A.I. as an Introduction to Research Methods in Computer Science

Raghuram Ramanujan
Dept. of Mathematics and Computer Science
Davidson College, Davidson, NC 28035
raramanujan@davidson.edu

Abstract

While many computer science programs offer courses on research methods, such classes typically tend to be aimed at graduate students. In this paper, we propose a novel means for introducing undergraduate students to research experiences in computer science — via an introductory Artificial Intelligence (A.I.) course. Students explore the content areas typically covered in an upper-level A.I. course (heuristic search, constraint satisfaction, game-playing etc.), while also learning about the mechanics of how empirical research is conducted in this field.

Introduction and Background

The benefits of exposing undergraduates to research experiences are well-documented. Students become better learners, acquire stronger technical and intellectual skills, and exhibit a greater desire to pursue graduate studies (Russell, Hancock, and McCullough 2007). Students also experience personal and professional growth — they report increased self-confidence, display greater resilience in the face of failure or obstacles, and demonstrate improved communication and interpersonal skills (Kinkead 2003; Lopatto 2010). Research experiences even aid with retention efforts (Nagda et al. 1998; Peckham et al. 2007). Given the significance of these findings, providing *every* computer science undergraduate with a taste of research seems like a worthy goal. In this paper, we propose a novel means for helping achieve this outcome — using an introductory A.I. course as a vehicle for exposing students to empirical research methods in computer science.

The idea of designing an introductory course on research methods in computer science is not novel in and of itself. Indeed, several such courses exist in universities around the country — for example, David Jensen's CMPSCI 691DD (Research Methods for Empirical Computer Science) at the University of Massachusetts and Cliff Shaffer's CS5014 (Research Methods in Computer Science) at Virginia Tech. However, both those courses (and indeed, most courses of this flavor) are graduate level courses, aimed at first-year Ph.D. students.

There has been prior work in attempting to incorporate a research component into an undergraduate level A.I. course. Most notable among these is the work of Goldsmith and Mattei who suggest using science fiction as a hook for encouraging students to engage more deeply with A.I. concepts (Goldsmith and Mattei 2011). By contrast, our course focuses more on quantitative analyses — students are tasked with conducting rigorous, controlled experiments to test hypotheses with a view to answering questions such as "Does algorithm A outperform algorithm B in this domain?". Our work is closest in spirit to that of Chiu and Wallace, who describe a similar approach to teaching a course on web data management (Chiu and Wallace 2013). In their course, students are assigned open-ended projects with a system evaluation requirement. Chiu and Wallace suggest that an introductory A.I. course would be well-served by a similar format and describe two projects that they have tested at their institution. In this paper, we describe an approach that takes their suggestion to its logical extreme, namely building an entire course around tackling open-ended research projects in A.I.

Course Structure

Davidson College is a small, private, liberal-arts college with a student-faculty ratio of about 10:1. Our *Artificial Intelligence* course is an upper-level elective, that mainly draws juniors and seniors. In the spring of 2015, the class enrolled 20 students. The course was designed around bi-weekly lectures, with assessment taking the form of five student projects, spaced apart evenly. There were no other homework assignments or exams in the class. In the remainder of this paper, we describe the structure and roles of these different facets of the course in greater detail.

A.I. Content

Our course covers systematic and local search, adversarial search, constraint satisfaction problems and combinatorial optimization, classical planning, and reinforcement learning. Notably, we do not delve into machine learning or natural language processing to avoid spreading ourselves "too thin" and to accommodate other content related to research methods. Class meetings comprise a mixture of discussions centered around pre-assigned readings, lectures, and problem-modeling and problem-solving exercises.

Research Methods Content

Early in the semester, the class spends a substantial portion of a lecture discussing the question of what constitutes research. This serves as an opportunity to dispel many myths (*"no, you don't have to discover something earth-shattering for it to count as research"*) as well as to discuss the computer science research landscape (for example, how research in theory differs from research in A.I.). Discussions about the nuts and bolts of research occur on a "just-in-time" basis during the rest of the semester and are centered around the following themes.

How To Read We discuss with students the need for peer review, introduce them to tools for searching the literature (Google Scholar and CiteSeer), and deconstruct the process of reading a technical paper.

How To Write On all projects, students are primarily evaluated on the basis of written work. Students are required to compose these in LaTeX, using the AAAI template files, so that their submissions look like professional research papers. We discuss the finer points of good technical writing and how to avoid plagiarism.

How To Present Data We hold class discussions on basic statistical analysis and common pitfalls to avoid — for example, computing confidence intervals, considering when aggregate statistics other than the mean of a variable (like the median) might be more useful, control variables and parameter tuning, and visualization choices (such as when to use a table versus a plot).

Projects

The primary means of assessment in our course are open-ended projects that are completed by pairs of students. While there is a rich tradition of using open-ended problems in computer science curricula, the projects used in this course have some distinctive characteristics:

1. They require discovery. Not every concept, algorithm, or technique needed to make inroads into a problem is covered in class; it is expected that students will take the initiative to search the literature, find related work and be self-directed learners.
2. They require analysis. There is often no "right way" to approach a problem and students are required to justify the design choices they make (for example, a modeling paradigm or a parameter setting), rather than picking among their options arbitrarily. Similarly, claims about performance need to be rigorously backed up with data.
3. They require communication of results. Student are required to submit a concise and cogent report detailing the problem they worked on and the approach they adopted, and their grade is dependent on this artifact alone.

In keeping with the spirit of true scientific discourse, a free exchange of ideas is encouraged in the class. Students are encouraged to discuss and share their findings (for example, particular sources or the efficacy of a certain problem-solving approach) with their peers, but to never share written work or code. Further, we avoid stigmatizing failure and encourage students to discuss their negative results openly and honestly in their papers, and our evaluation rubric prizes the process over results. In the spring of 2015, we used the following projects in our class that offer a scaffolded learning experience — students begin with projects that require replication of published results, and gradually move towards tackling more open-ended problems.

Project 1: Reproduce table 3.29 from page 104 of Russell and Norvig's seminal A.I. textbook (Russell and Norvig 2010), to compare various A* heuristics for the sliding 8-puzzle problem.

Project 2: Perform symbolic regression using genetic programming to discover equations describing natural laws, from raw data.

Project 3: Build and evaluate agents for playing rock-paper-scissors against sub-optimal opponents.

Project 4: Using anonymized data from the Davidson College registrar's office, devise a scheme for assigning students to courses to replace the existing lottery-based method.

The course culminates in a final project where teams investigate a topic or problem of their choosing. In addition to a final paper, students are also expected to present their results at an end-of-semester poster fair, that draws attendees from across the college. Students are asked to prepare two different oral presentations — one for a general audience, and one for a more technical audience (i.e., mathematicians and computer scientists). The final set of student projects in 2015 spanned a broad spectrum of A.I. topics, from segmentation of depth images, to multi-destination path planning, and the evolution of cooperation in iterated games.

References

Chiu, D., and Wallace, S. A. 2013. On the "science" in computer science: Integrating research preparedness in undergraduate CS. *J. Comput. Sci. Coll.* 29(1):157–163.

Goldsmith, J., and Mattei, N. 2011. Science fiction as an introduction to AI research. In *Proceedings of the Second AAAI Symposium on Educational Advances in Artificial Intelligence*.

Kinkead, J. 2003. Learning through inquiry: An overview of undergraduate research. *New Directions for Teaching and Learning* 2003(93):5–18.

Lopatto, D. 2010. Undergraduate research as a high-impact student experience. *Peer Review* 12(2):27–30.

Nagda, B.; Gregerman, S.; Jonides, J.; von Hippel, W.; and Lerner, J. 1998. Undergraduate student-faculty research partnerships affect student retention. *The Review of Higher Education* 22:55–72.

Peckham, J.; Stephenson, P.; Hervé, J.-Y.; Hutt, R.; and Encarnação, M. 2007. Increasing student retention in computer science through research programs for undergraduates. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '07, 124–128. New York, NY, USA: ACM.

Russell, S. J., and Norvig, P. 2010. *Artificial Intelligence - A Modern Approach (3. ed.)*. Pearson Education.

Russell, S. H.; Hancock, M. P.; and McCullough, J. 2007. Benefits of undergraduate research experiences. *Science* 316(5824):548–549.