CRCD in Machine Learning at the University of Central Florida Preliminary Experiences

M. Georgiopoulos (*), J. Castro (*), A. Wu (*), R. DeMara (*), E. Gelenbe (*), A. Gonzalez (*), M. Kysilka (**), M. Mollaghasemi (***)

(*) SEECS, University of Central Florida, Orlando, FL 32816
(**) Education Foundations Dept., University of Central Florida, Orlando, FL 32816
(**) Industrial Engineering Dept., University of Central Florida, Orlando, FL 32816
(407) 823-5338, michaelg@mail.ucf.edu

Categories & Subject Descriptors: I.2.6 [Computing Methodologies]: Artificial Intelligence---Learning.

General Terms: Algorithms.

Keywords: Curriculum Development, Learning, Engineering Education, Computer Science Education.

We will present work in progress on a Combined Research and Curriculum Development (CRCD) project at the University of Central Florida entitled "Advances in Machine Learning in Engineering and Science Education". The project is funded by a grant from the National Science Foundation[1] and involves the development of a curriculum model for the integration of machine learning research into science and engineering disciplines. This is a two-phase project. The first phase, which is where we are now, involves the integration of such modules into the lower level science and engineering courses, while the second phase involves the introduction of upper level machine learning courses.

We will focus on an overview of the project and its timeline, as well as report on its progress. We will then describe and demo some of the machine learning modules that we have developed at this stage of the project which have been used in the introductory computer engineering course. Some of these demos involve modules that make use of computer games to illustrate concepts of machine learning. Games can provide an effective means to demonstrate Artificial Intelligence (AI) strategies to students. Games are easier for students to comprehend and more manageable to code than complex applications, yet retain the essential features of AI mechanisms. For conveying machine learning concepts, the ancient Chinese game called Nim is ideal. By programming and tuning their own implementation of the Nim game, students can grasp and explore reinforcement learning techniques by writing a relatively manageable program. Nim is a two-player game sharing commonalities with casting lots and tic-tac-toe. A round of Nim begins with a collection of objects, typically referred to as *sticks,* placed in a pile accessible to both players. Players alternate moves retrieving 1, 2, or 3 sticks from the pile. The number of sticks removed can vary and is at the discretion of each player, but must consist of at least 1 stick, yet not exceed 3 sticks. The objective is to avoid picking up the last remaining stick in the pile. In other words, the player who picks up the final stick loses. For purposes of machine learning, the most important property of the Nim game is that there is a distinct trick to winning. In fact, two copies of Nim code played repeatedly against each other can readily learn this trick via reinforcement learning techniques. A computerized version of Nim[2] can learn this strategy by playing against a copy of itself. Suppose that the number of sticks is 17 and the maximum allowable sticks to be removed is 3. A 17x3 matrix records the state of knowledge that has been learned about the game if there are certain sticks left in the game and the player removes 1, 2 or 3 sticks. In the introductory computer science and computer engineering courses, students can readily implement simple reinforcement learning strategies in a high-level language such as C++, addressing an intellectually graspable, yet entertaining problem.

We will also demo additional modules that involve the introduction of concepts in artificial neural networks and genetic algorithms. One of the purposes of the neural network module is for the students to understand the basic concepts of a neural network including its definition, neuronal models, popular neural network structures, and their utility. Specifically the module addresses a specific type of neural architecture, the single layer perceptron. Students learn about its architecture and the perceptron learning algorithm and its modified version, the pocket algorithm[3], while at the same time learning about vectors/matrices, computing inner products, use of loops and conditional statements in the context of a machine learning topic. The module on Genetic Algorithms provides an introduction to the basic concepts and the workings of a Genetic Algorithm (GA). It consists of an interactive exercise that demonstrates the functionality of a GA by allowing students to participate in the creation and exploration of some of the basic dynamics of how a GA works.

All modules presented provide students with an introduction to some of the basic concepts of machine learning, while at the same time allowing them to learn and implement the various programming concepts and structures typically covered in the introductory computer science and computer engineering courses.

[1] This work was supported in part by a National Science Foundation grant CRDC-0203446.

ITiCSE'03, June 30–July 2, 2003, Thessaloniki, Greece
ACM 1-58113-672-2/03/0006.

[2] Shapiro,S. C., Nim Learning: A Learning Nim Player, http://www.cs.buffalo.edu/~shapiro/Courses/CSE572/nimlearn.ps

[3] Gallant, S. I., "Perceptron Based Algorithms," *IEEE Trans. on Neural Networks*, Vol. 1, No. 2, June 1990, pp. 179-191.