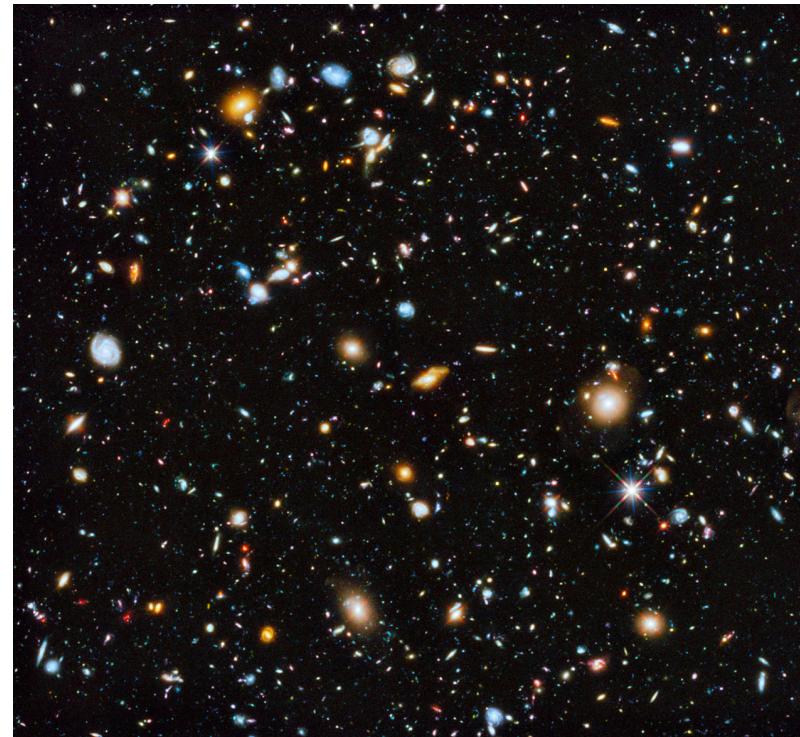




Lecture 1: Overview



Teaching staff

Percy Liang

Kelvin Guu (guest instructor)	Daniel Selsam (head CA)	Gregory Soh
Aaron Effron	Ajay Sohmshetty	Anand Dhoot
Anvita Gupta	Benoit Zhou	Chinmayee Shah
Chuma Kabaghe	David Golub	Andrew Han
Jacob Hoffman	Jaebum Lee	Megha Srivastava
Richard Diehl Martinez	Sudarshan Seshadri	Stephanie Dong
Vivekkumar Patel	Hao Wang	William Bakst
Yilong Li	Zhangyuan Wang	Steven Diamond
Kaiyi (Alexander) Fu	Amita Kamath	Yongshang Wu
Yianni Laloudakis	Benjamin Petit	Vivian Hsu



Roadmap

What is AI?

Course overview

Course logistics

Optimization



Microsoft creates AI that can read a document and answer questions about it as well as a person

January 15, 2018 | Allison Linn



Microsoft researchers achieve new conversational speech recognition milestone

August 20, 2017 | By Xuedong



June 24, 2014

DeepFace: Closing the Performance Gap in Face Recognition

Conference on Computer Vision and Pattern Recognition (CVPR)

By: Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf

Abstract

In modern face recognition, the conventional pipeline consists of alignment, feature extraction, and classification. We revisit both the alignment step and the representation modeling in order to apply a piecewise affine transformation to each facial region. This deep network involves multiple locally connected layers without weight sharing, rather than a single fully connected layer. We trained it on the largest facial dataset to-date, an identity dataset containing more than 4,000 identities.

If you think AI will never replace radiologists—you may want to think again

May 14, 2018 | Michael Walter | Artificial Intelligence

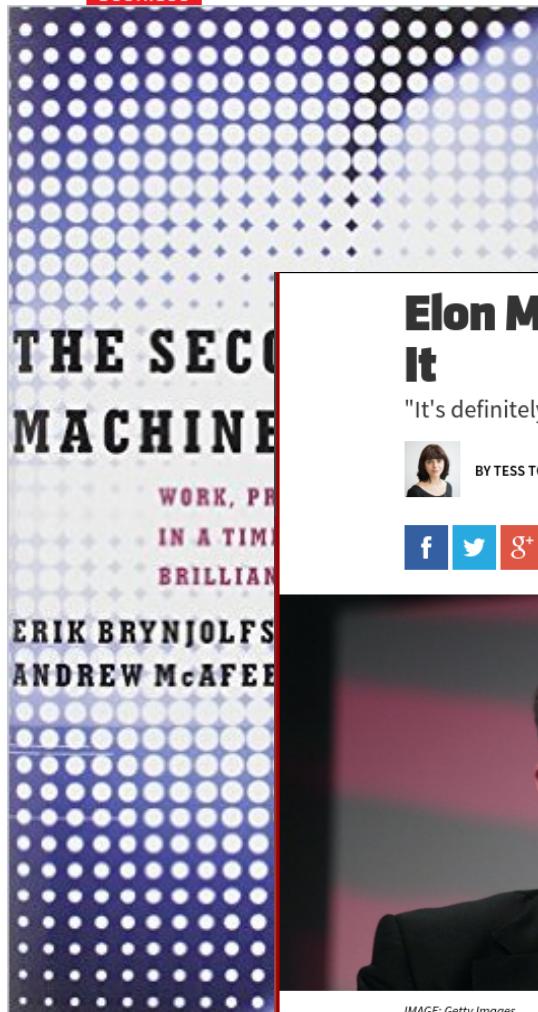


It's one of the most frequently discussed questions in radiology today: What kind of long-term impact will artificial intelligence (AI) have on radiologists?

Robert Schier, MD, a radiologist for RadNet, shared his own thoughts on the topic in a [new commentary](#) published by the *Journal of the American College of Radiology*—and he's not quite as optimistic as some of his colleagues throughout the industry.

- It is hard these days to escape hearing about AI — in the news, on social media, in cafe conversations. We see both reports triumphs of superhuman performance in games such as Jeopardy! (IBM Watson, 2011) and Go (DeepMind's AlphaGo, 2016), as well as on benchmark tasks such as reading comprehension, speech recognition, face recognition, and medical imaging (though it is important to realize that these are about performance on one benchmark, which is a far cry from the general problem).

BUSINESS



The advances we'

humanoid robots, speech recognition and sy
Jeopardy!-champion computers—are not the

IMAGE: Getty Images

Elon Musk has emerged as a leading voice in speaking out on the potential [dangers](#) of artificial intelligence, going so far as to call it the “biggest existential threat” to

Technology

Stephen Hawking warns artificial intelligence could end mankind

By Rory Cellan-Jones
Technology correspondent

2 December 2014 | Technology | [Comment](#)



Take a video tour of Facebook's Frank Gehry-Designed New York City Office

HIT THE ROAD



- We also see speculation about the future: that it will bring about sweeping societal change due to automation, resulting in massive job loss, not unlike the industrial revolution, or that AI could even surpass human-level intelligence and seek to take control.

Companies

 "An important shift from a mobile first world to an AI first world" [CEO Sundar Pichai @ Google I/O 2017]

 Created AI and Research group as 4th engineering division, now 8K people [2016]

 Created Facebook AI Research, Mark Zuckerberg very optimistic and invested

Others: IBM, Amazon, Apple, Uber, Salesforce, Baidu, Tencent, etc.

Governments



"AI holds the potential to be a major driver of economic growth and social progress" [White House report, 2016]



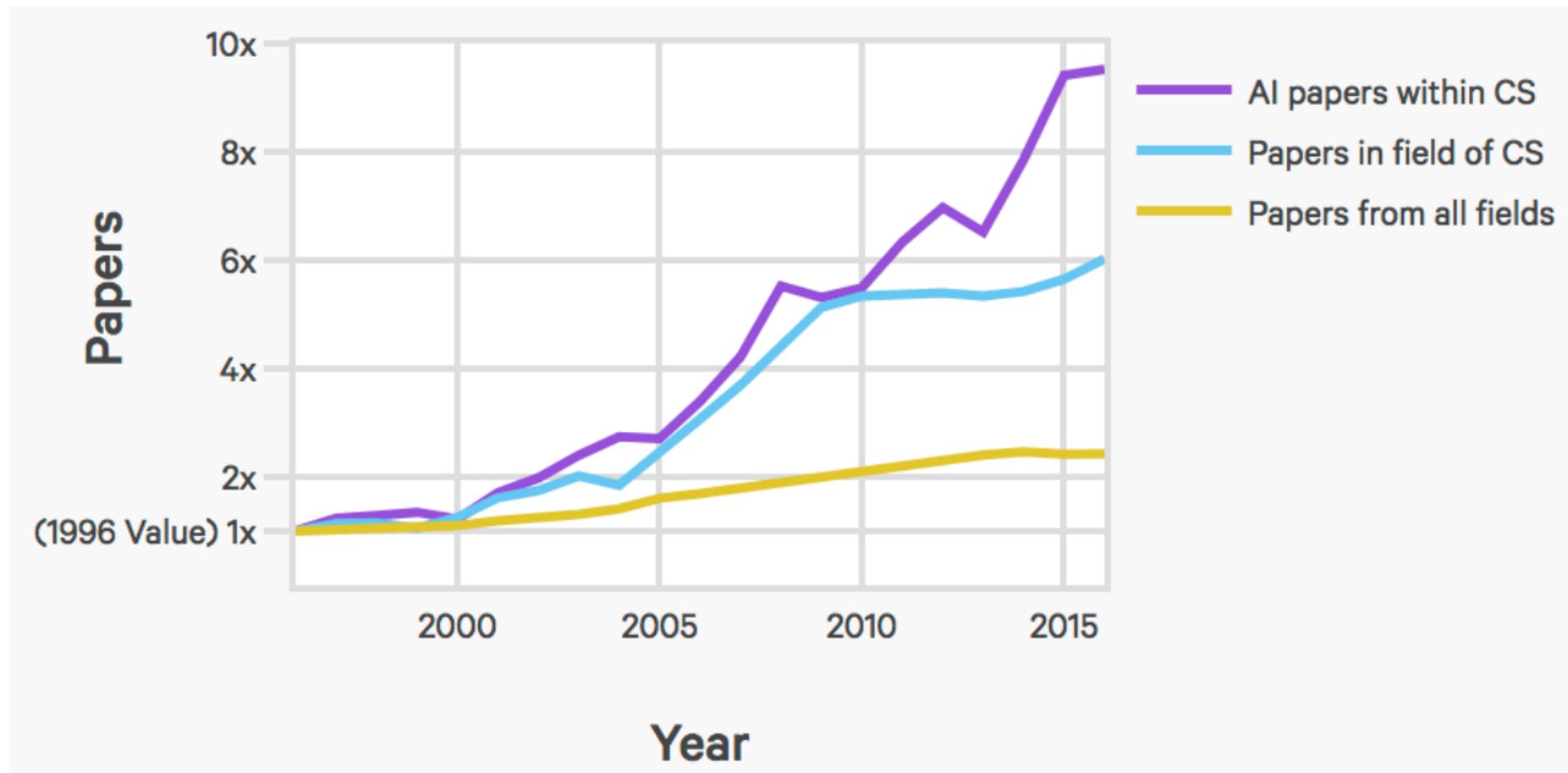
Released domestic strategic plan to become world leader in AI by 2030 [2017]



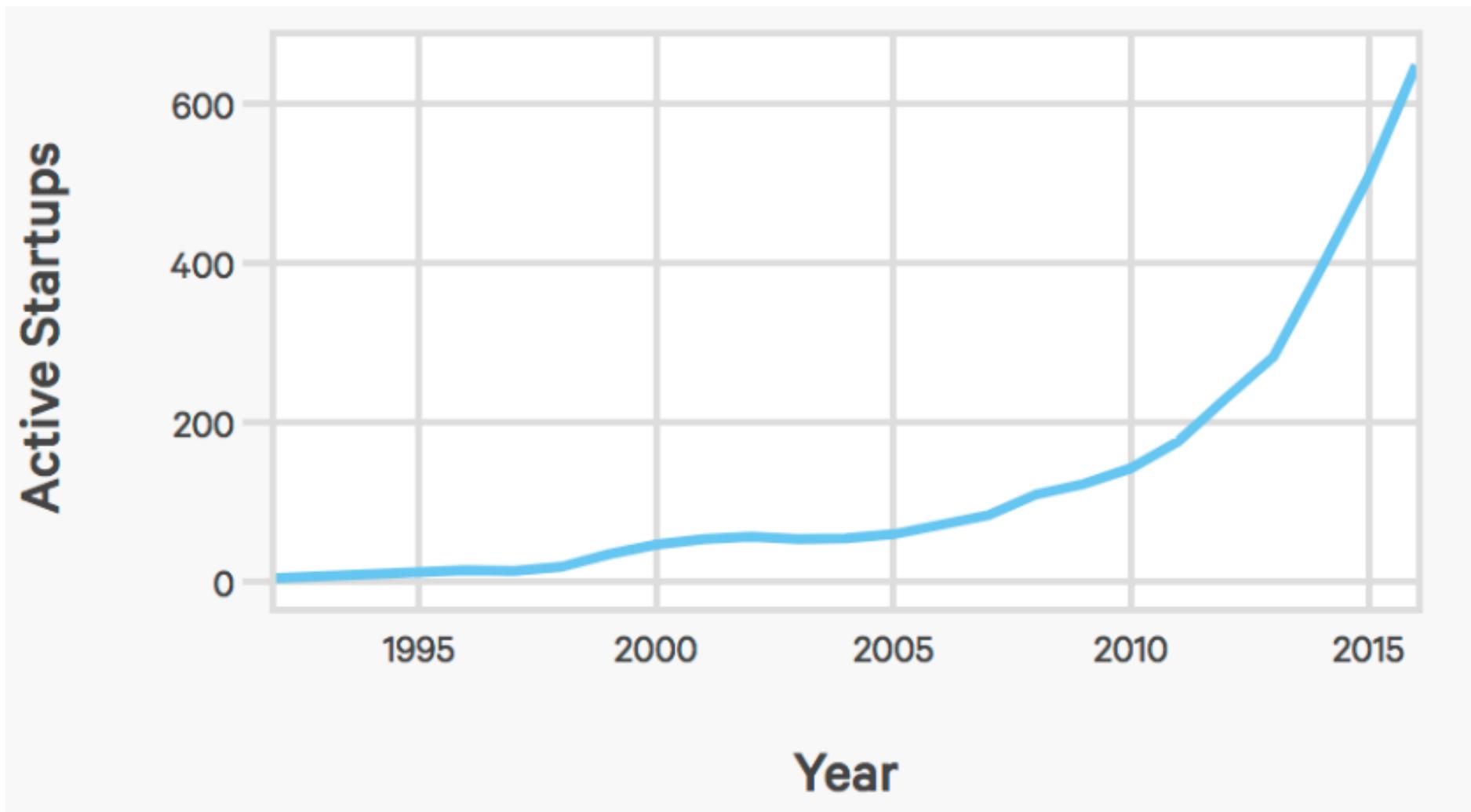
"Whoever becomes the leader in this sphere [AI] will become the ruler of the world" [Putin, 2017]

- While media hype is real, it is true that both companies and governments are heavily investing in AI. Both see AI as an integral part of their competitive strategy.

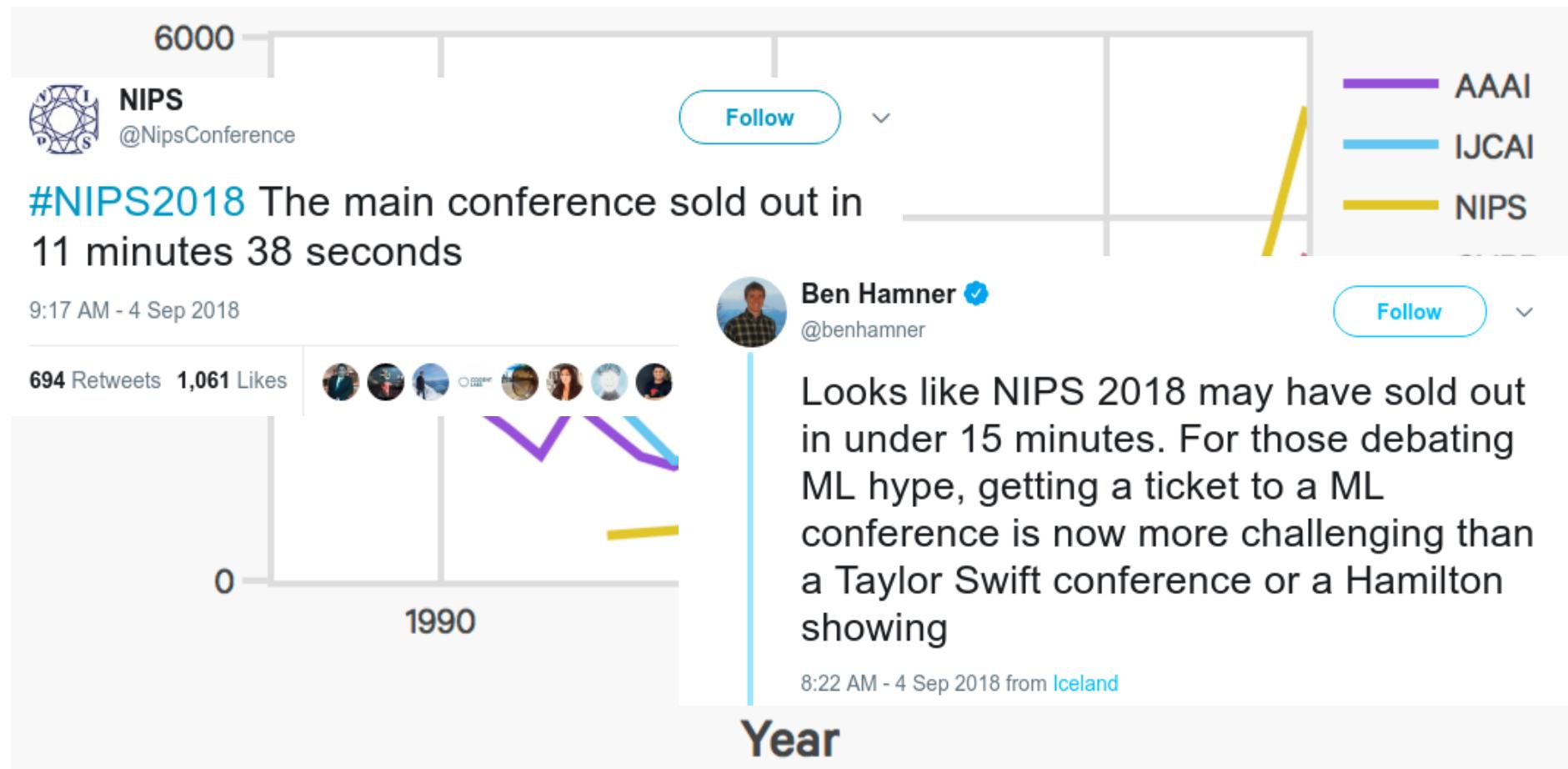
AI index: number of published AI papers



AI index: number of AI startups

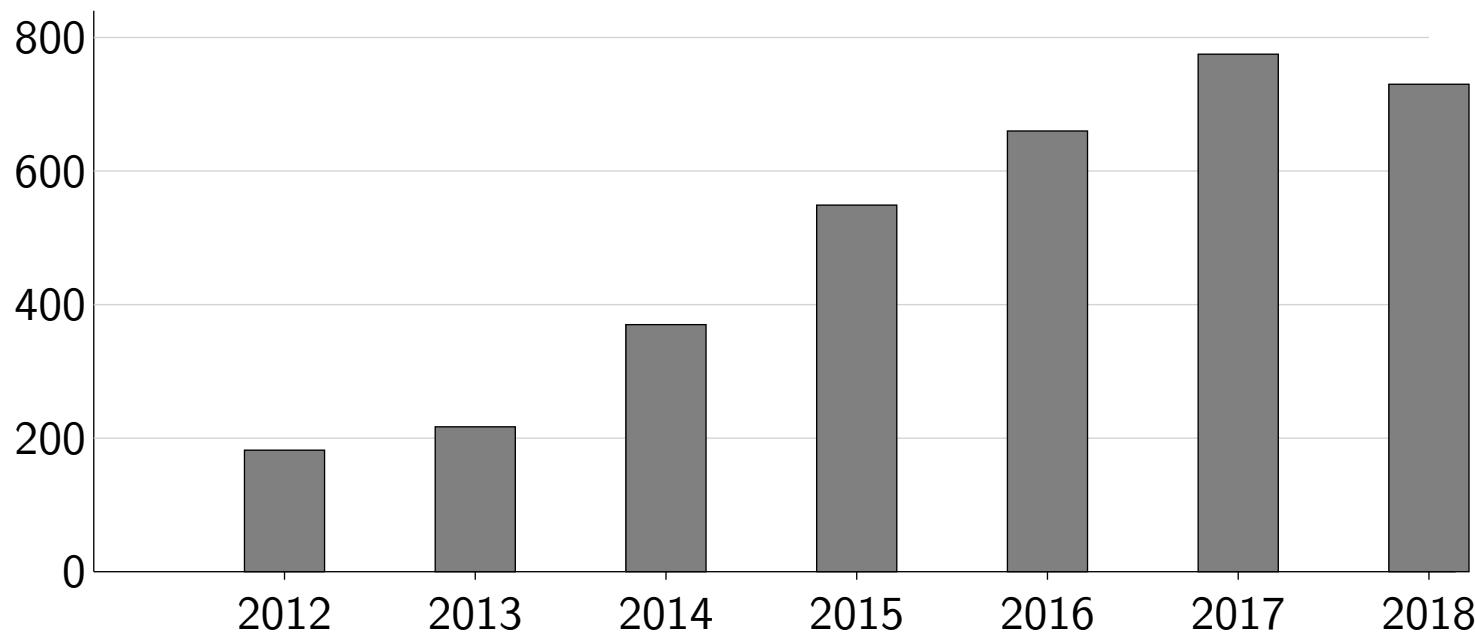


AI index: AI conference attendance



- The AI Index is an effort to track the progress of AI over time. In 2017, the AI Index published a report, showing essentially that all curves go up and to the right. Here are a few representative samples.

CS221 enrollments



Slowing down? Probably due to the CS221 spring offering...

- The reality is that there is a lot of uncertainty over what will happen, and there is a lot of nuance that's missing from these stories about what AI is truly capable of. The goal of this class is to help you understand these nuances, so that you can form your own opinion.

Ok, really, what is AI?

Two views of AI



AI agents: how can we re-create intelligence?



AI tools: how can we benefit society?

- There are two ways to look at AI philosophically.
- The first is what one would normally associate with the AI: the science and engineering of building "intelligent" agents. The inspiration of what constitutes intelligence comes from the types of capabilities that humans possess: the ability to perceive a very complex world and make enough sense of it to be able to manipulate it.
- The second views AI as a set of tools. We are simply trying to solve problems in the world, and AI techniques happen to be quite useful for that.
- However, both views boil down to many of the same day-to-day activities (e.g., collecting data and optimizing a training objective), the philosophical differences do change the way AI researchers approach and talk about their work. And moreover, the conflation of these two can generate a lot of confusion.



AI agents...

An intelligent agent

Perception

Robotics

Language



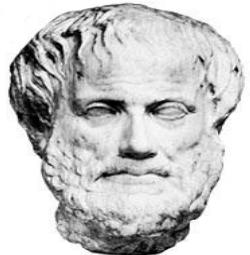
Knowledge

Reasoning

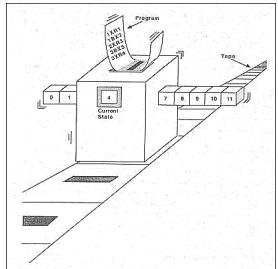
Learning

- The starting point for the agent-based view is ourselves.
- As humans, we have to be able perceive the world (computer vision), perform actions in it (robotics), and communicate with other agents.
- We also have knowledge about the world (from how to ride a bike to knowing the capital of France), and using this knowledge we can draw inferences and make decisions.
- Finally, last but not least, we learn and adapt over time. Indeed machine learning has become the primary driver of many of the AI applications we see today.

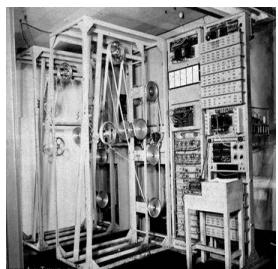
Pre-AI developments



Philosophy: **intelligence** can be achieved via mechanical computation (e.g., Aristotle)



Church-Turing thesis (1930s): any computable function is **computable** by a Turing machine



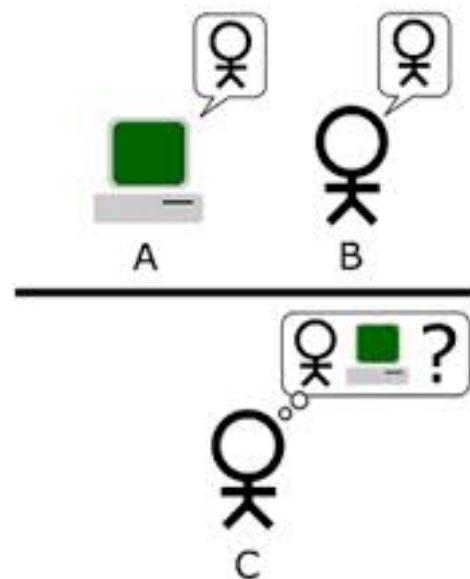
Real computers (1940s): actual **hardware** to do it: Heath Robinson, Z-3, ABC/ENIAC

- Why might one even think that it is even possible to capture this rich behavior?
- While AI is a relatively young field, one can trace back some of its roots back to Aristotle, who formulated a system of syllogisms that capture the reasoning process: how one can mechanically apply syllogisms to derive new conclusions.
- Alan Turing, who laid the conceptual foundations of computer science, developed the Turing machine, an abstract model of computation, which, based on the Church-Turing thesis, can implement any computable function.
- In the 1940s, actual devices that could actually carry out these computations started emerging.
- So perhaps one might be able to capture intelligent behavior via a computer. But how do we define success?

The Turing Test (1950)

[Turing, 1950. Computing Machinery and Intelligence]

"Can machines think?"



Q: Please write me a sonnet on the subject of the Forth Bridge.

A: Count me out on this one. I never could write poetry.

Q: Add 34957 to 70764.

A: (Pause about 30 seconds and then give as answer) 105621.

Tests behavior — simple and objective

- Can machines think? This is a question that has occupied philosophers since Descartes. But even the definitions of "thinking" and "machine" are not clear. Alan Turing, the renowned mathematician and code breaker who laid the foundations of computing, posed a simple test to sidestep these philosophical concerns.
- In the test, an interrogator converses with a man and a machine via a text-based channel. If the interrogator fails to guess which one is the machine, then the machine is said to have passed the Turing test. (This is a simplification but it suffices for our present purposes.)
- Although the Turing test is not without flaws (e.g., failure to capture visual and physical abilities, emphasis on deception), the beauty of the Turing test is its simplicity and objectivity. It is only a test of behavior, not of the internals of the machine. It doesn't care whether the machine is using logical methods or neural networks. This decoupling of what to solve from how to solve is an important theme in this class.

Birth of AI (1956)

Workshop at Dartmouth College; attendees: John McCarthy, Marvin Minsky, Claude Shannon, etc.



Aim for **general principles**:

Every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it.

A very brief history

- 1956: Dartmouth workshop, John McCarthy coined "AI"
- 1960: checkers playing program, Logical Theorist
- 1966: ALPAC report cuts off funding for translation
- 1974: Lighthill report cuts off funding in UK
- 1970-80s: expert systems (XCON, MYCIN) in industry
- 1980s: Fifth-Generation Computer System (Japan); Strategic Computing Initiative (DARPA)
- 1987: collapse of Lisp market, government funding cut
- 1990-: rise of machine learning
- 2010s: heavy industry investment in deep learning

- AI started out with a bang. People were ambitious and tried to develop things like General Problem Solver that could solve anything. Despite some successes, certain tasks such as machine translation were complete failures, which lead to the cutting of funding and the first AI winter. It happened again in the 1980s, this time with expert systems, though the aims were scoped more towards industrial impact. But again, expectations exceeded reality, leading to another AI winter. During these AI winters, people eschewed the phrase "artificial intelligence" as not to be labeled as a hype-driven lunatic.
- In the latest rebirth, we have new machine learning techniques, tons of data, and tons of computation. So each cycle, we are actually making progress. Will this time be different?
- We should be optimistic and inspired about the potential impact that advances in AI can bring. But at the same time, we need to be grounded and not be blown away by hype. This class is about providing that grounding, showing how AI problems can be treated rigorously and mathematically. After all, this class is called "Artificial Intelligence: Principles and Techniques".

2015 DARPA Robotics Challenge



- In the 2015 DARPA Robotics Challenge (DRC), robots were asked to perform a series of eight tasks motivated by a disaster relief scenario (e.g., getting out of a car and opening a door).
- While some teams did manage to successfully complete these tasks, many also failed spectacularly. One can certainly find videos of robots moving with considerable more grace, which is possible in controlled situations, but in unstructured environments, it is much harder.

Open-domain dialogue

A: How old are you?

B: I'm 16. Why are you asking?

A: I thought you were 12.

B: What made you think so?

A: I don't know what you are talking about.

B: You don't know what you are saying.

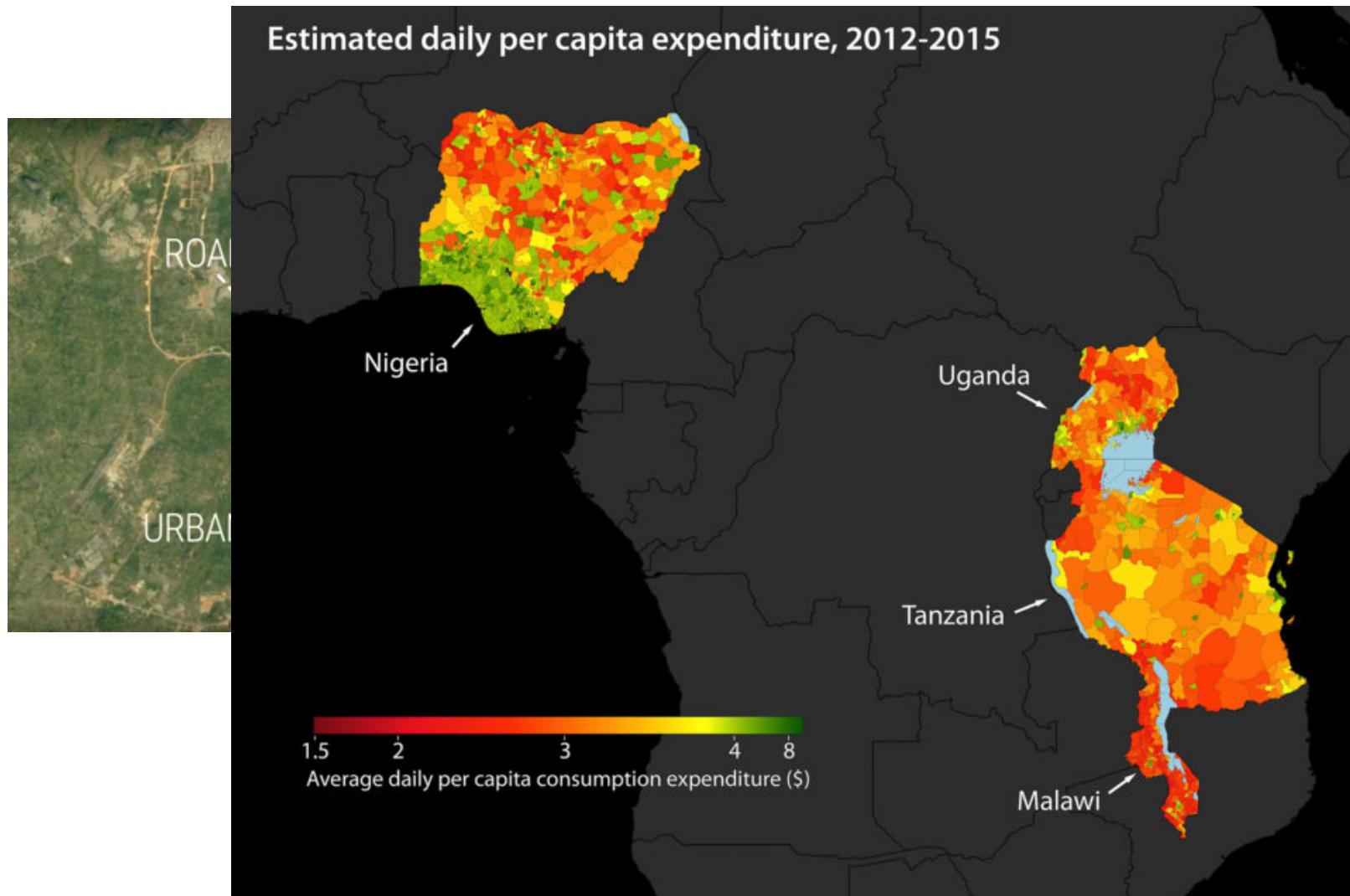
- We still don't have computers that we have a natural conversation with. While training deep neural networks on huge amounts of data has worked beautifully for speech recognition and machine translation, recent attempts in open-domain dialogue have not produced such sensible results. Models get confused by the sheer complexity of dialogue and often fall back to generic responses as shown here.



AI tools...

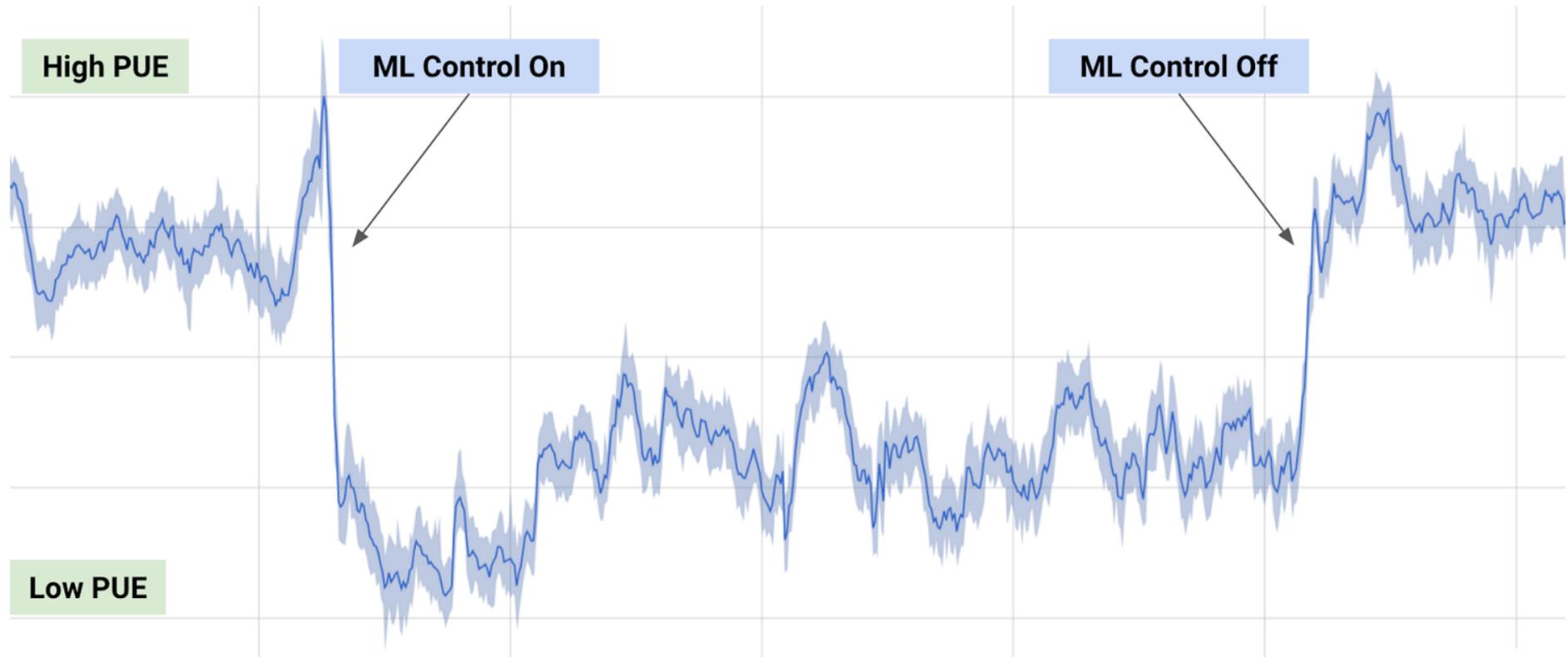
- The AI agents view is an inspiring quest to undercover the mysteries of intelligence and tackle the tasks that humans are good at. While there has been a lot of progress, we still have a long way to go along some dimensions: for example, the ability to learn quickly from few examples or the ability to perform commonsense reasoning.
- At the same time, the current level of technology is already being deployed widely in practice. These settings are often not particularly human-like (targeted advertising, news or product recommendation, web search, supply chain management, etc.)

Predicting poverty

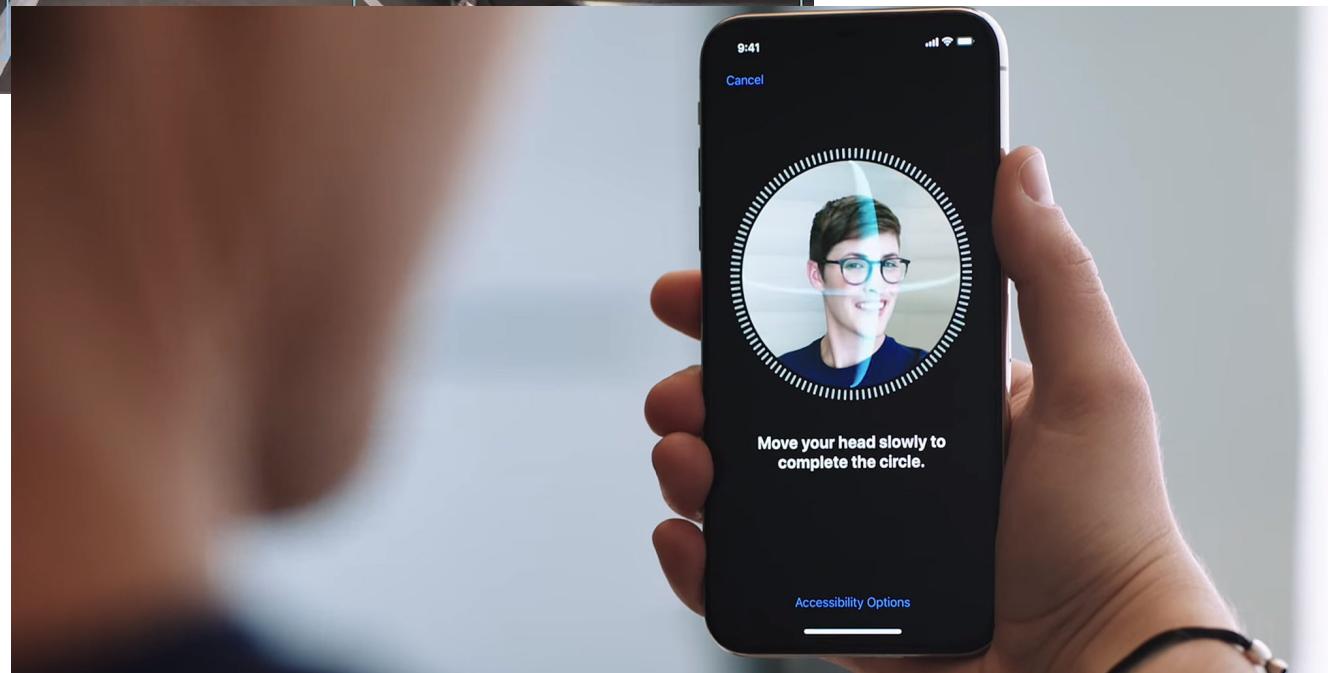
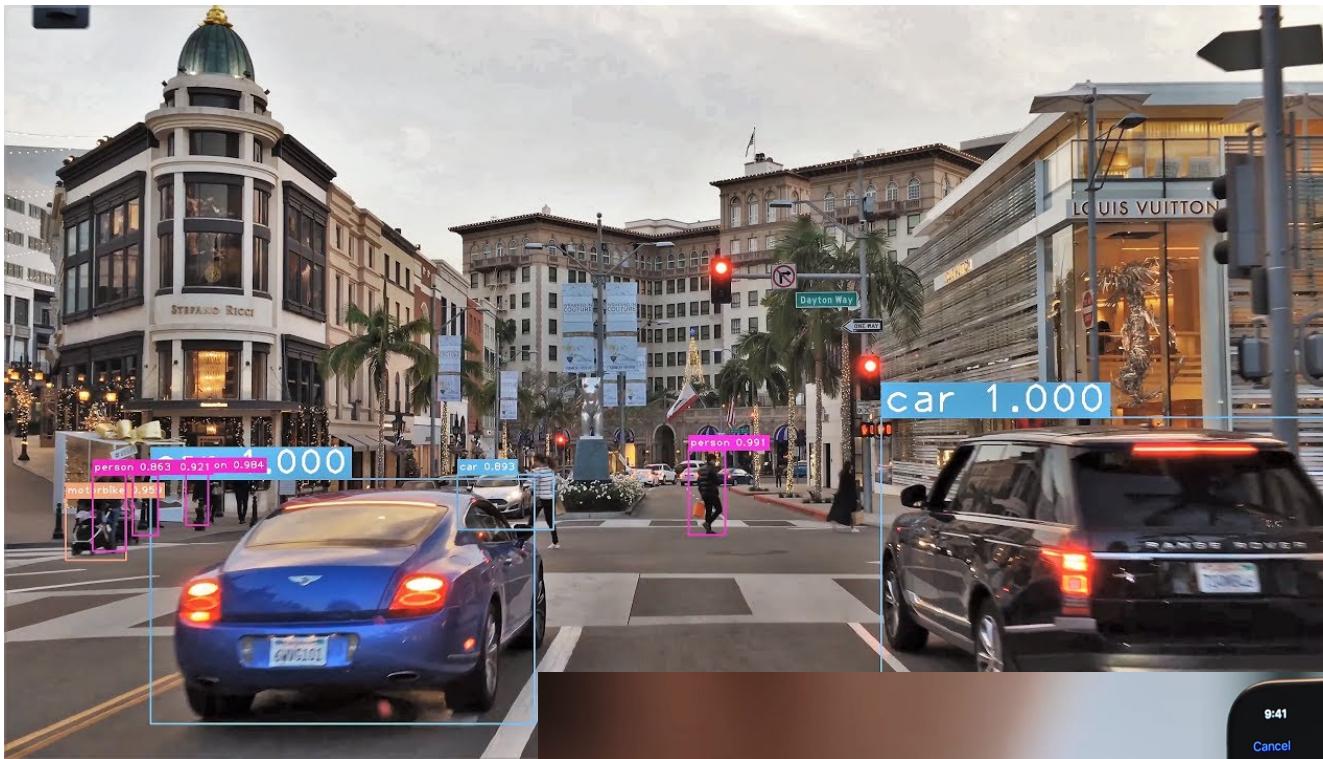


- The same computer vision techniques used to recognize objects can be used to tackle social problems. Poverty is a huge problem, and even identifying the areas of need is difficult due to the difficulty in getting reliable survey data. Recent work has shown that one can take satellite images (which are readily available) and predict various poverty indicators.

Saving energy by cooling datacenters



- Machine learning can also be used to optimize the energy efficiency of datacenters, which given the hunger for compute these days makes a big difference. Some recent work from DeepMind show how to significantly reduce Google's energy footprint by using machine learning to predict the power usage effectiveness from sensor measurements such as pump speeds, and using that to drive recommendations.

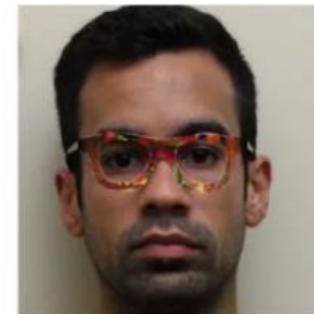


Security

[Evtimov+ 2017]



[Sharif+ 2016]



- Other applications such as self-driving cars and authentication have high-stakes, where errors could be much more damaging than getting the wrong movie recommendation. These applications present a set of security concerns.
- One can generate so-called **adversarial examples**, where by putting stickers on a stop sign, one can trick a computer vision system to mis-classify it as a speed limit sign. You can also purchase special glasses that fool a system to thinking that you're a celebrity.
- Even more fundamentally, these examples shows that current methods clearly are not learning "the right thing" as defined by the human visual system.

Bias in machine translation

The screenshot shows a machine translation interface with two columns. The left column is for Malay input, and the right column is for English output. The Malay input includes a dropdown menu set to "Malay - detected", a microphone icon, a double-headed arrow icon, and another dropdown menu set to "English". The English output includes a copy icon and a speaker icon.

Malay - detected	English
Dia bekerja sebagai jururawat.	She works as a nurse.
Dia bekerja sebagai pengaturcara. <small>Edit</small>	He works as a programmer.

society \Rightarrow data \Rightarrow predictions

Fairness in criminal risk assessment

- Northpointe: COMPAS predicts criminal risk score (1-10)
- ProPublica: given that an individual did not reoffend, blacks 2x likely to be (wrongly) classified 5 or above
- Northpointe: given a risk score of 7, 60% of whites reoffended, 60% of blacks reoffended

California just replaced cash bail with algorithms

By [Dave Gershman](#) • September 4, 2018



- A more subtle case is the issue of bias. One might naively think that since machine learning algorithms are based on mathematical principles, that they are somehow objective. However, machine learning predictions come from the training data, and the training data comes from society, so any biases in society are reflected in the data and propagated to predictions. The issue of bias is a real concern when machine learning is used to decide whether an individual should receive a loan or get a job.
- Unfortunately, the problem of fairness and bias is as much of a philosophical one as it is a technical one. There is no obvious "right thing to do", and it has even been shown mathematically it is impossible for a classifier to satisfy three reasonable fairness criteria (Kleinberg et al., 2016).



Question

What inspires you more?

Building agents with human-level intelligence

Developing tools that can benefit society



Summary so far

- AI dream of achieving human-level intelligence is ongoing
- Still lots of open research questions
- AI is having huge societal impact
- Need to think carefully about real-world consequences



Roadmap

What is AI?

Course overview

Course logistics

Optimization

How do we solve AI tasks?



```
# Data structures for supporting uniform cost search.
class PriorityQueue:
    def __init__(self):
        self.DONE = -100000
        self.heap = []
        self.stateToPriority = {} # Map from state to priority

    # Insert (state, prio) into the heap with priority (and�priorities). If
    # insert (state, p) in the heap or (newPriority) is smaller than the existing
    # priority,
    # then update the priority queue and update.
    def add(self, state, newPriority):
        oldPriority = self.stateToPriority.get(state)
        if oldPriority <= newPriority:
            self.stateToPriority[state] = newPriority
            heapq.heappush(self.heap, (newPriority, state))
        else:
            return False

    # Return (state with minimum priority, priority)
    def get(self):
        while len(self.heap) > 0:
            priority, state = heapq.heappop(self.heap)
            if priority == self.DONE:
                continue # Undeleted priority, skip
            self.stateToPriority[state] = self.DONE
            return (state, priority)
        return (None, None) # Nothing left...
```

```
# Simple examples of search problems to test your code for Problem 1.
# A simple rock-paper-scissors problem.
# From state 1, you can win to 2 or lose to 3 to move down, 2 to move up.
class NumberSearchProblem:
    def __init__(self):
        self.stateToPriority = {}
        self.stateToCost = {}
        self.stateToAction = {}

    def add(self, state, priority):
        self.stateToPriority[state] = priority
        self.stateToCost[state] = 0
        self.stateToAction[state] = None

    def moveUp(self, state):
        return [(('Up', state+1), 10, 'Up')]

    def moveDown(self, state):
        return [(('Down', state-1), 10, 'Down')]

    def win(self, state):
        return [(('Win', state+1), 10, 'Win')]

    def lose(self, state):
        return [(('Lose', state-1), 10, 'Lose')]

    def calculateCost(self, state):
        return self.stateToCost[state]

    def calculateAction(self, state):
        return self.stateToAction[state]
```

- How should we actually solve these AI tasks? The real world is complicated. At the end of the day, we need to write some code (and possibly build some hardware too). But there is a huge chasm.

Paradigm

Modeling

Inference

Learning

- In this class, we will adopt the **modeling-inference-learning** paradigm to help us navigate the solution space. In reality, the lines are blurry, but this paradigm serves as an ideal and a useful guiding principle.

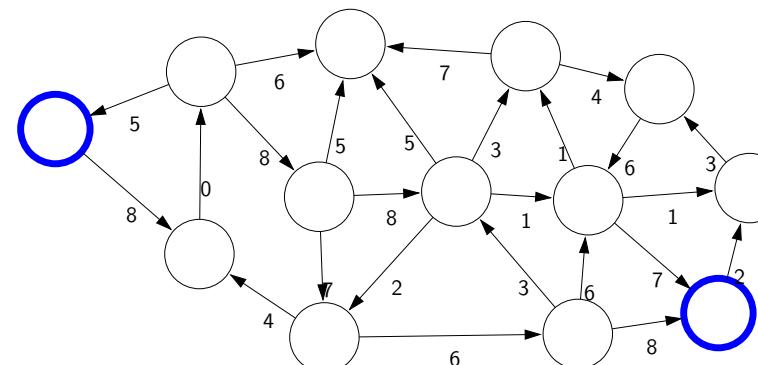
Paradigm: modeling

Real world



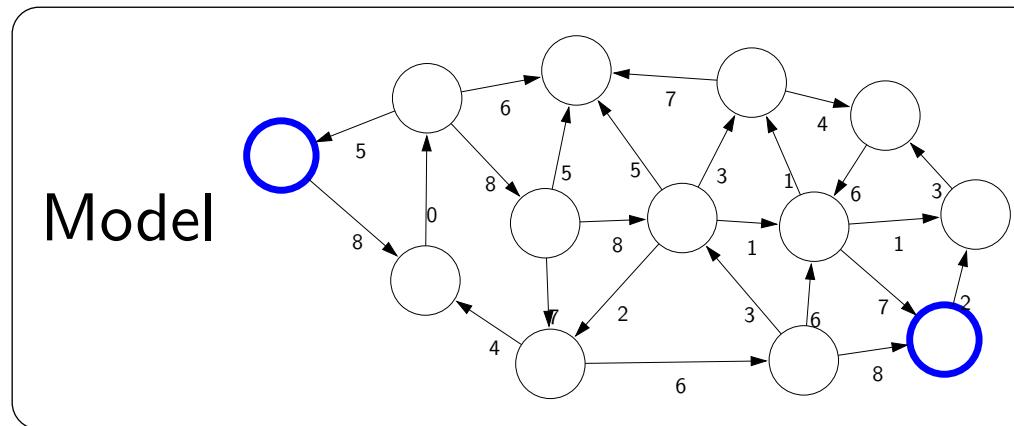
Modeling

Model

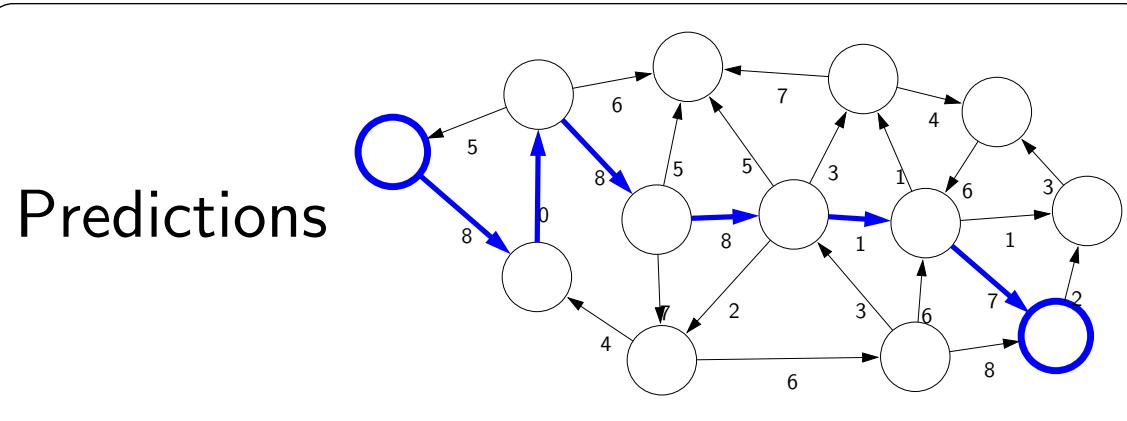


- The first pillar is modeling. Modeling takes messy real world problems and packages them into neat formal mathematical objects called **models**, which can be subject to rigorous analysis but is more amenable to what computers can operate on. However, modeling is lossy: not all of the richness of the real world can be captured, and therefore there is an art of modeling: what does one keep versus ignore? (An exception to this is games such as Chess or Go or Sodoku, where the real world is identical to the model.)
- As an example, suppose we're trying to have an AI that can navigate through a busy city. We might formulate this as a graph where nodes represent points in the city.

Paradigm: inference



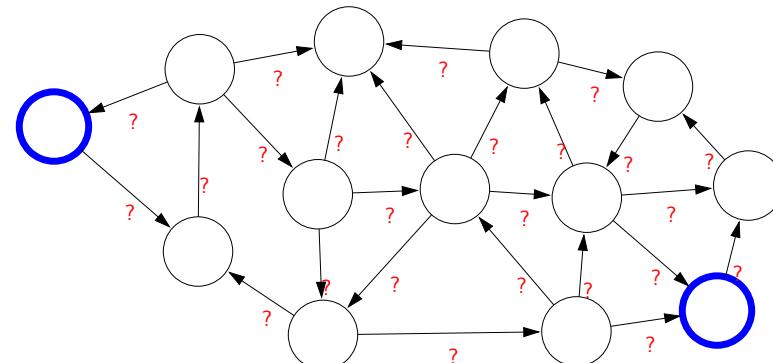
Inference



- The second pillar is inference. Given a model, the task of **inference** is to answer questions with respect to the model. For example, given the model of the city, one could ask questions such as: what is the shortest path? what is the cheapest path?
- For some models, computational complexity can be a concern (games such as Go), and usually approximations are needed.

Paradigm: learning

Model without parameters

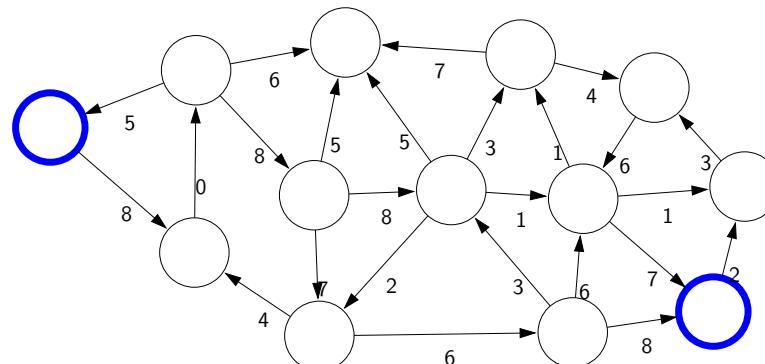


+data

Learning



Model with parameters



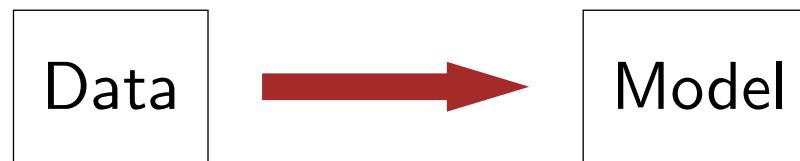
- But where does the model come from? Remember that the real world is rich, so if the model is to be faithful, the model has to be rich as well. But we can't possibly write down such a rich model manually.
- The idea behind (machine) **learning** is to instead get it from data. Instead of constructing a model, one constructs a skeleton of a model (more precisely, a model family), which is a model without parameters. And then if we have the right type of data, we can run a machine learning algorithm to tune the parameters of the model.

Course plan



- We now embark on our tour of the topics in this course. The topics correspond to types of models that we can use to represent real-world tasks. The topics will in a sense advance from low-level intelligence to high-level intelligence, evolving from models that simply make a reflex decision to models that are based on logical reasoning.

Machine learning



- The main driver of recent successes in AI
- Move from "code" to "data" to manage the information complexity
- Requires a leap of faith: **generalization**

- Supporting all of these models is **machine learning**, which has been arguably the most crucial ingredient powering recent successes in AI. Conceptually, machine learning allows us to shift the information complexity of the model from code to data, which is much easier to obtain (either naturally occurring or via crowdsourcing).
- The main conceptually magical part of learning is that if done properly, the trained model will be able to produce good predictions beyond the set of training examples. This leap of faith is called **generalization**, and is, explicitly or implicitly, at the heart of any machine learning algorithm. This can even be formalized using tools from probability and statistical learning theory.

Course plan

Reflex

"Low-level intelligence"

"High-level intelligence"

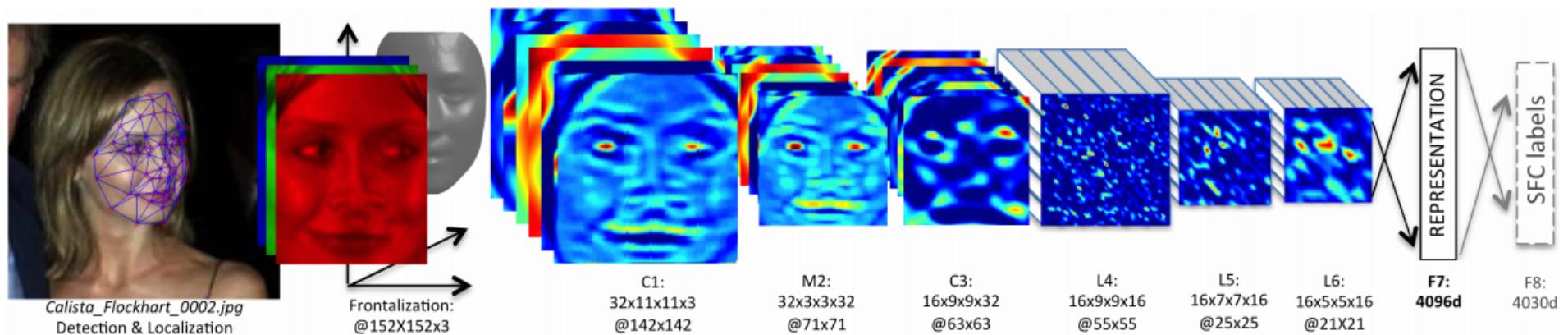
Machine learning

What is this animal?



Reflex-based models

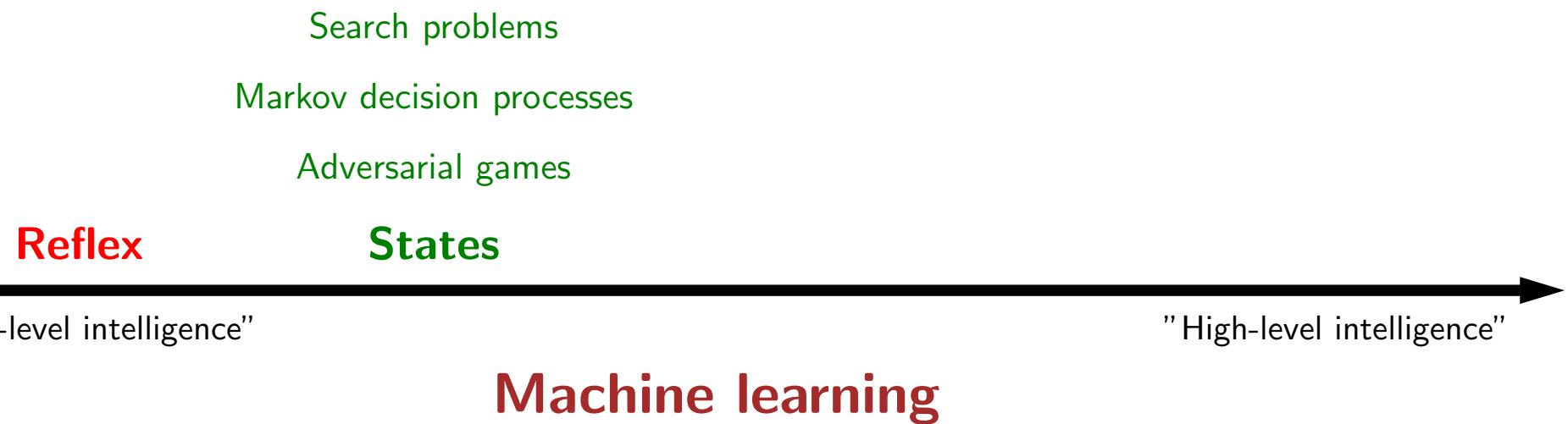
- Examples: linear classifiers, deep neural networks



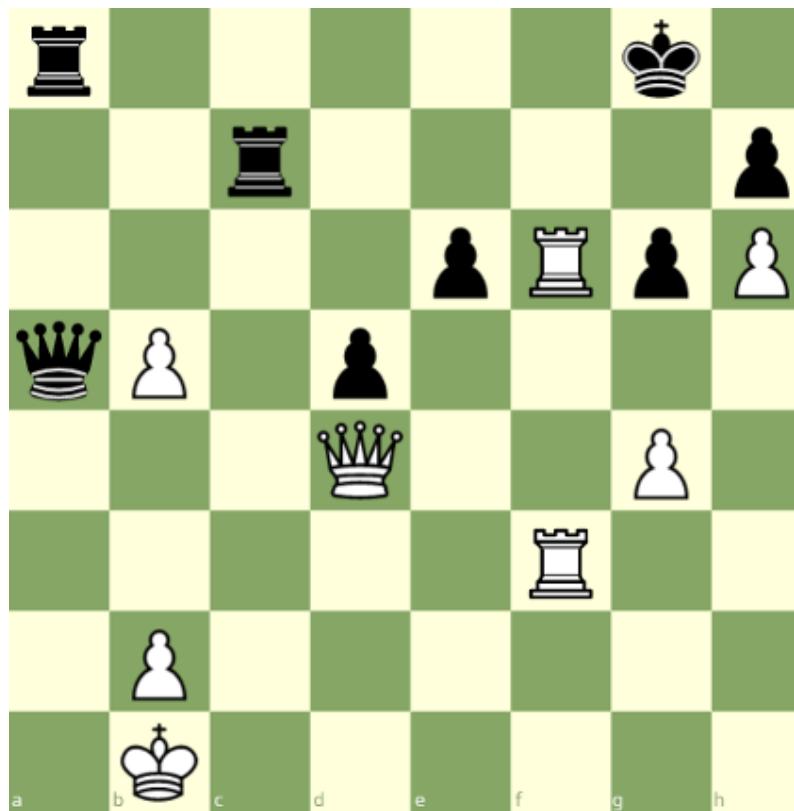
- Most common models in machine learning
- Fully feed-forward (no backtracking)

- The idea of a reflex-based model simply performs a fixed sequence of computations on a given input. Examples include most models found in machine learning from simple linear classifiers to deep neural networks. The main characteristic of reflex-based models is that their computations are feed-forward; one doesn't backtrack and consider alternative computations. Inference is trivial in these models because it is just running the fixed computations, which makes these models appealing.

Course plan

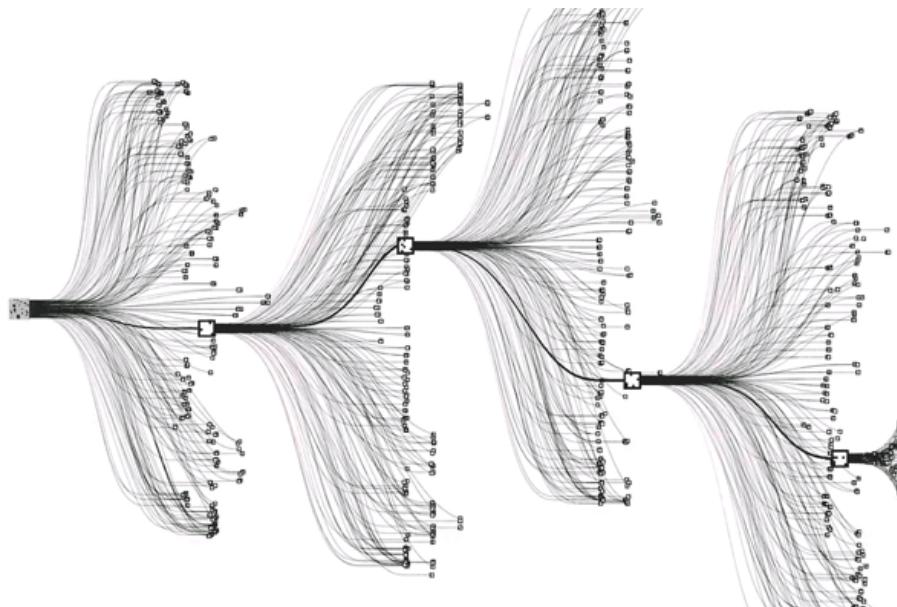


State-based models



White to move

State-based models



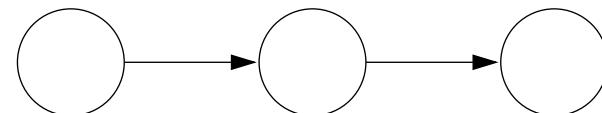
Applications:

- Games: Chess, Go, Pac-Man, Starcraft, etc.
- Robotics: motion planning
- Natural language generation: machine translation, image captioning

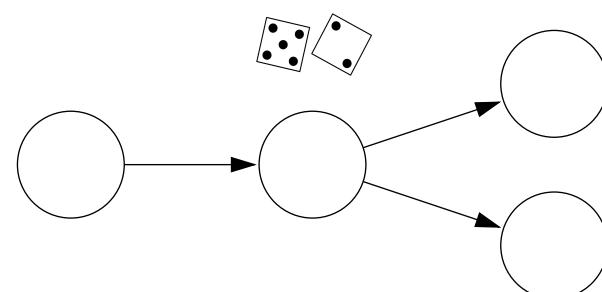
- Reflex-based models are too simple for tasks that require more forethought (e.g., in playing chess or planning a big trip). State-based models overcome this limitation.
- The key idea is, at a high-level, to model the **state** of a world and transitions between states which are triggered by actions. Concretely, one can think of states as nodes in a graph and transitions as edges. This reduction is useful because we understand graphs well and have a lot of efficient algorithms for operating on graphs.

State-based models

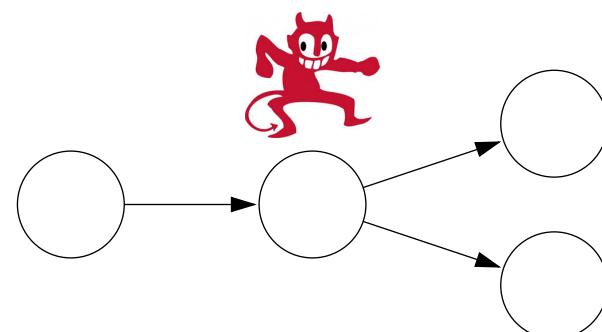
Search problems: you control everything



Markov decision processes: against nature (e.g., Blackjack)

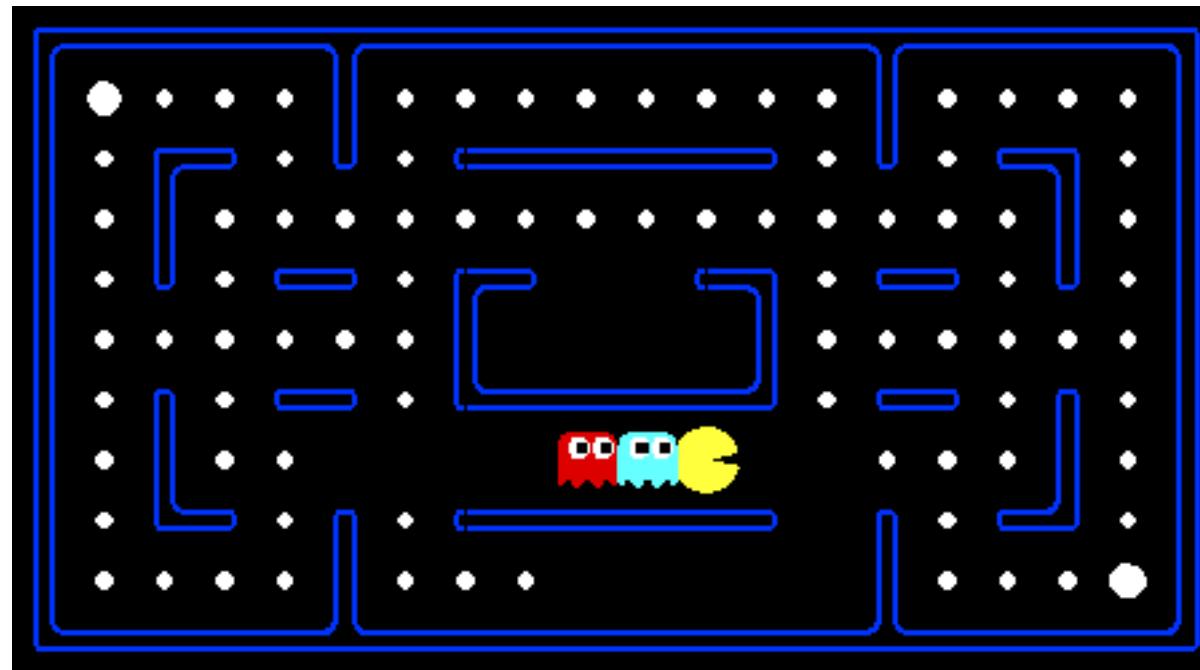


Adversarial games: against opponent (e.g., chess)



- Search problems are adequate models when you are operating in environment that has no uncertainty. However, in many realistic settings, there are other forces at play.
- **Markov decision processes** handle tasks with an element of chance (e.g., Blackjack), where the distribution of randomness is known (reinforcement learning can be employed if it is not).
- **Adversarial games**, as the name suggests, handle tasks where there is an opponent who is working against you (e.g., chess).

Pac-Man



[demo]



Question

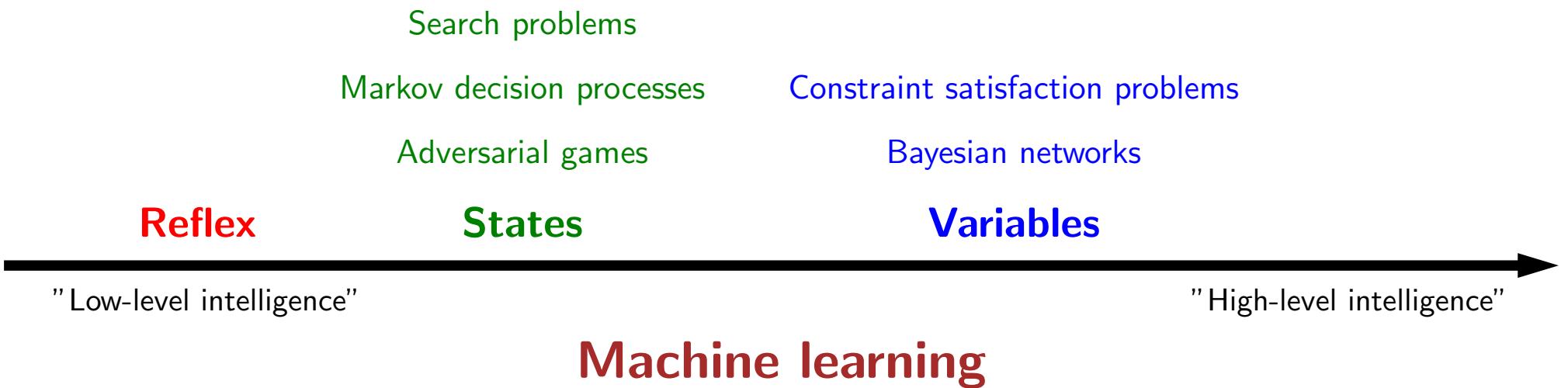
What kind of model is appropriate for playing Pac-Man against ghosts that move into each valid adjacent square with equal probability?

search problem

Markov decision process

adversarial game

Course plan



Sudoku

5	3		7					
6			1	9	5			
	9	8				6		
8			6				3	
4		8	3			1		
7			2			6		
	6			2	8			
		4	1	9			5	
		8		7	9			



5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

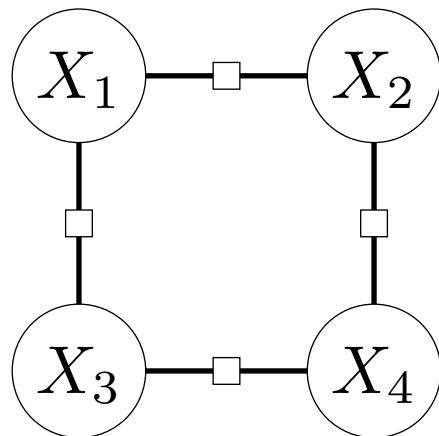
Goal: put digits in blank squares so each row, column, and 3x3 sub-block has digits 1–9

Note: order of filling squares doesn't matter in the evaluation criteria!

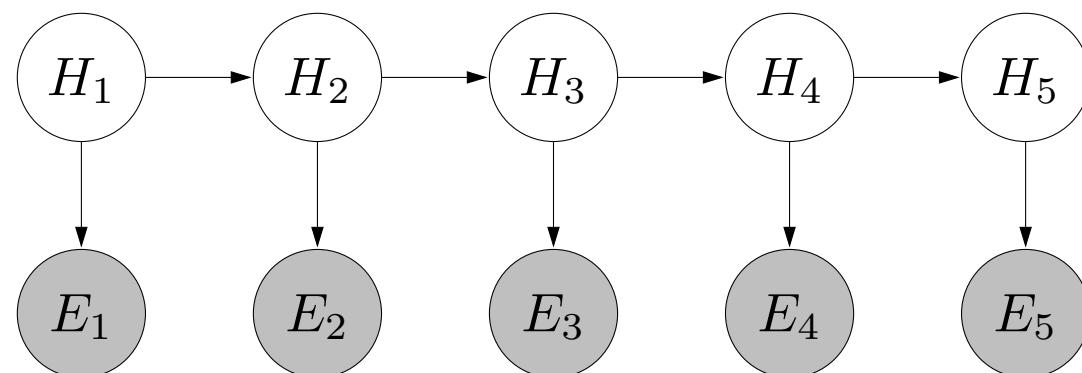
- In state-based models, solutions are procedural: they specify step by step instructions on how to go from A to B. In many applications, the order in which things are done isn't important.

Variable-based models

Constraint satisfaction problems: hard constraints (e.g., Sudoku, scheduling)

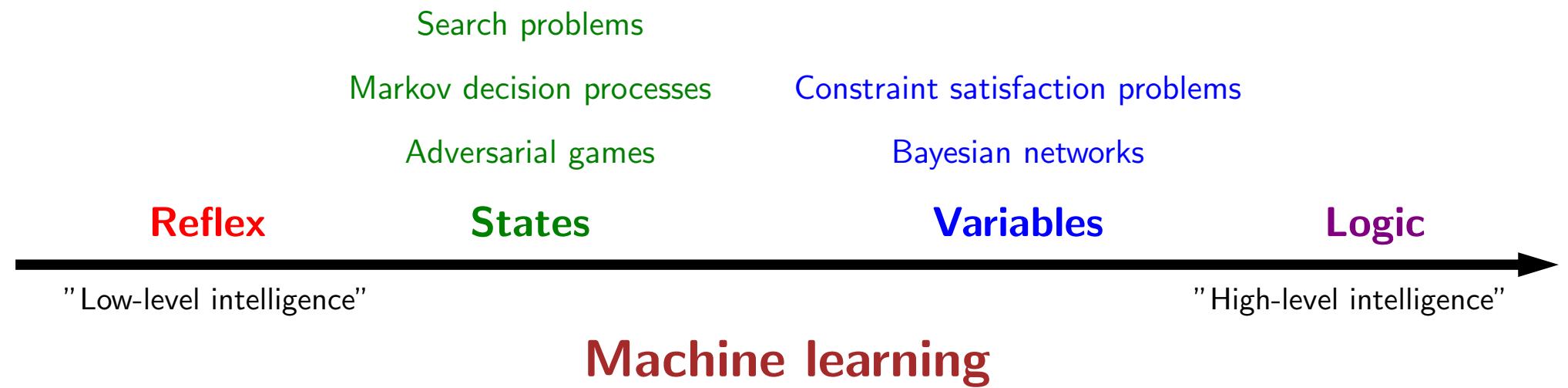


Bayesian networks: soft dependencies (e.g., tracking cars from sensors)



- **Constraint satisfaction problems** are variable-based models where we only have hard constraints. For example, in scheduling, we can't have two people in the same place at the same time.
- **Bayesian networks** are variable-based models where variables are random variables which are dependent on each other. For example, the true location of an airplane H_t and its radar reading E_t are related, as are the location H_t and the location at the last time step H_{t-1} . The exact dependency structure is given by the graph structure and formally defines a joint probability distribution over all the variables. This topic is studied thoroughly in probabilistic graphical models (CS228).

Course plan



Logic

- Dominated AI from 1960s-1980s, still useful in programming systems
- Powerful representation of knowledge and reasoning
- Brittle if done naively
- Open question: how to combine with machine learning?

- Our last stop on the tour is **logic**. Even more so than variable-based models, logic provides a compact language for modeling, which gives us more expressivity.
- It is interesting that historically, logic was one of the first things that AI researchers started with in the 1950s. While logical approaches were in a way quite sophisticated, they did not work well on complex real-world tasks with noise and uncertainty. On the other hand, methods based on probability and machine learning naturally handle noise and uncertainty, which is why they presently dominate the AI landscape. However, they have yet to be applied successfully to tasks that require really sophisticated reasoning.
- In this course, we will appreciate the two as not contradictory, but simply tackling different aspects of AI — in fact, in our schema, logic is a class of models which can be supported by machine learning. An active area of research is to combine the richness of logic with the robustness and agility of machine learning.

Motivation: virtual assistant

Tell information



Ask questions



Use natural language!

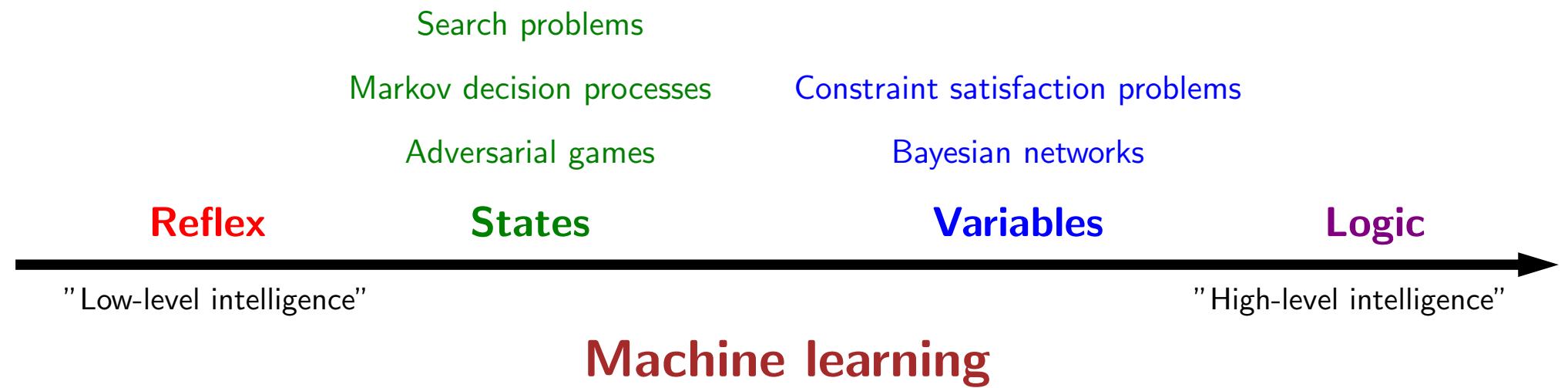
[demo]

Need to:

- Digest **heterogenous** information
- Reason **deeply** with that information

- One motivation for logic is a virtual assistant. At an abstract level, one fundamental thing a good personal assistant should be able to do is to take in information from people and be able to answer questions that require drawing inferences from these facts.
- In some sense, telling the system information is like machine learning, but it feels like a very different form of learning than seeing 10M images and their labels or 10M sentences and their translations. The type of information we get here is both more heterogenous, more abstract, and the expectation is that we process it more deeply (we don't want to have to tell our personal assistant 100 times that we prefer morning meetings).
- And how do we interact with our personal assistants? Let's use natural language, the very tool that was built for communication!

Course plan





Roadmap

What is AI?

Course overview

Course logistics

Optimization

Course objectives

Before you take the class, you should know... —

- Programming (CS 106A, CS 106B, CS 107)
- Discrete math, mathematical rigor (CS 103)
- Probability (CS 109)

At the end of this course, you should... —

- Be able to tackle real-world tasks with the appropriate techniques
- Be more proficient at math and programming



Coursework

- Homeworks (60%)
- Exam (20%)
- Project (20%)

Homeworks

- 8 homeworks, mix of written and programming problems, centers on an application

Introduction	foundations
Machine learning	sentiment classification
Search	text reconstruction
MDPs	blackjack
Games	Pac-Man
CSPs	course scheduling
Bayesian networks	car tracking
Logic	language and logic

- Some have competitions for extra credit
- When you submit, programming parts will be run on all test cases, but only get feedback on a subset

Exam

- Goal: test your ability to use knowledge to solve new problems, not know facts
- All written problems (look at past exam problems for style)
- Closed book except one page of notes
- Covers all material up to and including preceding week
- Tue Nov. 29 from 6pm to 9pm (3 hours)

Project

- Goal: choose any task you care about and apply techniques from class
- Work in groups of up to 3; find a group early, your responsibility to be in a good group
- Milestones: proposal, progress report, poster session, final report
- Task is completely open, but must follow well-defined steps: task definition, implement baselines/oracles, evaluate on dataset, literature review, error analysis (read website)
- Help: assigned a CA mentor, come to any office hours

Policies

Late days: 8 total late days, max two per assignment

Regrades: come in person to the owner CA of the homework

Piazza: ask questions on Piazza, don't email us directly

Piazza: extra credit for students who help answer questions

All details are on the course website

THE HONOR CODE

- Do collaborate and discuss together, but write up and code independently.
- Do not look at anyone else's writeup or code.
- Do not show anyone else your writeup or code or post it online (e.g., GitHub).
- When debugging, only look at input-output behavior.
- We will run MOSS periodically to detect plagiarism.



Roadmap

What is AI?

Course overview

Course logistics

Optimization

Optimization

Discrete optimization: a discrete object

$$\min_{p \in \text{Paths}} \text{Distance}(p)$$

Algorithmic tool: dynamic programming

Continuous optimization: a vector of real numbers

$$\min_{\mathbf{w} \in \mathbb{R}^d} \text{TrainingError}(\mathbf{w})$$

Algorithmic tool: gradient descent

- We are now done with the high-level motivation for the class. Let us now dive into some technical details. Let us focus on the inference and the learning aspect of the **modeling-inference-learning** paradigm.
- We will approach inference and learning from an **optimization** perspective, which allows us to decouple the mathematical specification of **what** we want to compute from the algorithms for **how** to compute it.
- In total generality, optimization problems ask that you find the x that lives in a constraint set C that makes the function $F(x)$ as small as possible.
- There are two types of optimization problems we'll consider: discrete optimization problems (mostly for inference) and continuous optimization problems (mostly for learning). Both are backed by a rich research field and are interesting topics in their own right. For this course, we will use the most basic tools from these topics: **dynamic programming** and **gradient descent**.
- Let us do two practice problems to illustrate each tool. For now, we are assuming that the model (optimization problem) is given and only focus on **algorithms**.



Problem: computing edit distance

Input: two strings, s and t

Output: minimum number of character insertions, deletions, and substitutions it takes to change s into t

Examples:

$$\text{"cat"}, \text{"cat"} \Rightarrow 0$$

$$\text{"cat"}, \text{"dog"} \Rightarrow 3$$

$$\text{"cat"}, \text{"at"} \Rightarrow 1$$

$$\text{"cat"}, \text{"cats"} \Rightarrow 1$$

$$\text{"a cat!"}, \text{"the cats!"} \Rightarrow 4$$

[live solution]

- Let's consider the formal task of computing the edit distance (or more precisely the Levenshtein distance) between two strings. These measures of dissimilarity have applications in spelling correction, computational biology (applied to DNA sequences).
- As a first step, you should think to break down the problem into subproblems. Observation 1: inserting into s is equivalent to deleting a letter from t (ensures subproblems get smaller). Observation 2: perform edits at the end of strings (might as well start there).
- Consider the last letter of s and t . If these are the same, then we don't need to edit these letters, and we can proceed to the second-to-last letters. If they are different, then we have three choices. (i) We can substitute the last letter of s with the last letter of t . (ii) We can delete the last letter of s . (iii) We can insert the last letter of t at the end of s .
- In each of those cases, we can reduce the problem into a smaller problem, but which one? We simply try all of them and take the one that yields the minimum cost!
- We can express this more formally with a mathematical recurrence. These types of recurrences will show up throughout the course, so it's a good idea to be comfortable with them. Before writing down the actual recurrence, the first step is to express the quantity that we wish to compute. In this case: let $d(m, n)$ be the edit distance between the first m letters of s and the first n letters of t . Then we have

$$d(m, n) = \begin{cases} m & \text{if } n = 0 \\ n & \text{if } m = 0 \\ d(m - 1, n - 1) & \text{if } s_m = t_n \\ 1 + \min\{d(m - 1, n - 1), d(m - 1, n), d(m, n - 1)\} & \text{otherwise.} \end{cases}$$

- Once you have the recurrence, you can code it up. The straightforward implementation will take exponential time, but you can **memoize** the results to make it $O(n^2)$ time. The end result is the dynamic programming solution: recurrence + memoization.



Problem: finding the least squares line

Input: set of pairs $\{(x_1, y_1), \dots, (x_n, y_n)\}$

Output: $w \in \mathbb{R}$ that minimizes the squared error

$$F(w) = \sum_{i=1}^n (x_i w - y_i)^2$$

Examples:

$$\{(2, 4)\} \Rightarrow 2$$

$$\{(2, 4), (4, 2)\} \Rightarrow ?$$

[live solution]

- The formal task is this: given a set of n two-dimensional points (x_i, y_i) which defines $F(w)$, compute the w that minimizes $F(w)$.
- **Linear regression** is an important problem in machine learning, which we will come to later. Here's a motivation for the problem: suppose you're trying to understand how your exam score (y) depends on the number of hours you study (x). Let's posit a linear relationship $y = wx$ (not exactly true in practice, but maybe good enough). Now we get a set of training examples, each of which is a (x_i, y_i) pair. The goal is to find the slope w that best fits the data.
- Back to algorithms for this formal task. We would like an algorithm for optimizing general types of $F(w)$. So let's **abstract away from the details**. Start at a guess of w (say $w = 0$), and then iteratively update w based on the derivative (gradient if w is a vector) of $F(w)$. The algorithm we will use is called **gradient descent**.
- If the derivative $F'(w) < 0$, then increase w ; if $F'(w) > 0$, decrease w ; otherwise, keep w still. This motivates the following update rule, which we perform over and over again: $w \leftarrow w - \eta F'(w)$, where $\eta > 0$ is a **step size** that controls how aggressively we change w .
- If η is too big, then w might bounce around and not converge. If η is too small, then w might not move very far to the optimum. Choosing the right value of η can be rather tricky. Theory can give rough guidance, but this is outside the scope of this class. Empirically, we will just try a few values and see which one works best. This will help us develop some intuition in the process.
- Now to specialize to our function, we just need to compute the derivative, which is an elementary calculus exercise: $F'(w) = \sum_{i=1}^n 2(x_i w - y_i)x_i$.



cs221.stanford.edu/q

Question

What was the most surprising thing you learned today?



Summary

- AI has high societal impact, our responsibility to steer it positively
- Modeling [reflex, states, variables, logic] + inference + learning
- Section this Friday: review of foundations
- Homework [foundations]: due next Tuesday 11pm
- Course will be fast-paced and exciting!