Computer Science Department - Rutgers University **Spring 2019**

CS 520 : Introduction to Artificial Intelligence

16:198:520

Time:	Wednesday, 3:20pm - 6:20pm
Place:	EN-B120
Instructor:	Wes Cowan
Office:	Hill Center, Office 268
E-Mail:	cwcowan at scarletmail.rutgers.edu (Instructor)
Office Hours:	Tuesdays 4-6, and by appointment.

Outline of the Course: This course is intended to provide a broad practical introduction to the concepts and methods in the field of artificial intelligence. "AI" represents a very diverse field of problems and techniques - we will focus on topics and ideas with broad application. The main focus and takeaway from this course should be the systematic representation of knowledge and the manipulation of these representations for general problem solving. If a problem can be represented systematically, any number of algorithmic techniques can be leveraged against it. Many of these algorithms will seem natural if not utterly naive - but applying them to appropriately represented and realized problems lends them power and generalizability.

The course will be divided roughly into three topics:

- Part I: Search - A Systematic Approach to Exploration and Discovery
  - Uninformed Search; Informed (Heuristic) Search; Adversarial Search; Local Search; Constraint Satisfaction; Logic and Satisfiability; Classical Planning
- Part II: Uncertainty - Representing and Coping with an Uncertain World
  - Probability; Inference; Bayesian Networks and Inference; Inference over Time; Utility and Decision Theory; Markov Decision Processes
- Part III: Learning - Adapting to and Understanding the World
  - A Model and Theory for Learning; Decision Trees; Perceptrons; Neural Networks; Non-Parametric Methods; Support Vector Machines; Statistical Learning; Clustering; Ensemble Learning:

Text:

The main text will be **Artificial Intelligence: A Modern Approach** by Norvig and Russell. The book is not strictly necessary, but is incredibly useful, especially if you plan to do future work in AI.

Prerequisites:

Students will be greatly assisted by familiarity and experience with basic algorithms and data structures, as well as a grounding in probability.

Grading:

Grades will have three components: i) projects (55%), ii) final (40%), and participation (5%, assigned at the discretion of the instructor). The grade is primarily project based, as this course is intended to give you a good practical grounding in the material and implementation is key. Additional bonus work may be assigned.

Rough Schedule of Topics

Note, this is nothing more than a tentative outline.

- Lec 1 Intro/Syllabus, Problems, Uninformed Search
- Lec 2 Heuristic Search, Adversarial Search, Local Search
- Lec 3 Logic, Satisfiability, Logical Inference
- Lec 4 Classical Planning, Probability, Inference
- Lec 5 Bayesian Networks, Inference on BNs, Exact vs Approx Inference
- Lec 6 Temporal Models, Prediction/Filtering, Smoothing/MLE
- Lec 7 Kalman/Particles, Utility/Decisions, Sequential Decisions / MDPs
- Lec 8 Solving MDPs, Models of Learning, Buffer
- Lec 9 Models of Learning, Linear Regression, Perceptrons
- Lec 10 Neural Networks, Back Propagation, CNNs and Computer Vision
- Lec 11 Buffer, GANs, RNNs, Others, Buffer
- Lec 12 Unsupervised Learning, Clustering, Kernel Methods, SVMs
- Lec 13 Statistical Learning, Ensemble Methods, NLP
- Lec 14 Ontologies, Buffer, Buffer

Summary of Projects:

There will be four main projects for this course, and they are meant to be the focus of your efforts and concerns. These are to be **group** projects, so get acquainted with each other. For each project, you are free to code in the language of your choice, so long as it is not deliberately perverse. Each consists of a significant coding portion, and a set of questions about the design and results of the code. In general, you are free to use outside libraries for supplemental parts of the projects (graphics, etc), however when it comes to the algorithms that are the focus of this course, **you will be expected to implement your own version of these algorithms in code**. Helper functions and existing libraries can be used as a scaffolding to build your code off of, but your final implementation should be your own code. You will additionally be expected to indicate the contributions of all members of your group. Particular emphasis will be placed on communicating your design and implementation ideas in a project writeup.