CS4100: Introduction to Artificial Intelligence
Fall 2022
Northeastern University

- **Instructor**: Steven Holtzen `s.holtzen@northeastern.edu`
  When emailing Steven or any of the TAs about course content, please include "CS4100" somewhere in the subject of your email.
- **Time**: 9:50AM – 11:30AM
- **Location**: Mugar Life Science Building 201
- **Instructor Office Hours**: Monday 11:45AM – 12:45PM EST at 310 West Village H
- **Teaching Assistants**
  - Alesia Chernikova `chernikova.a@northeastern.edu`
  - Harman Singh Farwah `farwah.h@northeastern.edu`
  - Jaison John `john.jai@northeastern.edu`
  - Anurag Sarkar `sarkar.an@northeastern.edu`
  - Divyanshu Sharma `sharma.divya@northeastern.edu`

  See Canvas for teaching assistant office hours.
- **Input/Output**: Important messages will be sent as Canvas Announcements; please ensure that you regularly check these messages. All assignments and projects are to be turned in on Canvas. We will have a course Piazza available here: `https://piazza.com/northeastern/fall2022/4100`. Please ask questions on Piazza: you will likely receive faster answers there, and others can benefit from answers to your questions. Questions can be asked anonymously.

1 Course Overview

This course will introduce students to the fundamentals of artificial intelligence, including the broad areas of search, decision-making under uncertainty, probabilistic graphical models (reasoning under uncertainty), and machine learning. The above topics only cover a small portion of the entirety of AI. However, by the end of the course, students will have developed a sufficiently broad set of technical tools, that will enable them to solve many real-world problems, self-learn additional techniques, and pursue further specialized courses in AI. The course material will focus on problem types, models, and algorithms. Applications will be discussed when relevant, but will not be the focus of the content. However, in the spirit of experiential learning, there will be significant opportunities for implementation and application through the programming assignments and final project.

Textbook and reference materials The following textbook is required: *Artificial Intelligence: A Modern Approach (AIMA; 4th edition)*, by Stuart Russell and Peter Norvig. This textbook serves as excellent reference material, and this course builds directly on the book's content (i.e., we will follow their notational conventions and occasionally use their exercises). The third edition is reasonably similar to the fourth edition (with different chapter/section numbers), and can be used as a substitute. This course is heavily influenced by the CS 188 course at UC Berkeley, developed by Dan Klein and Pieter Abbeel. They offer additional lecture slides, videos, and practice problems on similar topics at: `http://ai.berkeley.edu/course_schedule.html`. This course is also heavily based on prior offerings of CS4100/5100 at Northeastern, especially `https://www.ccs.neu.edu/home/rplatt/cs5100_spring2017/index.html`.

2 Prerequisites

This course will require that all students have the following background:

- Completed the course "CS3500: Object Oriented Design"
- The ability to complete the programming assignments in Python. If you have not used Python before, you must be willing to learn it.
- The course will require you to use basic probability (sets, counting, axioms of probability) and linear algebra (linear independence, standard matrix operations). If you do not have this background, you must be willing to learn it as we go.

3 Academic Honesty

Cheating and other acts of academic dishonesty will be referred to Khoury College. There are very serious penalties here, so please do not take any chances by copying any material from the Internet or from other past or present students of this course or related courses. In particular, when completing the programming assignments, it is important that you do not refer to any completed solutions that you find on the Internet. When in doubt, ask the instructor or consult the Northeastern academic honesty page here: `http://www.northeastern.edu/osccr/academic-integrity-policy/`

4 COVID-19 & Remote Policy

The instructor will follow university policies on whether or not the course is to be taught in-person. The class is assumed by default to be in-person: the instructor will make an announcement if it will not be in-person. When in-person, everyone must wear a face mask properly at all times while in the classroom. If you are not wearing a face mask, you will be asked to leave the classroom. If the class is to be remote, it will be taught online using Zoom, and a link will be available in Canvas under the "Zoom" tab.

5 Coursework & Grading Policy

See Canvas for a detailed course schedule with due dates for all graded material. This schedule is subject to change. The course will consist of the following graded material:

Type	**Frequency**	**Percent of Final Grade**
Problem Sets	About one per module	30%
Programming Assignments	4 total	20%
Exams	One	25%
Final Project	One	25%

- **Problem sets** are based on the previous week's material. Students may discuss the problems with other students, but must write up their own solutions. On each problem set, please also indicate who you discussed with (if any).
- **Programming assignments** are designed to let you see algorithms working in practice. Students should work on this by themselves. Resist the temptation to search for existing solutions – the process of implementation and debugging is critical to learning the material. We use software to detect copying any portion of the assignment from the Internet or from others in the class. If we detect cheating, you will get a zero on that assignment and we will report the incident to the University.
- **Exams** are non-cumulative and will cover all course content (including assignments and projects). The instructor will provide details on the exact content of the exams closer to the exam date. Exams will take place during the scheduled class period, and will take place on the dates on the detailed course schedule on Canvas.
- **Final project** is a self-directed project that can be done in groups of up to three students. Details are available on Canvas under the "Files" tab; see "final-project.pdf".

Late Work Policy All course assignments will be due at 11:59PM, and late work will be penalized according to the following scale:

- Less than 24 hours late: 5% penalty
- 24 – 48 hours late: 20% penalty
- More than 48 hours late: no longer accepted (at this point we will begin grading and require all work to be turned in)

If you require special accommodations or a grading extension, please email the instructor in advance of the deadline.

Grading thresholds Grades will be assigned according to the following standard letter-grade scale, where X is the final weighted percentage of total points in the course:

Range	> 93	[90,93)	[87, 90)	[83, 87)	[80, 83)	[77, 80)	[73, 77)	[70, 73)	[67, 70)	[60, 67)	<50
Grade	A	A-	B+	B	B-	C+	C	C-	D+	D	F

The instructor reserves the right to adjust final letter grades upwards in certain circumstances.

6 Learning Objectives

Module 1: Search – Sequential decision-making under certainty

- Concepts: Agents, environments, states, actions, graph search, tree search, heuristics. Further types of search problems: Constraint satisfaction, adversarial search (game tree). Properties of search algorithms: Time/space complexity, optimality, soundness, completeness
- Algorithms: BFS, DFS, IDS, UCS, Greedy search, A* search. Understand how the above algorithms are all unified by a priority queue. Further algorithms: Backtracking search, forward checking, minimax search, expectimax search
- Mastery objective: Given a sequential decision-making problem (with deterministic outcomes), formulate it as a search problem (by specifying formal components of a search problem), and solve the problem with an appropriate algorithm and heuristics (if applicable).

Module 2: Propositional Logic & Satisfiability

- Concepts: Logical encodings, reductions to propositional logic,
- Algorithms: Exhaustive search, DPLL, SAT-solvers
- Mastery objective: Given a constraint satisfaction problem, convert that problem into a logical encoding and use a SAT-solver to find a solution.

Module 3: MDPs – Sequential decision-making under uncertainty

- Concepts: Probability (expectation, conditional), utility, maximum expected utility. Markov decision process (MDP): Reward, return, value, policy, Bellman equation. Reinforcement learning (RL): Exploration vs. exploitation, model-based vs. model-free, Q-function (action-value function), temporal difference error, linear function approximation
- Algorithms: Value back-up (and-or tree), value iteration (dynamic programming), Q-learning
- Mastery objective: Given a sequential decision-making problem (with stochastic outcomes), formulate it as an MDP (by specifying formal components of an MDP), and solve the problem with an appropriate MDP/RL algorithm.

Module 4: Bayesian networks – Reasoning under probabilistic uncertainty

- Concepts: Probability (conditional, marginalization), Bayes's rule, Bayesian inference. Hidden Markov models (HMM): Belief, filtering, conditional independence, particle filtering. Bayesian networks: Representation, conditional independence assumptions, exact inference.
- Algorithms: Forward filtering, particle filtering, d-separation, variable elimination
- Mastery objective: Given an inference problem (with unobserved random variables), represent it as a graphical model (Bayesian network / HMM) with suitable conditional independence assumptions, and solve the inference query with an appropriate Bayesian inference algorithm.

Module 5: Learning – Acquisition of knowledge under uncertainty

- Concepts: Categories of machine learning, supervised learning, regression, classification Learning problem: Dataset, hypothesis, parameters, optimization, loss/error function, learning algorithm. Further concepts: Maximum likelihood, hyperparameters, model selection, train/validate/test.
- Models: Linear regression, logistic regression, (deep) neural networks. Algorithms: gradient descent, cross validation
- Mastery objective: Given a learning problem with an optimization objective, derive an appropriate learning algorithm, and understand how to apply the algorithm in practice.