Introductory AI for Both Computer Science and Neuroscience Students

Susan Eileen Fox
Macalester College
1600 Grand Avenue
Saint Paul, MN 55105
fox@macalester.edu

Abstract

Macalester College offers a single undergraduate elective in artificial intelligence. This course is cross-listed between Computer Science and Cognitive and Neuroscience Studies, and includes majors from both disciplines. Computer science students bring strong technical knowledge and skills but little awareness of the deeper issues of cognition and intelligence, while neuroscience students bring knowledge about biological and psychological models for human and animal behavior, but much less technical knowledge and experience. To meet the needs of these very different student populations, the course was redesigned from the ground up: choice of topics, classroom activities, kinds of assigned work. The resulting course retains strong technical content, and provides opportunities for all students to pursue topics related to their interests. It has maintained strong enrollment by computer science majors, even as the number of majors has declined, and has seen an increasing number of neuroscience majors each time it is offered.

Introduction

Undergraduate courses in artificial intelligence are typically taught as junior or senior level electives. As such, they can assume student proficiency in programming, and significant knowledge of algorithms and data structures. Popular textbooks such as Russell & Norvig (2003) cater to this kind of class, with extensive pseudocode and discussions of correctness, optimality, and algorithm efficiency. This kind of class serves undergraduate computer science majors very well, providing them with an overview of AI techniques that they can draw upon whatever career path they follow.

The "typical" AI course is not suitable for students with interests in AI, but who do not have extensive background in computer science. AI has strong ties to other disciplines, including neuroscience, cognitive science, and linguistics. Students from those disciplines could benefit from learning about AI, but typically cannot be expected to complete all the computer science prerequisites. Such students are often served by special-purpose courses that provide a shallow overview of AI techniques, or focus strictly on those of most interest to the related disciplines. A separate course is not always possible, however, particularly at small undergraduate institutions.

At Macalester we have developed an introductory AI course that is cross-listed with an interdisciplinary program in Cognitive and Neuroscience Studies. The course must meet the needs of computer science students as well as students whose primary interests lie in neuroscience, psychology, biology, philosophy, and linguistics. Computer science students have taken several programming courses, and typically complete a course on algorithm analysis. The cognitive and neuroscience students have taken one semester of introductory computer science, plus some relevant coursework in their own disciplines. We have tried to balance the needs of each population of students, while ensuring the course remains challenging and deep.

The course was re-designed to spend more time on AI topics of interest to cognitive science. It incorporates in-class activities that illustrate AI techniques in class on small, concrete examples. Homework assignments reinforce the understanding developed in class, with the same questions asked of all students in the class. Small "short-term projects" are tailored to the different student populations, with computer science students writing programs, and others using existing software or writing research papers. All students choose a semester-long project, done individually or in pairs, on a topic of personal interest. Semester projects must involve AI software or programming in some way, but range from expository research papers to full implementations of AI systems.

The course still involves the technical detail of a typical AI course: computer science students still program a range of AI techniques, and analyze their work in typical ways. However, the material is made accessible to non-computer science students through classwork and homework exercises. Topics are chosen to balance the needs and interests of both groups.

The re-design of the AI course was undertaken in collaboration with faculty involved in the Cognitive and Neuroscience Studies program. The course has been taught three times since the re-design process began (Fall 2002, Fall 2004, and Fall 2006). During that time, the enrollment in the course has remained constant despite a drop in the number of computer science majors. A large percentage of computer science majors still take the course, and the number of students enrolling in the CNS side of the course has increased dramatically, from 2 the first time, to 4 the second, to 13 in Fall 2006. All students demonstrate mastery of course topics, and have equally successful semester projects.

Integrating non-computer science students with knowledge and experience in related disciplines has benefitted both student populations. The non-computer science students bring a new perspective to class discussions, enthusiasm for a different set of AI topics, and innovative semester projects. They learn more about AI than they would in a shallow survey course. Computer science majors are exposed to the broader world of work in natural and artificial cognition, while retaining the chance to hone their programming skills and explore ”hard-core” AI techniques.

Background

Macalester College offers an inter-disciplinary program in Cognitive and Neuroscience Studies (CNS). Students in this program learn about the study of brain and mind in a variety of disciplines, including biology, psychology, philosophy, mathematics, and computer science. A separate Linguistics program also has a ”cognitive” track that overlaps with the CNS program. Each CNS student completes a broad core curriculum, drawn from all five disciplines listed above. The student then chooses a disciplinary focus and completes the major through advanced study in that discipline. While I often refer to the students here as ”neuroscience,” they actually lie along a broad continuum from cognitive science to true neuroscience.

The Math/CS component of the core curriculum includes introductory statistics, introductory computer science, and one of three relevant Math/CS courses. Artificial Intelligence is the most preferred of those three courses.

Macalester is a small, undergraduate, liberal-arts college. The Department of Mathematics and Computer Science houses programs in math, computer science, and statistics. Roughly 3.5 FTE are dedicated to the Computer Science program. Advanced electives like Artificial Intelligence are offered every other year. There is no possibility of offering a special version of the course for students interested in cognitive science, neuroscience, or linguistics. Starting in 2002, we integrated students interested in the CNS program into the AI course by cross-listing it with Cognitive and Neuroscience Studies and setting different prerequisites for the two listings. CNS students must have completed an introductory CS course, and they must have taken at least one other of the courses in the CNS core curriculum. In practice, most CNS students wait until their junior or senior years to take the AI course, and have completed three or four related courses.

Course Goals

Our goal in revising the AI course was to make a virtue of necessity: to include CNS students and make the course into a positive experience for everyone involved. Rather than having an ”easy” track and a ”hard” track, we wanted to encourage CS and CNS students to talk to each other, and to leverage the strengths of the neuroscience students to enrich the course as a whole.

I first focused on the choice of topics in revising the course. CNS students are particularly interested in topics like artificial life, neural networks, genetic algorithms, and Bayesian modeling. I also revised the course assignments, which were too focused on implementing AI techniques. Guidelines for the semester project were broadened to allow for less computationally-intensive topics.

As the course developed, I realized the need to change the classroom environment itself, to better support all students in the class. In Fall 2006, an average of two out of three classes included significant time spent on hands-on activities that provided immediate feedback to me and the students about what they understood and what they did not. Time was set aside to revisit topics after activities, to review the portions students found problematic.

The revised course is, I find, a better-balanced course, one that serves students with different interests, and yet provides a core of knowledge that both populations take away. As a class, we spend more time on computational modeling than before, and emphasize ”lower” tasks such as animal behavior and reactive systems, rather than knowledge representation and logical reasoning.

Obstacles and Rewards

CNS students typically have only one prior computer science course; we offer three different introductory courses that differ in how much programming students do, and what language they learn. In any case, the students have limited programming skills, and little experience with different kinds of algorithms, or even reading and understanding pseudocode. They cannot succeed in the class if required to write sizable programs on their own, and they need extra help to understand AI techniques described in algorithmic terms. Past attempts to pair CNS students with CS students on programming assignments were uneffective. The gap in skills often led CS students to dominate the work. In fact, I have found that groups of CNS students working on programming assignments learn more than in ”mixed” groups.

Computer science students often take AI because of the ”cool” applications, and are relatively unconcerned about issues of cognition and ”real” intelligence. Motivating them to think about the big picture in AI can be a challenge. CNS students naturally draw comparisons between artificial methods and human or animal behavior. They challenge the ”anything that works” attitude of some computer science students, while at the same time tying the course into real-world applications of AI methods.

Course Topics

It is never easy to select from the wide range of topics that fall under Artificial Intelligence to create a one-semester introduction to the field. The course I have created at Macalester still resembles a typical AI course (see list of topics in Figure 1). Many traditional AI topics remain in the syllabus. However, the amount of time spent on each topic has changed. We spend a significant amount of time on machine learning, including neural networks and genetic algorithms. We discuss traditional search algorithms such as

A* and minimax, but in an abbreviated timeframe. We spend relatively little time on logic and logical reasoning, but more on statistical methods that crop up commonly in modeling of physiological systems: Bayes nets, HMMs, and particle filtering. We emphasize behavior-based robotics over more traditional robotics topics.

Topic	Time
Reflex agents and behavior-based robotics	(3 weeks)
State-based agents, state space search, A*, Minimax	(2 weeks)
Propositional logic and inference, satisfiability	(1 week)
Machine learning: inductive, case-based, neural nets, genetic algorithms	(3 weeks)
Uncertainty: Bayes' Rule, naive bayes, Bayes nets, particle filtering	(2 weeks)
Natural Language Processing: traditional and statistical	(2 weeks)
Robotics revisited (localization through MCL)	(1 week)

Figure 1: List of topics in Introduction to Artificial Intelligence, Fall 2006

Dynamic Classroom Environment

In early versions of the AI course, I found that CNS students sometimes fell through the cracks. When faced with a difficult topic in class, they failed to ask questions, and also failed to master the techniques outside of class. Similarly, the class sometimes separated into two groups, CS and CNS, that had little to do with each other. It was clear to me that I needed to improve the dynamics in the classroom, to get students talking to each other, and to me. Beginning in 2004 and continuing in 2006, I began to develop a series of in-class activities for the AI topics we were covering.

The first day of class in Fall 2006 set the tone for the semester. Rather than giving students an hour-long lecture on what AI was all about, I planned a suite of activities, divided the class into groups, and had them rotate through the activities, interacting with me and each other. The activities were chosen to highlight different areas of AI, including natural language processing. logical reasoning, image processing, robotics, and planning (Figure 2). Some activities related to topics we would see later in the semester, and others were meant to suggest possible semester projects. The inspiration for using Clue for logical reasoning derives from Neller (Neller, Markov, & Russell 2006); the Hidden Pictures puzzles were from Highlights magazine online (Highlights 2006). The robot activity derives from Genesereth and Nilsson (Genesereth & Nilsson 1987), but appears in Russell and Norvig (Russell & Norvig 2003).

On the second day of class, we discussed the activities the students had engaged in, focusing on which were easy or hard for a human or animal to perform, and which would be easy or hard for a computer.

Area	Activity
Natural Language	Transcribe Gaelic video clip, translate the "real" transcription
Logical reasoning	Play Clue in a card-game form, and think about deductive reasoning involved
Image recognition	Find hidden pictures, what would be hard or easy for a computer?
Robotics	Play-act the components of a robot stacking two boxes
Route planning	Find a route from Saint Paul, MN to Bloomington, IN, using an atlas, given various criteria

Figure 2: First-day activities, Fall 2006

Throughout the course, I followed the same model for presenting new topics. I first introduced the topic, giving its context and one or two simple examples. After my introduction, the students broke up into groups of 3-5 and worked through a planned activity together, with my guidance answering questions, clarifying misconceptions, and posing new problems to more advanced groups. From observing how the groups grappled with the activity, I got a good sense for what the students understood and what was unclear. I would then take time that day or the following to review the parts that were problematic.

Activities centered on concrete examples, most of small enough size to be worked through in 15 to 20 minutes during class. On occasion an activity would take most or all of the class hour. When working with software, we would often spend one or two class days in a computer lab.

Two typical class activities are described below: working through the minimax algorithm for part of a Tic-Tac-Toe game, and acting out the particle filtering algorithm, with the class taking on the role of particles. In each of these activities, major misconceptions were revealed, and corrected before students attempted homework exercises on these topics.

Minimax and Tic-Tac-Toe: Tic-tac-toe was used to illustrate the minimax algorithm, following several days of examining single-player state space search problems. Students had completed a reading on the algorithm before class. I described the algorithm, walked through an example, and we discussed a simple evaluation function. Then students were asked to work through a new example (Figure 3).

I discovered during the exercise that students were unclear on the depth-first search order used by Minimax, and some wanted to take the average score for the parent nodes rather than the minimum or maximum. We discussed as a class the results of Minimax's biases: conservative play that avoids a loss rather than gambling on a win.

Particle Filtering Personified: We had examined Bayes networks and inference with them for a week, and were ready to discuss particle filtering for estimated inference. This can be a rather difficult algorithm for students to un-

Minimax activity:
The figure below shows the starting state somewhere in the middle of a game. The basic minimax algorithm searches from this state down every branch of the tree, using a depth-first search (assume it goes left-to-right through siblings).
Go over the tree and label each node with the value assigned to it by minimax, using the evaluation function we discussed.
When you finish, repeat the process for depth-limited Minimax with a depth of 3.

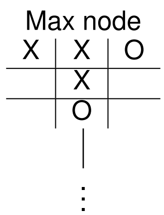

Figure 3: Activity for Minimax algorithm

derstand. I chose to use a localization-type task to illustrate particle filtering because students were very comfortable with robotics applications. I provided the algorithm in pseudocode, and went over the general idea prior to the exercise. As a class, then, we acted out the algorithm as described in Figure 4.

As we worked through the exercise, students asked questions about how sampling from the transition model worked, and why really bad sample points might persist through several iterations. After the exercise, we needed to sit down and work through the details of resampling, which were hidden from students during the exercise by the use of a program to generate the new sample set.

Course Assignments

The course assignments were designed to ensure (1) that everyone understood the basic AI methods and how to apply them, (2) that computer science students had experience implementing some typical AI techniques, (3) that neuroscience students examined how psychological or biological theories compare to their related AI counterparts, and (4) that each student had an opportunity to explore some specific topic in depth. AI courses are often surveys of the field Without some project aimed at deep understanding, the course becomes insubstantial and trivialized.

In order to achieve the goals articulated above, I designed three different kinds of assignments for the course: homework exercises, ”short-term projects”, and a semester project. Students had the option to work collaboratively on most projects, but were expected to do most exercises individually. Discussion of problems was explicitly permitted and encouraged.

1. The Robot will stand at one location in the classroom.
2. The Particles will spread themselves evenly around the room: the robot is equally likely to be in any location in the world. Each Particle will have a numbered card to hold up when asked.
3. We’ll then repeat the following process:
   (a) The Robot will take one step forward.
   (b) Each Particle will turn 45 degrees to the left.
   (c) Each Particle will then “sample from the transition model:” change yourself in a random way, but be guided by gaussian probability distributions.
   (d) The Robot will report two pieces of evidence about its location: how far away the nearest obstacle is at eye level, and how far away the nearest obstacle is at knee level (remember that the Particles don’t really exist in the world, so they aren’t obstacles, but desks, tables, and walls are).
   (e) Each Particle must determine how consistent the evidence is with its location (is there a high similarity, a medium similarity, or a low similarity).
   (f) Professor Fox will ask the high similarity Particles to hold up their numbers, and then the medium similarity, and the low similarity. These lists will be used to “resample” from the population, giving a new list of Particles. Some Particles may be selected more than once, and others may not be selected at all.
   (g) Professor Fox will move those Particles who were not selected to be copies of those selected more than once.

Figure 4: Particle filtering activity

Homework Exercises

Homework exercises were collections of relatively short problems, often taken from the course textbook, and designed to be solved by hand, rather than by programming. There were roughly 5 homework assignments each semester. Homework exercises reinforced the in-class activities, so students could see how well they understood the material when I was not at hand to help. Because they established the level of mastery of basic AI techniques, all students in the course completed the same exercises.

Typical homework assignments had 4 or 5 problems on them. Problems might include building a decision tree by hand for a simple domain, showing how a case-based reasoner would work for some specific cases, or building a Bayes net for a task, including estimated probabilities. Each of these tasks followed similar examples done together in class.

Short-term Projects

Short-term projects were designed to be more in-depth than exercises. Projects might include writing a program, using existing software, or writing a research paper. There were roughly four projects per semester. The goal of each assignment was to understand some AI technique, and to analyze it, either its performance in implementation, or in comparison to human or animal analogues. It was in these assignments that I distinguished explicitly between CS and CNS students. For some projects the programming component was simple enough that all students completed the same assignment. But for others, computer science students were expected to write complete programs using an AI technique, while CNS students researched the available literature from their own disciplines and wrote a paper comparing their research to the techniques studied in class.

Below, I describe two different homework projects used in Fall 2006. The first was the first assignment of the semester, a collaborative project that all students participated in. The second project illustrates separate tasks for CS versus CNS students.

Reflex Robots: The first project of the semester asked students to program simple Lego robots, using either Mindstorm or Handyboard controllers, to act as simple reflex robots, or to use a primitive behavior-based control structure. The class was divided into groups of four, each one assigned a robot. We spent three class days in the computer lab, learning to use Interactive C by programming the robots to behave like Braitenberg robots (Braitenberg 1984). The students then continued working on reflex robots to perform line-following and light-seeking activities.

This assignment illustrated the possibilities and the limitations of pure reflex agents, but also served as an "ice-breaker" as students got to know each other and got used to the idea of working together, in class and out.

State-Space Search and Game-Playing: This project offered a chance for computer science students to work on a more sophisticated programming task than the robot assignment. They chose between implementing an A*-based path planner for a map of our building, or implementing a minimax Tic-Tac-Toe player. It offered a chance for neuroscience students to reflect on how the course topics related to psychological research. They were asked to research the literature on human Chess players, or on human route-finding and spatial reasoning, and to write a paper comparing these approaches. I provided a few sources to get them started (Werner *et al.* 1997; Wiener & Mallot 2003; Ericsson & Lehmann 1996; Hyotyniemi & Saariluoma 1999; Gobet *et al.* 2001), but they were expected to explore further on their own.

Semester Projects

The semester project allowed a student or pair of students to select a single topic of interest and to complete a significant project around it. Students explored a narrowly focused problem in detail, simulating on a small scale the process of research in AI. The project required a review of literature, implementation of a new system or use of an existing piece of software, and thorough analysis of the results. The semester project included five different stages, starting with project proposal, moving through several project milestones, and culminating with a professional-style paper and public presentation to the class.

All students in the class needed to complete the project requirements. Within those requirements, however, there was broad leeway to tailor the project to each student's ability and interests. Computer science students were encouraged to implement a system of their own design, perhaps using some existing code. CNS students were encouraged to find existing software for a problem that interested them and work with it, focusing more on the background research and analysis aspects of the project. Some CNS students have always chosen to tackle significant programming projects, but most have chosen to use software packages instead.

Figure 5 lists a small number of topics chosen from all three past versions of this AI course. They illustrate the range of choices by both CS and CNS students. Computer science students have a *tendency* to choose game-related projects, while CNS students tend to compare human and machine approaches to the same problem. However, each group demonstrates a wide range of choices.

Topic	Major
Spam recognition using NN learning	CS
Vowel/sound classification with NNs	CNS
Natural and artificial face recognition	CNS
An expert system for playing Texas Hold'Em	CS
Artificial life herding simulation	CS
Cribbage-playing agent	CS
Neural modeling for prosthetics	CNS
Designing "Capture the Flag" agents	CNS

Figure 5: Project topics and the background of students involved

Outcomes and Conclusions

Over the six years and three versions of the integrated AI course, evidence of its success has accumulated. One measure of its success is the increasing number of neuroscience students enrolling in the course: the first time 2 enrolled, next time 4, and most recently 13 enrolled. Both populations of students have done equally well in the course over time. I have anecdotal evidence of students satisfaction with the course based on course evaluations: several students from Fall 2004 made a point of thanking me for a positive experience.

Results from Fall 2006 course evaluations indicated that the course was successful in engaging students with a wide array of AI topics. Some students felt the course involved too much work: I plan on reviewing the size and number of assignments before next time, or incorporating homework problems into class-time activities. Computer science students were positive about what they learned, across the board. CNS students ranged from very satisfied to wishing for an even stronger emphasis on cognitive and neuroscience-related topics.

From my perspective, the course has achieved our goals for it. Students from both populations bring energy and enthusiasm to the class, and a rich range of topics and interests. CS students take away experience with a range of important AI techniques, and a significant project that, more often than not, turns into their senior Capstone or Honors project. CNS students are exposed to the AI perspective on "intelligence," and they understand the uses and limitations of AI techniques. I would suggest that broadening participation in AI to include students with cognitive science or neuroscience skills can enhance the learning experience for all students.

References

Braitenberg, V. 1984. *Vehicles*. The MIT Press.

Ericsson, and Lehmann. 1996. Expert and exceptional performance: Evidence of maximal adaptation to task constraints. *Annual Review of Psychology*.

Genesereth, M. R., and Nilsson, N. J. 1987. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann.

Gobet; Lane; Croker; Cheng; Jones; Oliver; and Pine. 2001. Chunking mechanisms in human learning. *TRENDS in Cognitive Sciences*.

Highlights. 2006. Highlights for children.

Hyotyniemi, and Saariluoma. 1999. Chess – beyond the rules. *Games, Computers, and People*.

Neller, T.; Markov, Z.; and Russell, I. 2006. Clue deduction: Professor plum teaches logic. In *Proceedings of the 19th International FLAIRS Conference*. Florida Artificial Intelligence Research Society.

Russell, S., and Norvig, P. 2003. *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition.

Werner; Krieg-Bruckner; Mallot; Schweizer; and Freksa. 1997. Spatial cognition: The role of landmark, route, and survey knowledge in human and robot navigation. *Informatik*.

Wiener, and Mallot. 2003. 'fine-to-coarse' route planning and navigation in regionalized environments. *Spatial Cognition and Computation*.