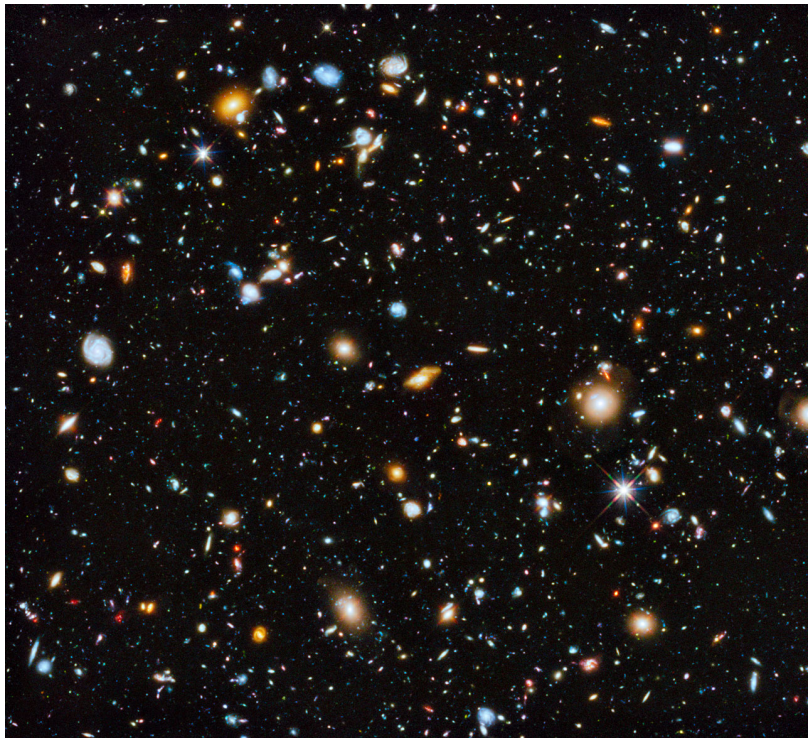
Lecture 1: Overview

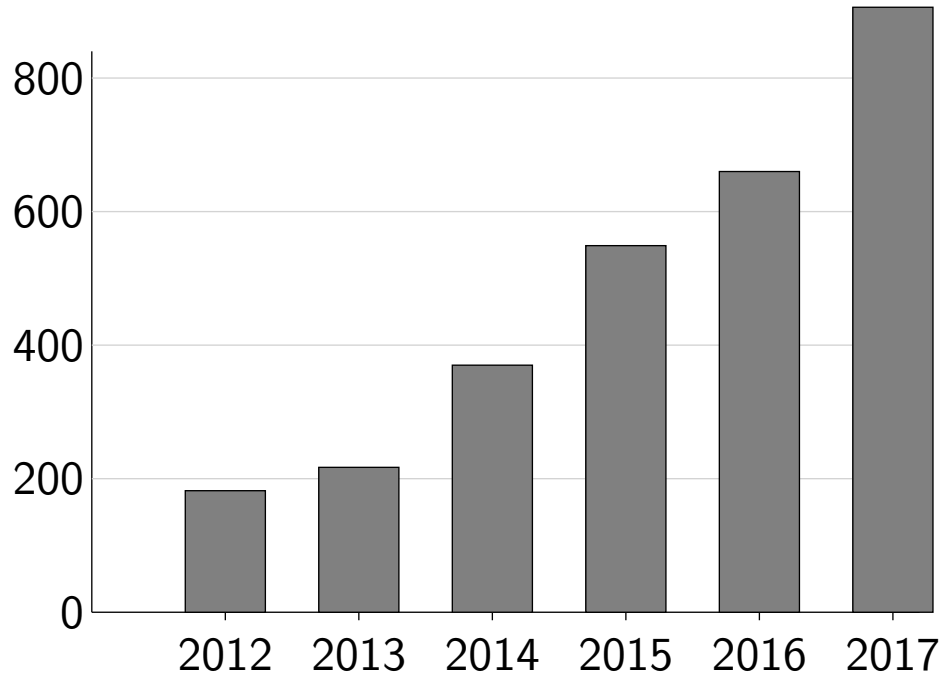


Teaching staff

Percy Liang and Stefano Ermon

Stephen Mussmann (head CA)
Zhi Bie
Orion Despo
Bryan He
Nish Khandwala
Dylan Liu
Reid Pryzant
Mohana Sathya Moorthy
Feiran Wang
Zach Barnes
Heather Blundell
Tony Duan
Mark Holmstrom
Hansohl Kim
Michelle Mei
Alisha Rege
Christopher Sauer
Yang Yuan
Bardia Beigi
Rickard Brüel Gabrielsson
David Eng
Helen Jiang
Sydney Li
Amani Peddada
Vig Sachidananda
Anna Wang

CS221 enrollments



CS221 breakdown by year

Freshman	1
Sophomore	38
Junior	163
Senior	175
GradYear1	209
GradYear2	190
GradYear3	66
GradYear4+	60

CS221 breakdown by majors

Psychology	1	Biology	6
Indiv Des Major-Engr	1	Music	6
Geological Sciences	1	Energy Resources Engineering	6
Biochemistry	1	Bioengineering	7
Japanese	1	Engineering	8
Philosophy & Rel Stud	1	Chemical Engineering	8
Human Biology	1	Materials Science & Engr	8
Law	1	Applied Physics	9
Education	1	Statistics	13
Philosophy	1	Math & Comp Science	14
Art History	1	Business Administration	15
Sociology	1	Aeronautics & Astro	15
Art Practice	2	Physics	16
Classics	2	Civil & Envir Engr	17
Geophysics	2	Comput & Math Engr	24
Environment and Resources	2	Mgmt Sci & Engineering	25
Mathematics	3	Symbolic Systems	27
Management	4	Mechanical Engineer	33
Biomedical Informatics	4	Undeclared	67
Petroleum Engineer	4	Graduate Non-Deg Option	97
Neurosciences	4	Electrical Engineering	123
Chemistry	4	Computer Science	309
Economics	6		

Roadmap

Why AI?

How do we approach it?

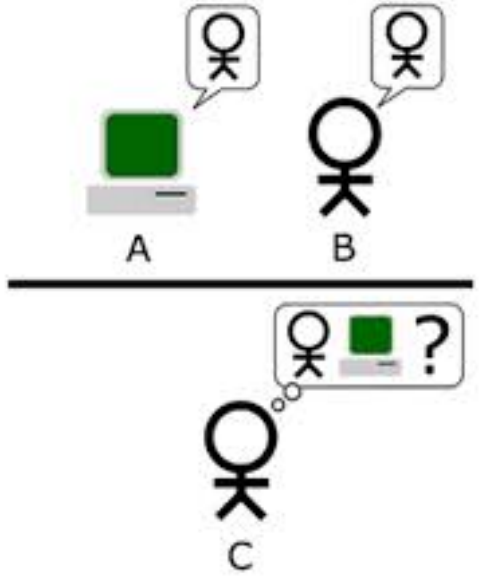
Course logistics

Optimization

What is AI?

The Turing Test (1950)

"Can machines think?"



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

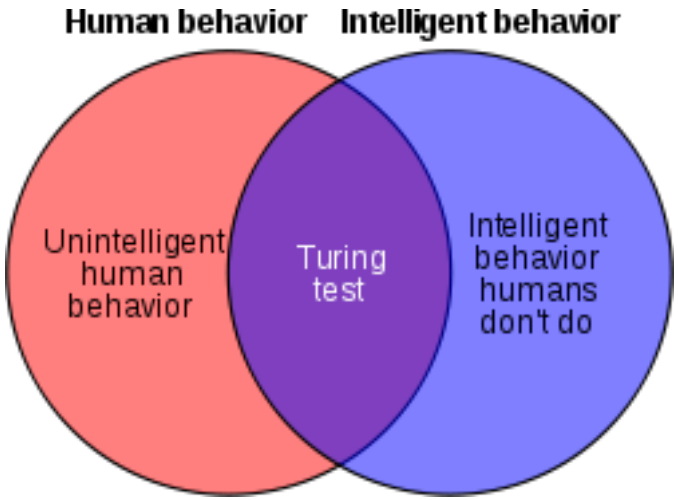
A: Count me out on this one. I never could write poetry.

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 Decartes. 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.

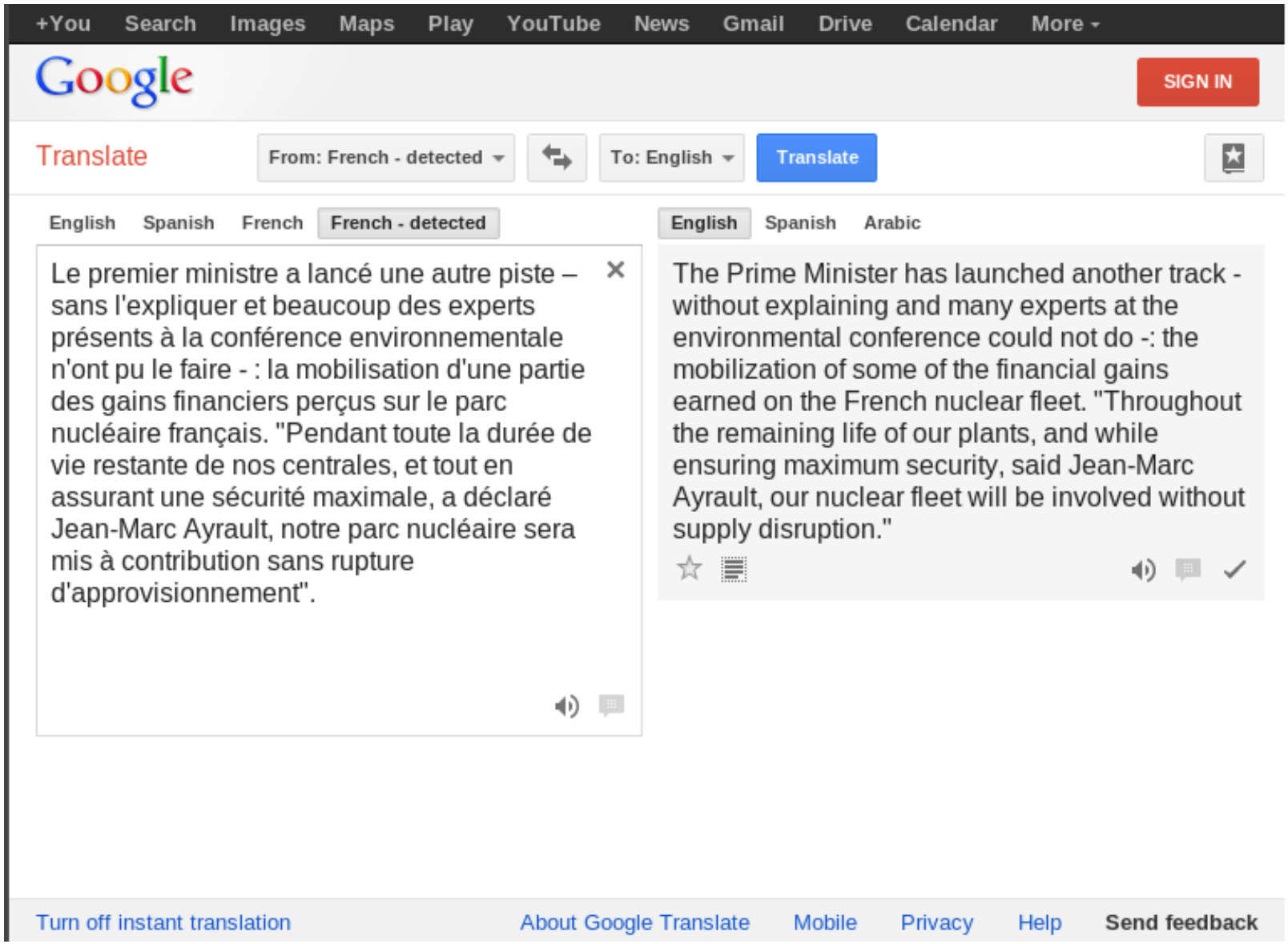

Human behavior
Intelligent behavior
Unintelligent human behavior
Turing test
Intelligent behavior humans don't do

- Perhaps imitating humans is really the wrong metric when it comes to thinking about intelligence. It is true that humans possess abilities (language, vision, motor control) which currently surpass the best machines, but on the other hand, machines clearly possess many advantages over humans (e.g., speed, memory). Why settle for human-level performance?
- The study of how humans think is fascinating and well-studied within the field of cognitive science. In this class, however, we will primarily be concerned with the engineering goal of building intelligent systems, drawing from humans only as a source of solvable problems and high-level motivation.

Some success stories...

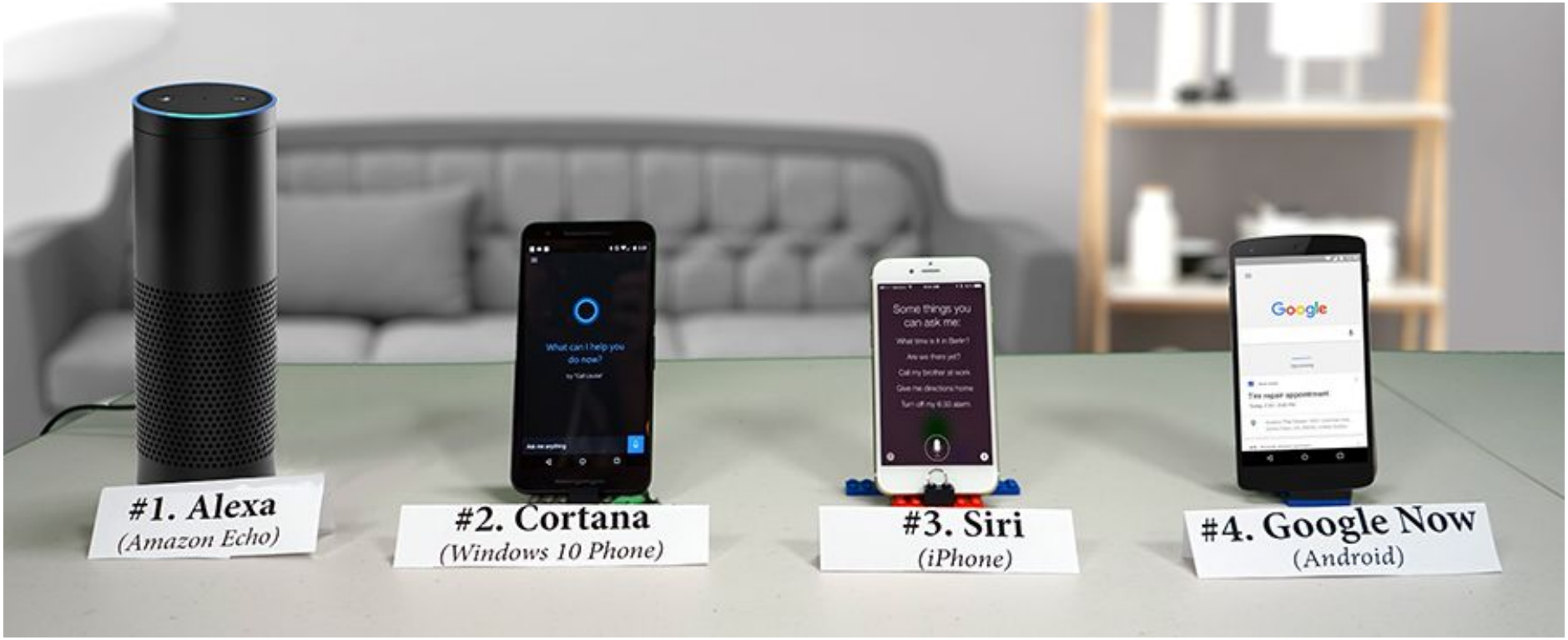
- Instead of asking what AI is, let us turn to the more pragmatic question of what AI can do. We will go through some examples where AI has been successful. Note that some of the examples are where AI is already widely deployed in practice, while others are fun but may not necessarily lead to something practically useful.

Machine translation



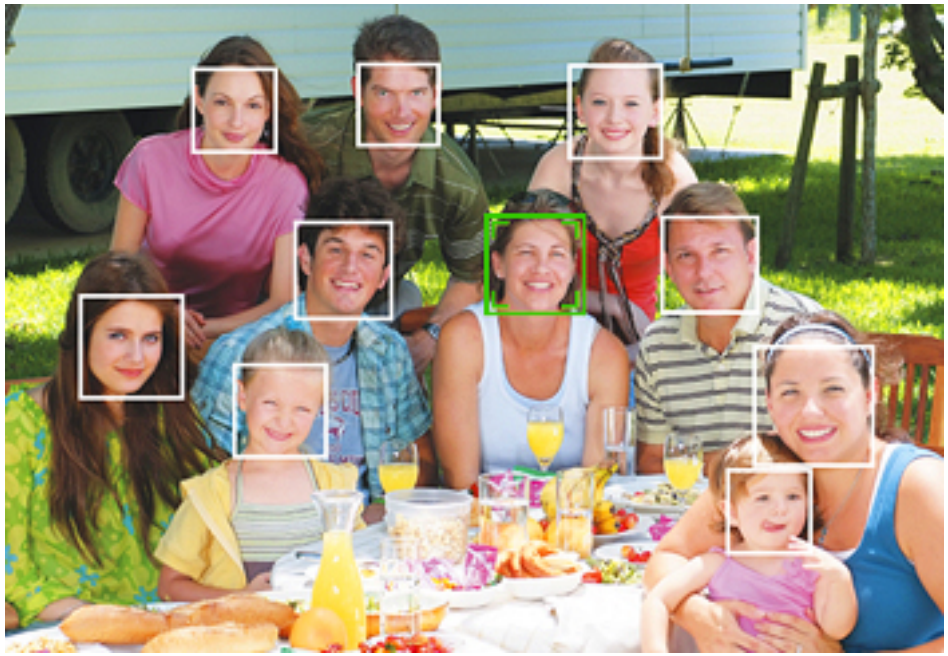
- Machine translation research started in the 1960s (the US government was quite keen on translating Russian into English). Over the subsequent decades, it went through quite a few rough turns.
- In the 1990s and 2000s, statistical machine translation, aided by large amounts of example translations, helped vastly improve translation quality.
- As of 2015, Google Translate supports 90 languages and serves over 200 million people daily. The translations are nowhere near perfect, but they are very useful.

Speech recognition



- Speech recognition is the problem of transcribing audio into words. It too has a long history dating back to the 1970s. But it wasn't until around 2009 that speech recognition started to work well due to the adoption of deep neural networks.
- In a very short period of time, companies such as Apple, Google, Microsoft all adopted this technology. Furthermore, with the rise of smartphones, speech recognition began paving way for the emergence of virtual assistants such as Apple's Siri, Google Now, Microsoft Cortana, Amazon Echo, and others.
- However, speech recognition is only one part of the story; the other is understanding the text, which is a much harder problem. Current systems don't handle much more than simple utterances and actions (e.g., setting an alarm, sending a text, etc.), but the area of natural language understanding is growing rapidly.

Face identification



human-level performance, but privacy issues?

- In 2014, Facebook Research published a paper describing their DeepFace face identification system. DeepFace is a 120 million parameter deep neural network and obtains 97.35% on the standard Labeled Faces in the Wild (LFW) dataset, which is comparable to human performance. Facebook definitely has an upper hand when it comes to amassing training data for this task: whenever a user tags a person in a photo, he or she is providing a training example.
- However, a powerful technology such as this comes with non-technical impliciations. Privacy advocates strongly oppose the deployment of pervasive identification, because it would enable some entity (be it a company or a government) to take arbitrary images and videos of crowds and identify every single person in it, which would effectively eliminate the ability to stay anonymous.

Autonomous driving

- And now we discuss some AI technologies which are promising but not quite ready for prime time.
- Research in autonomous cars started in the 1980s, but the technology wasn't there.
- Perhaps the first significant event was the 2005 DARPA Grand Challenge, in which the goal was to have a driverless car go through a 132-mile off-road course. Stanford finished in first place. The car was equipped with various sensors (laser, vision, radar), whose readings needed to be synthesized (using probabilistic techniques that we'll learn from this class) to localize the car and then generate control signals for the steering, throttle, and brake.
- In 2007, DARPA created an even harder Urban Challenge, which was won by CMU.
- In 2009, Google started a self-driving car program, and since then, their self-driving cars have driven over 1 million miles on freeways and streets.
- In January 2015, Uber hired about 50 people from CMU's robotics department to build self-driving cars.
- While there are still technological and policy issues to be worked out, the potential impact on transportation is huge.

[SQuAD dataset; Rajpurkar et al. 2016]

Reading comprehension

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?
gravity

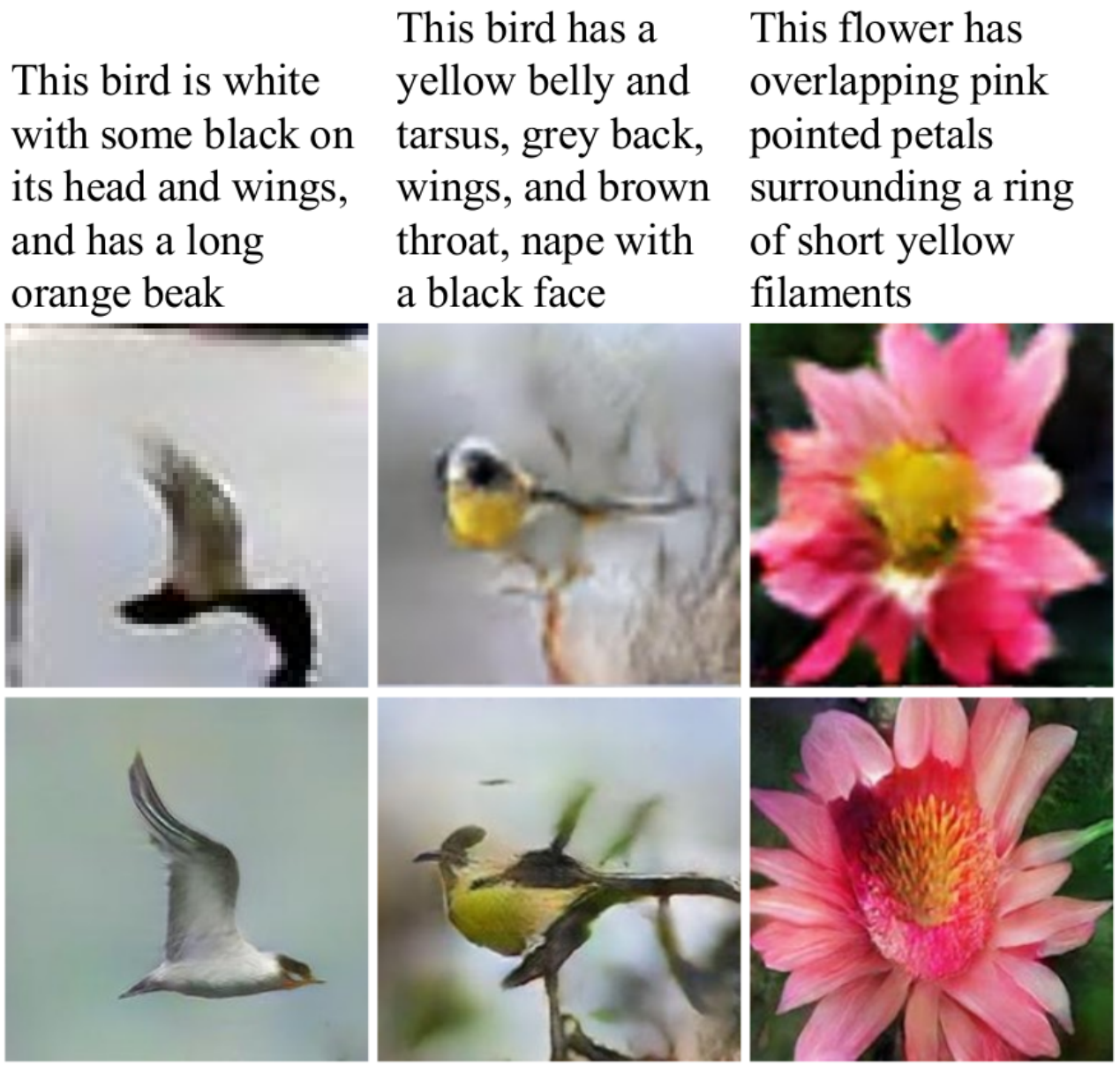
What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

- Natural language understanding generally is still widely regarded as an unsolved problem. One of the specific incarnations is the task of reading comprehension: given a passage, the goal is to answer a question about the passage (think standardized tests).
- One of the popular recent datasets for reading comprehension is SQuAD, which has 100K questions taken from Wikipedia. Current methods (see stanford-qa.com) do quite well on this dataset, but as a student can pass a standardized test without true understanding, recent work shows that such systems can get fooled by more probing questions.

[StackGANs; Zhang et al, 2016]

Image generation



- One particular hot topic in computer vision right now is generating photorealistic images from text. The results are becoming visually quite convincing, owing largely to advances such as Generative Adversarial Networks (GANs). However, keep in mind that it is hard to judge the quality of a system from looking at a single image, as the "copy a training example" strategy also works quite well.

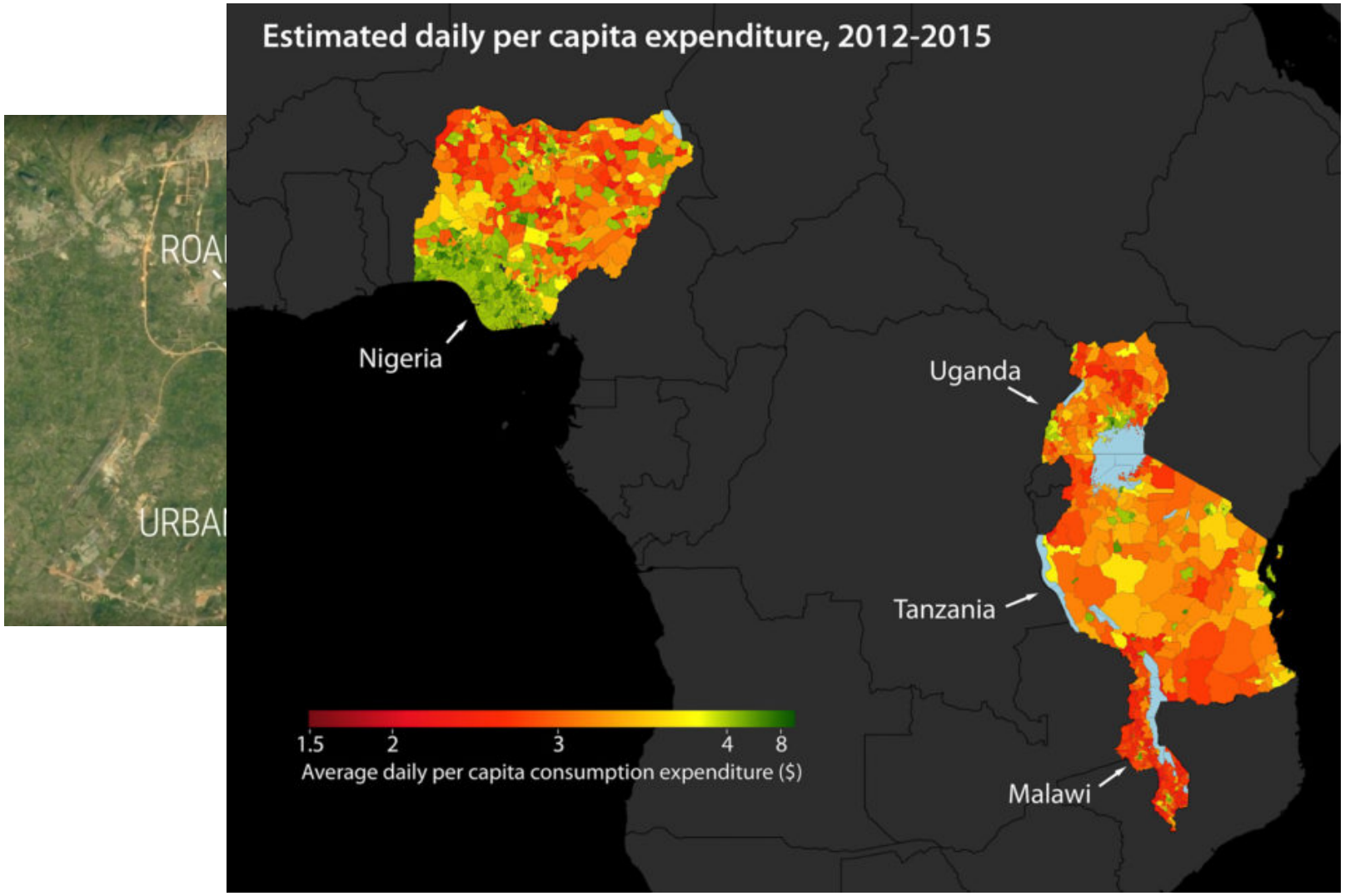
[from Justin Johnson's implementation of Gatys et al. 2015]

Artistic style transfer

- Another form of image generation is style transfer, in which we are given a "content image" and a "style image", and the goal is to generate a new image with the given contents and style. Though easier in many ways than generating an image from scratch, this leads to quite visually pleasing and stunning results.

[Jean et al. 2016]

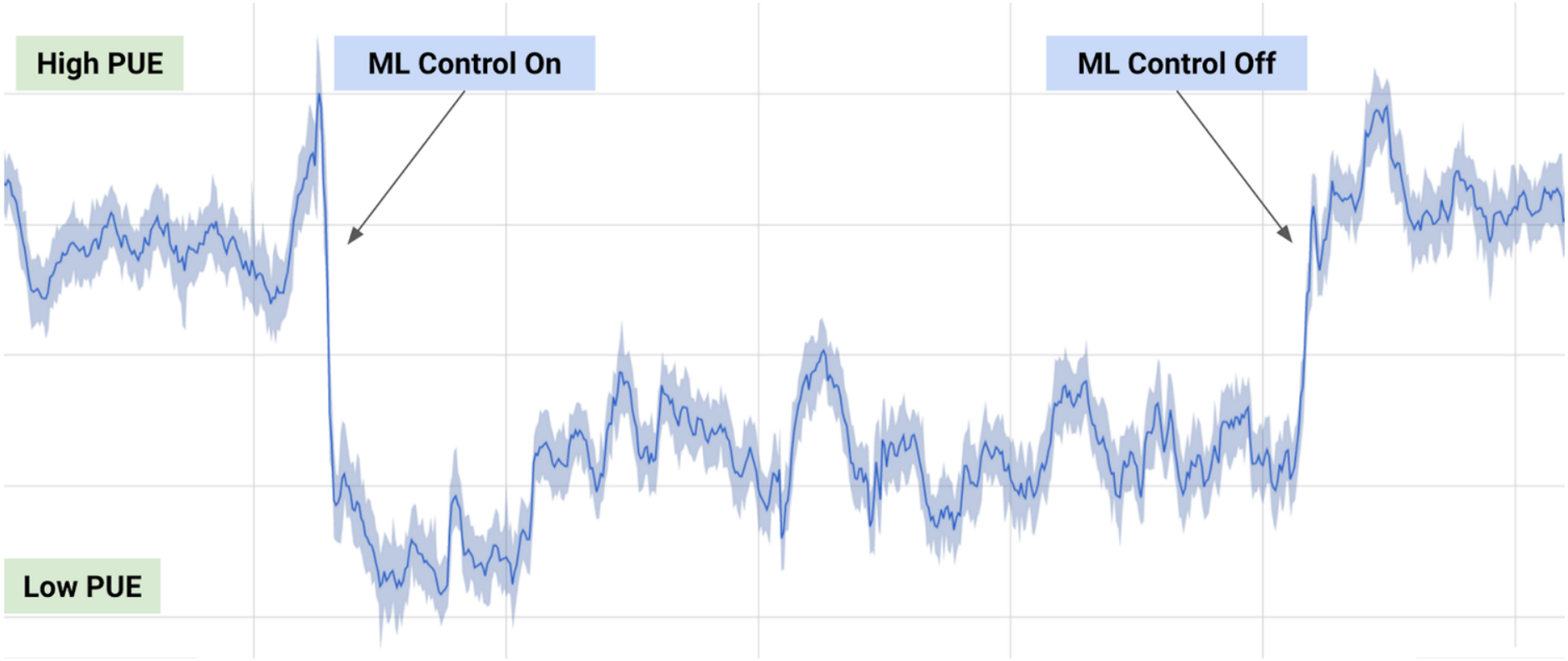
Predicting poverty



- Computer vision also 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.

[DeepMind]

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.

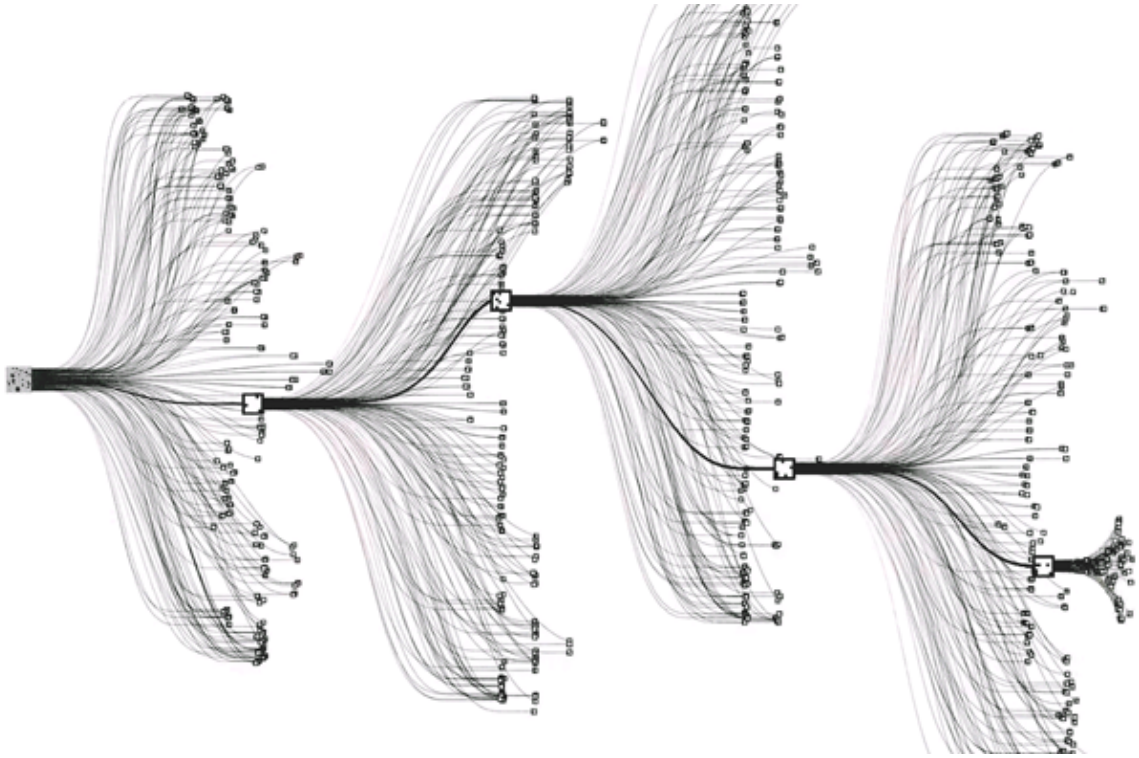
Humans versus machines

1997: Deep Blue (chess)

2011: IBM Watson (Jeopardy!)

- Perhaps the aspect of AI that captures the public's imagination the most is in defeating humans at their own games.
- In 1997, Deep Blue defeated Gary Kasparov, the world chess champion. In 2011, IBM Watson defeated two of the biggest winners (Brad Rutter and Ken Jennings) at the quiz show Jeopardy! (IBM seems to be pretty good at performing these kind of stunts.)
- One could have argued that Deep Blue won simply by the sheer force of its computational prowess, whereas winning Jeopardy! involved understanding natural language, and this defeat hit closer to home.

Humans versus machines

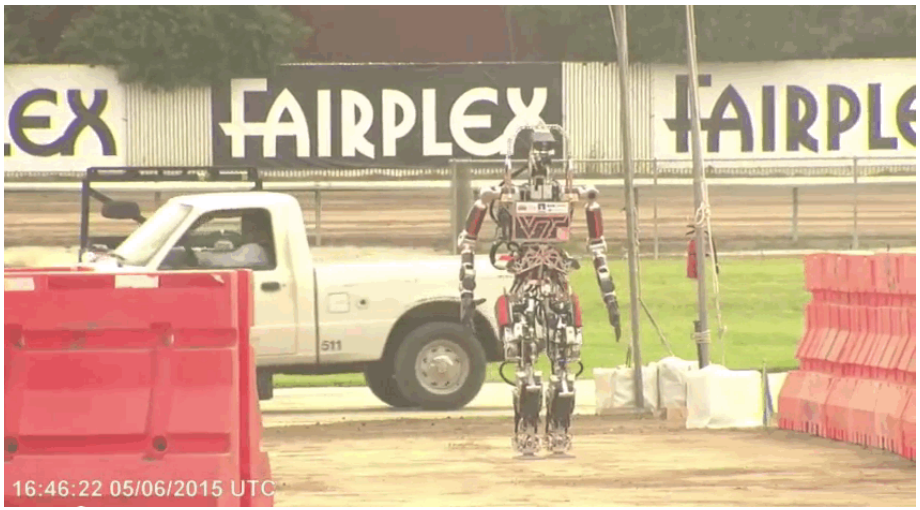


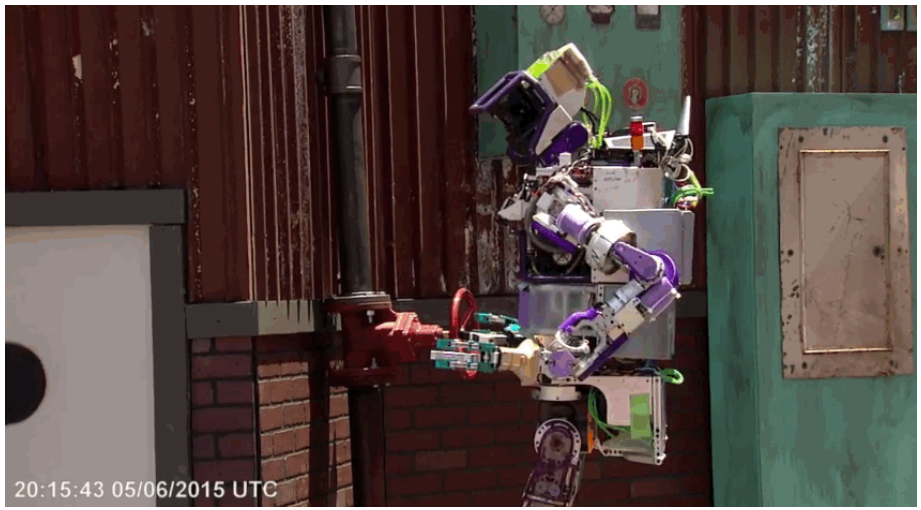
- March 2016 gave us another seminal result in game playing, this time in the ancient game of Go. Unlike chess, which fell to efficient search algorithms, Go styimed computer programs for a very long time, as the the space of possible moves in Go is much larger.
- Google DeepMind created a program called AlphaGo, which used deep neural networks and reinforcement learning (techniques we'll cover later in this class), to defeat Lee Sedol, a 9-dan professional, 4-1 in a stunning five-game match, surprising not only the master Go player but many AI researchers as well.
- Since then, the program has improved even more. In May 2017, AlphaGo defeated Ke Jie, who was ranked first in the world.

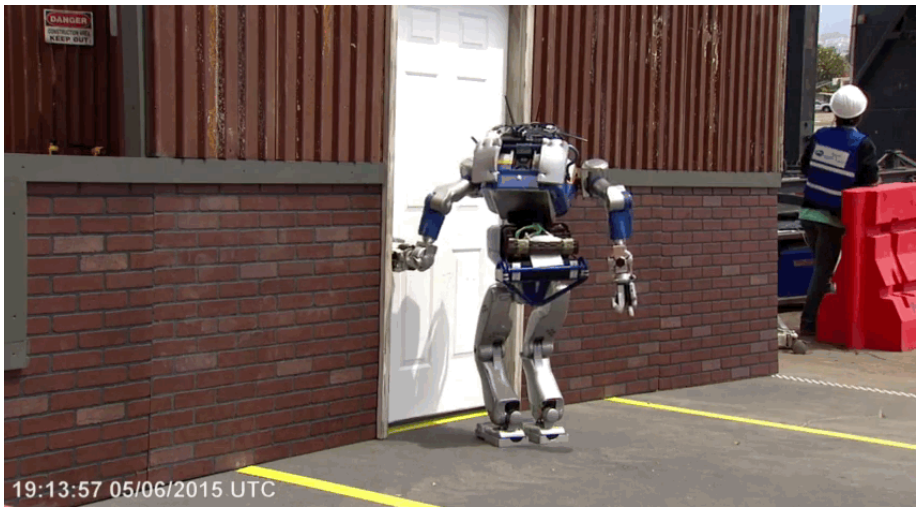
Some failures...

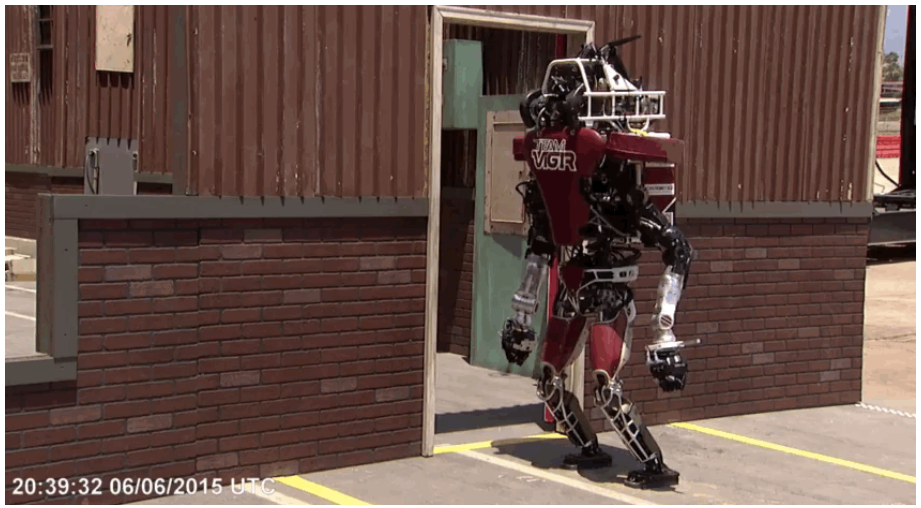
- It would be remiss not to also show the failure modes of AI, which can actually be quite serious and surprising. These failures show that there are still important and natural tasks that humans routinely perform with ease, that still pose difficult, open research problems.

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 completely 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.

[Li et al., 2016]

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.

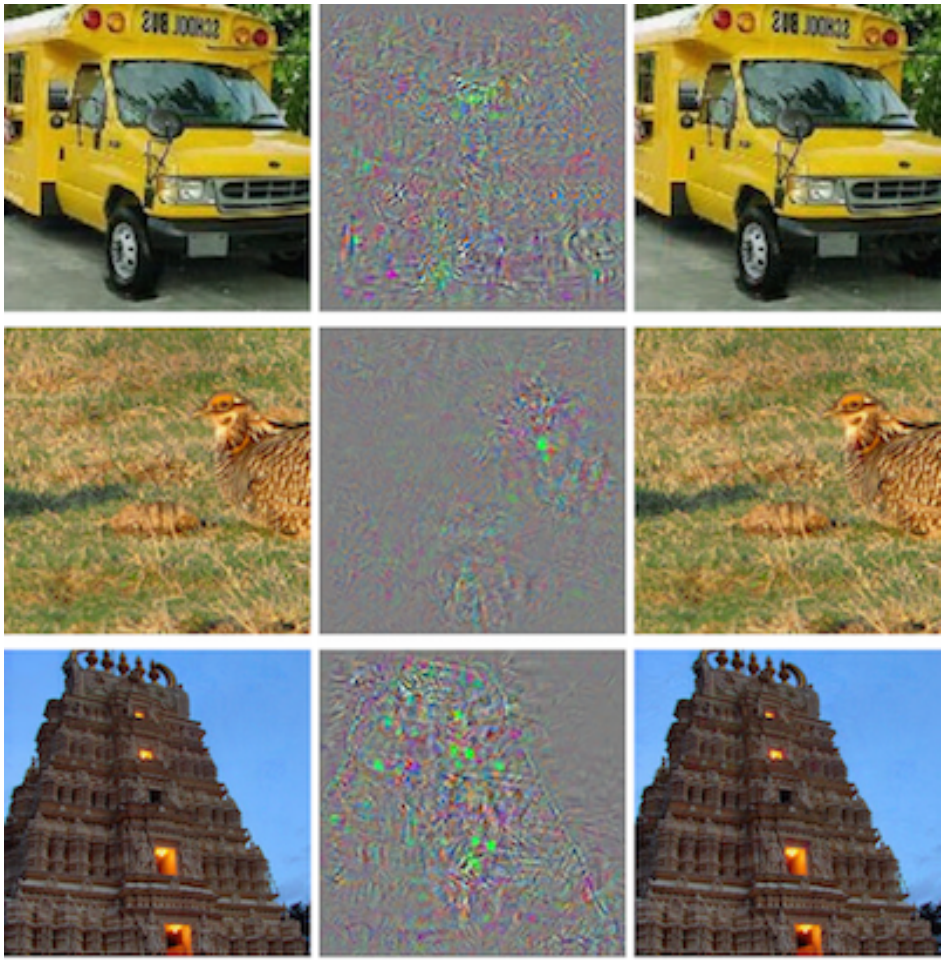
B: You don't know what you are saying.

- 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 sensible results. Models get confused by the sheer complexity of dialogue and often fall back to generic responses as shown here.

[Szegedy et al., 2013; Goodfellow et al., 2014]

Adversarial examples

AlexNet predicts correctly on the left

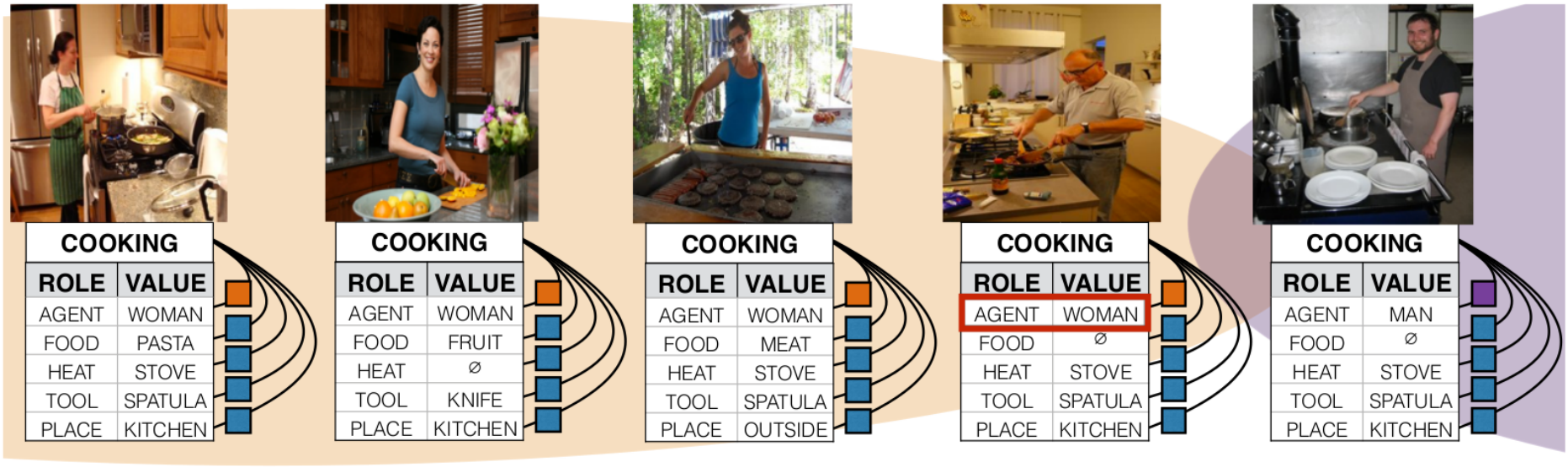


AlexNet predicts **ostrich** on the right

- Failures aren't just in extremely hard problems such as robotics and dialogue, but can be found in much easier tasks.
- An iconic example are adversarial examples where one can perturb an image by a carefully chosen, but imperceptible amount, and cause a state-of-the-art model to misclassify the image.
- These examples pose security problems as computer vision is adopted in self-driving cars and authentication systems. But more fundamentally, these examples shows that current methods clearly are not learning "the right thing" as defined by the human visual system.

[Zhao et al., 2017]

Bias



33% men in training set, only predict 16% men at test time

society $\Rightarrow$ data $\Rightarrow$ machine learning predictions

- 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.

In the spotlight...

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.

- Given the velocity of the recent developments in AI, AI has been embraced by the major tech companies, with very explicit endorsement from the top-down leadership.

PREPARING FOR THE FUTURE

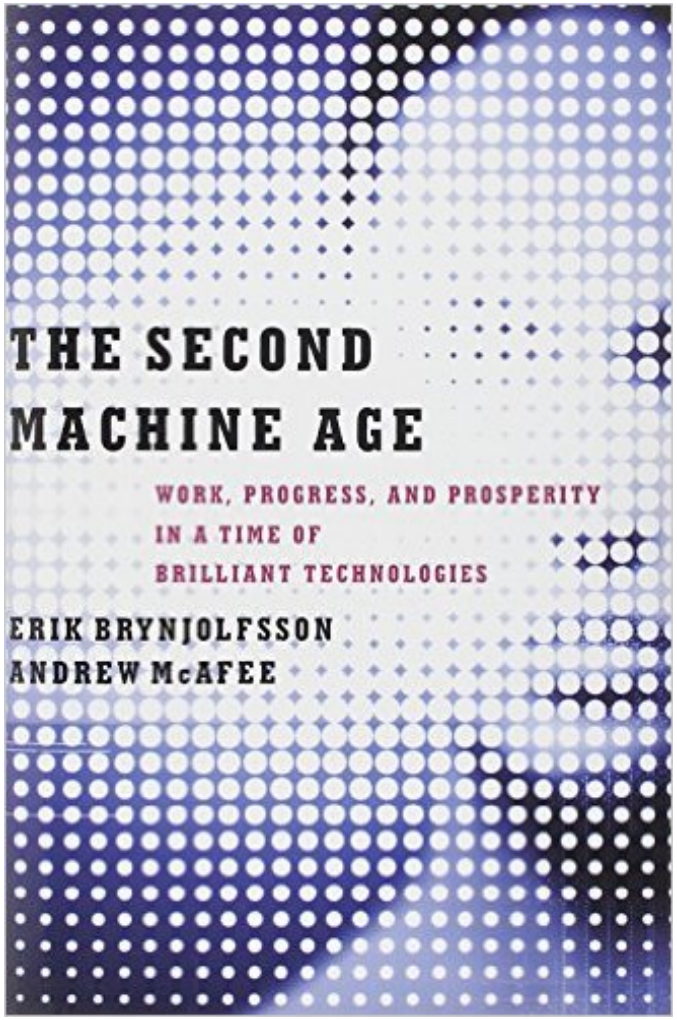
Governments

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

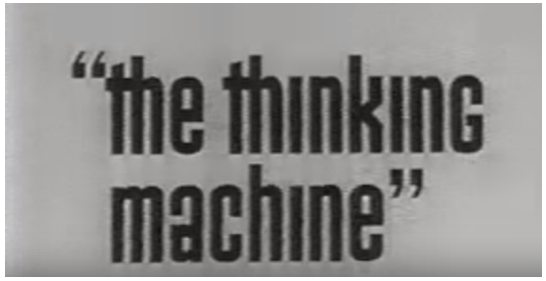
Released domestic strategic plan to become world leader in AI by 2030 [2017]

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

- Governments are noticing as well. In 2016, the White House put out a report describing the priorities of AI. China is investing extremely heavily in AI and is very ambitious about their goals.


THE SECOND
MACHINE AGE
WORK, PROGRESS, AND PROSPERITY
IN A TIME OF
BRILLIANT TECHNOLOGIES
ERIK BRYNJOLFSSON
ANDREW McAFEE

- Some even predict that AI will be as transformative on society as the agricultural and industrial revolutions. Just as the industrial revolution provided a solution to the problem of physical labor, AI promises to provide a solution to the problem of mental labor.



- 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 Initative (DARPA)
- 1987: collapse of Lisp market, government funding cut
- 1990-: rise of machine learning
- 2010s: heavy industry investment in deep learning

- But such optimism is not new. People in the 1960s, when computers were still fresh, had similar dreams. Ok, so maybe people misjudged the difficulty of the problem. But it happened again in the 1980s, 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".

cs221.stanford.edu/q

Question

Now what do you think AI will achieve by 2030?

- Hype will die down, will have limited impact
- Will be very useful, but only in narrow verticals
- Will match humans at many tasks but not all
- Will match or surpass humans at everything

Characteristics of AI tasks

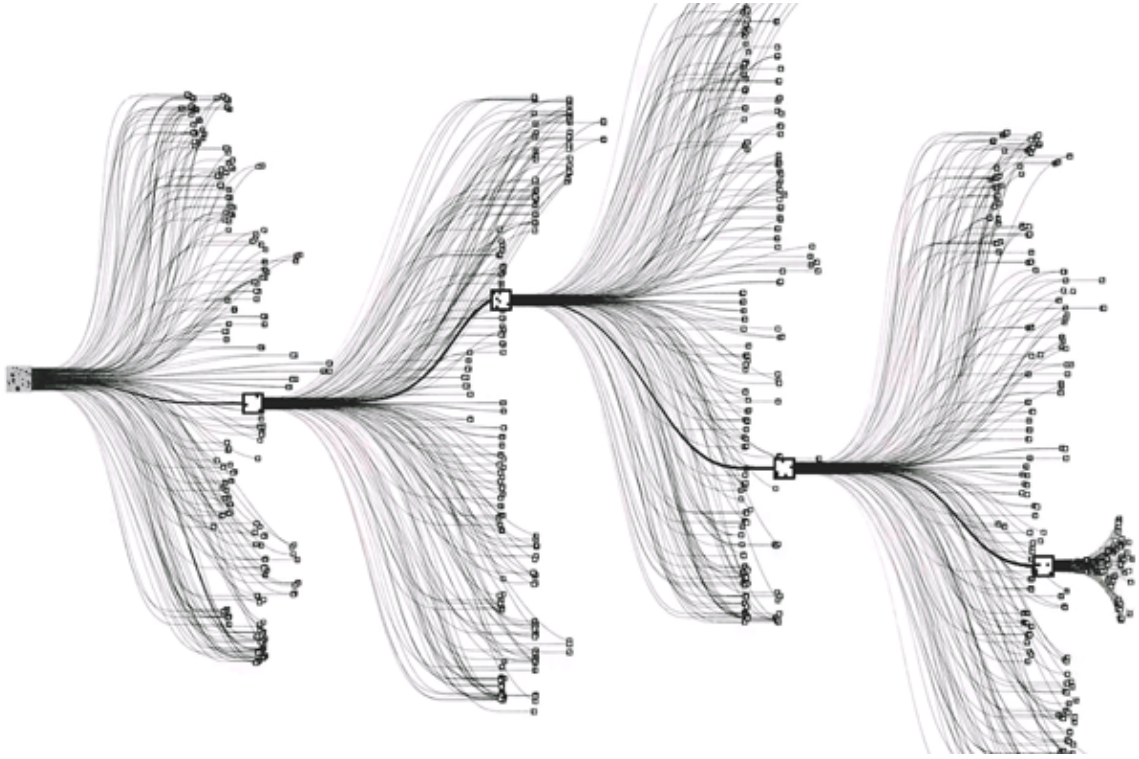
High societal impact (affect billions of people)

Diverse (language, games, robotics)

Complex (really hard)

- What's in common with all of these examples?
- It's clear that AI applications tend to be very **high impact**.
- They are also incredibly **diverse**, operating in very different domains, and requiring integration with many different modalities (natural language, vision, robotics). Throughout the course, we will see how we can start to tame this diversity with a few fundamental principles and techniques.
- Finally, these applications are also mind-bogglingly **complex** to the point where we shouldn't expect to find solutions that solve these problems perfectly.

Two sources of complexity...



Computational complexity: exponential explosion

- There are two sources of complexity in AI tasks.
- The first, which you, as computer scientists, should be familiar with, is **computational complexity**. We can solve useful problems in polynomial time, but most interesting AI problems — certainly the ones we looked at — are NP-hard. We will be constantly straddling the boundary between polynomial time and exponential time, or in many cases, going from exponential time with a bad exponent to exponential time with a less bad exponent.
- For example, in the game of Go, there are up to 361 legal moves per turn, and let us say that the average game is about 200 turns. Then, as a crude calculation, there might be 361^{200} game trajectories that a player would have to consider to play optimally. Of course, one could be more clever, but the number of possibilities would still remain huge.

这是什么意思?

Even infinite computation isn't enough...need to somehow *know* stuff.

Information complexity: need to acquire knowledge

- The second source of complexity, which you might not have thought of consciously, is **information complexity**.
- (Note that there are formal ways to characterize information based on Shannon entropy, but we are using the term information rather loosely here.) Suppose I gave you (really, your program) literally infinite computational resources, locked you (or your program) in a room, and asked you to translate a sentence. Or asked you to classify an image with the type of bird (it's a Weka from New Zealand, in case you're wondering).
- In each of these cases, increasing the amount of computation past a certain point simply won't help. In these problems, we simply need the information or knowledge about a foreign language or ornithology to make optimal decisions. But just like computation, we will be always information-limited and therefore have to simply cope with **uncertainty**.

Resources

Computation (time/memory)

Information (data)

- We can switch vantage points and think about resources to tackle the computational and information complexities.
- In terms of computation, **computers** (fast CPUs, GPUs, lots of memory, storage, network bandwidth) is a resource. In terms of information, **data** is a resource.
- Fortunately for AI, in the last two decades, the amount of computing power and data has skyrocketed, and this trend coincides with our ability to solve some of the challenging tasks that we discussed earlier.

Summary so far

- Potentially transformative impact on society
- Applications are diverse and complex
- Challenges: computational/information complexity

Roadmap

Why AI?

How do we approach it?

Course logistics

Optimization

How do we ~~solve~~ tackle these challenging AI tasks?

How?

- So having stated the motivation for working on AI and the challenges, how should we actually make progress?
- 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**

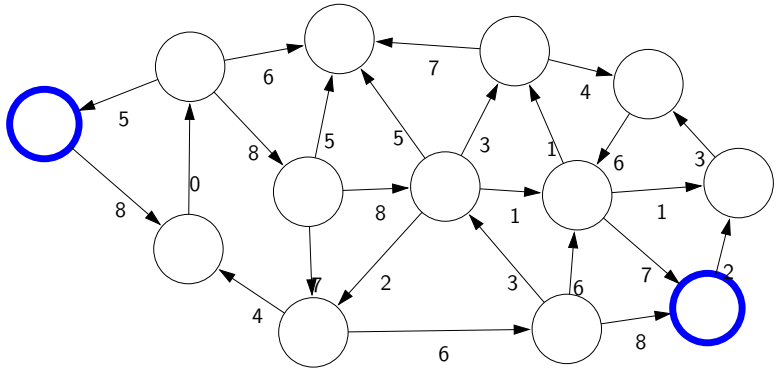
- In this class, we will adopt the **modeling-inference-learning** paradigm to help us navigate the solution space. In reality, the lines are blurry, but this paradigm serves as an ideal and a useful guiding principle.

Paradigm: modeling

Real world

Modeling

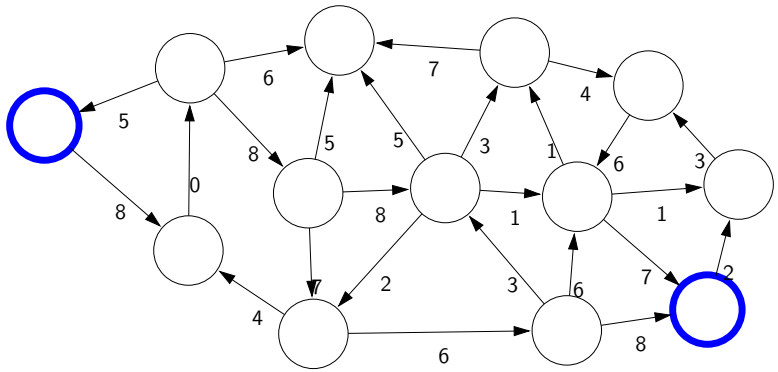
Model



- 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.

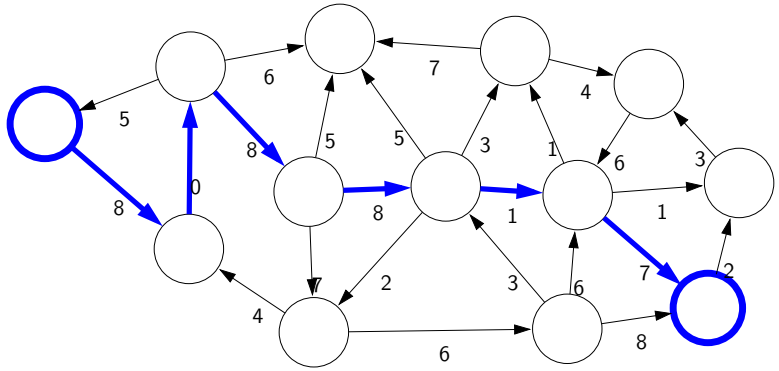
Paradigm: inference

Model



Inference

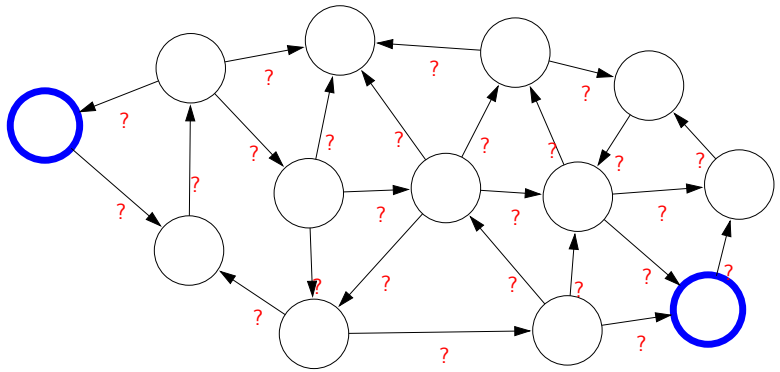
Predictions



- 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

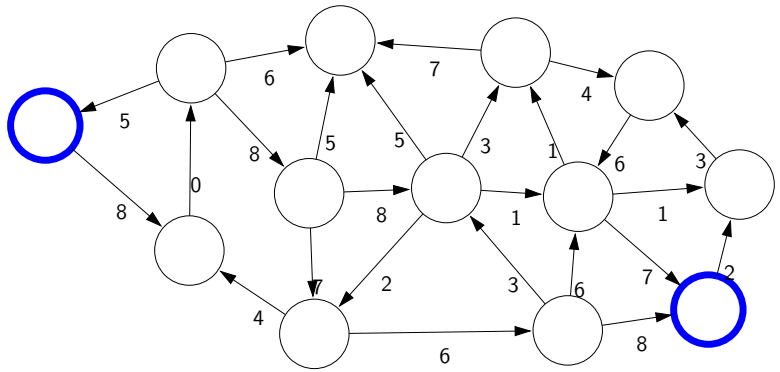
Model without parameters



+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. This is where information complexity rears its head. We can't possibly write down a 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.

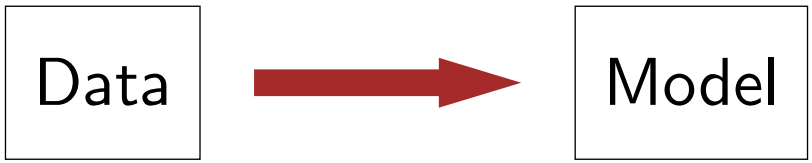
Course plan

"Low-level intelligence" "High-level intelligence"

Machine learning

- 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.

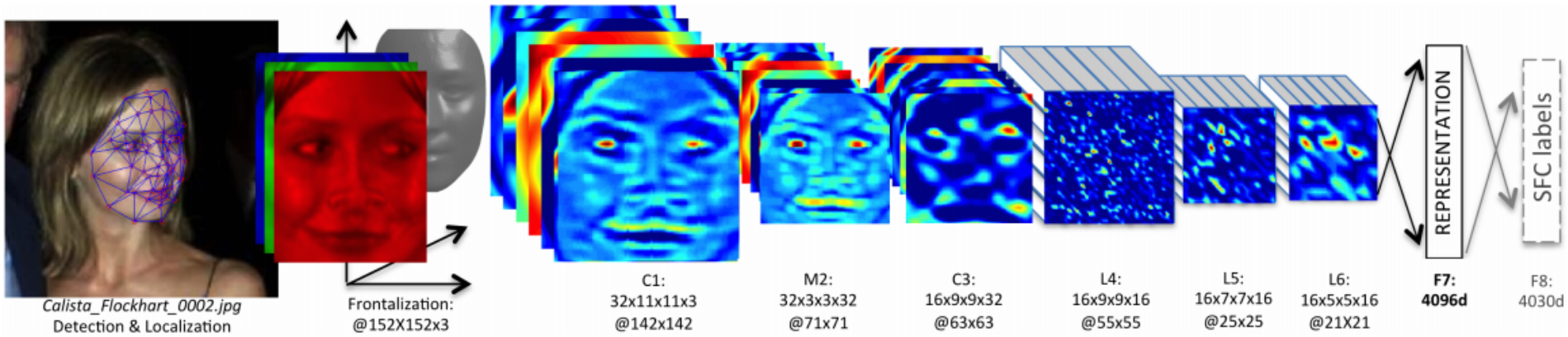
Course plan


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

Search problems

Markov decision processes

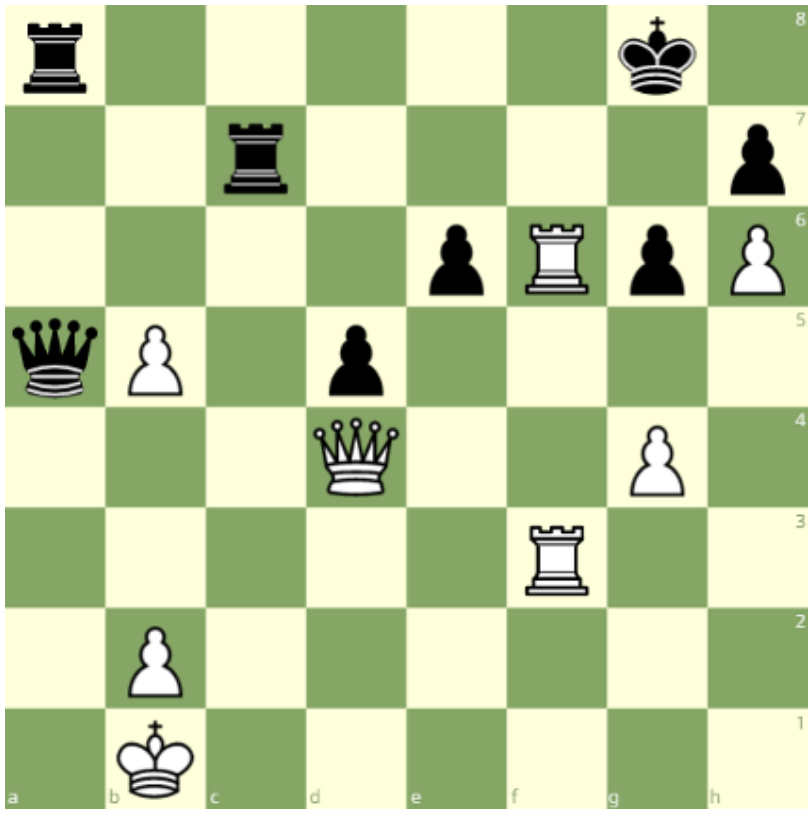
Adversarial games

Reflex **States**

"Low-level intelligence" "High-level intelligence"

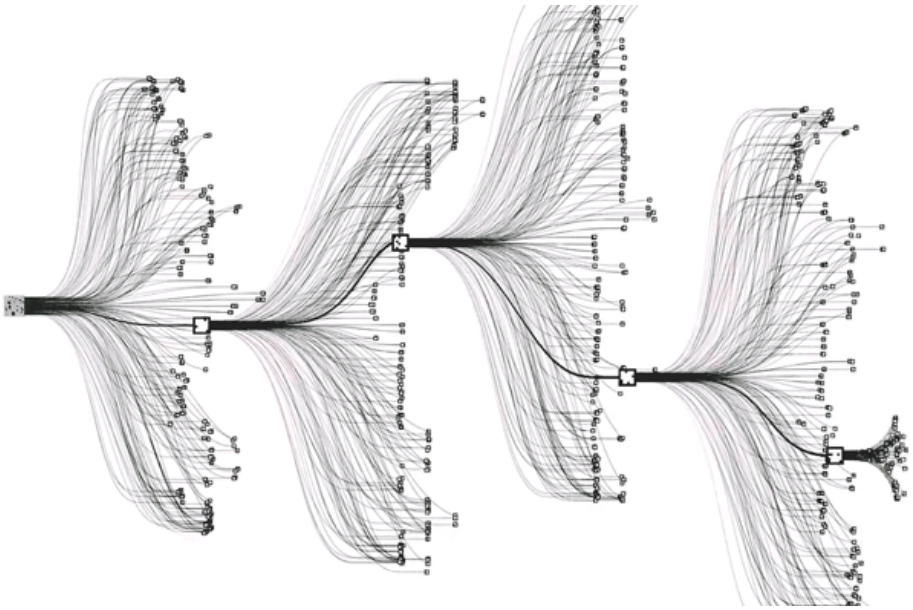
Machine learning

State-based models



White to move

State-based models



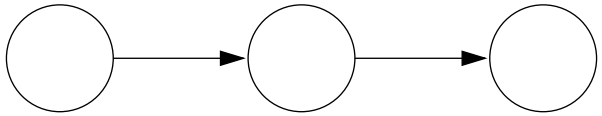
Applications:

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

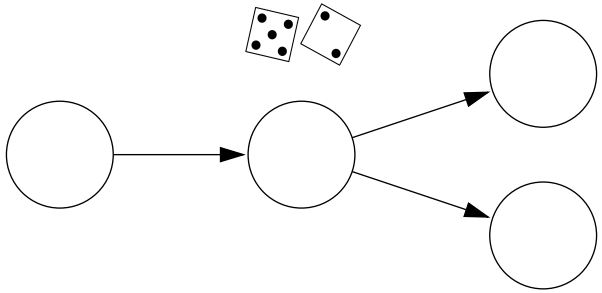
- Reflex-based models are too simple for tasks that require more forethought (e.g., in playing chess or planning a big trip). State-based models overcome this limitation.
- The key idea is, at a high-level, to model the **state** of a world and transitions between states which are triggered by actions. 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

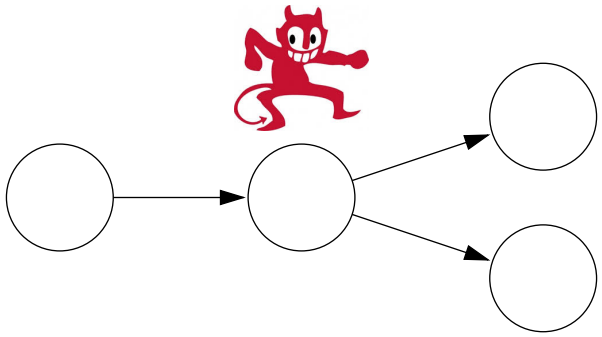
Search problems: you control everything



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

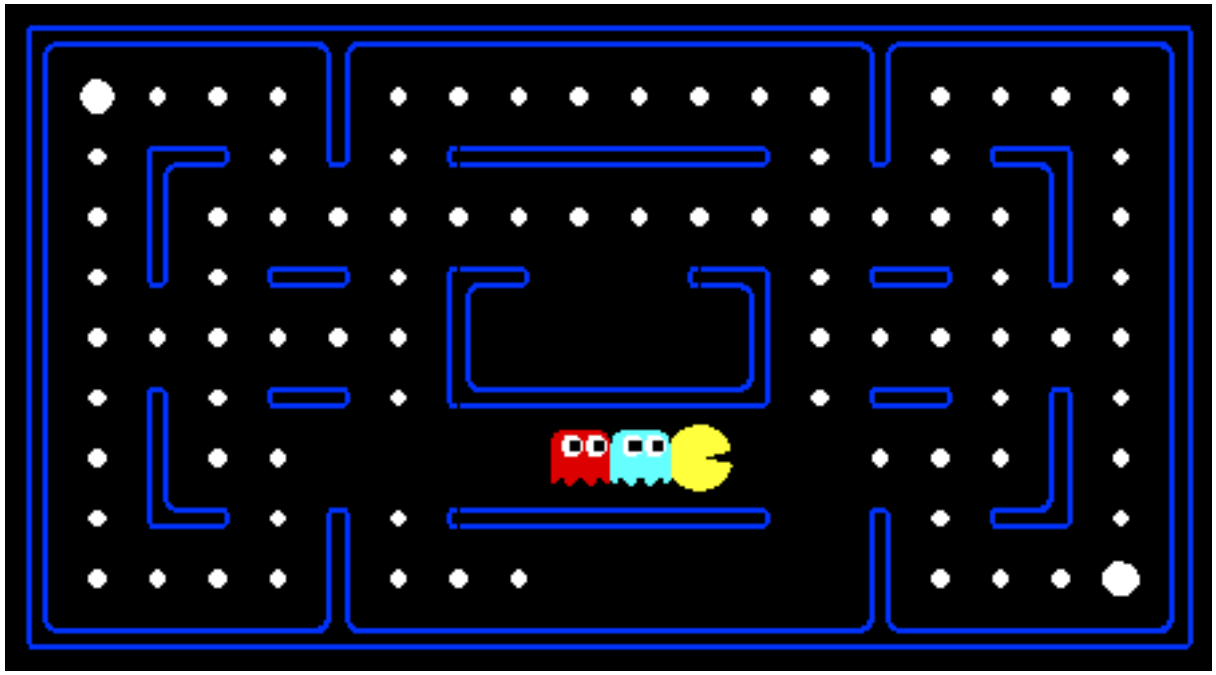


Adversarial games: against opponent (e.g., chess)



- 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]

cs221.stanford.edu/q

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

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Bayesian networks

Reflex **States** **Variables**

"Low-level intelligence" "High-level intelligence"

Machine learning

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

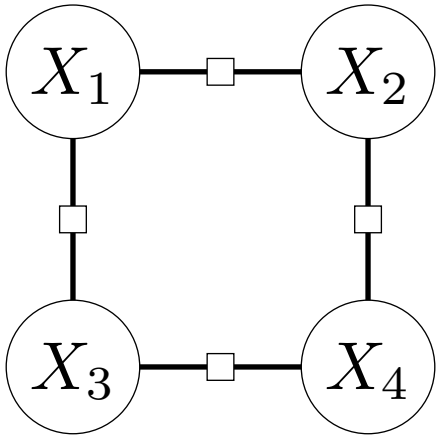
Goal: put digits in blank squares so each row, column, and 3x3 sub-block has digits 1–9

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

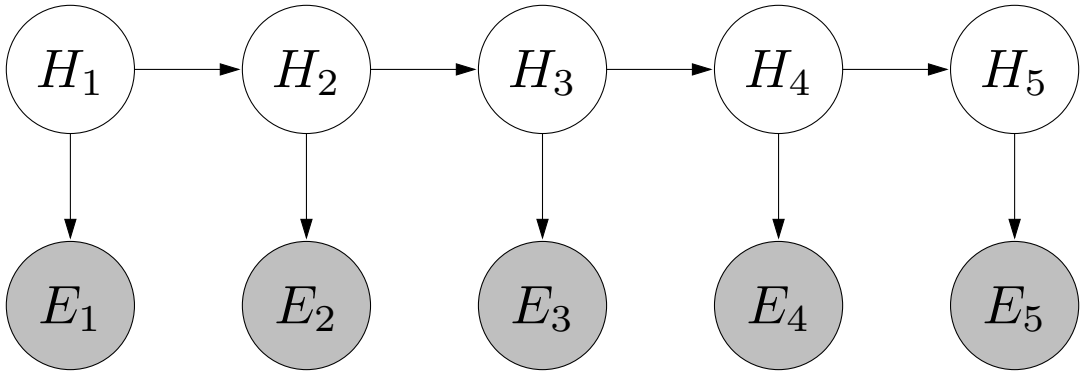
- In state-based models, solutions are procedural: they specify step by step instructions on how to go from A to B. In many applications, the order in which things are done isn't important.

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

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Bayesian networks

Reflex **States** **Variables** **Logic**

"Low-level intelligence" "High-level intelligence"

Machine learning

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

Search problems

Markov decision processes

Adversarial games

Constraint satisfaction problems

Bayesian networks

Reflex **States** **Variables** **Logic**

"Low-level intelligence" "High-level intelligence"

Machine learning

Roadmap

Why AI?

How do we approach it?

Course logistics

Optimization

Course objectives

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

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

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

- Be able to tackle real-world tasks with the appropriate models and algorithms
- 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 sanity checked on basic tests; your grade will be based on hidden test cases

Exam

- Goal: test your ability to use knowledge to solve new problems, not know facts
- All written problems, similar to written part of homeworks
- Closed book except one page of notes
- Covers all material up to and including preceding week
- Tue Nov. 28 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 plagarism.

Roadmap

Why AI?

How do we approach it?

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 provides both a mathematical specification of **what** we want to compute and the algorithms for **how** we 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:

"cat", "cat"	$\Rightarrow$	0
"cat", "dog"	$\Rightarrow$	3
"cat", "at"	$\Rightarrow$	1
"cat", "cats"	$\Rightarrow$	1
"a cat!", "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 $\{(x_1, y_1), \ldots, (x_n, y_n)\}$

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

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

Examples:

$$\begin{aligned} \{(2,4)\} &\Rightarrow 2 \\ \{(2,4),(4,2)\} &\Rightarrow ? \end{aligned}$$

[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 $F(w)$.
- A brief detour to explain the modeling that might lead to this formal task. **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 applications are high-impact and complex
- Modeling [reflex, states, variables, logic] + inference + learning
- Section this Thursday: review of foundations
- Homework [foundations]: due next Tuesday 11pm
- Course will be fast-paced and exciting!