure 2.5 was in the midst of scoring multiple points in rapid succession. Her facial expression during this time period shows relatively little variability in general, and no smile in particular.
Figure 2.6: An example of the “traces” collected of the student’s actions, the student’s video, and the teacher’s actions, all recorded in a synchronized manner.

2.6 Towards an automated affect-sensitive teaching system

The pilot experiment described above was conceived both to evaluate the hypotheses discussed in Section 2.3, and also to simultaneously collect training data that can be used to create an automated cognitive skills trainer. Recall that, in the WOZ (full) condition, the student interacts with an apparently “automated” iPad-based teacher, and that in this experimental condition no human was present. This interaction setting closely resembles the setting in which the student interacts with a truly automated trainer. Were training data collected from a 1-on-1 setting in which the student interacted with another human, the elicited affective states and behavior might be very different, and the collected training data might lead the automated system astray.

Given the “traces” of interactions between students and teachers recorded during the experiment (see Figure 2.6), there are several possible strategies for how to develop an affect-sensitive tutor, including rule-based expert systems, stochastic optimal control [6, 19], machine learning, or perhaps some combination of the three. In Woolf, et. al [99], for example, the authors combine manually coded rules with machine learning.

In our project we are pursuing a machine learning approach toward devel-
oping an affect-sensitive tutor:

1. Ask expert human teachers to label the key affective states of the student based both on the student’s actions and his/her video.

2. Perform automatic facial expression recognition on the student’s video, in order to convert the raw video into a form more amenable to automated teaching. Classifiers such as CERT [57] output the estimated intensity of a set of facial muscle movements.

3. Train affective state classifiers that map from the outputs of the facial expression classifier to the higher-level states labeled in the first step.

4. Use supervised learning to compute a policy, i.e., a map from a history of estimated affective states extracted from the live video, the student’s actions, and the teacher’s previous actions, into the teacher’s next action. The data necessary for training are available in the recorded traces.

2.7 Summary and further research

We have presented results from a pilot study assessing the importance of affect in automated teaching of cognitive skills. Results suggest that availability of affective state information may allow the teacher to achieve higher learning gains in the student. In addition, we have found evidence that smile during learning may indicate more embarrassment than achievement. Finally, we have proposed a methodology and software framework for collecting training data from the aforementioned experiment that can be used to train a fully automated, affect-sensitive tutoring agent. In future research we will extend the cognitive training experiment from 1 day to 6 days in an effort to elicit states with more variety, e.g., with more student fatigue.
2.8 Acknowledgement

Support for this work was provided by NSF grants SBE-0542013 and CNS-0454233, and an NSF Leadership Development Institute Fellowship from HBCU-UP to Dr. Serpell. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 2, in full, is a reprint of the material as it appears in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2011. Jacob Whitehill, Zewelanji Serpell, Aysha Foster, Yi-Ching Lin, Brittney Pearson, Marian Bartlett, and Javier Movellan. The dissertation author was the primary investigator and author of this paper.
Chapter 3

Automatic Recognition of Student Engagement

Abstract: In contrast to the more thoroughly studied problems of basic emotion (e.g., happy, sad) and facial action unit recognition [32], relatively little research has addressed the problem of automatically recognizing educationally relevant affective states such as frustration, boredom, fatigue, and engagement. In this chapter, we examine a dataset of 34 undergraduate subjects undergoing cognitive skills training on an Apple iPad and assess how well the degree of a student’s engagement, as perceived by external observers, can be estimated automatically using modern computer vision and machine learning methods. Results indicate that, for subject-independent binary classification (e.g., Engagement = 4 versus Engagement ≠ 4), machine classifiers are on par with humans in terms of accuracy. In addition, we find that 72% of the variance in perceived engagement judgments for video clips can be captured in the static pixels of the constituent frames, suggesting that frame-by-frame classification may be an effective methodology. On the real-valued estimation of the degree, the subject-independent Pearson correlation of machine-estimated labels with human labels was \( r = 0.50 \); inter-human accuracy was \( r = 0.71 \). Analysis of the errors suggest that improved accuracy in face detection and increased robustness to artifacts such as eye glasses may help to reduce this gap. Finally, we show that both human judgments and automatic estimates of engagement are correlated with task performance.
3.1 Introduction

In order to build an affect-sensitive ITS, one needs a method of perceiving affect, as well as a method of integrating the affective state estimate into the decision-making process of how to teach. Both are difficult problems; in this chapter we focus on the perception task. Solving the perception problem requires first identifying which affective states are important for a particular learning domain and then creating an automated classifier that uses auditory, visual, and perhaps even physiological sensor inputs to recognize those emotions. While heuristic rule-based approaches to automatic classification are possible, there is now a general consensus that machine learning approaches that learn from examples yield more accurate and flexible emotion classifiers. Hence, in order to create an automatic classifier of, say, student engagement, one must collect a training dataset, preferably of real students learning in realistic scenarios, e.g., interacting with an intelligent tutoring system or perhaps learning from a human instructor. The training data then need to be labeled for the degree, or perhaps the binary presence/absence, of the target emotion. In contrast to the more thoroughly studied domains of automatic basic emotion (happy, sad, angry, disgusted, fearful, surprised, or neutral) or facial action unit classification (from the Facial Action Coding System [32]), affective states that are relevant to learning such as frustration or engagement may be difficult to define clearly; hence, arriving at a sufficiently clear definition and devising an appropriate labeling procedure, including the timescale at which labeling takes place, is important for ensuring both the reliability and validity of the training labels. Finally, given a set of ecologically valid training data along with high-quality associated labels, one must then develop an algorithm to convert the sensor readings (audio, video, etc.) into an estimate of the affective state. As relatively little research has yet examined how to recognize the emotional states specific to students in real learning environments, it is an open question how well the state-of-the-art methods from the computer vision and facial expression recognition literature would perform on this task.
3.1.1 Recognizing student “engagement”

In this chapter we examine how to construct a real-time, fully automated system to estimate how “engaged” a student appears to be as perceived by an external observer, using visual features of the face. By “engaged”, we mean roughly the definition proposed by Matthews, i.e., “effortful striving directed toward task goals” [62]. We emphasize that our goal is not to “read the student’s mind” to detect his/her level of engagement; instead, we wish to distill the natural human ability of perceiving student engagement into an automatic classifier. Though humans too are imperfect in their judgments, it would already be a boon to contemporary intelligent tutoring systems to be able to match the perceptual prowess of an ordinary human observer.

Our primary purpose in developing an engagement recognizer is to provide an ITS with a useful real-time feedback signal with which to teach more effectively. For example, if the student appears non-engaged while performing the learning task, then the ITS might switch to a different task in an attempt to “perk up” the student. Or, the teacher might ask the student to concentrate harder on the task, and possibly even warn him/her that persistent non-engagement would be noted in the student’s final record. Besides serving as feedback to an ITS, an automated detector of perceived engagement could also be valuable to the student him/herself: if a student is preparing for a job or academic interview, for example, then how engaged the student appears is arguably even more important than how engaged a student feels.

Given the machine learning approach we employed, the first step to developing an engagement classifier was to collect a training corpus and label it for engagement; this required us to devise a procedure for labeling the data that was both efficient and gave reasonable inter-coder reliability. Next, given the training labels, we designed an automated system that takes either a video or image as input and outputs an estimate of the subject’s engagement. Finally, we assessed the degree to which the automated system’s outputs correlate both with human labels, and with standard educational measures such as test performance.

Compared to the existing literature on automatic engagement recognition,
the main contributions of this work are the following:

1. We propose and execute a procedure for annotating training data for a subjective emotional state such as “engagement” and evaluate it in terms of inter-coder reliability over different timescales of labeling.

2. We propose and implement an architecture for recognizing not just binary presence/absence, but also the real-valued degree to which a student appears to be engaged. On the binary task, the automated system matches, and in some cases slightly outperforms, human accuracy. On the real-valued task, the automated system shows reasonably high accuracy compared to human levels.

3. We show that perceived engagement, whether by a human observer or the automated system, correlates with student test performance.

4. We analyze how human observers make their decisions as to what constitutes each level of engagement.

This chapter is structured as follows: We first briefly review related work in Section 3.2. Next, in Section 3.3 we describe the training set that we collected as well as how we annotated it. Then, in Section 3.4, we propose a reasonable architecture of an automated engagement recognition system, implement it, and examine the most important parameters in terms of system accuracy. In Section 3.6 we examine the correlation between engagement estimates and other objective learning measurements. Finally in Section 3.7 we analyze the most important sources of error in the current system and suggest ways for creating better engagement classifiers in the future.

3.2 Related work

Student “engagement” has consistently been identified by the intelligent tutoring systems community as being a key affective state relevant to learning, and hence recognizing it automatically could help to improve the effectiveness of automated teaching. Engagement recognition is also of interest to the human-robot
interaction and human-computer interaction communities. Previous research on automatic student engagement recognition differs principally in how “engagement” is defined and also through what sensors and methods it is recognized. In terms of definition, there are four main categories: neurological and physiological approaches, self-report, external observation, and task performance.

From a neuroscience perspective, engagement might be defined in terms of activation of particular brain regions known to modulate attention and alertness [23]. It may then be possible to estimate the student’s engagement using electroencephalography (EEG) or other neuroimaging technique, as was investigated by [59, 70]. Physiological sensors too may facilitate engagement recognition if they can accurately measure physiological arousal, including blood pressure, of the student [18]. While such neurological and physiological approaches arguably drive as close as possible to a “gold standard” of engagement, it is possible that understanding the exact neural or physiological basis of attention and engagement is a much harder problem than recognizing a “softer” form of engagement defined, for example, in terms of self-report or task performance.

More common in the ITS community are the other kinds of engagement definitions: Engagement as defined by self-report is how engaged the student reports him/herself to be. The self-reported engagement score can be collected in an “emote aloud” fashion [28], or it could be collected after the learning task in a survey. Such surveys need not directly ask the student explicitly how “engaged” he/she feels but could instead map from the survey responses into an engagement measure, e.g., using factor analysis [62]. Given the self-reported engagement scores, one can attempt to recognize engagement automatically using facial expression analysis [63], EEG [39], or physiological sensors such as respiration, heart rate, and skin conductivity sensors [33].

While students themselves have probably more direct access to their own emotions than anyone else, it is not always practical to ask students how engaged they feel. Moreover, a student may feel embarrassed saying he was “very non-engaged” and may sometimes give inaccurate reports. An alternative measure of engagement is to ask an external observer how engaged the student appears to be.
based on live or recorded video of the student in the learning environment, combined perhaps with synchronized information on the student’s task performance. Given this definition of engagement, similar kinds of sensors and recognition approaches as for self-report can be applied to the automatic recognition problem, including facial expression analysis [99, 29, 63], posture analysis [75, 99], or EEG.

Finally, some engagement recognition systems treat “engagement” as a latent state that affects the student’s accuracy and perhaps response time in the learning task. For instance, a “non-engaged” student might be defined as a student who simply guesses randomly when answering a question. Using probabilistic inference, an ITS can estimate the student’s engagement level at run-time by assessing how well the student’s recent task performance can be explained by an “engaged” student compared to a “non-engaged” student [12, 46]. This technique has been dubbed “engagement tracing” [12], which is an allusion to the standard “knowledge tracing” technique in many ITS [49]. Estimating engagement in this manner does not even require any kind of “affective sensors” such as a web camera at all – engagement can be estimated using just the student’s task performance. However, it is also possible that, by additionally harnessing affective sensors, a more accurate estimate of the student’s engagement level could be obtained more quickly.

In the present study, we focus on automatically recognizing the appearance of a student’s “engagement” level, as defined by external observers, using a web camera and automatic face analysis methods.

3.3 Dataset collection and annotation for an automatic engagement classifier

Given our goal to train an automatic engagement recognizer that can approximate human judgments of how engaged a student appears, it is necessary to collect a training dataset. Our particular goal is to develop a system that can provide useful real-time observations to an intelligent tutoring system; hence, it makes sense to collect data from a setting in which students are interacting with
The data for this study were collected from 34 undergraduate students who participated in a 2010-2011 “Cognitive Games” experiment whose purpose was to measure the importance to teaching of seeing the student’s face [97]. In that experiment, video and synchronized task performance data were collected from subjects interacting with cognitive skills training software. Cognitive skills training has generated substantial interest in recent years; the goal is to boost students’ academic performance by first improving basic skills such as memory, processing speed, and logic and reasoning. A few prominent such systems include Brainskills (by Learning RX [54]) and FastForWord (by Scientific Learning [77]). The Cognitive Games experiment utilized custom-built cognitive skills training software, reminiscent of Brainskills, that was installed on an Apple iPad. A webcam was used to videorecord the students; it was placed immediately behind the iPad and aimed directly at the student’s face. The software consisted of three games – Set (very similar to the classic card game), Remember, and Sum – that trained logical, reasoning, perceptual, and memory skills. The dependent variables during the 2010-2011 experiment were pre- and post-test performance on the Set game.

Experimental data for the engagement study in this chapter were taken from 34 subjects from two pools: (a) the 26 subjects who participated in the Spring 2011 version of the Cognitive Games study at a Historically Black College/University (HBCU) in the southern United States. All of these subjects were African-American, and 20 were female. Additional data were collected from (b) the 8 subjects who participated in the Summer 2012 version of the Cognitive Games study at a university in California (UC), all of whom were either Asian-American or Caucasian-American, and 5 of whom were female. For the present study, the HBCU data served as the primary data source for training and testing the engagement recognizer. The UC dataset allowed us to assess how well the trained system would generalize to subjects of a different race – a known issue in modern computer vision systems.

In the experimental setup, each subject sat in a private room and played the cognitive skills software either alone or together with the experimenter. The iPad
was horizontally situated approximately 30 centimeters in front of the subject’s face and vertically so that the iPad was slightly below eye level. Behind the iPad pointing towards the subject was a Logitech web camera recording the entire session.

During each session, the subject gave informed consent and then watched a 3 minute video on the iPad explaining the objectives of the three games and how to play them. The subject then took a 3 minute pre-test on the Set game to measure baseline performance. Test performance was measured as the number of valid “sets” of 3 cards (according to the game rules) that the student could form within 3 minutes. The particular cards dealt during testing were the same for all subjects. After the pre-test, the subject then underwent 35 minutes of cognitive skills training using the game software. The software was controlled by the human trainer, who either sat next to the student (in the 1-on-1 condition) or monitored the experiment remotely from a separate room (in the Wizard-of-Oz conditions). The trainer’s goal was to help the student maximize his/her test performance on Set. After the training period, the subject took a post-test on Set and then was done.

3.3.1 Data annotation

From scanning the recorded videos of the cognitive training sessions, it was clear that there was considerable variation across subjects of their degree of engagement. For example, some subjects were highly attentive to the cognitive
games almost the entire time, whereas one subject literally fell asleep during the experiment. There was also variation within subjects; for instance, one subject who usually appeared highly engaged spent a few seconds looking away from the iPad while answering his cellular phone.

Given these recorded videos, the next step was to annotate (label) them for “engagement”. Since our goal was to build an engagement recognizer to estimate how engaged the student appears to be, we organized a team of labelers, consisting of undergraduate and graduate students from computer science, cognitive science, and psychology from the two universities where data were collected. These labelers viewed and rated the videos for the appearance of engagement. In pilot experimentation we tried three different approaches to labeling:

1. Watching video clips and giving continuous engagement labels by adjusting a “dial” (in practice, just the Up/Down arrow keys).

2. Watching video clips and giving a single number to rate the entire video.

3. Viewing static images and giving a single number to rate the entire video.

We found approach (1) very difficult to execute in practice. One problem was the tendency to habituate to each subject’s recent level of engagement, and to adjust the current rating relative to that subject’s average engagement level of the recent past. This could yield labels that are not directly comparable between subjects or even within subjects. Another problem was how to rate short events, e.g., brief eye closure or looks to the side: should these brief moments be labeled as “non-engagement”, or should they be overlooked as normal behavior if the subject otherwise appears highly engaged? Finally, it was difficult to provide continuous labels that were synchronized in time with the video; proper synchronization would require first scanning the video for interesting events, and then re-watching it and carefully adjusting the engagement up or down at each moment in time. We found the labeling task was easier using approaches (2) and (3), provided that clear instructions were given as to what constitutes “engagement”.
3.3.2 Engagement categories and instructions

Given the approach of giving a single engagement number to an entire video clip or image, we decided on the following approximate scale to rate engagement:

1: Not engaged at all – e.g., looking away from computer and obviously not thinking about task, eyes completely closed.

2: Nominally engaged – e.g., eyes barely open, clearly not “into” the task.

3: Engaged in task – student requires no admonition to “stay on task”.

4: Very engaged – student could be “commended” for his/her level of engagement in task.

X: The clip/frame was very unclear, or contains no person at all.

Example images and mean face images for each engagement level are shown in Figures 3.3 and 3.2, respectively.

Labelers were instructed to label clips/images for “How engaged does the subject appear to be?” The key here is the word appear – we purposely did not want labelers to try to infer what was “really” going on inside the students’ minds because this left the labeling problem too open-ended. This has the consequence that, if a subject blinked, then he/she was labeled as very non-engaged.
Figure 3.3: Sample faces for each engagement level from the HBCU subjects. All subjects gave written consent to publication of their face images.
51

(Engagement = 1) because, at that instant, he/she appeared to be non-engaged. In practice, we found that this made the labeling task clearer to the labelers and still yielded informative engagement labels. If the engagement scores of multiple frames are averaged over the course of a video clip (see Section 3.3.4), momentary blinks will not greatly affect the average score anyway. In addition, labelers were told to judge engagement based on the knowledge that subjects were interacting with game software on an iPad directly in front of them. Any gaze around the room or to another person (i.e., the experimenter) should be considered non-engagement (rating of 1) because it implied the subject was not engaging with the iPad. The goal here was to help the system generalize to a variety of settings where students should be looking directly in front of them.

3.3.3 Timescale

An important variable in annotating video is the timescale at which labeling takes place. For approach (2), we experimented with two different time scales: clips of 60 sec and clips of 10 sec. Approach (3) (single images) can be seen as the lower limit of the length of a video clip. In a pilot experiment we compared these three approaches (two timescales for video, plus single images) for inter-coder reliability. As performance metric we used Cohen’s $\kappa$ averaged over all labelers in a leave-one-labeler-out fashion (see Appendix). Since the engagement labels belong to an ordinal scale and are not simply categories, we used a weighted $\kappa$ with quadratic weights to penalize label disagreement.

For the 60 sec labeling task, all the video sessions (~ 45 minutes/subject) from the HBCU subjects were watched from start to end in 60 sec clips, and labelers entered a single engagement score after viewing each clip. For the 10 sec labeling task, 505 video clips of 10 sec each were extracted at random timepoints from the session videos and shown to the labelers in random (in terms of both time and subject) order. Between the 60 sec clips and the 10 sec labeling tasks, we found the 10 sec labeling task more intuitive. When viewing the longer clips, it was difficult to know what label to give if the subject appeared non-engaged early on but appeared highly engaged at the end. The inter-coder reliability of the 60
sec clip labeling task was $\kappa = 0.39$ (across two labelers); for the 10 sec clip labeling task $\kappa = 0.68$.

For approach (3), we created custom labeling software in which labelers annotated batches of 100 images each. The images for each batch were video frames extracted at random timepoints from the session videos. Each batch contained a random set of images spanning multiple timepoints from multiple subjects. Labelers rated each image individually but could view many images and their assigned labels simultaneously on the screen. The labeling software also provided a Sort button to sort the images in ascending order by their engagement label. In practice, we found this to be an intuitive and efficient method of labeling images for the appearance of engagement. The inter-coder reliability for image-based labeling was $\kappa = 0.56$. If we exclude the single labeler with the lowest inter-subject agreement, then $\kappa = 0.60$.

In terms of timescale, it seems that short video clips give the best reliability of the three timescales. However, the reliability of image-based labeling is not far behind that of the 10 sec video clips. Furthermore, it is possible that, if we average together the labels for individual image frames that were sampled from consecutive timepoints from a single video, then the reliability of these averaged sets of frames might be even higher. We examined this issue in the subsection below.

### 3.3.4 Static pixels versus dynamics

One interesting question is how much information about students’ engagement is captured in the static pixels of the individual video frames compared to the dynamics of the motion. We conducted a pilot study to examine this question. In particular, we randomly selected 120 video clips (10 sec each) across many subjects and split each clip into 40 frames spaced 0.25 sec apart. These frames were then shuffled both in time and across subjects. A human labeler then labeled these image frames for the appearance of engagement, as described in “approach (3)” of Section 3.3.1. Finally, the engagement values assigned to all the frames for a particular clip were reassembled and averaged; this average served as an estimate of the “true” engagement score given by that same labeler when viewing that video
clip as described in “approach (2)” above. We found that, with respect to the true engagement scores, the estimated scores gave a $\kappa = 0.78$, and the fraction of explained variance was 0.72. Though not perfect, this “reconstruction” accuracy is quite high, and suggests that most of the information about the appearance of engagement is contained in the static pixels, not the motion per se.

In addition to computing the reconstruction accuracy, we also examined the video clips in which the reconstructed engagement scores differed the most from the true scores. In particular, we ranked the 120 labeled video clips in decreasing order of absolute deviation of the estimated label (by averaging the frame-based labels) from the “true” label given to the video clip viewed as a whole. We then examined these clips and attempted to explain the discrepancy:

In the first clip (greatest absolute deviation), the subject was swaying her head from side to side as if listening to music (although she was not). It is likely that the coder treated this as non-engaged behavior. Clearly, this is a behavior that cannot be easily captured from static frame judgments – it requires some means of recognizing the dynamics. However, it was also an anomalous case.

In the second clip, the subject turned his head to the side to look at the experimenter, who was talking to him for several seconds. In the frame-level judgments, this was perceived as off-task, and hence non-engaged behavior; this corresponds to the instructions given to the coders that they rate engagement under the assumption that the subject should always be looking towards the iPad. For the video clip label, however, the coder judged the student to be highly engaged because he was intently listening to the experimenter. This is an example of inconsistency on the part of the coder as to what constitutes engagement and does not necessarily indicate a problem with splitting the clips into frames.

Finally, in several clips the subjects shifted their eye gaze downward several times to look at the bottom of the iPad screen. At a frame level, it was difficult to distinguish the subject looking at the bottom of the iPad from the subject looking to his/her own lap, which would be considered non-engagement. This is an example of a student behavior that can be more accurately labeled from video clips compared to frames. In most videos, however, the mislabeling of the subjects’
downward gaze was occasional and effectively filtered out by simple averaging.

The relatively high accuracy of estimating video-based labels from frame-based labels suggests an approach for how to construct an automatic classifier of engagement: Instead of analyzing video clips as video, break them up into their video frames, and then somehow combine engagement estimates for each frame. In the next section, we describe our proposed architecture for automatic engagement recognition based on this frame-by-frame design.

### 3.4 Automatic recognition architectures

Based on the finding from Section 3.3.4 that video clip-based labels can be estimated with high fidelity simply by averaging frame-based labels, we focus our study on frame-by-frame recognition of student engagement. This means that that many techniques developed for emotion and facial action unit classification can be applied to the engagement recognition problem. In this chapter we proposed

**Figure 3.4**: Automatic engagement recognition pipeline.
a 3-stage pipeline (see Figure 3.4):

1. Face registration: the face and facial feature positions are localized in the image; the face box coordinates are computed; and the face patch is cropped from the image [57]. We experimented with $36 \times 36$ and $48 \times 48$ pixel face resolution.

2. The cropped face patch is classified by four binary classifiers, one for each engagement category $e \in \{1, 2, 3, 4\}$.

3. The outputs of the binary classifiers are fed to a regressor to estimate the image’s engagement level.

Stage (1) is standard for automatic face analysis tools, and our particular approach is described in [57]. Stage (2) is discussed in the next subsection, and stage (3) is discussed in Section 3.4.8.

### 3.4.1 Binary classification

We trained four binary classifiers of engagement – one for each of the four levels described in Section 3.3.1. The task of each of these classifiers is to discriminate an image (or video frame) that belongs to engagement level $e$ from an image that belongs to some other engagement level $e' \neq e$. We call these detectors 1-v-rest, 2-v-rest, etc. We compared three commonly used and demonstrably effective feature type + classifier combinations from the automatic facial expression recognition literature:

- GentleBoost with Box Filter features (GB(BF)): this is the approach popularized Viola and Jones in [92] for face detection.

- Support vector machines with Gabor Energy Filters (SVM(GEF)): this approach has achieved some of the highest accuracies in the literature for facial action and basic emotion classification [57].

- Multivariate logistic regression with expression outputs from the Computer Expression Recognition Toolbox [57] (MLR(CERT)): here, we attempt to
harness an existing automated system for facial expression analysis to train the student engagement classifiers.

We describe each approach in more detail below:

**GB(BF)**

Box Filter (BF) features were shown in [95] to be highly effective for automatic smile detection. At run-time, BF features are extremely fast to extract using the “integral image” technique [92]. At training time, however, the number of BF features relative to the image resolution is very high compared to other image representations (e.g., a Gabor decomposition), which can lead to overfitting. BF features are typically combined with a boosted classifier such as Adaboost [36] or GentleBoost (GB) [37], which performs both feature selection during training and actual classification at run-time. In our GentleBoost implementation, each weak learner consists of a non-parametric regressor smoothed with a Gaussian kernel of bandwidth $\sigma$, to estimate the log-likelihood ratio of the class label given the feature value. Each GentleBoost classifier was trained for 100 boosting rounds. For the features, we included 6 types of Box Filters in total, comprising two-, three-, and four-rectangle features similar to those used in [92], and an additional two-rectangle “center-surround” feature.

**SVM(GEF)**

Gabor Energy Filters (GEF) [65] are bandpass filters with a tunable spatial orientation and frequency. They model the complex cells of the primate’s visual cortex. Gabor Energy Filters have a proven record in a wide variety of face processing applications, including face recognition [52] and facial expression recognition [57]. In machine learning applications GEF features are often classified by a soft-margin linear support vector machine (SVM) with parameter $C$ specifying how much misclassified training examples should penalize the objective function. In our implementation, we applied a “bank” of 40 Gabor Energy Filters consisting of 8 orientations (spaced at 22.5deg intervals) and 5 spatial frequencies spaced at half-octaves.
MLR(CERT)

The Facial Action Coding System [32] is a comprehensive framework for objectively describing facial expression in terms of Action Units, which measure the intensity of over 40 distinct facial muscles. Manual FACS coding has previously been used to study student engagement and other emotions relevant to automated teaching [48, 63]. In our study, since we are interested in automatic engagement recognition, we employ the Computer Expression Recognition Toolbox (CERT), which is a software tool developed by our laboratory to estimate facial action intensities automatically [57]. Although the accuracies of the individual facial action classifiers vary, we have found CERT to be useful for a variety of facial analysis tasks, including the discrimination of real from faked pain [56], driver fatigue detection [93], and estimation of students’ perception of curriculum difficulty [94]. CERT outputs intensity estimates of 20 facial actions as well as the 3-D pose of the head (yaw, pitch, and roll). For engagement recognition we classify the CERT outputs using multivariate logistic regression (MLR), trained with an $L_2$ regularizer on the weight vector of strength $\alpha$. We use the absolute value of the yaw, pitch, and roll to provide invariance to the direction of the pose change.

Internally, CERT uses the SVM(GEF) approach described above. Since CERT was trained on 280 subjects, which is substantially higher than the number of subjects collected for this study, it is possible that CERT’s outputs will provide an identity-independent representation of the students’ faces, which may boost generalization performance.

3.4.2 Data selection

We used the following procedure to select training and testing data for each binary classifier to distinguish e-v-rest:

1. For each of the labeled HBCU images, we considered the set of all labels given to that image by all the labelers. If any labeler marked the frame as X (no face, or very unclear), then the image was discarded.

2. If the minimum and maximum label given to an image differed by more than
1 (e.g., one labeler assigns a label of 1 and another assigns a label of 3), then the image was discarded.

3. If the automatic face detector (from CERT [57]) failed to detect a face, or if the largest detected face was less than 36 pixels wide (usually indicative of a false alarm), the image was discarded.

4. Otherwise, the “ground truth” label for that image was computed by rounding the average label for that image (e.g., 2.4 rounds to 2; 2.5 rounds to 3). If the rounded label equalled $e$, then that image was considered a positive example for that classifier’s training set; otherwise, it was considered a negative example.

In total there were 14204 frames from the HBCU dataset selected using this approach.

3.4.3 Cross-validation

We used 4-fold subject-independent cross-validation to measure the accuracy of each trained binary classifier. Specifically, the set of all labeled frames was partitioned into 4 folds such that no subject appeared in more than one fold; hence, the cross-validation estimate of performance gives a sense of how well the classifier would perform on a novel subject on which the classifier was not trained.

3.4.4 Accuracy metric

We use the 2AFC [100] metric to measure accuracy, which expresses the probability of correctly discriminating a positive example from a negative example in a 2-alternative forced choice classification task. Under mild assumptions the 2AFC is equivalent to the area under the Receiver Operating Characteristics curve, which is commonly used in the facial expression recognition literature. To assess the machine’s accuracy relative to inter-human accuracy, we computed the 2AFC for human labelers as well, using the same image selection criteria as described in Section 3.4.2.
3.4.5 Hyperparameter selection

Each of the classifiers listed above has a hyperparameter associated with it (either $\sigma$, $C$, or $\alpha$). The choice of hyperparameter can impact the test accuracy substantially, and it is a common pitfall to give an overly optimistic estimate of a classifier’s accuracy by manually tuning the hyperparameter based on the test set performance. To avoid this pitfall, we instead optimize the hyperparameters using only the training set by further dividing each training set into 4 subject-independent inner cross-validation folds in a double cross-validation paradigm. We selected hyperparameters from the following sets of values: $\sigma \in \{10^{-2}, 10^{-1.5}, \ldots, 10^0\}$, $C \in \{0.1, 0.5, 2.5, 12.5, 62.5, 312.5\}$, and $\alpha \in \{10^{-5}, 10^{-4}, \ldots, 10^5\}$.

3.4.6 Results: binary classification

Classification results are shown in Table 3.1 for cropped face resolution of $48 \times 48$ pixels. Accuracy at $36 \times 36$ pixel resolution was slightly lower. All results are for subject-independent classification. From the upper table, we see that the binary classification accuracy given by the machine classifiers is very similar to inter-human accuracy. All of the three architectures tested delivered similar performance averaged across the four tasks (1-v-rest, 2-v-rest, etc.). However, MLR(CERT) performed noticeably worse for 1-v-rest, and noticeably better for 4-v-rest. As we discuss in Section 3.5, many images labeled as Engagement = 1 exhibit eye closure; in addition, Engagement = 4 can be discriminated using pose information. It is possible that CERT’s eye closure detector is relatively inaccurate, and in comparison the GB(BF) and SVM(Gabor) approaches are able to learn an accurate eye closure detector from the training data themselves. On the other hand, CERT may have an advantage in terms of pose information because CERT’s pose detector was trained on tens of thousands of subjects.

Overall we find the results encouraging that machine classification of engagement can reach inter-human levels of accuracy. However, another important problem beyond binary classification is estimating the real-valued degree of engagement, which we examine in Section 3.4.8.
Table 3.1: **Top**: Subject-independent, within-dataset (HBCU), image-based engagement recognition accuracy for each engagement level $e \in \{1, 2, 3, 4\}$ using each of the three classification architectures, along with inter-human classification accuracy. **Bottom**: Engagement recognition accuracy on a different dataset (UC) not used for training.

<table>
<thead>
<tr>
<th>Task</th>
<th>MLR (CERT)</th>
<th>GB (BF)</th>
<th>SVM (Gabor)</th>
<th>Human</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-v-rest</td>
<td>0.8322</td>
<td>0.9697</td>
<td>0.9139</td>
<td>0.9132</td>
</tr>
<tr>
<td>2-v-rest</td>
<td>0.7554</td>
<td>0.7688</td>
<td>0.7109</td>
<td>0.6736</td>
</tr>
<tr>
<td>3-v-rest</td>
<td>0.5842</td>
<td>0.6107</td>
<td>0.6303</td>
<td>0.6272</td>
</tr>
<tr>
<td>4-v-rest</td>
<td>0.7011</td>
<td>0.6101</td>
<td>0.6600</td>
<td>0.6563</td>
</tr>
<tr>
<td>Avg</td>
<td>0.7182</td>
<td>0.7398</td>
<td>0.7288</td>
<td>0.7176</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>MLR (CERT)</th>
<th>GB (BF)</th>
<th>SVM (Gabor)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-v-rest</td>
<td>0.7745</td>
<td>0.9234</td>
<td>0.8307</td>
</tr>
<tr>
<td>2-v-rest</td>
<td>0.6179</td>
<td>0.7049</td>
<td>0.6678</td>
</tr>
<tr>
<td>3-v-rest</td>
<td>0.5053</td>
<td>0.5262</td>
<td>0.5696</td>
</tr>
<tr>
<td>4-v-rest</td>
<td>0.6176</td>
<td>0.6540</td>
<td>0.6965</td>
</tr>
<tr>
<td>Avg</td>
<td>0.6288</td>
<td>0.7021</td>
<td>0.6911</td>
</tr>
</tbody>
</table>
3.4.7 Generalization to a different dataset

A well-known issue for contemporary face analysis systems is to generalize to people of a different race from the people in the training set; in particular, modern face detectors often have difficulty detecting people with dark skin [95]. For our study, we collected data both at HBCU, where all the subjects were African-American, as well as UC, where all the subjects were either Asian-American or Caucasian-American. This gives us the opportunity to assess how well a classifier trained on one dataset generalizes to the other. Here, we measure performance of the binary classifiers described above that were trained on HBCU when classifying subjects from UC.

Results are shown in Table 3.1 for each feature type and classifier combination. For GB(BF) and SVM(GEF), the degradation in performance was mild. Interestingly, the MLR(CERT) architecture generalized the worst, even though CERT was trained on a much larger number of subjects (several hundred, compared to just 26 for this study) and outputs only a small number of features. It is possible that the head pose features that are measured by CERT and are useful for the HBCU dataset do not generalize to the UC dataset.

3.4.8 Regression

After performing binary classification of the input image for each engagement level \( e \in \{1, 2, 3, 4\} \), the final stage of the pipeline is to combine the classifier outputs into a real-valued estimate of the student’s engagement. Here, as a first implementation, we use standard linear regression using the raw binary classifier outputs as features. Note, however, that more sophisticated methods of “averaging” are possible, such as using a non-linear classification method over histograms of frame-based engagement scores instead of simply taking the sample mean.

3.4.9 Results: regression

We chose the SVM(GEF) architecture and trained a linear regressor to map from the four binary classifiers’ outputs to a real-valued engagement estimate.
Subject-independent 4-fold cross-validation accuracy, measured using Pearson’s correlation $r$, was 0.50. For comparison, inter-human accuracy on the same task was 0.71.

The fact that human accuracy approximately equal to machine accuracy on the binary classification tasks, whereas it is higher for the regression task, suggests that there are some images on which the binary engagement classifiers make “egregious mistakes”, e.g., the automated system believes that a student whom humans labeled as a 4 was a 1. In other words, the machine may make fewer binary classification mistakes than the humans do, but the mistakes it makes are much “worse”. We discuss this further in Section 3.7.

### 3.5 Reverse-engineering the human labelers

Given that our goal in this project is to recognize student engagement as perceived by an external observer, it is interesting and instructive to analyze how the human labelers formed their judgments. From Figure 3.3 we can see clearly that eye closure is a strong indicator of low engagement, as are large head pose deviations from frontal. At engagement level 2, which we call “nominally engaged”, the subject is often resting his/her head on one hand. This is also reflected in the slight smearing of the subject’s left cheek in the Engagement = 2 image from the HBCU subjects and more pronouncedly from the UC subjects in Figure 3.2. Note the subtle difference between resting the head on the hand and placing the hand in front of the face, as exhibited by the third subject from the left in the Engagement = 4 row in Figure 3.3. The difference between groups Engagement = 3 and Engagement = 4 is subtle but noticeable, particularly in the intensity of the subjects’ eye gaze towards the screen and possible furrowing of the eye brow when Engagement = 4. Finally, Figure 3.3 also suggests that subjects in Engagement = 4 lean forward slightly towards the iPad.

We can also use the weights assigned to the CERT features that were learned by the MLR(CERT) classifiers to assess quantitatively how the human labelers judged engagement – if the MLR weight assigned to AU 45 (eye closure) had a large
Figure 3.5: Weights associated with different Action Units (AUs) to discriminate Engagement = 4 from Engagement ≠ 4, along with examples of AUs 1 and 10. Pictures courtesy of Carnegie Mellon University’s Automatic Face Analysis group webpage.

<table>
<thead>
<tr>
<th>CERT feature</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs(Roll)</td>
<td>−0.5659</td>
</tr>
<tr>
<td>AU 10</td>
<td>+0.5089</td>
</tr>
<tr>
<td>AU 1</td>
<td>−0.4430</td>
</tr>
<tr>
<td>AU 45</td>
<td>−0.2851</td>
</tr>
<tr>
<td>abs(Pitch)</td>
<td>−0.2644</td>
</tr>
</tbody>
</table>

magnitude, for example, then that would suggest that eye closure was an important factor in how humans labeled the dataset on which that MLR classifier was trained. In particular, when we examine the 5 MLR weights of highest magnitude that were learned by the 4-v-rest MLR(CERT) classifier, we obtain the results shown in Figure 3.5. (Here, we trained a 4-v-rest classifier on all the subjects’ data because we were not interested in cross-validation performance.) The most discriminating feature was the absolute value of roll (in-plane rotation of the face), with which Engagement = 4 was negatively associated (weight of −0.5659). It is possible that the hand-resting-on-hand that is prominent for Engagement = 2 also induces roll in the head, and that the MLR(CERT) classifier learned this trend. The second most discriminating facial action was Action Unit 10 (upper lip raiser), which was positively correlated with Engagement = 4; speculatively, this AU may suggest that frustration or even annoyance at the learning task can be correlated with high levels of engagement. AU 1 (inner brow raiser), AU 45 (eye closure), and the absolute value of pitch (tilting of the head up and down) were also negatively correlated with Engagement = 4.
Table 3.2: Correlation of student engagement with test scores. Correlations with a * are statistically significant ($p < 0.05$).

<table>
<thead>
<tr>
<th>Correlations of Engagement with Test Scores</th>
<th>Pre-test</th>
<th>Post-test</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Human labelers</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean engagement label</td>
<td>0.52*</td>
<td>0.37</td>
</tr>
<tr>
<td>$P(\text{Engagement} = 4)$</td>
<td>0.57*</td>
<td>0.47*</td>
</tr>
<tr>
<td><strong>Automatic classifier</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P(\text{Engagement} = 4)$</td>
<td>0.64*</td>
<td>0.27</td>
</tr>
</tbody>
</table>

3.6 Comparison to objective measurements

Our primary purpose in developing an engagement recognition system is to provide an ITS with a real-time feedback signal that agrees with human perception of student engagement. However, it would also be desirable for both human perceptions and automatic judgments of student engagement to be predictive of certain objective measurements of the learning session. In this section we investigate the correlation between human and automatic perceptions of engagement with student test performance and learning.

3.6.1 Test performance

Using the student pre- and post-test data collected during the Cognitive Games experiment, we assessed whether perceived student engagement is correlated with test performance. Note that it is not obvious a priori what the correlation should be – a student may appear non-engaged because the task is too difficult and he/she has given up, or because the task is too easy and he/she is bored. In addition, the reasons for a positive correlation between test performance and engagement can also vary. For instance, high pre-test performance might motivate students to try hard during the remainder of the session, while low pre-test performance might discourage students from trying. Alternatively, students who are non-engaged from the onset, perhaps because they did not understand the task instructions, might expend very little effort both during the pre-test and during
the rest of the session.

Human labels

We first compared human judgments of engagement with test performance by computing the mean engagement label over all labeled frames for each subject in the HBCU dataset, and then correlating these mean engagement labels with pre-test and post-test scores. The Pearson correlation between engagement and pre-test was $r = 0.52 \ (p < 0.05)$ and between engagement and post-test was $r = 0.37$ (not statistically significant).

We also examined which of the 4 engagement levels was most predictive of task performance by correlating the fraction of frames labeled as Engagement = 1, Engagement = 2, etc., with student test performance. Only Engagement = 4 was positively correlated with pre-test ($r = 0.57, p < 0.05$) and post-test ($r = 0.47, p < 0.05$) performance. In fact, the fraction of frames for which a student appeared to be in engagement level 4 was a better predictor than the mean engagement predictor described above. All the other engagement levels $e < 4$ were negatively (though non-significantly) correlated with test performance, suggesting that Engagement = 4 is the only “positive” engagement state.

For comparison, the correlation between students’ pre-test and post-test scores was $r = 0.44$ – in other words, human perceptions of student Engagement = 4 as labeled on a frame-by-frame basis is a better predictor of post-test performance than is the student’s pre-test score.

Automatic estimates

We also computed the correlation between automatic judgments of engagement and student pre- and post-test performance. Since the best predictor of test performance from human judgments was from the fraction of frames labeled as Engagement = 4, we focused on the output of the 4-v-rest classifier. In particular, we correlated the fraction of frames over each subject’s entire video session that the 4-v-rest detector predicted to be a “positive” frame by thresholding with $\tau$, where $\tau$ is the median detector output over all subjects’ frames. In other words, frames
on which the detector’s output exceeded $\tau$ was considered to be a “positive” frame for engagement level 4. The correlation with this automatic Engagement = 4 predictor and pre-test performance was 0.64 ($p < 0.05$); for post-test performance, it was $r = 0.27$ (not statistically significant).

### 3.6.2 Learning

In addition to raw test performance, we also examined correlations between engagement and learning. One simple definition of “learning” is post-test minus pre-test scores. The correlation between mean engagement for each subject, as labeled by humans, and post-test minus pre-test, was practically 0. The correlation between fraction of frames labeled as Engagement = 4 and post-test minus test was 0.08 and was not statistically significant.

On the other hand, the simple post-test minus pre-test measure does not account for possible ceiling effects in testing – it is possible that increasing test performance from, say, 10 to 12 is much harder than from 0 to 2. Hence, as an alternative measure of learning, we also examined $\exp(\text{post-test}) - \exp(\text{pre-test})$ which gives a larger weight to score increases starting from a higher baseline. Using this new learning metric, the correlation between mean label and learning is $r = 0.32$, and the correlation between fraction of frames labeled as Engagement = 4 and learning was $r = 0.37$. Though neither correlation is statistically significant, these results still mildly support the possibility of a ceiling effect in the Set task on which students were tested.

### 3.7 Error analysis

In this section we consider some of the most important sources of error in the automated engagement detector. Identifying the important classes of incorrectly classified images may suggest avenues for further research that will help to improve future engagement recognition systems.

We examined the most egregious (i.e., largest absolute difference between human labels and machine-estimated label) mistakes made by the automatic en-
Figure 3.6: Representative images that the automated engagement detector misclassified. **Left**: inaccuracy face or facial feature detection. **Middle**: Thick-rimmed eyeglasses that was incorrectly interpreted as eye closure. **Right**: Subtle distinction between looking down and eye closure.

Engagement regressor (see Section 3.4.8) within the two cross-validation folds in which accuracy was the lowest. There were three chief sources of errors: (1) The face detector or facial feature detector did not accurately find the face or facial features (Figure 3.6 **left**). Given an inaccurate facebox, it is natural that all downstream processing, including engagement recognition, can be highly inaccurate. (2) The presence of thick-rimmed eyeglasses can skew the output of the eye closure detector, which is a key input to the classifier of Engagement=1. One female subject (Figure 3.6 **middle**) in particular wore such eyewear and though the human labels indicated she was highly engaged, the automatic classifier believed her eyes were mostly closed. (3) It was sometimes difficult to distinguish a subject looking down to the bottom of the iPad from looking completely down to his/her lap in a non-engaged manner (Figure 3.6 **right**). This resulted in many frames being misclassified for one male subject in particular. Interestingly, if we re-compute the system’s accuracy after removing this subject from the test set, then the cross-validation accuracy increases substantially to $r = 0.56$.

Problems (1) and (2) should likely diminish as face and facial feature detectors become more accurate and robust to different head poses, eyewear, and skin color. Problem (3) may possibly improve with a larger training set.
3.8 Conclusion

We have presented an architecture for an automatic recognition system of how “engaged” a student who is interacting with an intelligent tutoring system appears to be. Analysis of human engagement labels suggests that most of the information is captured from static pixels, not the video dynamics, so that frame-by-frame analysis is possible.

On binary classification tasks (e.g., Engagement = 4 versus Engagement ≠ 4), the machine classifier’s accuracy is comparable to humans’. On the real-valued regression task, machine accuracy (Pearson $r = 0.50$) is reasonable but needs improvement compared to inter-human agreement ($r = 0.71$). Analysis of the most egregious classification mistakes suggest that accuracy can be improved through more accurate face and facial feature detection as well as greater robustness to artifacts such as eyeglasses. In addition, when we consider that modern expression recognition systems are typically trained in hundreds [57] if not thousands [95] of subjects, it seems reasonable to assume that higher accuracy can be achieved by growing the training set. Even at the current accuracy levels, however, the automatic engagement detector developed in this study is already predictive of objective measurements such as test performance, and it may serve as a useful feedback signal to automated teaching systems.

3.9 Appendix: calculating inter-human accuracy

The automatic classifiers we develop for this chapter are trained on and evaluated against the average label, across all human labelers, given to each image. In order to enable a fair comparison between inter-human accuracy and machine-human accuracy, we assess the accuracy (using Cohen’s $\kappa$, Pearson’s $r$, or the 2AFC metric) of each human labeler $l$ by comparing his/her labels to the average label, over all other labelers $l' \neq l$, given to each image. We then average the individual accuracy scores over all labelers and report this as the inter-human reliability. Note that this “leave-one-labeler-out” agreement is typically higher than the average pair-wise agreement.
3.10 Acknowledgments

Support for this work was provided by NSF grants IIS 0968573 SOCS, SBE-0542013, and CNS-0454233, and by the Temporal Dynamics of Learning Center (TDLC) at UCSD.

Chapter 3, in full, is currently being prepared for submission for publication of the material. Jacob Whitehill, Zewelanji Serpell, Yi-Ching Lin, Aysha Foster, and Javier Movellan. The dissertation author was the primary investigator and author of this material.
Chapter 4

Measuring the Perceived Difficulty of a Lecture Using Automatic Facial Expression Recognition

Abstract: In this chapter we show how automatic real-time facial expression recognition can be used to estimate the difficulty level, as perceived by an individual student, of a delivered lecture. We also show that facial expression is predictive of an individual student’s preferred rate of curriculum presentation at each moment in time. On a video lecture viewing task, training on less than two minutes of recorded facial expression data and testing on a separate validation set, our system predicted the subjects’ self-reported difficulty scores with mean accuracy of 0.42 (Pearson $r$) and their preferred viewing speeds with mean accuracy of 0.29. Our techniques are fully automatic and have potential applications for both intelligent tutoring systems (ITS) and standard classroom environments.
4.1 Introduction

One of the fundamental challenges faced by teachers – whether human or robot – is determining how well his/her students are receiving a lecture at any given moment. Each individual student may be content, confused, bored, or excited by the lesson at a particular point in time, and one student’s perception of the lecture may not necessarily be shared by his/her peers. While explicit feedback signals to the teacher such as a question or a request to repeat a sentence are useful, they are limited in their effectiveness for several reasons: If a student is confused, he may feel embarrassment in asking a question. If the student is bored, it may be inappropriate to ask the teacher to speed up the rate of presentation. Some research has also shown that students are not always aware of when they need help [2]. Finally, even when students do ask questions, this feedback may, in a sense, come too late – the student may already have missed an important point, and the teacher must spend lesson time to clear up the misunderstanding.

If, instead, the student could provide feedback at an earlier time, perhaps even subconsciously, then moments of frustration, confusion, and even boredom could potentially be avoided. Such feedback is particularly useful for automated tutoring systems. For example, an interactive tutoring system could dynamically adjust the speed of the instruction to increase when the student’s understanding is solid and to slow down during an unfamiliar topic.

In this chapter we explore one such kind of feedback signal based on automatic recognition of a student’s facial expression. Recent advances in the fields of pattern recognition, computer vision, and machine learning have made automatic facial expression recognition in real-time a viable resource for intelligent tutoring systems (ITS). The field of ITS has already begun to make use of this technology, especially for the task of predicting the student’s affective state (e.g., [47, 72, 27, 76]). This chapter investigates the potential usefulness of automatic expression recognition for two different tasks: (1) measuring the difficulty as perceived by students of a delivered lecture, and (2) determining the preferred speed at which lesson material should be presented. To this end, we conducted a pilot experiment in which subjects viewed a video lecture at an adjustable speed while
their facial expressions were recognized automatically and recorded. Using the “difficulty” scores that the subjects report, the correlations between facial expression and difficulty, and between facial expression and preferred viewing speed, can be assessed.

The rest of this chapter is organized as follows: In Section 4.2, we briefly describe the automatic expression recognition system that we employ in our study. Section 4.3 describes the experiment we perform on human subjects, and Section 4.4 presents the results. We end with some concluding remarks about facial expression recognition for ITS.

4.2 Facial Expression Recognition

Facial expression is one of the most powerful and immediate means for humans to communicate their emotions, cognitive states, intentions, and opinions to each other [31]. In recent years, researchers have made considerable progress in developing automatic expressions classifiers [87, 11, 69]. Some expression recognition systems classify the face into the set of “prototypical” emotions such as happy, sad, angry, etc. [55]. Others attempt to recognize the individual muscle movements that the face can produce [10] in order to provide an objective description of the face. The best known psychological framework for describing nearly the entirety of facial movements is the Facial Action Coding System (FACS) [32].

4.2.1 FACS

FACS was developed by Ekman and Friesen as a method to code facial expressions comprehensively and objectively [32]. Trained FACS coders decompose facial expressions in terms of the apparent intensity of 46 component movements, which roughly correspond to individual facial muscles. These elementary movements are called action units (AU) and can be regarded as the “phonemes” of facial expressions. Figure 4.1 illustrates the FACS coding of a facial expression. The numbers identify the action unit, which approximately corresponds to one facial muscle; the letter (A-E) identifies the level of activation.
Figure 4.1: Example of comprehensive FACS coding of a facial expression. The numbers identify the action unit, which approximately corresponds to one facial muscle; the letter (A-E) identifies the level of activation.

4.2.2 Automatic Facial Expression Recognition

We use the automatic facial expression recognition system presented in [10] for our experiments. This machine learning-based system analyzes each video frame independently. It first finds the face, including the location of the eyes, mouth, and nose for registration, and then employs support vector machines and Gabor energy filters for expression recognition. The version of the system employed here recognizes the following AUs: 1 (inner brow raiser), 2 (outer brow raiser), 4 (brow lowerer), 5 (upper eye lid raiser), 9 (nose wrinkleer), 10 (upper lip raiser), 12 (lip corner puller), 14 (dimpler), 15 (lip corner depressor), 17 (chin raiser), 20 (lip stretcher), and 45 (blink), as well as a detector of social smiles.

4.3 Experiment

The goal of our experiment was to assess whether there exist significant correlations between certain AUs and the perceived difficulty as well as the preferred viewing speed of a video lecture. To this end, we composed a short composite “lecture” video consisting of seven individual movie clips about a disparate range of topics. The individual clips were excerpts taken from public-domain videos from the Internet. In order, they were:
Figure 4.2: Representative video frames from each of the 7 video clips contained in our “lecture” movie.

1. An introductory university physics lecture (46 sec).
2. A university lecture on Sigmund Freud (36 sec).
3. A soundless tutorial on Vedic mathematics (46 sec).
4. A university lecture on philosophy (20 sec).
5. A barely audible sound clip (with a static picture backdrop) of Sigmund Freud (16 sec).
6. A teenage girl speaking quickly while telling a humorous story (21 sec).
7. Another excerpt on physics taken from the same source as the first clip (15 sec).

Representative video frames of all 7 video clips are shown in Figure 4.2.

4.3.1 Procedure

Each subject performed the following tasks in order:

1. Watch the video lecture. The playback speed could be adjusted continuously by the subject. Facial expression data were recorded.

2. Take the quiz. The quiz consisted of 6 questions about specific details of the lecture.

3. Self-report on the difficulty. The video lecture was re-played at a fixed speed of 1.0.
For watching the lecture at an adjustable speed we created a special viewing program in which the user can press Up to increase the speed, Down to decrease the speed, and Left to rewind by two seconds. Rewinding the video also set the speed back to the default rate (1.0). The video player was equipped with an automatic pitch equalizer so that, even at high speeds, the lecture audio was reasonably intelligible. Subjects practiced using the speed controls on a separate demo video prior to beginning the actual study. In order to encourage subjects to use their time efficiently and thus to avail themselves of the speed control, we informed them prior to the first viewing that they would take a quiz afterwards, and that their performance on the quiz would be penalized by the amount of time they needed to watch the video. We also started a visible, automatic “shut-off” timer when they started watching the lecture to give the impression of additional time pressure. In actuality, the timer provided enough time to watch the whole lecture at normal speed, and the quiz was never graded – these props were meant only to encourage the subjects to modulate the viewing speed efficiently.

While watching the video lecture for the first time, the subject’s facial expression data were recorded automatically through a standard Web camera using the automatic face and expression recognition system described in [10]. The experiment was performed in an ordinary office environment inside our laboratory without any special lighting conditions. After watching the video and taking the quiz, subjects were then informed that they would watch the lecture for a second time. During the second viewing, subjects could not change the speed (it was fixed at 1.0), but they instead rated frame-by-frame how difficult they found the movie to be on an integral scale of 0 to 10 using the keyboard (A for “harder”, Z for “easier”). This form of continuous audience response labeling was originally developed for consumer research [34]. Subjects were told to consider both acoustic as well as conceptual difficulty when assessing the difficulty of the lecture material. Facial expression information was not collected during the second viewing.

In our experimental design, the fact that subjects adjusted the viewing speed of the lecture video while viewing it may have affected their perception of how difficult the lecture was to understand. Our reason for designing the experiment
in this way was to capture both speed control and difficulty information from all subjects. However, we believe that the ability to adjust the speed of the lecture would, if anything, cause the self-reported Difficulty values to be more “flat,” thus increasing the challenge of the prediction task (predict Difficulty from Expression).

4.3.2 Human Subjects

Eight subjects (five female, three male) participated in our pilot experiment. Subjects ranged in age from early 20’s to mid 30’s and were either undergraduate students, graduate students, or administrative or technical staff at our university. Five were native English speakers (American), and three were non-native (one was Northern European, one was Southern European, and one was East Asian). Each subject was paid $15 for his/her participation, which required about 20 minutes in total.

None of the subjects was aware of the purpose of the study or that facial expression data would be captured. Prior to starting the experiment, subjects were informed only that they would be watching a video at a controllable speed and that they would be quizzed afterward. They were not informed of rating the difficulty of the experiment or of watching the video at second time until after the quiz. Subjects were not requested to restrict head movement in any way (though all remained seated throughout the entire video lecture), and the resulting variability in head pose, while presenting no fundamental difficulty for our expression recognition system, may have added some amount of noise. Due to the need to manually adjust the viewing angle of the camera for facial expression recording, it is possible that subjects inferred that their facial behavior would be analyzed.

4.3.3 Data Collection and Processing

While the subjects watched the video, their faces were analyzed in real-time using the expression recognition system presented in [10]. The output of 12 action unit detectors (AUs 1, 2, 4, 5, 9, 10, 12, 14, 15, 17, 20, 45) as well as the smile
Table 4.1: List of FACS Action Units (AUs) employed in this study.

<table>
<thead>
<tr>
<th>Description of Facial Action Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>AU #</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>45</td>
</tr>
<tr>
<td>Smile</td>
</tr>
</tbody>
</table>

detector were time-stamped and saved to disk. The muscle movements to which the above-listed AUs correspond are shown in Table 4.1. Speed adjustment events (Up, Down, and Rewind) were used to compute an overall Speed data series. A Difficulty data series was likewise computed using the difficulty adjustment keyboard events (A and Z). Since all Expression, Difficulty, and Speed events were timestamped, and since the video player itself timestamped the display time of each video frame, we were able to time-align pairwise the Expression and Difficulty, and Expression and Speed time series, and then analyze them for correlations.

4.4 Results

We performed correlation analyses between individual AUs and both the Difficulty and Speed time series. We also performed multiple regression over all AUs to predict both the Difficulty and Speed time series. Local quadratic regression was employed to smooth the AU values. The smoothing width for each subject was taken as the average length of time for which the user left the Difficulty value unchanged during the second viewing of the video. The exact number of data
Table 4.2: *Middle column:* The three significant correlations with the highest magnitude between difficulty and AU value for each subject. *Right column:* the overall correlation between predicted and self-reported Difficulty value, when using linear regression over the whole set of AUs for prediction.

<table>
<thead>
<tr>
<th>Subj</th>
<th>3 AUs Most Correlated with Self-Reported Difficulty</th>
<th>Overall Corr. (r)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4 (+.42), 9 (-.40), 2 (-.35)</td>
<td>0.84</td>
</tr>
<tr>
<td>2</td>
<td>5 (-.34), 15 (-.30), 17 (-.25)</td>
<td>0.73</td>
</tr>
<tr>
<td>3</td>
<td>20 (+.66), 5 (+.45), 45 (-.42)</td>
<td>0.76</td>
</tr>
<tr>
<td>4</td>
<td>20 (-.51), 5 (-.47), 9 (-.47)</td>
<td>0.85</td>
</tr>
<tr>
<td>5</td>
<td>10 (-.31), 12 (-.28), 2 (-.25)</td>
<td>0.60</td>
</tr>
<tr>
<td>6</td>
<td>5 (-.65), 4 (-.55), 15 (-.49)</td>
<td>0.88</td>
</tr>
<tr>
<td>7</td>
<td>17 (-.53), 1 (-.47), 14 (-.43)</td>
<td>0.74</td>
</tr>
<tr>
<td>8</td>
<td>17 (-.22), 5 (+.19), 45 (+.18)</td>
<td>0.56</td>
</tr>
<tr>
<td>Avg</td>
<td></td>
<td><strong>0.75</strong></td>
</tr>
</tbody>
</table>

points in the Expression data series varied between subjects since they required different amounts of time to watch the video, but for all subjects at least 790 data points (approximately 4 per second) were available for calculating correlations.

For each subject there were a number of AUs that were significantly correlated (we required $p < 0.05$) with perceived difficulty, and also a number of AUs correlated with viewing speed. We report the 3 AUs with the highest correlation magnitude for each prediction task (Difficulty, Viewing Speed). Results are shown in Tables 4.2 and 4.3.

These results indicate substantial inter-subject variability on which AUs correlated with perceived difficulty, and on which AUs correlated with viewing speed. The only AU which showed both a significant and consistent correlation (though not necessarily in the top 3) with difficulty was AU 45 (blink) – for 6 out of 8 subjects their difficulty labels were negatively correlated with blink, meaning these subjects blinked less during the more difficult sections of video. This finding is consistent with evidence from experimental psychology that blink rate decreases when interest or mental load is high [43, 85]. To our surprise, AU 4 (brow lowerer), which is associated with concentration and consternation, was not consistently
Table 4.3: The three significant correlations with highest magnitude between preferred viewing speed and AU value for each subject.

<table>
<thead>
<tr>
<th>Subj.</th>
<th>3 AUs Most Correlated with Viewing Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9 (+.29), 45 (+.26), 4 (-.24)</td>
</tr>
<tr>
<td>2</td>
<td>17 (+.21), 2 (-.16), Smile (+.16)</td>
</tr>
<tr>
<td>3</td>
<td>14 (-.46), 2 (-.44), 1 (-.42)</td>
</tr>
<tr>
<td>4</td>
<td>20 (+.42), 2 (-.37), 17 (-.36)</td>
</tr>
<tr>
<td>5</td>
<td>1 (-.21), 20 (-.20), 15 (-.19)</td>
</tr>
<tr>
<td>6</td>
<td>9 (-.48), 4 (+.40), 15 (+.39)</td>
</tr>
<tr>
<td>7</td>
<td>17 (+.35), 14 (+.34), Smile (+.32)</td>
</tr>
<tr>
<td>8</td>
<td>15 (-.53), 17 (-.47), 12 (-.46)</td>
</tr>
</tbody>
</table>

positively correlated with difficulty.

### 4.4.1 Predicting Difficulty from Expression Data

To assess how much overall signal is available in the AU outputs for predicting self-reported difficulty values, we performed linear regression over all AUs and targeted Difficulty labels as the dependent variable. The correlations between the predicted difficulty values and the self-reported values are shown in right column of Table 4.2. A graphical representation of the predicted difficulty for Subject 6 is shown in Figure 4.3. The average correlation between predicted difficulty values and self-reported values of 0.75 suggests that AU outputs are a valuable signal for predicting a student’s perception of difficulty. In Section 4.4.2, we extend this analysis to the case where a Difficulty model is learned from a training set separate from the validation data.

### 4.4.2 Learning to Predict

Given the high inter-subject variability in which AUs correlated with difficulty and with viewing speed, it seems likely that subject-specific models will need to be trained in order for facial expression recognition to be useful for predicting difficulty and viewing speed. We thus trained a linear regression model to predict
both Difficulty and Viewing Speed scores for each subject. In our model we re-
gressed over both the AU outputs themselves and their temporal first derivatives.
The derivatives might be useful since it is conceivable that sudden changes in ex-
pression could be predictive of changes in difficulty and viewing speed. We also
performed a variable amount of smoothing, and we introduced a variable amount
of time lag into the entire set of captured AU values to account for a possible delay
between watching the video and reacting to it with facial expression. The smooth-
ing and lag parameters were optimized using the training data, as explained later
in this section.

For assessing the model’s ability to learn, we divided the time-aligned AU
and Difficulty data into disjoint training and validation sets: Each subject’s data
were divided into 16 alternating bands of approximately 15 seconds each. The first
band was used for training, the second for validation, the third for training, and
so on.

Given the set of training data (AUs, their derivatives, and Difficulty values
over all training bands), linear regression was performed to predict the Difficulty
values in the training set. A grid search over the lag and smoothing parameters
was performed to minimize the training error. Given the trained regression model
and optimized parameters, the validation performance on the validation bands was
then computed. This procedure was conducted separately for each subject.
Results are shown in Table 4.4. For all subjects except Subject 2, the model was able to predict both the validation Difficulty and Viewing Speed scores with a correlation significantly ($p < 0.05$) above 0. Upon inspecting the AU available for Subject 2, we noticed that the face detection component of the expression recognition system could not find the face for a large stretches of time (the subject may have moved his head slightly out of the camera’s view); this effectively decreases the amount of expression data for training and makes the learning task more difficult.

The average validation correlation across all subjects between the model’s difficulty output and the self-reported difficulty scores was 0.42. This result is significantly above 0 (Wilcoxon sign rank test, $p < 0.05$), which would be the expected correlation if the expression data contained no useful signal for difficulty prediction. The average validation correlation for predicting preferred viewing speed was 0.29, which was likewise significantly above 0 (Wilcoxon sign rank test, $p < 0.05$), regardless of whether Subject 2 was included or not. While these results show room for improvement, they are nonetheless an encouraging indicator of the utility of facial expression for difficulty prediction, preferred speed estimation, and other important tasks in the ITS domain.

4.5 Conclusions

Our empirical results indicate that facial expression is a valuable input signal for two concrete tasks important to intelligent tutoring systems: estimating how difficult the student finds a lesson to be, and estimating how fast or slow the student would prefer to watch a lecture. Currently available automatic expression recognition systems can already be used to improve the quality of interactive tutoring programs. One particular application we are currently developing is a “smart video player” which modulates the video speed in real-time based on the user’s facial expression so that the rate of lesson presentation is optimal for the current user.
Table 4.4: Accuracy (Pearson $r$) of predicting the perceived Difficulty, as well as the preferred viewing Speed, of a lecture video from automatic facial expression recognition channels. All results were computed on a validation set not used for training.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Difficulty</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.41</td>
<td>0.23</td>
</tr>
<tr>
<td>2</td>
<td>0.28</td>
<td>0.04</td>
</tr>
<tr>
<td>3</td>
<td>0.44</td>
<td>0.32</td>
</tr>
<tr>
<td>4</td>
<td>0.85</td>
<td>0.11</td>
</tr>
<tr>
<td>5</td>
<td>0.27</td>
<td>0.44</td>
</tr>
<tr>
<td>6</td>
<td>0.56</td>
<td>0.28</td>
</tr>
<tr>
<td>7</td>
<td>0.32</td>
<td>0.19</td>
</tr>
<tr>
<td>8</td>
<td>0.24</td>
<td>0.68</td>
</tr>
</tbody>
</table>

| Avg     | 0.42       | 0.29  |

4.6 Acknowledgement

Support for this work was provided in part by NSF grants SBE-0542013 and CNS-0454233. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 4, in full, is a reprint of the material as it appears in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2008. Jacob Whitehill, Marian Bartlett, and Javier Movellan. The dissertation author was the primary investigator and author of this paper.
Chapter 5

Teaching Word Meanings from Visual Examples

Abstract: In recent years the intelligent tutoring systems (ITS) community has gained renewed interest in the use of stochastic optimal control theory to develop control policies for automated teaching. The Partially Observable Markov Decision Process (POMDP), in particular, is a useful framework for integrating noisy sensor observations from the student, including keyboard presses, touch events, and emotion data captured through a webcam, into the decision-making process in order to minimize some cost. However, given the well-known intractability issues of computing optimal POMDP policies exactly, more research is needed on how to find approximately optimal policies that work well in practice in ITS. In this paper we present a prototype ITS that teaches students foreign language vocabulary by image association, in the manner of Rosetta Stone [73] and Duolingo [30]. The system’s controller was developed by modeling the student as a Bayesian learner and then employing a policy gradient approach to optimize the teacher’s control policy. In contrast to previously used forward-search methods, this approach shifts the computational burden of planning offline, thus allowing for deeper search and possibly better policies. In an experiment on 90 human subjects in which the independent variable was time-to-mastery, the system’s optimized control policy compares favorably both with a random baseline controller and a hand-crafted control policy. In addition, we propose and demonstrate in
simulation a simple architecture for how affective sensor inputs can be integrated into the decision-making process in order to increase learning efficiency.

5.1 Introduction

One of the chief challenges facing the intelligent tutoring systems (ITS) community today is how to build automated teaching systems that are responsive to the student’s emotional state. One part of this challenge is to recognize important emotional states in the student automatically given data sources such as webcam images or physiological sensors, and the other is to somehow use these affective state estimates in an intelligent way so as to boost student learning. For the latter challenge, the field of stochastic optimal control may hold promise. The Partially Observable Markov Decision Process (POMDP), in particular, is a probabilistic framework for optimal decision-making under uncertainty of the student’s state. POMDPs offer the possibility of integrating affective state inputs into the decision-making process in a principled manner so as to minimize the long-term cost of teaching. To “solve” a POMDP is to find an optimal control policy, i.e., a decision function that maps the teacher’s belief about the student’s state into the next action the teacher should execute in order to minimize the expected cost of learning. Computing optimal POMDP control policies exactly is well known to be computationally intractable in most cases. The key to progress thus consists in developing approaches to finding approximately optimal policies that work well in practice.

In general, there are two classes of POMDP solution methods: model-based approaches and model-free approaches. Model-based approaches assume that a probabilistic model of how the student learns in response to actions taken by the teacher is known. Given a sufficiently accurate model, policies can be simulated, evaluated, and optimized offline, and once they perform well enough in simulation, they can be applied to real students. In contrast, model-free approaches are based on the premise that constructing a detailed model of the student may be an even harder problem than finding a good control policy itself. Model-free approaches
can attempt to identify good policy parameters by running experiments online with real human students, evaluating how well the students learn under a given policy, and then tuning the policy’s parameters to hopefully increase performance.

In this paper, we propose a novel method of constructing an automated teaching system using model-based control. In some model-based approaches to automated control, the model (e.g., a Hidden Markov Model) is learnt entirely from data. Here, we take a different approach and posit that the student can be modeled as a Bayesian learner who updates her beliefs about the curriculum she is learning according to probability theory. We then “soften” this assumption by introducing several parameters which we learn from data collected of real students. Our target teaching application is foreign word learning by image association, in the manner of Rosetta Stone [73] language software and Web-based Duo Lingo [30]. In this language learning method, students learn words’ meanings not by translating them into their native language, but instead by viewing examples that represent the foreign words. An example application of this approach is in early childhood education, in which the learners may not even have a fully developed native language. Figure 5.1 shows a robot, RUBI, developed at our laboratory [64], serving in the University of California, San Diego (UCSD) Early Childhood Education Center (ECEC), along with children playing educational games with RUBI. In this study, we aim both to develop algorithms that could improve the teaching effectiveness of RUBI’s word learning games, and more broadly to contribute to the science of how to create automated teachers using control-theoretic methods.

5.1.1 Contributions

The main contributions of this paper are the following:

- We formalize the problem of teaching words by image association by modeling the student as a Bayesian learner and by modeling the teaching task as a POMDP.

- We propose a method of solving this POMDP using policy gradient descent on features of the particle distribution used to represent the teacher’s belief
of the student’s state. We then validate this approach in an experiment on 90 human subjects and show that the optimized controller performs favorably compared to two baseline heuristics. The proposed policy computation approach may also find use in other teaching problems in which a reasonable model of the student is known a priori.

- We propose, and illustrate using simulation, a simple architecture for how the prototype language teacher developed in this paper could be extended to use affective observations of the student to teach more effectively.

The rest of this paper proceeds as follows: in Section 5.2 we describe related work, including other methods to developing automated teachers based on optimal control and other probabilistic models of word learning. In Section 5.3 we describe the learning setting of our target application, namely word learning by image association. Section 5.4 gives a brief review of POMDPs and describes in broad terms how they will be applied to our learning setting. After these preliminaries, we then delve into the core contributions of the paper, namely the model of the student as a Bayesian learner (Section 5.5), the teacher model (Section 5.6), and the method of computing a control policy (Section 5.7). We validate our proposed method of finding a policy in an experiment in Section 5.9. Lastly, we sketch
a simple architecture for how to use POMDPs to create an affect-sensitive word teacher in Section 5.10 and then conclude in Section 5.11.

5.1.2 Notation

We use upper-case letters to represent random variables and lower-case letters to represent particular values. We define $y_{1:t} = y_1, \ldots, y_t$. We consistently use indices $i, j, k$, and $t$ to index concepts, words, images, and time, respectively. We use $p$ to index individual particles in the particle filter. Finally, we refer to the student using the pronoun “she”, but the model should also generalize to male students without modification.

5.2 Related work

Automated teaching machines have been pursued for over half a century. Some of the earliest such systems were developed at Stanford University and focused on “flashcard”-style learning of foreign vocabulary words [7, 61] where the student was shown a foreign word in association with its meaning in her native language. The student in such systems was typically modeled as a 2-state “all-or-none” learner [14], and the role of the teacher was to execute actions so that the student exited the tutoring session in the “learned” state for each word with high probability. Interestingly, automated teaching in this form was one of the first control problems to be formalized as a Partially Observable Markov Decision Process (POMDP) [80]. However, the early research on optimal teaching focused on either deriving analytical solutions, which is possible only in a very few cases, or on computing exact solutions numerically using dynamic programming, which becomes computationally intractable except for very small teaching problems. By the mid 1970s, the optimal control approach to teaching mostly died off.

Around 1980, John Anderson and colleagues at Carnegie-Mellon University pioneered the “cognitive tutor” movement, based loosely on the ACT-R theory of cognition [3, 4]. The aim of cognitive tutors was to teach more complex skills than basic vocabulary, including algebra, geometry, and computer programming,
The focus was less on developing better controllers and more on how to decompose such high-level skills into lower-level “items” that can be learnt independently. In a geometry tutor, for example, one such item might be the ability to apply the Side-Angle-Side theorem for triangle congruence. After breaking down the learning problem into independent items, the same all-or-none learning model used for vocabulary teaching could then also be used to teach complex skills [21]. The cognitive tutor movement gave rise to some of the most successful automated teaching systems to date, including the Algebra Tutor [49], Geometry Tutor, and LISP Tutor [5].

Since the early 2000s, the ITS community has experienced growing interest in developing affect-sensitive tutoring systems that are responsive to the student’s emotional state [26, 99], as well as renewed interest in using principled probabilistic approaches to developing controllers [8, 35, 66, 19]. These developments are likely due to the significant progress made contemporaneously in the fields of machine learning and reinforcement learning, respectively. New reinforcement learning algorithms, including approximate methods for solving POMDPs such as belief compression [74], point-based value iteration [83], and policy gradient approaches such as [98], may hold promise both for developing automated teaching controllers in general and for creating affect-sensitive tutoring systems in particular. To date, however, no one has developed an automated affect-sensitive teacher that incorporates affective sensor inputs into the decision-making process using a principled approach – the few existing affect-aware systems use heuristic rule-based systems to alter the teacher’s behavior upon recognizing certain emotional states in the student.

Algorithmically, the most similar prior work to the language teaching system developed in this paper is by [71], who developed a prototype automated teacher based on POMDPs to teach simple math concepts (e.g., to associate variable $x$ with value 7) to a student who is modeled as Bayesian learners. Besides the target tutoring task (language versus math concepts), the main differences between our work and theirs is the policy gradient approach we use to compute the control policy instead of online forward-search, as well as the probabilistic per-
ception model we use to describe which images belong to which concepts (e.g., image \(y\) represents “mouse” with probability 0.3 and “cheese” with probability 0.2) instead of “binary” concept membership (e.g., \(y\) either does or does not represent “mouse”). Our work also draws inspiration from the work of [24], who used Markov Decision Processes to construct an automated teacher of the grammar of an artificial language.

The work in this paper is also related to the field of concept learning. Concept learning is concerned with modeling how a student infers from examples the identity of a hidden concept, which is typically defined as a subset of some collection of examples (images, words, etc.), or a subregion of a hypercube. In our case, each foreign word (e.g., Katze) corresponds to a single named concept (e.g., “cat”). The student has a belief for each image \(k\) about whether \(k\) belongs to that concept, as expressed by the distribution \(P(c \mid k)\). The teacher’s job is to convey to the student the particular concept \(c\) that is associated with the target word. The Bayesian model of the student in our model was inspired by the works of [86, 101, 68], and [67], all of whom modeled concept learning as a Bayesian inference problem. The Bayesian “size principle” [86], in particular, can account for why humans prefer more specific concepts to more general ones even when both concepts can generate the same examples; this was applied to how humans identify rectangular subregions [86], sets of numbers [68], and the meanings of words within a taxonomy [101]. Finally, in addition to concept learning, some research has also tackled the problem of concept teaching. [78], in particular, proposed a “pedagogical sampling” model to explain both how students learn the locations of rectangular subregion concepts from examples, and how teachers select examples to teach the concept optimally.

5.3 Learning setting

In this paper we investigate how to construct an automated teacher to teach students foreign language vocabulary by image association. Learning a foreign language is a daunting task for many people, and software programs such as Rosetta
Stone [73] and Web-based Duolingo [30] have emerged that attempt to make learning more engaging and natural by “immersing” the student in an environment in which she infers the meanings of foreign words directly from images and videos instead of “translating” the foreign words into her native language. For instance, in the German version of Rosetta Stone, the student may be shown an image of a girl drinking a cup of milk, followed by an image showing a man drinking tea. In conjunction with these media, the word *trinken* would be displayed. From these image+word pairs, the student can infer that *trinken* means “drink”. [91] found that 55 hours of this image association approach to language instruction produced language learning gains that were equivalent to 1 semester of university instruction in Spanish.

The specific learning setting we consider is the following: a student is trying to learn a vocabulary of $n$ words (e.g., *trinken*), each of which can mean any one of $m$ possible concepts (e.g., “drink”); we refer to the words by their indices $\{1, \ldots, n\}$ and the concepts by their indices $\{1, \ldots, m\}$. The ground truth definition of each word $j$ is known by the teacher and denoted by variable $W_j \in \{1, \ldots, m\}$; there is no restriction against synonyms. Each of the concepts could be a noun (e.g., man, carrot, lust, irony), adjective (e.g., red, huge, coercive), or verb (e.g., run, eat, defenestrate) – anything that a student might perceive to be represented in an image.

The student learns the mapping from words into concepts by *image association* – if the teacher shows the student a word $j$ along with an example image that the student perceives to represent a concept $i$, then the student will associate concept $i$ with word $j$. There are $l$ different images $\{1, \ldots, l\}$ that the teacher might show the student. Each image can represent one of several possible concepts. For example, the left image in Figure 5.3 could represent “man”, “salad”, “fruit”, “eat”, and possibly even “pink shirt”, but it probably does not represent “office” or “wombat”. In addition, an image can represent concepts with different probabilities – “eat” might be represented with probability 0.5, whereas “pink shirt” might be represented with probability 0.05. The precise semantics of these probabilities will be defined in Section 5.2. The student associates word $j$ with
concept $i$ according to the “strength”, as expressed by this probability, with which the image represents each concept $i$.

The task of the teacher in our setting is to help the student learn the mapping from words to concepts as quickly as possible. At each timestep, the teacher can execute one of three different kinds of actions:

- **Teach word** $j$: the teacher shows the student an image $k$ and indicates whether or not $k$ represents word $j$ (i.e., whether $k$ represents the concept to which word $j$ corresponds); this causes the student to revise her belief about the meaning of word $j$.

- **Ask word** $j$: the teacher presents the student with two different images $k_1$ and $k_2$ and asks the student, “Which of the two images is more likely to represent word $j$?” The student then responds with either a 0 or 1 corresponding to her answer. This kind of question is a 2-alternative forced choice and, in our model of the student, does not impart any information about the true meaning of word $j$. It does, however, help to reduce the teacher’s uncertainty about the student’s beliefs.

- **Test**: the teacher gives the student a set of questions, each of which asks the student to select (from a list) the true meaning for a particular word $j$. If the student passes the test, then the learning session is over; otherwise, the learning session proceeds with the teacher’s new knowledge of the student’s beliefs. The “test” action both reduces the teacher’s uncertainty, and also helps the student to graduate from the session; however, it is typically relatively expensive (in terms of time) compared to the “teach” and “ask” actions.

The teacher must decide which action to execute at each timestep $t$ based on a control policy. This policy is computed using a model of the teaching session, in particular, a Partially Observable Markov Decision Process. We describe POMDPs briefly in the next section.
5.4 POMDPs

We pose the teaching-by-image-association task as a discrete-time, continuous-state, discrete-action Partially Observable Markov Decision Process (POMDP), which is a probabilistic framework for minimizing an agent’s long-term accumulated cost of interacting with an environment whose state is only partially known to the agent. Here, the “agent” is the teacher, and the “environment” is the student. Formally, a POMDP consists of a state space $S$, an action space $U$, an observation space $O$, a state transition dynamics model $P(s_{t+1} \mid s_t, u_t)$, an observation likelihood model $P(o_t \mid s_t, u_t)$, a prior belief $P(s_1)$ of the state, a time horizon $\tau$, a discount factor $\gamma$, and a cost function $C$. The premise is that, a time $t$, the teacher chooses an action $U_t$ based on its control policy. Executing this action then causes student to transition from state $S_t = s$ to $S_{t+1} = s'$, and also to output a response (“observation”, from the teacher’s perspective) $O_t = o$, e.g., the answer to a question posed by the teacher. The teacher uses both $u$ and $o$ to update its belief about the student’s state, and it then chooses the next action $U_{t+1}$, etc. This process repeats until $t = \tau$. Associated with each student state $s_t$ and each action $u_t$ is a cost, defined by cost function $c(s, u)$. An optimal policy $\pi^*$ is one that minimizes the expected sum of costs from $t = 1$ to $t = \tau$:

$$\pi^* = \arg \min_\pi \mathbb{E} \left[ \sum_{t=1}^{\tau} \gamma^t c(s_t, u_t) \mid \pi, P(s_1) \right]$$

The discount factor $\gamma$ specifies how much costs in the long-term future are weighted compared to costs in the near future.

In our setting, the state space will consist of all possible beliefs that the student might have about the words’ meanings, along with values for certain learning parameters of the student (described later). The observation space will consist of all possible answers (either binary numbers, or vectors of binary numbers) that the student might give in response to a question or a test given by the teacher. The action space consists of all possible word+image combinations for “teach” actions, all possible word+image pair combinations for “ask” images, and all possible $d$-element subsets of $\{1, \ldots, n\}$ for “test” actions. For our application, we define the cost function in terms of the expected length of time (in seconds) needed to
execute each action; these costs will be estimated empirically from data collected of human subjects in Section 5.9.

In the sections below we define the POMDP parameters that capture our learning setting. In Section 5.5, we first consider the learning setting from the student’s point of view and define the state and observation spaces as well as the transition dynamics and observation likelihood models. In particular, we model the student as a Bayesian learner who conducts probabilistic inference given the evidence revealed to him to deduce the meaning of each word. Then, in Section 5.6, we consider the teaching problem from the teacher’s perspective and define the action space and describe how the teacher updates its belief about the student’s state.

5.5 Modeling the student as a Bayesian learner

Let us consider the learning problem from the student’s perspective. Our goal in this section will be to develop the transition dynamics and observation likelihood model of the student, both of which will be required when defining the POMDP from the teacher’s perspective in Section 5.6.

The student’s task in our setting is to infer the values of variables \( W_1, \ldots, W_n \).
given the information she receives from the teacher.\(^1\) At each timestep, the teacher may decide either to teach, ask a question, or give a test. First, let us assume that taking a test and answering a question do not change the learner’s beliefs. Then, the only actions that can alter the student’s beliefs are the “teach” actions.

At each timestep \(t\), the teacher shows the student an image \(Y_t \in \{1, \ldots, l\}\) along with a single word specified by \(Q_t \in \{1, \ldots, n\}\). (See the probabilistic graphical model in Figure 5.2.) In addition, the teacher gives the “answer” to whether or not \(Y_t\) represents word \(Q_t\); this answer is represented by variable \(A_{tq_t} \in \{0, 1\}\). Hence, at each timestep \(t\), the learner observes two nodes: \(Y_t\) and \(A_{tq_t}\). For instance, if the teacher shows word 3 to the student at time \(t = 6\) and says that the image does in fact represent word 3, then \(Q_6 = 3\), and the student observes that node \(A_{6,3} = 1\). The other \(n - 1\) “answer” nodes \(A_{t,j \neq q_t}\), which represent whether the other words are represented by image \(Y_t\), are not observed by the learner because the teacher reveals only one “answer” node per timestep. (This assumption simplifies the student’s inference process.) Node \(C_t\), which is also not observed by the learner, represents \textit{which single concept the student believes the teacher to perceive in} \(Y_t\ \textit{at time} \ t\). In other words, the student learns to associate words with meanings based on what she believes the teacher to perceive in the example images – this could theoretically differ from what the student him/herself perceives. The student does not know exactly what the teacher perceives, but she has a belief \(P(c \mid k)\) for each image \(k\) which models the human student’s natural perceptual capability to map from images into concepts.

Under our model of the student, the value of answer node \(A_{tq_t}\) is determined by the concept \(C_t\) and by the meaning of the word \(Q_t\), represented by \(W_{q_t}\). The answer is 1 (“word \(Q_t\) is represented by image \(Y_t\)”’) if and only if the image represents the concept \(W_{q_t}\) that the word means. Specifically, we define

\[
P(A_{tj} = 1 \mid C_t = i, W_j = i) = 1
\]

\[
P(A_{tj} = 1 \mid C_t = i, W_j \neq i) = 0
\]

Based on all the images \(Y_{1:t}\) and answers \(A_{1q_1}, \ldots, A_{tq_t}\) that the student observes,

\(^1\)In contrast to the “pedagogical sampling” model of [78], our model student does not assume that the examples word+image pairs selected by the teacher are necessarily good examples.
she can infer the meanings of the words by updating her prior belief distribution according to Bayesian inference as described in the next subsection.

### 5.5.1 Inference

Let $M_t$ be an $n \times m$ matrix specifying the student’s belief at time $t$ about the words’ meanings, where entry $M_{tji}$ specifies the student’s posterior belief $P(W_j = i \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t})$ that word $j$ means concept $i$ given the images and answers she has observed.

Before deriving the belief update equation for each word $j$, we first note that the joint posterior distribution of the meanings of all words is equal to the product of the marginal posterior distributions due to the theorem in Appendix 5.12.1; in other words, the student can update her belief about the meaning of each word independently:

$$P(w_1, \ldots, w_n \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t}) = \prod_j P(w_j \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t})$$

Now, consider the marginal posterior distribution $P(W_j = i \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t})$, and suppose that at timestep $t$ the teacher teaches a word $q_t \neq j$. Then by Appendix 5.12.2,

$$m_{t+1,ji} \doteq P(W_j = i \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t}) = P(W_j = i \mid y_{1:t}, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})$$

$$= P(W_j = i \mid y_{1:t-1}, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})$$

(cond. indep. from graphical model)

$$= m_{tji}$$

In other words, the posterior distribution of $W_j$ is equal to the prior distribution for every timestep $t$ when the teacher teaches a word $q_t \neq j$. 

On the other hand, if the teacher teaches word \( q_t = j \) at timestep \( t \), then

\[
P(W_j = i \mid y_1:t, a_{1q_1}, \ldots, a_{tq_t})
\]

\[
\propto P(a_{tq_t} \mid W_j = i, y_1:t, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})P(W_j = i \mid y_1:t, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})
\]

\[
= P(a_{tj} \mid W_j = i, y_1:t, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})P(W_j = i \mid y_1:t-1, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})
\]

\[
= P(a_{tj} \mid W_j = i, y_t)P(W_j = i \mid y_1:t-1, a_{1q_1}, \ldots, a_{t-1,q_{t-1}})
\]

(by cond. indep. from graphical model)

\[
= m_{tji}P(a_{tj} \mid W_j = i, y_t)
\]

To compute \( P(a_{tj} \mid W_j = i, y_t) \), we handle the case that \( A_{tj} = 1 \) (i.e., \( Y_t \) represents word \( Q_t \)) and \( A_{tj} = 0 \) (i.e., \( Y_t \) does not represent word \( Q_t \)) separately. For the former case,

\[
P(A_{tj} = 1 \mid W_j = i, y_t)
\]

\[
= \sum_{i' = 1}^m P(A_{tj} = 1 \mid C_t = i', W_j = i, y_t)P(C_t = i' \mid W_j = i, y_t)
\]

\[
= \sum_{i' = 1}^m P(A_{tj} = 1 \mid C_t = i', W_j = i)P(C_t = i' \mid y_t)
\]

\[
= P(C_t = i \mid y_t)
\]

since \( i' = i \) is the only value of \( i' \) that contributes positive probability mass to the sum. The latter case \( (A_{tj} = 0) \) is then simply \( 1 - P(C_t = i \mid y_t) \).

Combining these two cases, we get:

\[
m_{t+1,ji} \propto m_{tji}P(C_t = i \mid y_t)^{a_{tj}}(1 - P(C_t = i \mid y_t))^{(1-a_{tj})} \quad (5.1)
\]

In other words, if the teacher teaches word \( j \) at timestep \( t \), then the student updates her belief about the meaning of that word: if the teacher says image \( y_t \) does represent word \( j \) \( (A_{tj} = 1) \), then the student increases the probability that word \( j \) means any concept \( i \) that is shown in the image with high probability. If, on the other hand, the teacher said \( y_t \) does not represent word \( j \) \( (A_{tj} = 0) \), then the student decreases the probability that word \( j \) means any concept \( i \) that is shown in the image with high probability.
Figure 5.3: Top: two example images that could be used to teach the Hungarian word \textit{férfi}. (Bottom-left): prior belief about the meaning of the word \textit{férfi}. (Bottom-middle): posterior belief about the meaning of word \textit{férfi} after seeing the left image in the figure. (Bottom-right): posterior belief after seeing both images.

For convenience later, let us define the function $f(m_t, \beta_t, y_t, q_t, a_{tq})$ whose output is an $n \times m$ matrix. The $ji$th entry of the output of $f$ is denoted as $f_{ji}$ and defined such that:

$$f_{ji}(m_t, \beta_t, y_t, q_t, a_{tq}) = \begin{cases} m_{t+1,ji} & \text{if } j = q_t \\ m_{tji} & \text{if } j \neq q_t \end{cases}$$

Given this definition of $f$, we can write

$$m_{t+1} = f(m_t, \beta_t, y_t, q_t, a_{tq})$$

**Example**

Suppose the student is learning the meaning of a single (Hungarian) word \textit{férfi}. For this example, suppose the set of concepts that the student considers as possible meanings for \textit{férfi} are \{ man, salad, eat, fruit, smile, coffee, drink, breakfast \}. Hence, $n = 1$ and $m = 8$. In more realistic settings, the concept
set might be much larger; in practice, we might define the concept set by showing
the \( l \) images in the image set to human labelers, asking them what concepts they
perceive in the images, and then taking the union, over all \( l \) images, of the concepts
that the labelers reported. For concreteness, we assume that the student’s prior
belief about the meaning of \( f\acute{e}r\acute{f}i \) is uniform over the concept set, as shown in Figure
5.3 (bottom-left).

Now, suppose that at timestep \( t = 1 \) the teacher shows the student the im-
age \( y \) portrayed in the left image in Figure 5.3 and says that \( f\acute{e}r\acute{f}i \) is represented by
\( y \), i.e., \( A_{1,1} = 1 \). The most prominent concepts represented by the left image (in our
example) are “eat”, followed by “man”, and so on. After analyzing the image for
its concepts, the student will then conduct inference on \( W_1 \) based on Equation 5.1
and obtains the posterior belief about \( W_1 \) shown in Figure 5.3 (bottom-middle).
At this point, the student is certain that \( f\acute{e}r\acute{f}i \) does not mean “coffee” or “drink”
because these concepts were not represented by the image; however, there are still
several candidate meanings, including “man”, “salad”, etc. If the teacher subse-
quently (\( t = 2 \)) shows the right image in Figure 5.3 and says that this image also
represents \( f\acute{e}r\acute{f}i \), then the student will once analyze the image for what concepts
she perceives in it and then update her belief to arrive at the distribution shown in
Figure 5.3 (bottom-right). At this point, the student believes that \( f\acute{e}r\acute{f}i \) probably
means “man”, but with low probability it might mean “breakfast”.

5.5.2 Adding noise

The model of the student described above assumes that humans are “per-
fectly Bayesian” and update their beliefs exactly according to the derived equations
above, which is unrealistic [67]. In order to model students more accurately, it is
important to add realistic sources of noise to the student’s belief updates. We
incorporate the following two kinds of noise into our student model: absorption
and belief update strength.
Absorption

We allow for the possibility that sometimes the student does not “absorb” an example word+image pair. This could be because the student was too tired to process the example and update her belief, or because the student left the room momentarily to go to the restroom and did not see what the teacher showed. If a student does not “absorb” the example at time \( t \), then neither \( Y_t \) nor \( A_{tq} \) is observed – it is as if this timestep did not exist. The student is modeled to absorb a “teach” action at time \( t \) with probability \( \alpha_t \).

Modeling absorption is useful for another reason too: According to the student model in its pure form, the teacher can cause the student to become arbitrarily certain about her belief about a particular word by showing the same image to represent the same word over and over again because the student will update her belief each time she sees the image. In reality, a student is unlikely to change her belief much after seeing the image for the first time. To account for this, we augment the student model with an “absorption matrix”: if the teacher teaches word \( j \) at time \( t \) using image \( k \), and if the student “absorbed” that teaching event, then showing that same image \( k \) in conjunction with that same word \( j \) at some later timestep \( t' > t \) has no effect on the student’s belief. We represent the set of images that the student has absorbed for each word with the absorption matrix \( v_t \), whose \( jk \)th entry \( v_{tjk} \) represents whether the student has absorbed image \( k \) in conjunction with word \( j \). Once an image is absorbed, it is never “forgotten”.

For convenience later, we define the function \( h(v_t, j, k) \) to update the absorption matrix whenever the student absorbs an image \( k \) used to teach word \( j \). Function \( h \) outputs an \( n \times l \) matrix whose \( j', k' \)th entry is 1 if either \( v_{lkj} = 1 \) or \( (j = j' \text{ and } k = k') \); otherwise, the entry is 0. We can then write

\[
v_{t+1} = h(v_{t}, j, k)
\]

if the student absorbed the teaching action at timestep \( t \).
Belief update strength

Assuming the student absorbs a particular image at a particular timestep, the student will update her belief according to Equation 5.1. However, we assume that the student may only “partially” update her belief according to belief update strength parameter $\beta_t$; hence, we revise the belief update equation to be:

$$m_{t+1,ji} \propto m_{tji} \left[ P(C_t = i \mid y)^{a_{ij}} \left( 1 - P(C_t = i \mid y)^{1-a_{ij}} \right) \right]^{\beta_t} \quad (5.2)$$

As $\beta_t \to 0$, the student updates her belief less and less. If $\beta_t = 1$, then the student is “perfectly Bayesian” and updates her belief according to Equation 5.1. Note that this form of noise has the same expressive power as the noise model in [67]; however, the scale of $\beta_t$ is different from the noise parameter $\mu$ in their model.

In our model, both $\alpha_t$ and $\beta_t$ are assumed to be sampled from a finite set of values over the interval $(0, 1]$.

5.5.3 Dynamics model

Given the model of how the student conducts inference derived above, we can now specify the student’s state transition dynamics model. Let us define the student’s state $S_t$ to be the student’s belief $M_t$ along with the absorption probability $\alpha_t$, belief update strength $\beta_t$, and absorption matrix $V_t$. Then $S_t \equiv [M_t, \alpha_t, \beta_t, V_t]$. The transition dynamics model is the probability distribution $P(s_{t+1} \mid s_t, u_t)$ where variable $U_t$ is the teacher’s action at time $t$. We define

$$P(s_{t+1} \mid s_t, u_t) \quad (5.3)$$

$$= P(m_{t+1}, \alpha_{t+1}, \beta_{t+1}, v_{t+1} \mid m_t, \alpha_t, \beta_t, v_t, u_t) \quad (5.4)$$

$$= P(m_{t+1}, v_{t+1} \mid m_t, \alpha_t, \beta_t, v_t, u_t)P(\alpha_{t+1}, \beta_{t+1} \mid m_{t+1}, v_{t+1}, m_t, \alpha_t, \beta_t, v_t, u_t) \quad (5.5)$$

For the second probability in Equation 5.5, we choose a simple form in which $\alpha_{t+1}, \beta_{t+1}$ depend only on $\alpha_t, \beta_t$:

$$P(\alpha_{t+1}, \beta_{t+1} \mid m_{t+1}, v_{t+1}, m_t, \alpha_t, \beta_t, v_t, u_t) = P(\alpha_{t+1}, \beta_{t+1} \mid \alpha_t, \beta_t)$$
In fact, for the word learning experiment we conducted on human subjects (see Section 5.9), we assumed $\alpha_t$ and $\beta_t$ were static parameters, so that:

$$P(\alpha_{t+1}, \beta_{t+1} \mid m_{t+1}, v_{t+1}, m_t, \alpha_t, \beta_t, v_t, u_t) = \delta(\alpha_{t+1}, \alpha_t)\delta(\beta_{t+1}, \beta_t)$$

where $\delta(x, y)$ equals 1 if $x = y$ and 0 otherwise.

For the first probability in Equation 5.5, we assume that “ask” and “test” actions do not affect the student’s belief (or the absorption matrix), i.e.,

$$P(m_{t+1}, v_{t+1} \mid m_t, \alpha_t, \beta_t, v_t, u_t) = \delta(m_{t+1}, m_t)\delta(v_{t+1}, v_t)$$

if $\text{type}(u_t) \in \{\text{“ask”, “test”}\}$

For “teach” actions, $u_t = [q_t, y_t, a_t]$ to encapsulate the image, word and answer that the teacher shows the student. When computing the posterior probability of $M_{t+1}$ we must consider whether the student had already “absorbed” the image, and whether she does so at time $t$:

$$P(m_{t+1}, v_{t+1} \mid m_t, \alpha_t, \beta_t, v_t, u_t) \quad (5.6)$$

$$= P(m_{t+1}, v_{t+1} \mid m_t, \alpha_t, \beta_t, v_t, Q_t = j, Y_t = k, a_{tj}) \quad (5.7)$$

$$= \begin{cases} 
1 & \text{if } v_{tjk} = 1 \text{ and } m_{t+1} = m_t \text{ and } v_{t+1} = v_t \\
\alpha & \text{if } v_{tjk} = 0 \text{ and } m_{t+1} = f(m_t, \beta_t, k, j, a_{tj}) \text{ and } v_{t+1} = h(v_t, j, k) \\
1 - \alpha & \text{if } v_{tjk} = 0 \text{ and } m_{t+1} = m_t \text{ and } v_{t+1} = v_t \\
0 & \text{otherwise} 
\end{cases} \quad (5.8)$$

where function $f$ was defined in Section 5.5.1 and $h$ was defined in Section 5.5.2. The first case in Equation 5.8 is when the student had already absorbed image $k$ for word $j$. The second case is when the student had not yet absorbed image $k$ for word $j$, and when she does so at time $t$. The third case is when the student fails to absorb the image+word pair at time $t$. The fourth case simply states that no other state transitions are possible under the model.

### 5.5.4 Observation model

The subsections above described how the student updates her belief when the teacher executes “teach” actions. Sometimes, however, the teacher may ask
the student a question, or may give him/her a test. The probability distribution \( P(o_t \mid s_t, u_t) \) over the different outputs \( o_t \) (“observations”) a student might give in response to “ask” and “teach” constitutes the observation model of the student. For “teach” actions, we assume \( P(o_t \mid s_t, u_t) \) is uninformative, e.g., \( P(O_t = 1 \mid s_t, u_t) = 1 \). For “ask” and “test” actions, we defined the observation likelihoods below.

**Response to “ask” actions**

When the teacher executes an “ask” action at timestep \( t \), it presents the student with two alternative images \( Y_{t_1} = k_1 \) and \( Y_{t_2} = k_2 \) along with a word \( Q_t = j \) and asks the student which of the two images more probably represents word \( j \). The student then gives a binary response \( O_t \) to indicate which image (e.g., “first image” or “second image”) is the correct answer. To simplify notation just for this section, let us define \( A_1 \) (and \( A_2 \)) to represent whether or not image \( Y_{t_1} \) (or \( Y_{t_2} \), respectively) represents word \( j \). In our model, given the student’s belief \( m_t \) at time \( t \), the student responds with \( O_t = 1 \) (“first image”) with probability:

\[
P(O_t = 1 \mid y_{t_1}, y_{t_2}, m_t, Q_t = j) = \frac{P(A_1 = 1 \mid y_{t_1}, m_{tj})}{P(A_1 = 1 \mid y_{t_1}, m_{tj}) + P(A_2 = 1 \mid y_{t_2}, m_{tj})}
\]

(5.9)

In other words, the student responds probabilistically based on the relative likelihoods of the two images representing the query word \( j \). Note that other observation models are also conceivable; for example, the student might respond deterministically based on the greater of the two probabilities.
To evaluate $P(A_1 = 1 \mid y_{t_1}, m_{t_j})$, we note that:

\[
P(A_1 = 1 \mid y_{t_1}, m_{t_j}) = \sum_i m_{tji} P(A_1 = 1 \mid W_j = i, y_{t_1}, m_{t_j}) \]

where we used the fact that $P(A_1 = 1 \mid C_t = i', W_j = i) > 0$ only if $i' = i$.

**Responses to “test” actions**

Sometimes the teacher will give the student a test, consisting of $d$ questions of the form, “Which of the concepts $\{1, \ldots, m\}$ is the true meaning of word $j$?” To simplify notation just for this section, we denote these test words as $q_1, \ldots, q_d$.

When answering such a question, we assume that the student selects concept $i$ for word $j$ with probability $m_{tji}$, where $m_t$ is the student’s belief matrix at the time when the test was given. Hence, the student’s response $O_t$ is a random vector with $d$ elements such that:

\[
P(o_t \mid s_t, q_1, \ldots, q_d) = \prod_{j=1}^{d} P(O_{tj} = i \mid s_t, q_j)
= \prod_{j=1}^{d} m_{tqji}
\]

where $O_{tj}$ is the $j$th component of the student’s response to a test given at time $t$, i.e., the student’s response to the $j$th test equation. Notice that this observation model not only predicts whether the student will answer a test question correctly or incorrectly, but also predicts the specific word the student chooses if she is incorrect.
5.6 Teacher model

Having defined the model of the student, we now consider the word teaching problem from the perspective of the teacher. We model the teaching task from the teacher’s perspective using a Partially Observable Markov Decision Process (POMDP) whose graphical model is shown in Figure 5.4.

The goal of the teacher is to execute “teach”, “ask”, and “test” actions in some sequence so that the student passes the test as quickly as possible. This implicitly requires that the student learn the words, i.e., that the student to update her belief $M_t$ to match the words’ true definitions. The teacher knows the true meanings $W_1, \ldots, W_n$ of all the words $1, \ldots, n$. Although the teacher is assumed to know the model of the student (including absorption and belief update strength), i.e., the process by which the student updates her beliefs when shown a sequence of word+image pairs, the teacher does not know the exact state $S_t$ of the student $S_t$. Instead, the teacher must make teaching decisions based on $B_t$, which is the teacher’s belief (a probability distribution) at time $t$ over the student’s state $S_t$, given the sequence of actions and observations through time $t - 1$. The teacher computes $B_{t+1}$ based on its prior belief $B_t$, the teacher’s action $U_t$, and the student’s
response $O_t$. The teacher is also assumed to know the student’s perceptual belief $P(c \mid k)$ for each image $k$. In other words, the teacher knows the student’s belief about the teacher’s perception of concepts in the images.

$U_t$ is the action executed by the teacher at time $t$. $U_t$ contains multiple components depending on $U_t$’s associated type:

- For “teach” actions, the teacher shows the student a particular “query” word $Q_t \in \{1, \ldots, n\}$ along with a particular image $Y_t$ and an “answer” $A_t$. The “answer” $A_t$, which can be 1 or 0, indicates whether $Y_t$ represents $Q_t$ or does not represent $Q_t$, respectively. In this sense, the teacher can provide either examples or “non-examples” to the student that help to teach the word’s meaning. For “teach” actions, $U_t = [Y_t, Q_t, A_t]$.

- For “ask” actions, the teacher presents the student with two images $Y_{t_1}$ and $Y_{t_2}$ along with a query word $Q_t$ and asks the student which image more probably represents word $Q_t$. The student’s answer is given by $O_t$ (see below). Hence, for “ask” actions, $U_t = [Y_{t_1}, Y_{t_2}, Q_t]$.

- For “test” actions, the teacher gives the student a list of words and for each word a list of possible definitions (concepts). For “test” actions, $U_t = [Q_{t_1}, \ldots, Q_{t_d}]$, where $d$ is the number of test questions.

$O_t$ is the response (“observation”) received from the student in response to the teacher’s action $U_t$. For “teach” actions, $O_t$ is uninformative. For “ask” actions, $O_t$ is either 0 or 1 to specify which image the student selected. For “test” actions, $O_t \in \{1, \ldots, m\}^d$ contains the student’s answers to the test questions.

Finally, node $\pi$ is the teacher’s policy, which together with the teacher’s current belief $B_t$ dictates what the teacher’s next action will be. In our implementation, $\pi$ is a stochastic logistic policy parameterized by a weight vector for each possible action. The policy $\pi$ is implicitly determined by the words that the teacher must teach, along with the particular example images from which the teacher can choose – for example, if the image set contains only images that “weakly” represent the words, then the teacher may have to show the student more of them to convey their meaning. The policy will be selected (in Section 5.7) so as to (locally)
minimize the expected cost of teaching, which we define as \( \sum_{t=1}^{\tau} \gamma^t c(u_t) \) where \( c(u_t) \) is the expected amount of time, in seconds, necessary to execute action \( u_t \). We assume that all actions of the same type (“teach”, “ask”, and “test”) take the same amount of time and estimate these times from data of real students. We set \( \tau = 500 \) and \( \gamma = 1 \); hence, this is an undiscounted, finite-horizon control problem.

5.6.1 Representing and updating \( B_t \)

During the entire teaching session, the teacher must maintain and update a probability distribution \( b_t \) over the student’s state \( S_t \). After executing action \( U_t \) and receiving observation \( O_t \) from the student, the teacher computes its new belief \( B_{t+1} \) about the student’s new state \( S_{t+1} \):

\[
    b_{t+1} = P(s_{t+1} | u_{1:t}, o_{1:t})
    \propto P(s_{t+1}, o_t | u_{1:t}, o_{1:t-1})
    = \sum_{s_t} P(s_{t+1}, o_t | s_t, u_{1:t}, o_{1:t-1}) P(s_t | u_{1:t}, o_{1:t-1})
    = \sum_{s_t} P(s_{t+1} | s_t, u_t) P(o_t | s_t, u_t) P(s_t | u_{1:t-1}, o_{1:t-1})
    = \sum_{s_t} P(s_{t+1} | s_t, u_t) P(o_t | s_t, u_t) b_t
\]

The first term in the summation represents the transition dynamics of the student’s state, and the second term represents the observation likelihood model of the student. These terms were both developed in Section 5.5. The third term in the summation is the teacher’s prior belief \( b_t \).

Unfortunately, the student’s state contains \( M_t \), which itself is a real-valued matrix, and hence representing \( b_t \) in exact form is infeasible. Instead, we use a particle filter with importance resampling [40] to represent this distribution approximately. This approach was previously applied to an automated teaching problem in [71]. Each particle stores a possible student state \( s_t \) comprising the student’s belief \( m_t \), values for \( \alpha_t \) and \( \beta_t \), and an absorption matrix \( v_t \). Associated with the \( p \)th particle is a weight \( w_p \), such that the sum of the weights over all particles is unity. We denote the student’s belief \( m_t \) as represented by the \( p \)th particle as \( m_{tp} \).
α_{pt} and β_{pt} have analogous meanings. At each timestep, when the teacher executes action \( u_t \) and then receives an observation \( o_t \) from the student, each particle is updated according to the dynamics model in Section 5.5.3 and then reweighted according to how well it explains \( o_t \) according to the observation model in Section 5.5.4. The particle weights are then re-normalized to 1.

5.7 Computing a policy

Now that the teaching problem has been defined as a POMDP, we can consider the various methods for solving it, i.e., for computing a policy \( \pi \) that maps the teacher’s belief (more specifically, features of the teacher’s particles) at time \( t \) into its next action. One simple approach to making decisions which works with both finite and infinite (as in our case) state spaces is forward search; this was the approach used to develop a math concept teacher in [71]. At timestep \( t \), the teacher considers every possible action \( u_t \) it could take, and for each \( u_t \), it considers every possible subsequent trajectory of subsequent states, actions, and observations that might ensue for \( h \) timesteps (the planning horizon) into the future. The teacher chooses action \( u_t \) so as to minimize the expected cost, summed over \( h \) timesteps, of executing action \( u_t \) first. While simple to implement, this approach is limited: Since the “tree” of possible trajectories grows exponentially in size with the planning horizon \( h \), and since this forward search must be conducted at runtime when there are hard computational constraints (the teacher should not take too long to decide its next action), \( h \) must typically be kept quite small (in [71], \( h \) was 2). This leads to the concrete problem in our case that the teacher will never decide to execute a “test” action because the immediate cost of testing is typically an order of magnitude higher than the “teach” actions. Given a small planning horizon \( h \), it is always cheaper simply to execute \( h \) “teach” actions in succession rather than execute one “test” action – hence, the student will never be given a chance to graduate from the learning session.

Due to the small planning horizon issue, we instead chose a policy gradient approach to optimizing the control policy, which requires a control policy that
can be expressed in some parametric form. Optimization of the policy parameters can be performed offline when computational constraints are less pressing. In addition, instead of exhaustively searching through a tree exponentially large in the planning horizon, policy gradient approaches sample many linear trajectories of length $\tau$, where $\tau$ is the time horizon of the POMDP itself. Hence, a policy optimized with policy gradient can choose actions whose benefit might not be realized until much further in the future. Nevertheless, even the policy gradient approach we use does not completely solve the computational issues involved in computing a policy. To simplify computation further, we employ a two-tiered approach: In the first tier (“macro-controller”), we use a stochastic logistic policy to decide, based on certain features of the teacher’s particles, whether to test, teach word $j$, or ask a question about word $j$. This macro-controller does not decide which image(s) to show for word $j$. The second tier (“micro-controller”), given the decision to teach/ask about word $j$, decides which particular images would be best based on an information-gain heuristic.

5.7.1 Macro-controller

We use a stochastic logistic policy that maps the teacher’s particles not into a single action, but instead into a probability distribution over actions. We define

$$\pi(x_t) \doteq P(U_t = u \mid x_t) \propto \exp(x_t^\top w_u)$$

where $U_t$ is the action executed at time $t$, $x_t$ is a vector encoding certain features of the teacher’s particles at time $t$, and $w_u$ is a weight vector associated with action $u \in \mathcal{U}$, where $\mathcal{U}$ is the action space of the macro-controller. We defined

$$\mathcal{U} = \{\text{test}, \text{teach}_1, \ldots, \text{teach}_n, \text{ask}_1, \ldots, \text{ask}_n\}$$

For simplicity, the “test” actions always asked the student about all $n$ words; hence, there is only one “test” action in $\mathcal{U}$. Also, we restricted the “teach” actions to only show positive examples for each word, i.e., $A_{tq_t}$ is always 1.

We define $x_t$ to consist of the following features:
• For each word $j$, the expected (w.r.t. the teacher’s particles) “goodness”
\[ \sum_p w_p g(m_{ptj}) \] of the student’s belief about word $j$, where *goodness* is defined as
\[ g(m_{tj}) = m_{tji} \text{ for } i = W_j \]
(5.10)
In other words, the goodness of the student’s belief about the meaning of word $j$ is the probability she assigns to the correct concept.

• The expected (w.r.t. the teacher’s particles) total uncertainty
\[ \sum_p w_p \sum_j u(m_{ptj}) \]
over the student’s belief, summed over all words, where the uncertainty $u$ is defined as
\[ u(m_{ptj}) \doteq (m_{ptj} - \overline{m_{tj}}) \top (m_{ptj} - \overline{m_{tj}}) \]
and where
\[ \overline{m_{tj}} \doteq \sum_p w_p m_{ptj} \]
In other words, $u(m_{ptj})$ expresses how far particle $p$’s opinion about $M_t$ is from the “mean belief” $\overline{m_{pt}}$. Note that the uncertainty is over the teacher’s belief about the student’s belief; it is not over the student’s belief itself. (The latter uncertainty is instead captured by the “goodness” defined above.)

• A bias term (constant 1).

To compute the policy weight vectors, we use the REINFORCE [98] policy gradient technique. Since we have a model of the student, we can run simulations to estimate the value and gradient of a particular policy as parameterized by \( \{w_u\}_{u \in \mathcal{U}} \). Whenever the macro-controller decides either to teach or ask about word $j$, it queries the micro-controller (see below) to determine the particular image(s) it should present in conjunction with $j$. For each gradient estimate, we ran 500 simulations of at most 500 timesteps (a simulation can end early if the simulated student passes the test) using 100 particles. The learning rate was set to 0.005, and the policy was optimized over 400 gradient descent steps.
At run-time, given the particles expressing the teacher’s belief \( b_t \), policy \( \pi \) computes the probability over each action. The particular action chosen by the teacher at that timestep is then sampled according to these probabilities.

### 5.7.2 Micro-controller

Given the decision at time \( t \) to teach (or ask about) word \( j \), the teacher needs a mechanism to select an image (or pair of images) for \( j \). For the micro-controller we use a 1-step lookahead search based on a information-maximization heuristic. In particular, for the “teach” actions, the teacher selects image \( k \) so as to maximize the expected (w.r.t. the particles) increase, from time \( t \) to \( t + 1 \), in the goodness of the student’s belief about word \( j \), where goodness is defined by Equation 5.10 above.

For “ask” actions, the teacher chooses \( k_1, k_2 \) so as to maximize the expected reduction in the teacher’s “total uncertainty” from time \( t \) to \( t + 1 \). We define “total uncertainty” as:

\[
\sum_p w_p \sum_j u(m_{ptj}) + \sum_p w_p (\alpha_{pt} - \overline{\alpha}_t)^2 + \sum_p w_p (\beta_{pt} - \overline{\beta}_t)^2
\]

where \( \overline{\alpha}_t \equiv \sum_p w_p \alpha_{pt} \) and \( \overline{\beta}_t \equiv \sum_p w_p \beta_{pt} \). This metric includes uncertainty not just over the student’s belief, but also over student parameters \( \alpha_t \) and \( \beta_t \).

### 5.8 Procedure to train automatic teacher

Given the models of the student and teacher, the particle-based belief update mechanism, and the method of computing a teaching policy, we are now ready to put all the parts together and create a prototype automated language teacher. The procedure for creating a teacher of \( n \) words is the following:

1. Collect database of images \( \mathcal{Y} = \{1, \ldots, l\} \) with which to teach the words.

2. Query human subjects on which concepts they believe are represented by the images, i.e., estimate \( P(c \mid k) \) for each image \( k \in \mathcal{Y} \).
Table 5.1: List of words and associated meanings taught during a word learning experiment.

<table>
<thead>
<tr>
<th>Word</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>duzetuzi</td>
<td>man</td>
</tr>
<tr>
<td>fota</td>
<td>woman</td>
</tr>
<tr>
<td>nokidono</td>
<td>boy</td>
</tr>
<tr>
<td>mininami</td>
<td>girl</td>
</tr>
<tr>
<td>pipesu</td>
<td>dog</td>
</tr>
<tr>
<td>mekizo</td>
<td>cat</td>
</tr>
<tr>
<td>xisaxepe</td>
<td>bird</td>
</tr>
<tr>
<td>botazi</td>
<td>rabbit</td>
</tr>
<tr>
<td>koto</td>
<td>eat</td>
</tr>
<tr>
<td>notesabi</td>
<td>drink</td>
</tr>
</tbody>
</table>

3. Collect data from human subjects to estimate the time costs (in seconds) for “teach”, “ask”, and “test” actions, as well as student model parameters $P(\alpha), P(\beta)$ by teaching students using an arbitrary teaching policy.

4. Execute gradient descent on policy parameters $W$ to locally minimize the expected teaching cost using the REINFORCE [98] policy gradient algorithm.

We implemented this procedure and tested the efficacy of the resulting teaching policy; the experiment is described in the next section.

5.9 Experiment

We evaluated the procedure described in Section 5.8 using 10 words from a synthetically generated foreign language with the associated meanings listed in Table 5.1. The concept set consisted of all of the 10 meanings listed above plus an additional “other” concept whose purpose was to account for everything that subjects might perceive in the images that is not one of the words’ meanings. Hence, there were $n = 10$ foreign words and $m = 11$ concepts in total.

We collected an image set of 56 images, each of which showed one of the humans/animals in the concept set that was either eating or drinking. The images are shown in Figure 5.5.
Figure 5.5: Image set used to teach foreign words that mean “man”, “cat”, “eat”, “drink”, etc. All images were found using Google Image Search.
We used a custom-built image labeling tool that asks the labeler to indicate, for each image, “how strongly or weakly the image represents each of the listed concepts” using a slider bar for each of the 11 concepts (including “other”). The scale of each slider is from 0 to 1 and can thus be used as a reasonable estimate of \( P(c | k) \) for each concept and image combination.

Next, we estimated the time costs and student model parameters from 42 subjects from the Amazon Mechanical Turk. The subjects for this parameter estimation phase were taught with one of three possible teaching policies described below: a RandomWordTeacher, a HandCraftedTeacher, and OptimizedTeacher. (The parameters of the teachers during this data collection phase were set by hand.) Based on these pilot subjects, we computed the average time costs to be \( c(\text{teach}) = 10.74 \), \( c(\text{ask}) = 7.53 \), and \( c(\text{test}) = 106.46 \) seconds, respectively. For simplicity, we assumed \( \alpha_t, \beta_t \) were constant across the learning session (but possibly different for each student).

Finally, we conducted policy gradient on the stochastic logistic policy’s weight vectors using the costs and parameters estimated in the previous step. We set the time horizon \( \tau = 500 \) and the learning rate to 0.005. In practice, we found that the choice for the learning rate was important – if it was too high, then policy gradient descent sometimes converged to a nonsensical solution such as never testing the student at all. We executed gradient descent for 400 iterations. We call the resultant policy the OptimizedTeacher.

### 5.9.1 How the OptimizedTeacher behaves

The parameter set \( \{w_u\}_{u \in \mathcal{U}} \), concatenated row-wise into a matrix, is shown in Figure 5.6. For a vocabulary of \( n = 10 \) words there are 21 rows – one “teach” and “ask” action for each word, plus a “test” action. \( g(j) \) represents the goodness of the student’s belief about word \( j \) as estimated by the teacher’s particles. At run-time, each vector \( w_u \) is multiplied (inner-product) by the feature vector \( x_t \) computed from the teacher’s particles, which in turn gives a scalar proportional to the probability of executing action \( u \) under the policy. In the figure, dark colors represent low weights, and light colors represent high weights. First, notice that
Figure 5.6: Policy of the OptimizedTeacher. Each row corresponds to the policy weight vector $w_a$ for the action specified on the left, e.g., “Teach $j$” means teach the word indexed by $j$. Dark colors correspond to low values of the associated weight vector; light colors represent high values.
the “bias” feature (last column) for the “test” action (last row) is very dark, i.e., has a very low weight – this means that the teacher tends not to execute “test” actions very often. In addition, notice that the “uncertainty” feature for “test” is also quite dark – the teacher does not like to test when it is uncertain about the student’s beliefs. Finally, notice how the “goodness” features 1,...,10 for the “test” action are very light – if the teacher believes the student’s beliefs are good, then this increases the probability that the teacher will “test”.

Next, observe the “staircase” effect along the diagonal – for every “teach” action $j$, the teacher tends not to teach word $j$ if the student’s belief for word $j$ is already good – this makes sense since teaching the student a word she already knows would be a waste of time.

Finally, the “ask” actions tend to be executed rarely, as evidenced by the dark color associated with most of their features. This is likely because the binary questions that we constrained the teacher to ask provide relatively little information, especially compared to a “test”, which contains $d$ questions at once.

In order to measure the teaching effectiveness of the computed OptimizedTeacher, we compared it to two other teaching strategies: a HandCraftedTeacher and a RandomWordTeacher. The performance metric was the average amount of time, in seconds, that a student would take to learn the words and pass the test.

5.9.2 RandomWordTeacher

We designed a teaching policy called the RandomWordTeacher that selects a word uniformly at random from the vocabulary $\{1, \ldots, n\}$ at each round. Then, to teach word $j$, the RandomWordTeacher randomly selects an image $k$ with probability proportional to $P(c \mid k)$ where $W_j = c$. The RandomWordTeacher only shows positive examples, i.e., $A_{tq}$ is always 1. Every $\delta$ rounds, the RandomWordTeacher gives a test to the student; if the student passes (accuracy at least 90% correct), then the teaching session is over. Otherwise, teaching continues for another $\delta$ rounds, then the teacher gives another test, and so on. The maximum length of the teaching session is $\tau = 500$ rounds total (including tests). Parameter $\delta \in \{5, 10, \ldots, 45, 50\}$ was optimized in simulation and set to $\delta = 35$. 
Note that the RandomWordTeacher does not select an image to teach word $j$ uniformly at random – this would be almost nonsensical and could frequently result in showing, for example, an image of a rabbit to teach that meaning of the word *fota* equals “woman”. Instead, the RandomWordTeacher actually uses part of the student model of the Optimized Teacher, namely the perceptual belief $P(c \mid k)$.

### 5.9.3 HandCraftedTeacher

The HandCraftedTeacher is a reasonable teaching policy that we crafted by hand and which uses the student’s responses to the test questions to teach more effectively. In other words, it is a closed-loop teacher.

The HandCraftedTeacher keeps track of how many times it has taught each vocabulary word. At each round, it computes the set of words that have been taught for fewer than $\epsilon = 3$ times (parameter optimized in simulation) and chooses one of those words uniformly at random. For the chosen word, the HandCraftedTeacher then chooses an image with probability proportional to $P(c \mid k)$ where $W_j = c$. The HandCraftedTeacher only shows positive examples, i.e., $A_{eq}$ is always 1.

As soon as the HandCraftedTeacher has determined that each of the $n$ words has been taught $\epsilon$ times, it gives the student a test. If the student passes, then the session is over; otherwise, the HandCraftedTeacher sets the counter of every word that the student answered incorrectly on the test to 0. It then resumes selecting a word that has been taught fewer than $\epsilon$ times. Note that this teaching strategy was designed not to teach the same words needlessly when other words have not been taught at all; it focuses the teaching on those words that the student did not learn well based on test performance; and, like the RandomWordTeacher, it capitalizes on the perceptual model $P(c \mid k)$ to teach each word.
5.9.4 Experimental conditions

We constructed a simple teaching interface using Javascript and conducted a study on 90 human subjects from the Amazon Mechanical Turk. Because we were interested more in how the student learns the words’ meanings rather than how well students can remembered them, we allowed students to use a “notepad” during the learning session, implemented using a textbox within the teaching system.

In our experiment there were three experimental conditions to which subjects were randomly assigned: OptimizedTeacher ($N = 29$), HandCraftedTeacher ($N = 35$), and RandomWordTeacher ($N = 26$). Subjects were paid $0.15 for completing the experiment, defined as passing the test (accuracy at least 90%). Because the payment was fixed, subjects were economically motivated to learn as efficiently as possible. In pilot experimentation, we found that offering a monetary reward for faster learning (in addition to the base payment for participation) had little effect on subjects’ performance.

5.9.5 Results

Results are shown in Figure 5.7. The OptimizedTeacher delivered the best performance with an average time to learn the words and pass the test of 539.81 seconds; this was statistically significantly better ($p < 0.01$) than the RandomWordTeacher condition, in which subjects took an average of 735.77 seconds – a 24% improvement. The OptimizedTeacher also performed slightly better than the HandCraftedTeacher (mean time to completion of 560.99 seconds).

The results suggest that the procedure from Section 5.8 can produce an automated teacher that performs equally well as a reasonable hand-crafted policy. The advantage of the policy computation procedure proposed in this paper is that, should the image statistics or time costs change, e.g., for a different vocabulary of words, or for a different population of students, then the same procedure outlined in Section 5.8 could be executed to automatically account for the change when computing a new policy. In addition, having an explicit model of the student’s beliefs is useful when creating an affect-sensitive teacher, as explored in Section 5.10. Finally, the performance of the OptimizedTeacher also lends support to
the paradigm of modeling the student as a Bayesian learner: even though human students may not be perfect computers of Bayesian belief updates, the student model was sufficiently accurate to teach students in a reasonable manner.

### 5.9.6 Correlation between time and information gain

Since the human subjects in the experiment were economically motivated to complete the task quickly, it is possible that they themselves could compensate for any poor teaching examples chosen by teachers. For example, if the student already learned that the word *pipesu* means dog, and yet the automated teacher chose to show him/her even more examples to convey *pipesu*’s meaning, then perhaps the student would simply “click through” that example quickly to reach more informative word+image combinations.

To test this hypothesis, we computed the average Pearson correlation over all subjects between the time (in seconds) spent on each round in which the teacher executed a “teach” action, and the expected increase in belief “goodness” (Equation 5.10) given the chosen image+word combination, by the student during that round. The average correlation, which was statistically significant ($p < 0.01$), was $r = 0.28$. This suggests that the students’ own behavior may have moderated the
influence of the particular teaching strategy.

5.10 Incorporating affect

A key challenge for the ITS community today is to design mechanisms to incorporate “affective” observations of the student, such as those captured by a web camera, into the decision-making process. POMDPs are a natural framework for using affective sensor inputs, and here we propose and demonstrate in simulation a simple architecture for how this might be done.

Consider a learning setting in which the student goes through phases in which she is “trying” or “not trying”. When the student is trying, she processes each image+word combination fully, i.e., her belief update strength $\beta_t$ is high. When asked a question about a word+image pair or on a test, she tries to answer correctly and answers according to her beliefs. In contrast, when a student is not trying, her belief update strength and answer discriminability are low.

As a teacher, it is important to know when the student is receptive to learning. It is also important to attribute mistakes on a test, or incorrect answers to posed questions, to the proper cause: did the student really not know the correct answer, or was she simply not trying? In other words, it is valuable, in this example, for the teacher to have a good estimate of $\beta_t$.

Note that the teacher already has some ability to estimate $\beta_t$ through the student’s responses to questions and tests. If the student previously answered several questions about the same word correctly, and yet now she answers them incorrectly, then this is probably because she has stopped trying. However, questions and tests will likely be issued relatively rarely due to their cost; hence, the teacher’s estimate of $\beta_t$ may be very uncertain.

Now, suppose the teacher has access to some kind of “affective sensor”, e.g., an estimate of how hard the student is trying as measured by a classifier that processes images captured by a web camera. These sensor inputs could be used to estimate $\beta_t$ much more precisely. If the teacher also had some mechanism with which to re-“engage” the student in the task, and if the teacher executed this
action judiciously, then it could potentially teach more effectively.

5.10.1 Simulation

To illustrate the potential utility of incorporating affective state estimates into the decision-making process, we conducted a simulation, using the same words and image statistics from Section 5.9, comparing two automated teachers that teaches students who can “stop trying” as described above. In particular, we let $\beta_t \in \{0.25, 0.95\}$ for each $t$, and we suppose that over time the student tends to stop trying, i.e., with probability 0.05, she will transition from $\beta_t = 0.95$ to $\beta_{t+1} = 0.25$. In order to help the teacher teach more effectively, we endow both teachers with an additional “engage” action, which causes the student to enter the “trying” state ($\beta_{t+1} = 0.95$) with some probability 1. However, the “engage” action also incurs a cost – we suppose that it is 4 times more expensive than a “teach” action. In order to encourage the “engage” action to be selected when $\beta_t$ was estimated by the teacher to be low, we added an added an extra element to the feature vector $x_t$ of the particles, namely the expected value (w.r.t. the teacher’s particles) of $\beta_t$: $\sum_p w_p \beta_{pt}$.

We compare one teacher with “affective sensors” to another teacher that does not have affective sensors. The teacher with affective sensors receives at every timestep $t$ an “affective observation” $z_t$ (in addition to the standard observation $o_t$) that depends only on $\beta_t$. Here, we model $z_t$ as being generated from $\beta_t$ plus some Gaussian noise:

$$P(z_t \mid m_t, o_t, \beta_t, u_t) \sim \mathcal{N}(\mu = \beta_t; \sigma^2 = 0.25^2)$$

where $\mathcal{N}$ is the normal distribution. At every timestep, the affective teacher updates its belief about the student’s state at time $t+1$ (see Section 5.6.1) based not
just on $o_t$, but now also on $z_t$:

\[
\begin{align*}
   b_{t+1} & \doteq P(s_{t+1} | u_{1:t}, o_{1:t}, z_{1:t}) \\
   & \propto P(s_{t+1}, o_t, z_t | u_{1:t}, o_{1:t-1}, z_{1:t-1}) \\
   & = \sum_{s_t} P(s_{t+1}, o_t, z_t | s_t, u_{1:t}, o_{1:t-1}, z_{1:t-1}) P(s_t | u_{1:t}, o_{1:t-1}, z_{1:t-1}) \\
   & = \sum_{s_t} P(s_{t+1} | s_t, u_t) P(o_t, z_t | s_t, u_t) P(s_t | u_{1:t-1}, o_{1:t-1}, z_{1:t-1}) \\
   & = \sum_{s_t} P(s_{t+1} | s_t, u_t) P(o_t | s_t, u_t) P(z_t | s_t, u_t) b_t \\
   & = \sum_{s_t} P(s_{t+1} | s_t, u_t) P(o_t | s_t, u_t) P(z_t | \beta_t) b_t
\end{align*}
\]

The value of the affective sensors can thus be defined by this additional term $P(z_t | \beta_t)$ which helps to (greatly) constrain the possible states $s_t$ that could explain the data observed by the teacher.

We used policy gradient descent (as before) to optimize the policies of both the teacher with affective sensors and the teacher without them. The policies learnt for both teachers in this simulation were similar to the policy shown in Figure 5.6. For the “engage” action, the weight on the feature expressing the expected $\beta_t$ is strongly negative, so that the teacher tends to execute “engage” when the teacher believes that $\beta_t$ is low (i.e., student has stopped trying). In addition, the weight for the “test” action associated with $\beta_t$ is also strongly negative, which encodes the fact that asking a student to take a test when she is not trying is probably a bad idea. These trends apply to the policies of both teachers – one with affective sensors, one without. The key difference between them is that the teacher with extra sensors can estimate $\beta_t$ more accurately than the other teacher.

In simulation, the utility of the benefit of having affective sensors, as measured by the student’s learning gains as a function of time, is displayed in Figure 5.8 (left). Results were averaged over 1000 simulation runs. The shaded area for each teacher represent the teacher’s belief of the student’s belief goodness, summed over all words, plus or minus one standard deviation for each timestep $t$. Notice how the mean teacher’s belief of the student’s belief goodness is not only higher for the teacher with affective sensors, but its uncertainty is also lower. Figure 5.8
Figure 5.8: Simulation results comparing a teacher with “affective sensors” to one without them. The affect-sensitive teacher is able to teach the student more quickly (left), allowing the student to pass the test, on average, more quickly (right). (right) shows the average amount of time for a student to pass the test for each teacher. In agreement with the other graph, the teacher with affective sensors helps its students to graduate more quickly on average. Finally, the average cost of the teaching session (until either $\tau = 200$ or the student passes the test) was also lower for the teacher with affective sensors: 89.50sec for the affect-sensitive teacher versus 133.25sec for the teacher without the extra sensors.

5.11 Summary

We have developed an automated foreign language teacher that teaches words by image association, i.e., in the manner of Rosetta Stone language software [73]. The student in this learning setting was modeled as a Bayesian learner; the teaching problem was modeled as a POMDP; and the system’s controller was optimized using a hierarchical policy gradient descent approach over features of the teacher’s belief as represented by a particle filter. Results on a human subjects experiment indicate that the policy computed using the proposed procedure can deliver policies that perform favorably compared to reasonable baseline controllers.
Finally, we illustrate how the developed prototype teacher could be extended to incorporate affective observations of the student, e.g., from a webcam, in order to teach even more effectively.

5.12 Appendix: Conditional independence proofs

Below we prove two conditional independence theorems about the random variables shown in the graphical model of Figure 5.2.

5.12.1 d-Separation of $W_j$ from $W_{j'}$ for $j' \neq j$

We show that

$$P(w_1, \ldots, w_n \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t}) = \prod_j P(w_j \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t})$$

All we must show is that $W_j$ is conditionally independent of $W_{j'}$ given $Y_{1:t}$ and $A_{1q_1}, \ldots, A_{tq_t}$ for all $j' \neq j$. To prove this conditional independence we will use graph theory and show that every undirected path between $W_j$ and $W_{j'}$ is d-separated. (For brevity, we will omit the word “undirected”.) First, note that it suffices to show that every path without cycles between $W_j$ and $W_{j'}$ is d-separated: any path with cycles can be trivially converted to a path without cycles, and if the simplified path (with no cycles) is d-separated, then so too will be the original path.

Our proof proceeds in two parts. In Part 1, we will show that any path $P$ from $W_j$ to $W_{j'}$ must contain two nodes $A_{\nu_j}$ and $A_{\nu_j'}$ for some timestep $\nu$ such that $j' \neq j$. In Part 2 we will use the fact that, at every round $\nu$, the student observes at most one node $A_{\nu_j}$ to show that the other, unobserved node $A_{\nu_j'}$ d-separates $W_j$ from $W_{j'}$.

**Part 1:** Notice that the only nodes to which $W_j$ is connected are $A_{1j}, \ldots, A_{ij}$; hence $P$ must start with the nodes $W_j A_{\nu_j}$ for some $\nu$. Now, consider the next node in $P$ after $A_{\nu_j}$: the only nodes to which $A_{\nu_j}$ is connected are $C_{\nu}$ and $W_j$. We can ignore the latter possibility because a path that starts out as $W_j A_{\nu_j} W_j$ would contain a cycle. Hence, $P$ must start out as $W_j A_{\nu_j} C_{\nu}$. From the
graphical model it is clear that any path from $C_\nu$ to $W_{j'}$ must eventually proceed through some node $A_{\nu j'}$. The only remaining question is whether $j' = j$ or $j' \neq j$. However, we can discard the former possibility because that would result in a cycle. Hence, every path $P$ from $W_j$ to $W_{j'}$ must contain two nodes $A_{\nu j}$ and $A_{\nu j'}$ such that $j' \neq j$ for some timestep $\nu$.

**Part 2:** To finally prove d-separation between $W_j$ and $W_{j'}$, we note that, at each timestep $\nu$, at most one of $A_{\nu j}$ and $A_{\nu j'}$ can be observed by the student because only one query is “answered” at each timestep. Since at least one of those two nodes is unobserved, and since none of the $A$ nodes has any descendants, then $P$ is d-separated (by the “inverted fork” rule) by either $A_{\nu j}$ or $A_{\nu j'}$. Since this is true of any path $P$ without cycles, we conclude that $W_j$ is d-separated from $W_{j'}$.

### 5.12.2 d-Separation of $W_j$ from $A_{\nu q_\nu}$ for all $\nu$ such that $q_\nu \neq j$

We prove that

$$P(w_j \mid y_{1:t}, a_{1q_1}, \ldots, a_{tq_t}) = P(w_j \mid y_{1:t}, \{a_{\nu j}\}_{\nu \neq j})$$

In other words, belief about $W_j$ is not affected by answers to queries about other words $j' \neq j$. Consider any timestep $\nu$ for which $q_\nu \neq j$, and consider any path $P$ from $A_{\nu q_\nu}$ to $W_j$. In order for $P$ to reach $W_j$, it must pass through either $W_{q_\nu}$ or through $A_{\nu j}$. In the former case, we already know (from Appendix 5.12.1, where the set of observed nodes was the same) that every path $P'$ from $W_j$ to $W_{j'}$ is d-separated for all $j' \neq j$; hence, path $P$ would also be d-separated. In the latter case, since $A_{\nu q_\nu}$ is observed, and since at most one $A$ node is observed at any one timestep, then node $A_{\nu j}$ cannot be observed; hence, $A_{\nu j}$ d-separates $A_{\nu q_\nu}$ from $W_j$. In either case, $W_j$ is d-separated from $A_{\nu q_\nu}$ for any $\nu$ such that $q_\nu \neq j$. Hence, $W_j$ is conditionally independent of $A_{\nu q_\nu}$ for any $\nu$ such that $q_\nu \neq j$.

### 5.13 Acknowledgement

Chapter 5, in full, is currently being prepared for submission for publication of the material. Jacob Whitehill and Javier Movellan. The dissertation author was
the primary investigator and author of this material.
Bibliography


