Wrapup

Final Exam

- Monday, May 1, 5:30-8pm
- Either here (FJ-D) or FJ-B (to be determined)
- Cumulative, but emphasizes material post-midterm.
- Study old homework assignments, including programming projects.

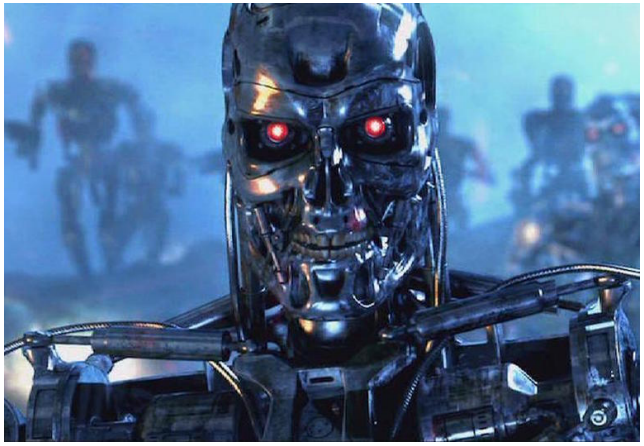
Victory Lap

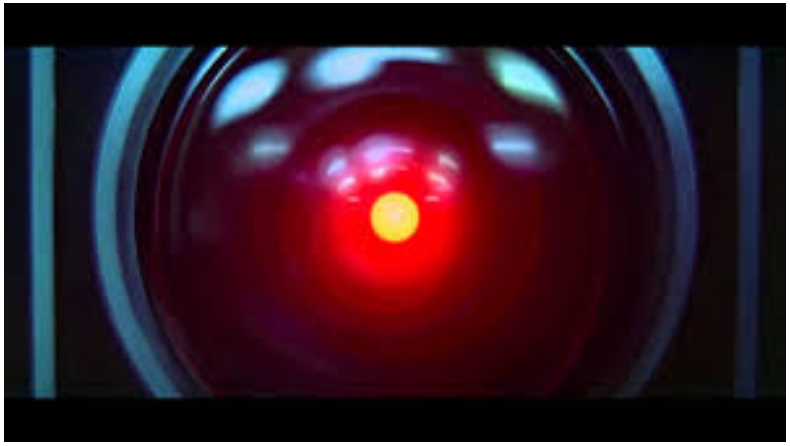
A victory lap is an
extra trip
around the track

– By the exhausted victors (us) ☺

Review course goals

– See if we met them





Artificial Intelligence

Goals

- Give you a toolbox of AI techniques.
- Show you when each technique is most appropriate.

Tools and techniques

- State space search
- Adversarial search
- Probability
- Bayes nets
- Naïve Bayes
- Hypothesis choosing (ML/MAP)
- Markov chains & hidden Markov models
- Reinforcement learning
- Neural nets

Environments

- Fully-observable vs partially-observable
- Single agent vs multiple agents
- Deterministic vs stochastic
- Episodic vs sequential
- Static or dynamic
- Discrete or continuous

Models, Inference, and Learning

- A ***model*** is an abstract way of representing a problem, including its environment, how the environment works, and the possible solutions to the problem.
  - Often includes data structures and/or mathematical relationships.
  - Examples: state spaces, game trees, Bayes nets (including Naïve Bayes classifiers, Markov chains, and HMMs), MDPs, neural networks.
- *A model is how we represent the world and how it works.*

Models, Inference, and Learning

- An ***inference algorithm*** draws conclusions or makes inferences based on the model.
  - Search (uniform cost search, greedy best first search, minimax, alpha-beta pruning), exact inference algorithm for Bayes nets, ML & MAP, inference algorithm in Markov chains, forward algorithm, backward algorithm, calculating output of neural network, value iteration.
- Inference algorithms answer questions about an existing model of the world
  - *(they don't change the model, they just use it)*

Models, Inference, and Learning

- A ***learning algorithm*** tries to deduce the structure or parameters of the model itself from auxiliary data (often examples).
  - Training a Naïve Bayes classifier by estimating the prior and feature probabilities.
  - Training a neural network by using the backpropagation algorithm to learn the weights.
  - Q-learning.
- *Learning algorithms produce or modify a model of the world.*
- (Studied further in machine learning courses.)

State Space Search

- Represent a partial solution to the problem as a “state.”
- Use an algorithms to find the “best” path through the state space.
- Pros: Often easy to formulate the model: states and actions.
- Cons: Often slow with a mediocre heuristic, state space is often too big to store explicitly in memory.
- Environment needed: Fully observable, single agent, deterministic, static.

Aside: What is a state?

- A (agent) state is an abstraction of the agent's *current* knowledge about the world.
  - In state space search, this is the set of variables describing what the agent knows at a certain time.
  - Suppose you were doing state space search by hand, and you had to stop in the middle. A friend is going to take over for you. What knowledge (separate from the environmental model) would you have to tell them to allow them to continue?

Aside: What is a state?

- You have a graph G = (V, E) and an integer n. Find a set of *n* vertices V' such that that the set of vertices either in V' or adjacent to a vertex in V' is as large as possible.
- How do you represent a state?
- How do you represent the actions?

Adversarial Search

- Still uses a “state,” only we aren’t usually interested in the entire “best” path, just the “best” next move.
- Can use minimax and alpha-beta pruning to search the game tree.
- Pros: “The” model & algorithm(s) for 2-player games.
- Cons: Can’t represent entire tree in memory, very slow for large games, still requires heuristics for deep trees.
- Environment needed: Fully observable, multi-agent (2 opponents), deterministic, static.

Probability

- Way of representing uncertainty in a model or algorithm.
- Many modern AI techniques based on rules of probability.
  - Often can give better results than heuristic approaches, where any numbers used may not be derived from any mathematical rules.
- Algorithms for ML and MAP hypothesis choosing.

Bayesian Networks

- A representation of the conditional independences that hold among a set of random variables.
- Lets you compute the probability of any event, given any observation (setting) of a set of other variables.
- Pros: Simple representation, grounded in math
- Cons: Hard to learn, exact inference can be slow, scientist must develop set of appropriate variables.

Naïve Bayes

- Particular kind of Bayes net with nice properties.
- Assumes conditional independence among all pieces of evidence/features/data.
- Useful where you need to choose a hypothesis, but don't necessarily care about the actual posterior probability (often the conditional independence assumption messes that up).
- Pros: Very simple, parameters of model easy to learn, fast algorithms for inference and learning.
- Cons: Can make gross oversimplifications, probability estimates may not be very accurate (though hypothesis often is).
- Environment needed: Fully observable, (single agent), (deterministic?), static.

Markov chains and HMMs

- Another type of Bayes net!
- Makes Markov assumption: probability distribution of next state depends only upon current state. (Sometimes called Markov property)
- Used for sequential or temporal data.
- Pros: Only model so far that takes time into account, efficient algorithms for inference and learning.
- Cons: Again, might be overly simplistic for some applications.
- Environment needed: Fully/partially observable, single agent, stochastic, static.

Reinforcement learning

- Model: MDP
- Inference: Bellman equations, value iteration
- Learning: Q-learning, lots of others…
- Pros: Simple representation, good for cases where you'll be in the same state many times.
- Cons: Slooooooooooow, must be able to get experience by repeating same situations over and over.
- Environment needed: Fully (partially) observable, single/multi agent, stochastic, static (dynamic).

All Markov Models

		Do we have control over the state transitions?	
		No	**Yes**
Are the states completely observable?	**Yes**	Markov chain	MDP (Markov decision process)
	No	HMM (Hidden Markov model)	POMDP (Partially-observable Markov decision process)

Neural networks

- Models: choice of activation function, # of hidden layers and # of nodes, what inputs look like.
- Inference: Calculating output of NN from given inputs.
- Learning: perceptron learning algorithm (single layer), backpropagation algorithm (multi-layer), all kinds of more modern algs (deep learning resurgence).
- Pros: Modern NNs are very accurate.
- Cons: can be hard or slow to train, need lots of training data.

Comparison of models

- Some model-algorithm combinations can solve “any” problem:
  - State-space search
  - (assuming fully-observable and deterministic environment)
- But often they either require
  - lots of engineering on the human’s part
  - and/or are intractable on real-world problems

Comparison of models

- Other model-algorithm combinations solve problems very quickly:
  - e.g., Naïve Bayes and HMMs
- But they only work for problems that fit the model well.

- Being good in AI involves picking the right combination of model and algorithm.

Future

- Other algorithms:
  - local search/optimization, constraint satisfaction problems, formal logic, planning, knowledge representation, so much more Bayes net/NN stuff, most of machine learning, ...
- Other application areas:
  - robotics, speech/natural language processing, computer vision, ...
- What's hot now: NNs and deep learning
- What will be hot in ten years: who knows?

What next?

- Take these ideas and use them in practice!
  - (But only where it makes sense.)

- Stay in touch
  - Tell me when this class helps you out with something cool (seriously).
  - Ask me cool AI questions (may not always know the answer, but I can tell you where to find it).
  - Don't be a stranger: let me know how the rest of your time at Rhodes (and beyond!) goes… I really do like to know.