Model AI Assignments 2016

Todd W. Neller
Gettysburg College
tneller@gettysburg.edu

Laura E. Brown
Michigan Technological University
lebrown@mtu.edu

James B. Marshall
Sarah Lawrence College
jmarshall@sarahlawrence.edu

Lisa Torrey
St. Lawrence University
ltorrey@stlawu.edu

Nate Derbinsky
Wentworth Institute of Technology
derbinskyn@wit.edu

Andrew A. Ward
andrew.ward.cs@gmail.com

Thomas E. Allen, Judy Goldsmith, and **Nahom Muluneh**
University of Kentucky
{teal223,goldsmit}@cs.uky.edu

Abstract

The Model AI Assignments session seeks to gather and disseminate the best assignment designs of the Artificial Intelligence (AI) Education community. Recognizing that assignments form the core of student learning experience, we here present abstracts of six AI assignments from the 2016 session that are easily adoptable, playfully engaging, and flexible for a variety of instructor needs. Assignment specifications and supporting resources may be found at http://modelai.gettysburg.edu.

A Genetic Algorithm for Robby the Robot

James Marshall

In this assignment, students implement a genetic algorithm in Python to evolve cleaning strategies for Robby the Robot, as described in Chapter 9 of Melanie Mitchell's book *Complexity: A Guided Tour* (Mitchell 2009). Robby's task is to collect empty soda cans that lie scattered around his rectangular grid world, by following instructions encoded as 243-character genome strings. An easy-to-use graphical simulator written in Python is included, which avoids the need for students to write one themselves, allowing them to focus on implementing the core ideas of the GA. Robby's actions are controlled through method calls to the simulator. The simulator makes it easy to watch Robby's behavior improve dramatically over time as the GA evolves, providing students with immediate and satisfying visual confirmation that simulated evolution really works. Many potential avenues exist for experimenting with or extending the GA, such as by varying crossover and mutation rates or population size, adding elitism, trying out different selection strategies, analyzing evolved strategies by observing the effect of changing certain "genes", monitoring the amount of "genetic drift" in the population over time for certain genes, etc. This assignment can serve as a sequel to the Model AI Assignment "A Simple Genetic Algorithm".

An Introduction to k-Means Clustering

Todd W. Neller and Laura E. Brown

An informal survey of attendees at EAAI-2014 yielded the insight that unsupervised learning was a topic perceived as most in need of curricular support materials. Within unsupervised learning, clustering was deemed most important, and among clustering algorithms, k-Means Clustering was deemed most essential. We have thus developed and gathered significant resources to support the teaching of k-Means Clustering.

Included in our offering are:

- a set of programming exercises with a programming-contest style input/output specification for basic and iterated k-Means Clustering, as well as implementation of a simplified gap statistic for determining the number of clusters k,
- sample datasets that do or do not conform to the assumptions of k-Means Clustering,
- Matlab/Octave scripts for visualizing data and clustering results for 2D data,
- a short Weka tutorial illustrating feature selection for clustering,
- a PowerPoint presentation including worked examples and many visual illustrative examples,
- suggested readings for the different learning objectives with a mapping to Computer Science 2013 curricular topic areas,
- a notation translation guide for suggested readings, and

- links to online demos, videos, and relevant MOOCs.

It is our hope that instructors will pick and choose from these resources we and others have developed to meet pressing pedagogical needs in this important area of AI.

Python Console-Animation Suite

Lisa Torrey

This is a suite of exercises that apply five core AI algorithms to five classic puzzles and games within a consistent Python framework. The algorithms are breadth-first search, A* search, simulated annealing, minimax, and Q-learning. The application domains are a maze, a sliding-block puzzle, the N-queens problem, tic-tac-toe, and Pac-Man. The exercises have all been used as in-class examples or short homework assignments in an undergraduate introductory AI course. They keep students engaged by working on a range of popular problems, and by adopting a consistent approach, they reduce cognitive load. Console-based animation allows for visualization without the overhead of a graphics library. The exercises are accessible to any student who can work with classes, strings, tuples, lists, sets, and dictionaries in Python.

An Introduction to Classification: A CS2 Object-Oriented Programming Project

Nate Derbinsky

Through a series of scaffolded steps, this CS2 project asks students to develop a flexible framework for evaluating classification algorithms. In each phase of the project, students are provided documentation, unit tests, and supporting compiled code. The focus is on object-oriented concepts, such that the application can "mix and match" any classification algorithm with any training/testing-pair evaluation dataset. The purpose of the project is for the students to apply principles of object-oriented programming to a relatively large-scale, real-world problem. Along the way students learn about basic machine learning concepts; issues and tradeoffs related to knowledge representation (e.g. the efficiency of sparse vs. dense feature vectors); and implement two basic supervised learning algorithms (1-NN, ZeroR), as well as a simple form of reservoir sampling.

A Simple Genetic Algorithm

James Marshall

Students implement a simple genetic algorithm in Python to evolve binary strings of 0s and 1s. The assignment is based on an exercise from *An Introduction to Genetic Algorithms* (Mitchell 1996). Because GAs are inherently probabilistic, debugging them can prove challenging and frustrating to students. Therefore a robust automated tester program is included, which can be used by students and instructors alike to detect bugs in students' code. The assignment provides significant structure by breaking the GA implementation into 8 building blocks, corresponding to 8 functions that can be written and tested in an incremental, modular fashion. The required input/output behavior of each function is specified in the assignment description. The tester program is capable of detecting many subtle problems in students' code, including: not performing crossover entirely correctly, performing mutation with a higher or lower probability than expected, selecting genomes from the population in a way that does not correspond to their fitness, etc. Students can use the tester to receive immediate feedback as they develop their code. The tester also makes grading the assignment considerably easier for the instructor. This assignment can serve as a prequel to the Model AI Assignment "A Genetic Algorithm for Robby the Robot".

Adventures with Prolog: Entering the Dungeon Lord's Lair

Thomas E. Allen, Andrew A. Ward, Judy Goldsmith, and Nahom Muluneh

Our Artificial Intelligence survey course includes units on Propositional Logic, Predicate Logic, Situation Calculus, and Resolution. As an application of this material, we introduce Prolog. Our students, mostly undergraduate juniors and seniors, have had success with Python and C++. Often, however, they find Prolog bewildering. "How do I write a loop?" they ask us. "How do I assign X a new value?!"

We present an assignment with a theme inspired by Tolkien and "D&D" style role-playing games: Students implement a simple maze and a character that navigates through it, avoiding enemies and obstacles and searching for treasure. In the first assignment, students only implement the maze and show that their character can navigate through it in a manner that does not involve infinite recursive descent. In the second, we introduce treasure guarded by enemies. By breaking the assignment in two, and going through solutions of the first part, students are more likely to succeed in the second part, and thus feel a sense of accomplishment. The assignment is also easily adapted from semester to semester to incorporate different themes and challenges.

References

Mitchell, M. 1996. *An Introduction to Genetic Algorithms*. Cambridge, Massachusetts, USA: MIT Press / Bradford Books.

Mitchell, M. 2009. *Complexity: A Guided Tour*. New York: Oxford University Press.