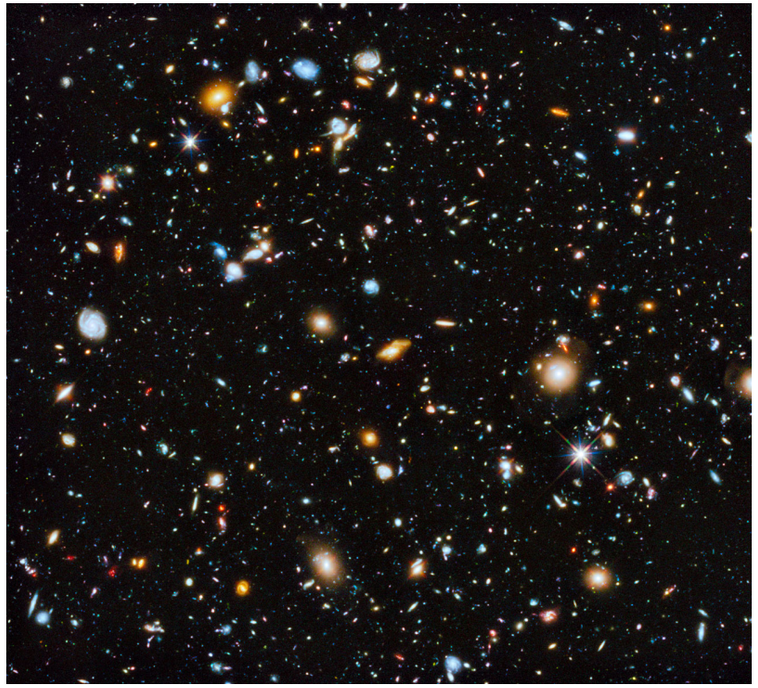
Overview



- Hello and welcome! I'm Dorsa, who along with Percy, will be your instructors for CS221 this quarter.

[image credit: Andrew Brodhead]

- I hope that everyone is staying safe and generally doing okay.
- Needless to say, this is an incredibly tough time, and we have all experienced a difficult year. I am excited that we are finally seeing some light at the end of the tunnel, and coming out of this all stronger.
- I know that many of you are adversely affected, trying to just hanging in there, so thank you for joining us today.
- We are going to hold this class fully online this quarter, but we now have a bit of experience on what works well and what does not, and we hope to use that experience to have a successful offering of CS 221 this quarter.

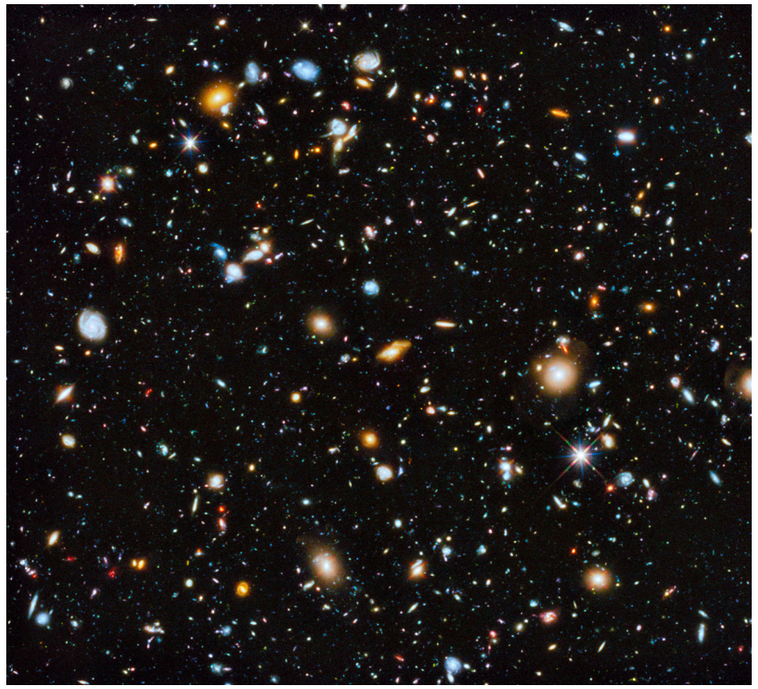
Percy Liang

Dorsa Sadigh

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 with others, only look at input-output behavior.
- We will run MOSS periodically to detect code plagiarism.

General: Course content



- We now talk about the topics in this course.

Bridging the gap

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

    # Insert [state] into the heap with priority [newPriority] if
    # [state] isn't in the heap or [newPriority] is smaller than the existing
    # priority.
    # Return whether the priority queue was updated.
    def update(self, state, newPriority):
        oldPriority = self.priorities.get(state)
        if oldPriority == None or newPriority < oldPriority
            self.priorities[state] = newPriority
            heapq.heappush(self.heap, (newPriority, state))
            return True
        return False

    # Returns (state with minimum priority, priority)
    # or (None, None) if the priority queue is empty.
    def removeMin(self):
        while len(self.heap) > 0
            priority, state = heapq.heappop(self.heap)
            if self.priorities[state] == self.DONE: continue  # Outdated priority, skip
            self.priorities[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 search problem on the number line:
# Start at 0, want to go to 10, costs 1 to move down, 2 to move up.
class NumberLineSearchProblem:
    def startState(self): return 0
    def isGoal(self, state): return state == 10
    def succAndCost(self, state): return [('West', state-1, 1), ('East', state+1, 2)]

# Function to create search problems from a graph.
# You can use this to test your algorithm.
def createSearchProblemFromString(start, goal, description)
    # Parse the graph
    graph = collections.defaultdict(list)
    for line in description.split("\n"):
        if len(line) == 0 or line.startswith('#'): continue
        # Edge from state a to state b.
        a, b, cost = line.split(' ')
        cost = float(cost)
        # Action is the same as the destination state (b).
        graph[a].append((b, b, cost))
```

- Real-world AI problems are complex with lots of unknowns.
- Suppose we're trying to build a system that can navigate through a busy city.
- At the end of the day, we need to write some code to solve this task (and possibly build some hardware, too).
- But there is a huge chasm between the problem and the solution.

Paradigm

Modeling

Inference **Learning**

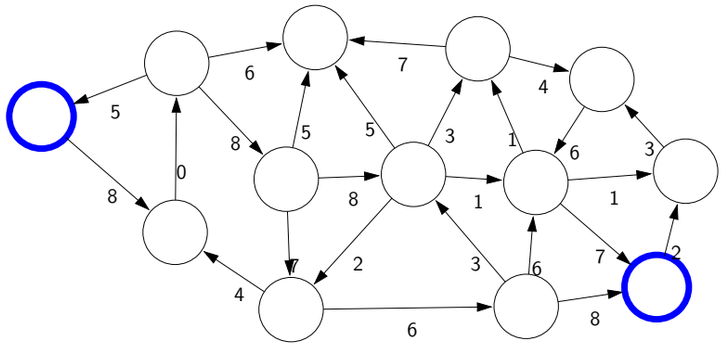
- In this course, we will adopt the **modeling-inference-learning** paradigm to help us navigate the solution space. In reality, the lines are blurry, but this paradigm serves as an ideal and a useful guiding principle.
- There are three pillars, modeling, inference, and learning.

Paradigm: modeling

Real world

Modeling

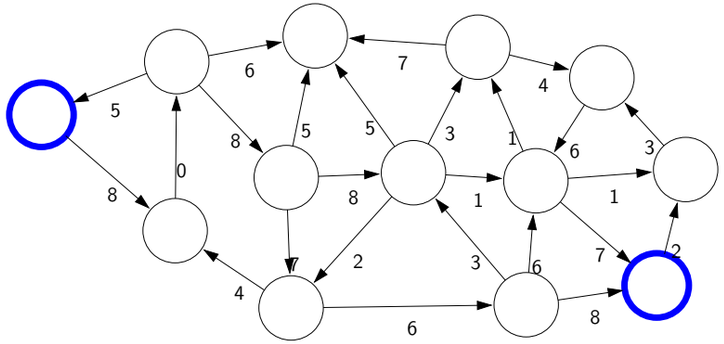
Model



- 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 and can be operated on by computers.
- 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 are games such as Chess, Go, or Sudoku, where the real world is identical to the model.)
- We might formulate the route finding problem as a graph where nodes represent points in the city, edges represent the roads, and the cost of an edge represents the traffic on that road.

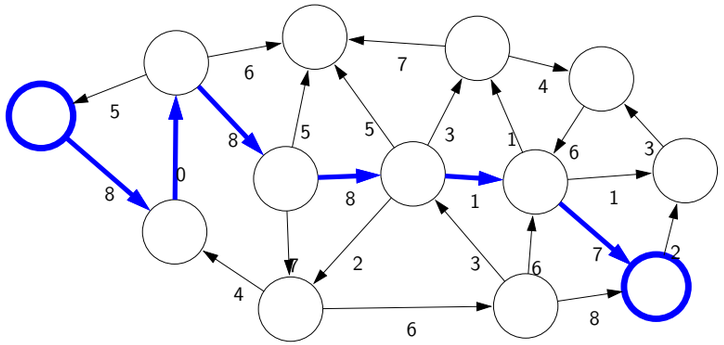
Paradigm: inference

Model



Inference

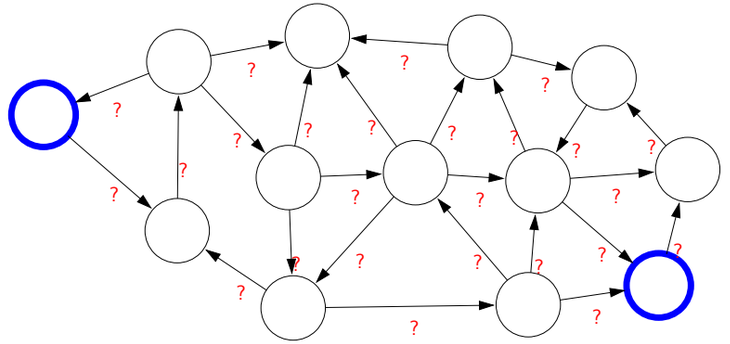
Predictions



- 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?
- The focus of inference is usually on efficient algorithms that can answer these questions.
- For some models, computational complexity can be a concern (games such as Go), and usually approximations are needed.

Paradigm: learning

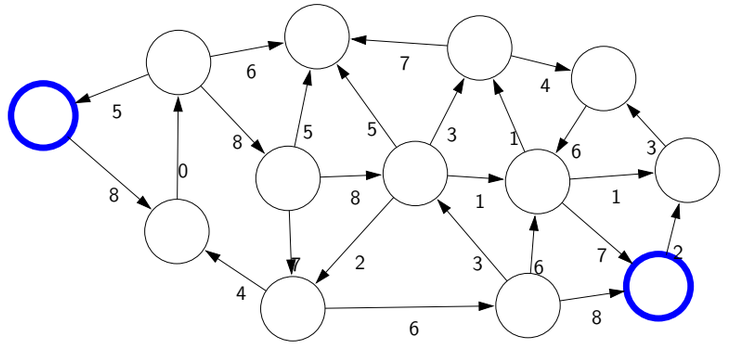
Model without parameters



+data

Learning

Model with parameters



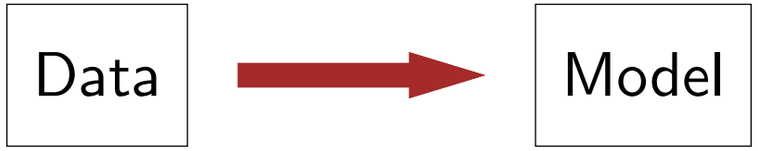
- But where does the model come from? Remember that the real world is rich, so if the model is to be faithful, the model has to be rich as well. But we can't possibly write down such a rich model manually.
- The idea behind (machine) **learning** is to instead get it from data. Instead of constructing 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.
- Note that learning here is not tied to a particular model family (e.g., neural networks), but it is more of a philosophy.
- This general paradigm will allow us to bridge the gap between logical AI and statistical AI.

Course plan

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"
- 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. From a systems engineering perspective, 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 be formalized using tools from probability and statistical learning theory.

Course plan

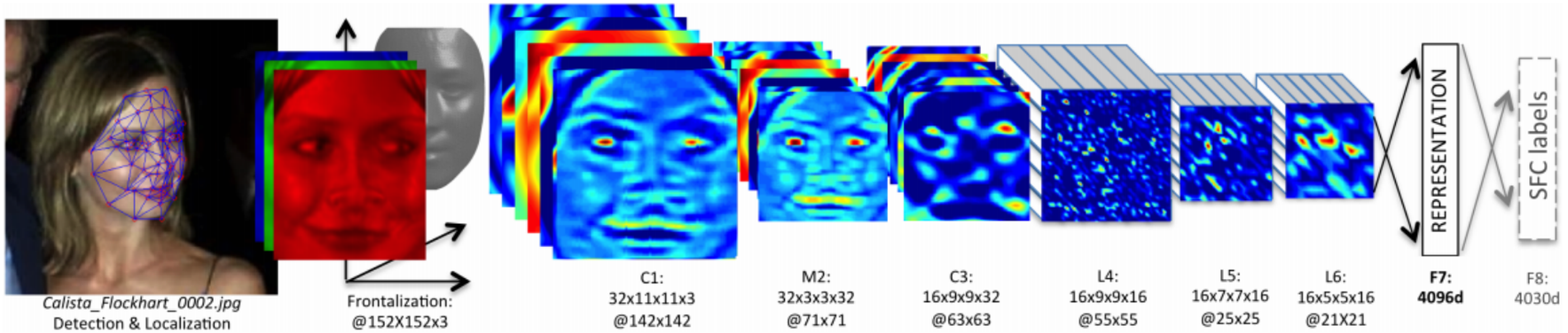
Machine learning

What is this animal?

- Most of you could probably recognize the zebra in that split second.

Reflex-based models

- Examples: linear classifiers, deep neural networks



- Most common models in machine learning

- Fully feed-forward (no backtracking)

- 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

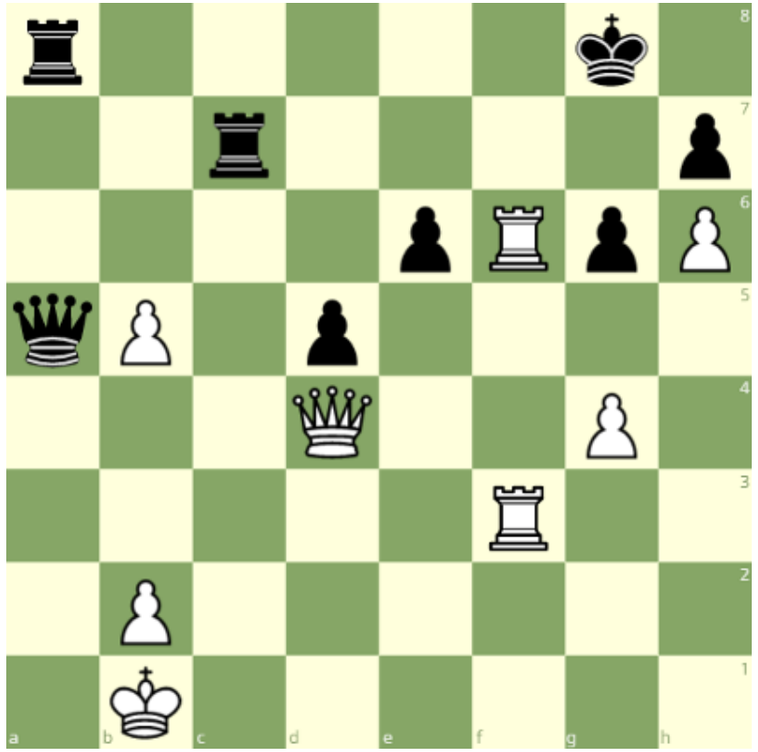
Reflex **States**

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

Machine learning

- Next, we will consider state-based models.

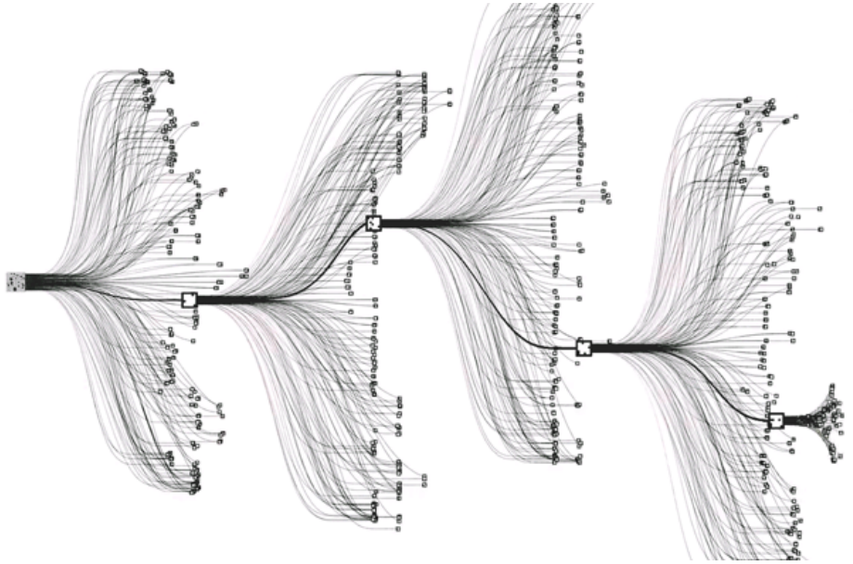
State-based models



White to move

- Consider the task of figuring out what move white should make given a particular chess position.
- Most of us will find this more challenging than recognizing the zebra.

State-based models



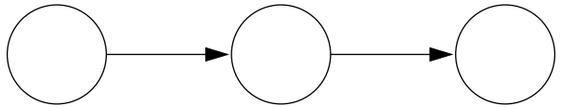
Applications:

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

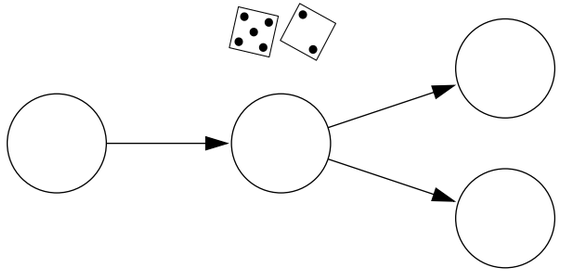
- Reflex-based models are too simple for tasks that require more forethought (e.g., in playing chess or planning a big trip). State-based models overcome this limitation.
- The key idea is, at a high-level, to model the **state** of a world and transitions between states which are triggered by actions.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

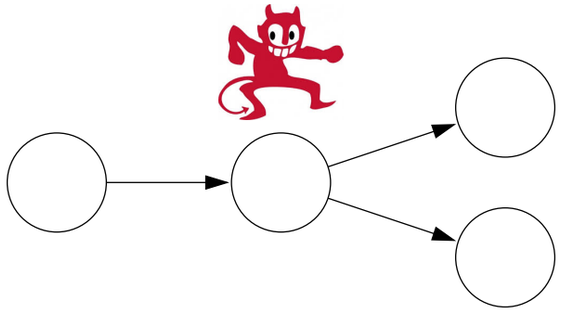
Search problems: you control everything



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

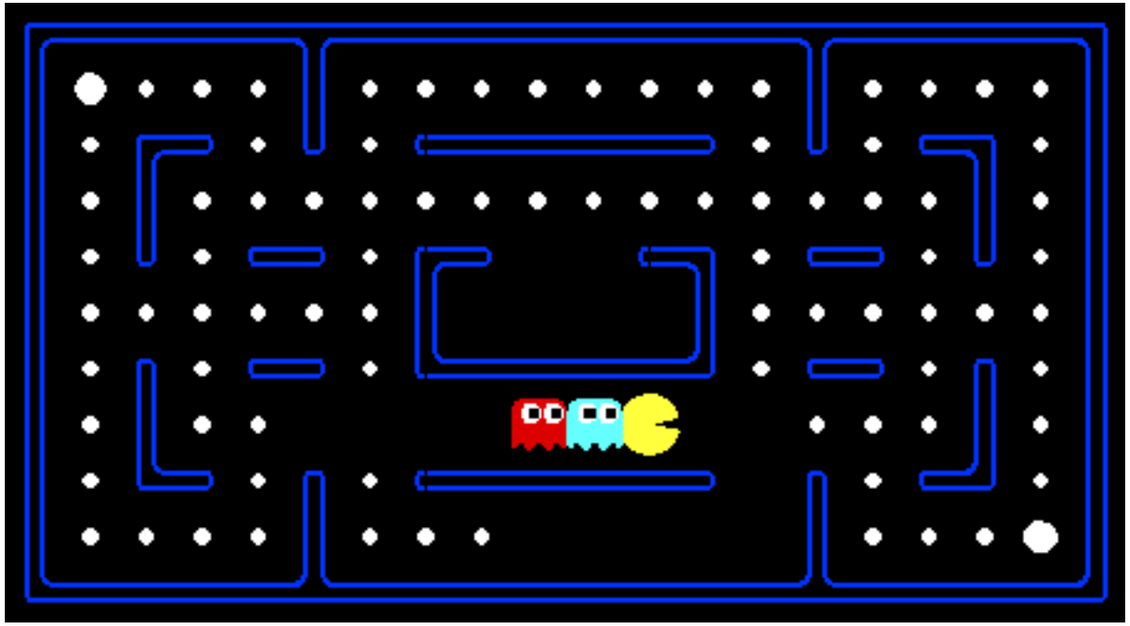


Adversarial games: against opponent (e.g., chess)



- **Search problems** are adequate models when you are operating in an 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).

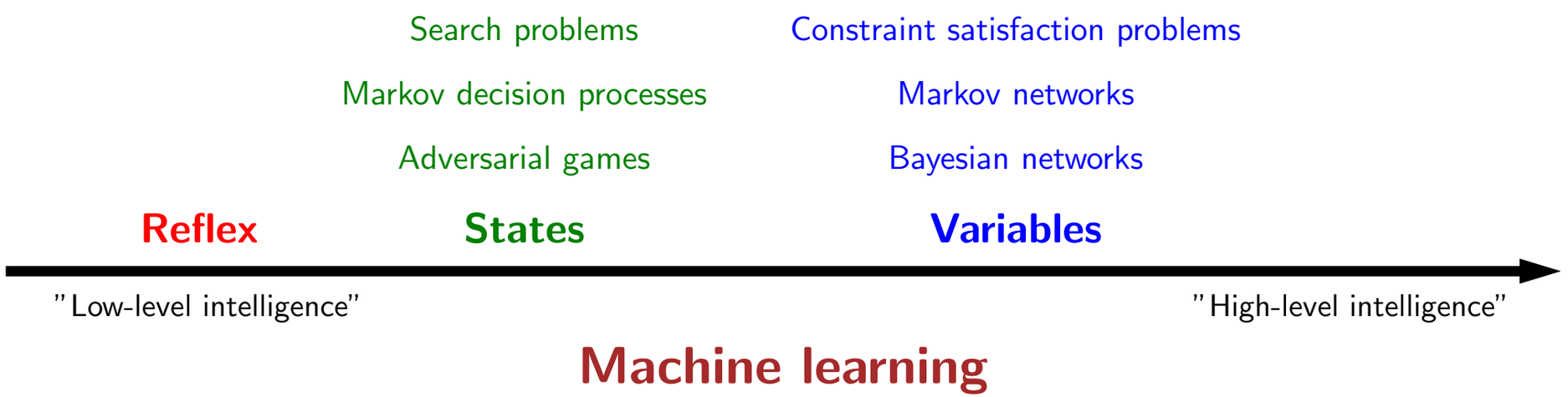
Pac-Man



[demo]

- In one of the homeworks, you will build an agent that can play Pac-Man.
- To whet your appetite, this is what it will look like.
- (demo)
- Think about what the states of this model should be, and how you might come up with the optimal strategy.

Course plan


Search problems
Constraint satisfaction problems
Markov decision processes
Markov networks
Adversarial games
Bayesian networks
Reflex
States
Variables
"Low-level intelligence"
"High-level intelligence"
Machine learning

- Next, we will talk about variable-based models.

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

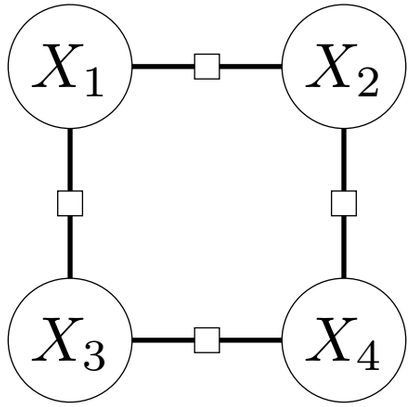
Goal: put digits in blank squares so each row, column, and 3x3 sub-block has digits 1–9

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

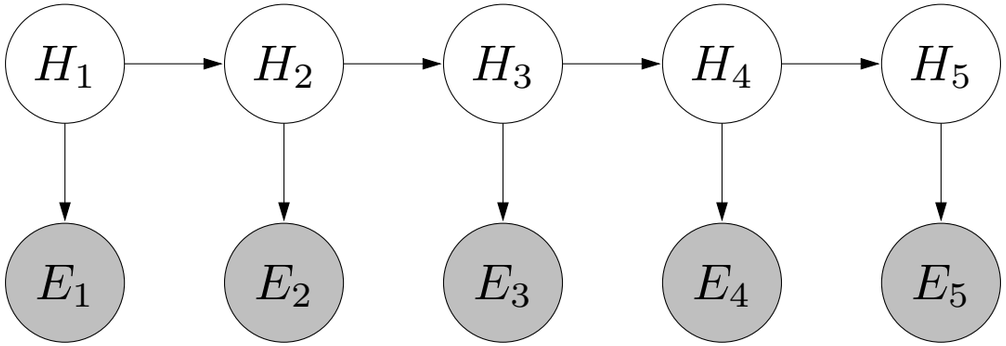
- In state-based models, solutions are procedural: they specify step by step instructions on how to go from A to B.
- In some applications, the order in which things are done isn't important.
- For example, in Sodoku, where the goal is to put digits in the blank squares to satisfy uniqueness constraints, all that matters is the final configuration of numbers; you can fill them in in any order.
- This flexibility allows us to devise more efficient algorithms.

Variable-based models

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

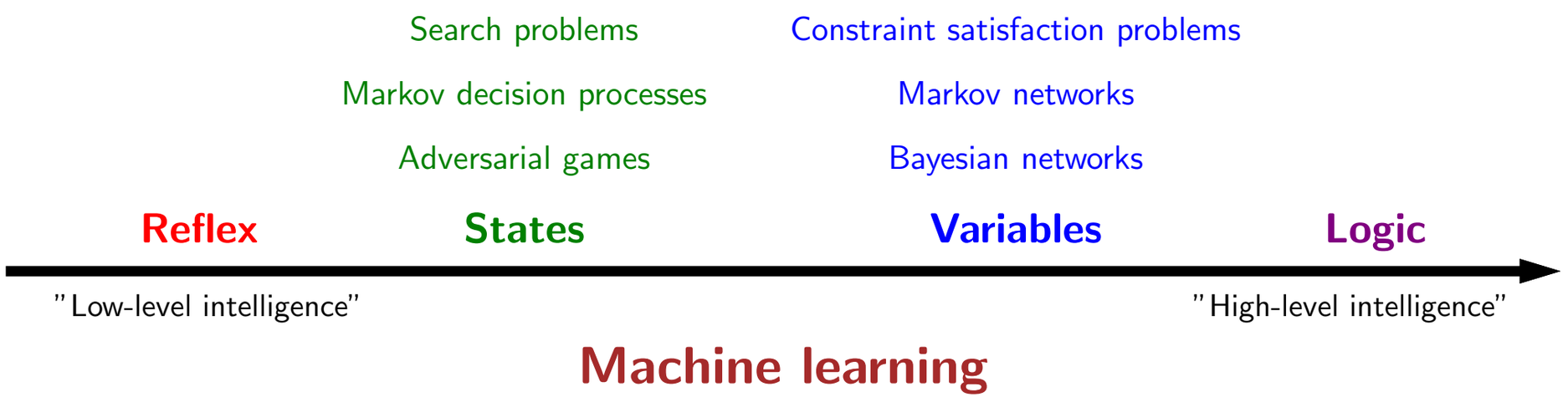


Bayesian networks: soft dependencies (e.g., tracking cars from sensors)



- **Constraint satisfaction problems** are variable-based models where we only have hard constraints. For example, in scheduling, one person can't be in two places at once.
- **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 and it formally defines a joint probability distribution over all of the variables.

Course plan


Search problems
Constraint satisfaction problems
Markov decision processes
Markov networks
Adversarial games
Bayesian networks
Reflex
States
Variables
Logic
"Low-level intelligence"
"High-level intelligence"
Machine learning

- The last topic is logic.

Motivation: virtual assistant

Tell information

Ask questions

Use natural language!

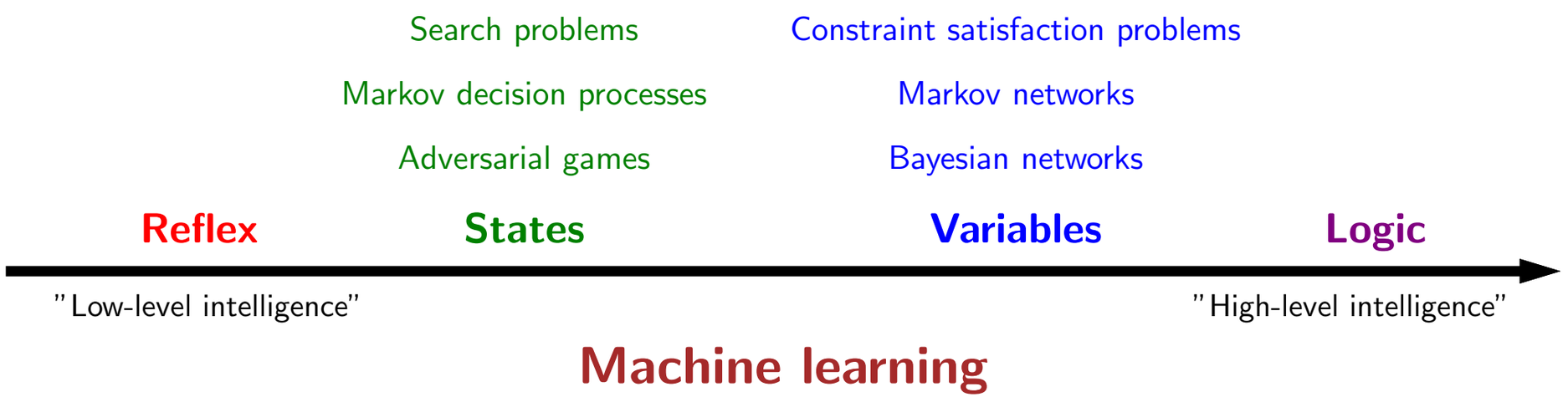
[demo]

Need to:

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

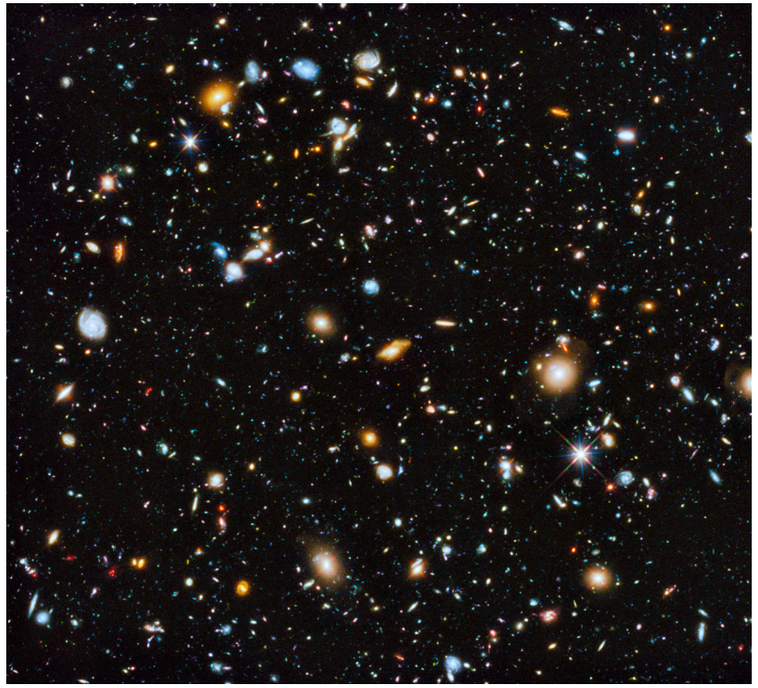
- One motivation for logic is a virtual assistant. A good assistant should be able to remember what you told it and answer questions that require drawing inferences from its knowledge. And you'd want to interact with it using natural language, the tool that humans invented for communication.
- I'll show you a demo which you'll have an opportunity to play with in the final homework.
- (demo)
- Interacting with this system feels very different than a typical machine learning-based system. First, it is adaptive, whereas most ML systems are a fixed function. Second, the types of information we provide it and the types of questions are more heterogenous and more abstract, and we expect the system to realize the full consequences of every single word. (We don't want to tell our personal assistant 100 times that we prefer morning meetings.)
- One often contrasts logical AI and statistical AI. In this course, we will treat the two as not contradictory but rather complementary. Logic provides a class of models which is more expressive, but still needs to be supported by the robustness of machine learning.

Course plan


Search problems
Constraint satisfaction problems
Markov decision processes
Markov networks
Adversarial games
Bayesian networks
Reflex
States
Variables
Logic
"Low-level intelligence"
"High-level intelligence"
Machine learning

- And this concludes our tour of the topics.
- To summarize, we will discuss models of increasing expressive power (from reflex to state-based models to variable-based models to logic), and for each, we will instantiate the modeling-inference-learning paradigm.

General: AI history



- I will present a short history of AI, which will necessarily be simplified and incomplete. But, I hope that it will give you a general appreciation of AI's multi-faceted history.

. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

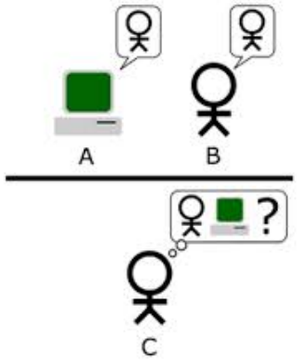
PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

By A. M. Turing

1. *The Imitation Game.*

I propose to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to



objective specification

Many people think that a very abstract activity, like the playing of chess, would be best. It can also be maintained that it is best to provide the machine with the best **sense organs** *that money can buy, and then teach it to understand and speak English. This process could follow the normal* **teaching of a child**. *Things would be pointed out and named, etc. Again I do not know what the right answer is, but I think both approaches should be tried.*

- A natural place to start talking about the history of AI is Alan Turing's landmark paper published in 1950 called Computing Machines and Intelligence.
- In this paper, Turing asked the question, "Can machines think?" and answered it with the Imitation Game, more commonly known as the Turing Test. As some of you might know, a machine is said to pass the Turing test if it can convince a human judge that it's actually a human through natural language dialogue.
- This paper is remarkable not because it built a system or proposed any methods, but because it framed the philosophical discussions for decades to come. You have to appreciate how difficult a notion like intelligence is to pin down. So this was really the first actionable, formal answer to the question, "Can machines think?"
- Whether passing the Turing test is something that should be directly worked on is questionable and controversial, but the philosophical implications are quite thought-provoking.
- For us, one important takeaway of the Turing test is the separation of the **objective** specification of what we want a system to do (the "what") from the methods that might get us there (the "how"). This decoupling is a major theme throughout this course.
- At the end of the paper, Turing discusses two possible approaches. The first is based on solving abstract problems like chess, which is the route taken by symbolic AI. The second is where you build a machine and teach like a child, which is the route taken by neural and statistical AI.
- I will now tell three stories of symbolic, neural, and statistical AI.

1956

- 1956 is the beginninig of our first story.

Birth of AI

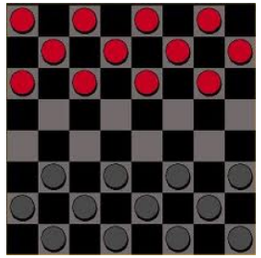
1956: John McCarthy organized workshop at Dartmouth College

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

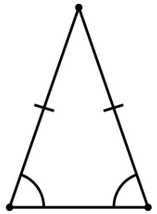
general principles

- It is the year that the name **artificial intelligence** was coined.
- John McCarthy, who later founded the Stanford AI lab, organized a workshop at Dartmouth College that summer.
- In addition to McCarthy, the workshop was attended by Marvin Minsky, Allen Newell, Herbert Simon, etc., all of whom went on to make seminal contributions in AI.
- The participants laid out a bold proposal: to build a system that could capture every aspect of intelligence. They were after **generality**.
- Indeed, during this post-war era, computers were just coming on the scene. It was a very exciting time and people were ambitious.

Birth of AI, early successes



Checkers (1952): Samuel's program learned weights and played at strong amateur level



Problem solving (1955): Newell & Simon's Logic Theorist: prove theorems in Principia Mathematica using search + heuristics; later, General Problem Solver (GPS)

- A few notable systems were created during this time.
- Arthur Samuel wrote a program that could play checkers at a strong amateur level.
- Alan Newell and Herbert Simon's Logic Theorist could prove theorems. For one theorem, it actually found a proof that was more elegant than the human-written proof. They tried to publish a paper on the result, but the paper got rejected because it was not a new theorem. Perhaps the reviewers failed to realize that the third author was actually a computer program.
- Later, they developed the General Problem Solver, which promised to solve any problem (which could be suitably encoded in logic), again carrying forward the ambitious "general intelligence" agenda.

Overwhelming optimism...

Machines will be capable, within twenty years, of doing any work a man can do.

Within 10 years the problems of artificial intelligence will be substantially solved.

I visualize a time when we will be to robots what dogs are to humans, and I'm rooting for the machines.

- With these initial success, it was a time of high optimism, with all the leaders of the field, all impressive thinkers, predicting that AI would be "solved" in a matter of years.

...underwhelming results

Example: machine translation

The spirit is willing but the flesh is weak.

(Russian)

The vodka is good but the meat is rotten.

1966: ALPAC report cut off government funding for MT, first AI winter

- Despite the successes, certain tasks such as machine translation were complete failures.
- There is a folklore story of how the sentence "The spirit is willing but the flesh is weak" was translated into Russian and then back to English, leading to the amusing translation "The vodka is good but the meat is rotten".
- However, this translation was not so amusing to government agencies funding the research. In 1966, the ALPAC report resulted in funding being cut off for machine translation.
- This marked the beginning of the first AI winter.

Implications of early era

Problems:

- **Limited computation**: search space grew exponentially, outpacing hardware
- **Limited information**: complexity of AI problems (number of words, objects, concepts in the world)

Useful contributions (John McCarthy):

- Lisp
- Garbage collection
- Time-sharing

- What went wrong? Two things.
- The first was computation. Most of the approaches casted problems as logical reasoning, which required a search over an exponentially large search space. The hardware at the time was simply too limited.
- The second is information. Even if researchers had infinite computation, AI would not have been solved. There are simply too many concepts, words, and objects in the world, and this information has to somehow be put into the AI system.
- Though the grand ambitions were not realized, some generally useful technologies came out of the effort. Lisp was way ahead of its time in terms of having advanced language features. People programming in high-level languages like Python take garbage collection for granted. And the idea that a single computer could simultaneously be used by multiple people (time sharing) was prescient.

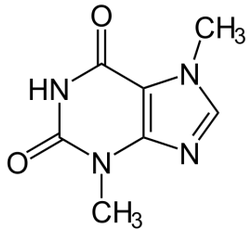
Knowledge-based systems (70-80s)

Expert systems: elicit specific domain knowledge from experts in form of rules:

```
if [premises] then [conclusion]
```

- In the 1970s and 80s, AI researchers looked to knowledge as a way to combat both the computation and information limitations of the previous era.
- At this time, expert systems became fashionable, where a domain expert would encode their domain expertise in these systems, usually in the form of if-then rules.

Knowledge-based systems (70-80s)



DENDRAL: infer molecular structure from mass spectrometry

MYCIN: diagnose blood infections, recommend antibiotics

XCON: convert customer orders into parts specification

- There was also a noticeable shift in focus. Instead of the solve-it-all optimism from the 1950s and 60s, researchers focused on building narrow practical systems in targeted domains.
- Famous examples from this era included systems for chemistry, medical diagnosis, and business operations.

Knowledge-based systems

Wins:

- Knowledge helped both the **information** and **computation** gap
- First **real application** that impacted industry

Problems:

- Deterministic rules couldn't handle the **uncertainty** of the real world
- Rules quickly became too **complex** to create and maintain

A number of people have suggested to me that large programs like the SHRDLU program for understanding natural language represent a kind of dead end in AI programming. Complex interactions between its components give the program much of its power, but at the same time they present a formidable obstacle to understanding and extending it. In order to grasp any part, it is necessary to understand how it fits with other parts, presents a dense mass, with no easy footholds. Even having written the program, I find it near the limit of what I can keep in mind at once. — Terry Winograd

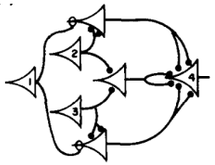
1987: Collapse of Lisp machines and second AI winter

- What knowledge (in addition to the restriction to narrow domains) did was not only providing information to the system, but it also helped alleviate the need for as much computation, by placing constraints on the space of possibilities.
- Also, this was the first time AI had a real impact on industry, rather than being just an academic's playground.
- However, knowledge engineering ran into major limitations. First, deterministic rules failed to capture the uncertainty in the real world, thought there were attempts to patch this heuristically as an afterthought.
- Second, these systems were just too much work to create and maintain, making it hard to scale up to more complex problems.
- Terry Winograd built a famous dialogue system called SHRDLU summed up well by the sentiment in this quote: the complex interactions between all the components made it too hard for mortals to even grasp. After that, he moved to Stanford and became an HCI professor.
- During the 80s, there was again a lot of overpromising and underdelivering, the field collapsed again. It seemed like history was repeating itself.
- We will now leave the story of symbolic AI, which dominated AI for multiple decades...

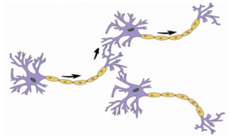
1943

- ...and go back in time to to 1943 to tell the story of neural AI.

Artificial neural networks



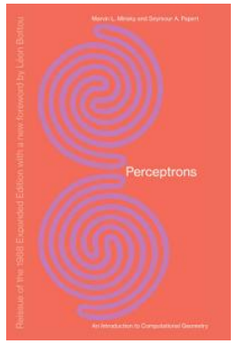
1943: artificial neural networks, relate neural circuitry and mathematical logic (McCulloch/Pitts)



1949: "cells that fire together wire together" learning rule (Hebb)

1958: Perceptron algorithm for linear classifiers (Rosenblatt)

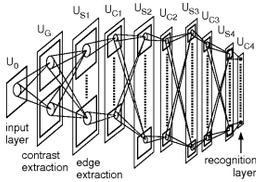
1959: ADALINE device for linear regression (Widrow/Hoff)



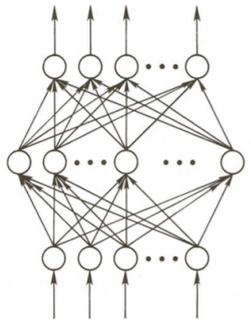
1969: Perceptrons book showed that linear models could not solve XOR, killed neural nets research (Minsky/Papert)

- In 1943, neurophysiologist Warren McCulloch and logician Walter Pitts devised a simple mathematical model of a neuron, giving birth to the field of (artificial) neural networks.
- They showed how this model could compute arbitrary logical functions (and, or, not, etc.), but did not suggest a method for learning this model.
- In 1949, neuropsychologist Donald Hebb introduced the first learning rule. It was based on the intuition that cells that fire together wire together. This rule was nice in that it was local, but it was unstable and so didn't really work.
- In 1958, Frank Rosenblatt developed the Perceptron algorithm for learning single-layer networks (a.k.a. linear classifiers), and built a device that could recognize simple images.
- In 1959, Bernard Widrow and Ted Hoff came up with ADALINE, a different learning rule corresponding to linear regression. A multi-layer generalization called MADALINE was used later to eliminate echo on phone lines, one of the first real-world applications of neural networks.
- 1969 was an important year. Marvin Minsky and Seymour Papert published a book that explored various mathematical properties of Perceptrons. One of the (trivial) results was that the single-layer version could not represent the XOR function. Even though this says nothing about the capabilities of deeper networks, the book is largely credited with the demise of neural networks research, and the continued rise of symbolic AI.

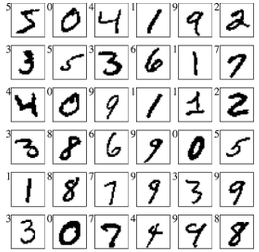
Revival of connectionism



1980: Neocognitron, a.k.a. convolutional neural networks for images (Fukushima)



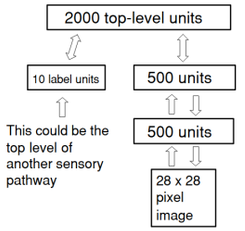
1986: popularization of backpropagation for training multi-layer networks (Rumelhardt, Hinton, Williams)



1989: applied convolutional neural networks to recognizing handwritten digits for USPS (LeCun)

- In the 1980s, there was a renewed interest in neural networks under the banner of connectionism, and there were many new links to psychology and cognitive science.
- The Neocognitron developed by Kunihiko Fukushima was the first convolutional neural network, with multiple layers and pooling. It was trained in a rather heuristic way.
- Donald Rumelhardt, Geoff Hinton, and Ronald Williams rediscovered (yet again) and popularized backpropagation as a way to train multi-layer neural networks, and showed that the hidden units could capture interesting representations.
- Yann LeCun built a system based on convolutional neural networks to recognize handwritten digits. This was deployed by the USPS to recognize zip codes, marking one of the first success stories of neural networks.

Deep learning



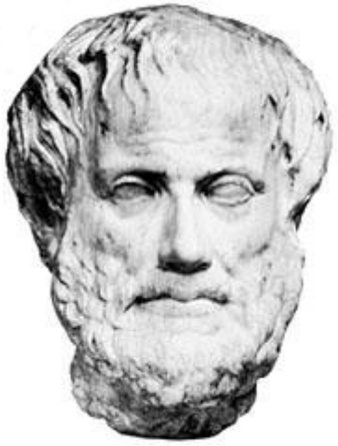
2006: unsupervised layerwise pre-training of deep networks (Hinton et al.)

2012: AlexNet obtains huge gains in object recognition; transformed computer vision community overnight

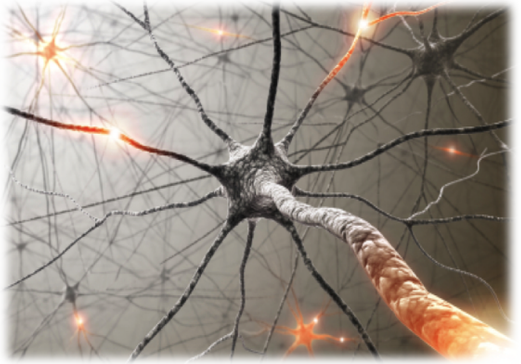
2016: AlphaGo uses deep reinforcement learning, defeat world champion Lee Sedol in Go

- But until the mid-2000s, neural network research was still quite niche, and they were still notoriously hard to train. In 2006, this started changing when Geoff Hinton and colleagues published a paper showing how deep networks could be trained in an unsupervised manner, and then fine-tuned on a small amount of labeled data. The term deep learning started around this time. This "pre-training" technique is ubiquitous today.
- The real break for neural networks came in the 2010s. In 2012, Alex Krizhvesky, Ilya Sustkever, and Geoff Hinton trained a landmark convolutional neural network called AlexNet, which resulted in massive improvements on the ImageNet benchmark, turning the skeptical computer vision community into believers almost instantaneously.
- In 2016, DeepMind's AlphaGo was another turning point. By defeating humans at Go, a feat that many experts thought was still a few decades away, deep learning firmly established itself as the dominant paradigm in AI.

Two intellectual traditions



symbolic AI



neural AI

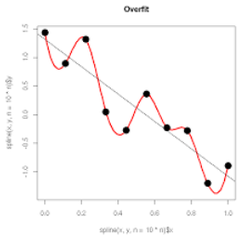
Food for thought: deep philosphical differences, but deeper connections (McCulloch/Pitts, AlphaGo)?

- So far, we've seen two intellectual traditions, symbolic AI, with roots in logic and neural AI, with roots in neuroscience.
- While the two have fought fiercely over deep philosophical differences, perhaps there are deeper connections.
- For example, McCulloch and Pitts' work from 1943 can be viewed as the root of deep learning, but that paper is mostly about how to implement logical operations.
- The game of Go can be perfectly characterized by a set of simple logic rules. But AlphaGo did not tackle the problem directly using logic and instead leveraged the pattern matching capabilities of artificial neural networks.

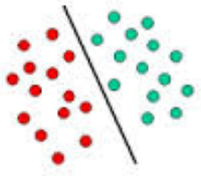
1801

- But there's a third and final story we must tell to complete the picture. This story is not really about AI per se, but rather the influx of certain other areas that have helped build a solid mathematical foundation for AI. This **statistical AI** perspective is also how we will frame the topics in this course.

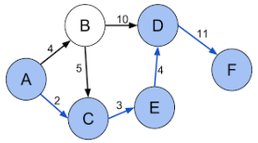
Early ideas from outside AI



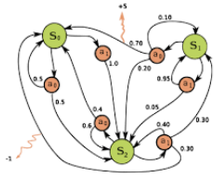
1801: linear regression (Gauss, Legendre)



1936: linear classification (Fisher)



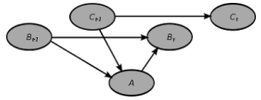
1956: Uniform cost search for shortest paths (Dijkstra)



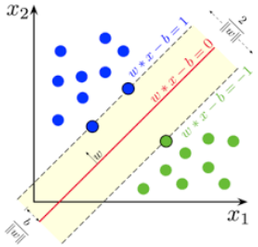
1957: Markov decision processes (Bellman)

- The idea of fitting models from data, which is at the heart of machine learning and modern AI, goes back to as far as Gauss and Legendre, who developed the principle of least squares for linear regression.
- Classification (linear discriminant analysis) was developed by Fisher in statistics.
- In general, machine learning has quite a bit of overlap with the statistics and data mining communities, who worked on solving concrete problems without the lofty goals of "intelligence".
- Besides machine learning, AI consists of sequential decision making problems. Along these lines, there's Dijkstra's algorithm for finding shortest paths for deterministic settings.
- Bellman developed Markov decision processes in the context of control theory, which handles uncertainty in the world.
- Note that these developments largely predated AI.

Statistical machine learning



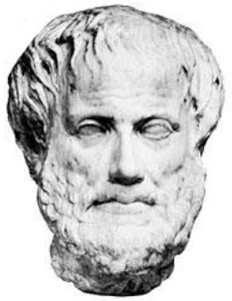
1985: Bayesian networks (Pearl)



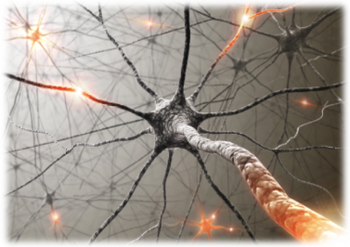
1995: Support vector machines (Cortes/Vapnik)

- You might have noticed that our story of symbolic AI ended at the end of the 1980s, but neural AI only became widespread in the 2010s.
- This is because for much of the 1990s and 2000s, the term AI wasn't actually used as much as it is today, partly to put distance between the most recent failed attempts in symbolic AI and partly because the goals were more down-to-earth.
- People talked about **machine learning** instead, and during that time period, machine learning was dominated by two paradigms.
- The first is Bayesian networks, developed by Judea Pearl, which provides an elegant framework for **reasoning under uncertainty**, something that symbolic AI didn't have a satisfying answer for.
- The second is Support Vector Machines (SVMs), which originated from statistical learning theory and optimization. SVMs were easier to tune than neural networks and became the favored tool in machine learning.

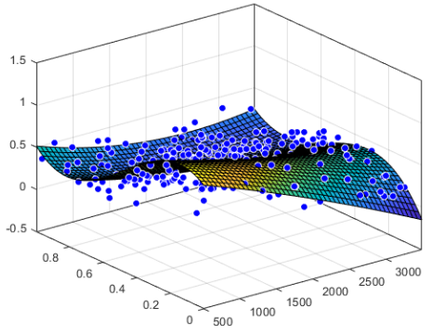
Three intellectual traditions



symbolic AI



neural AI



statistical AI

- This concludes our tour of the three stories that make up what AI is today.
- **Symbolic AI** took a top-down approach and failed to fulfill its original promise. But it offered a vision and did built impressive artifacts for ambitious problems like question answering and dialogue systems along the way.
- **Neural AI** took a completely different approach, proceeding bottom-up, starting with simple perceptual tasks, which the symbolic AI community wasn't interested in. It offered a class of models, deep neural networks, that with today's data and computing resources, has proven capable of conquering ambitious problems.
- Finally, **statistical AI** foremost offers mathematical rigor and clarity. For example, we define an objective function separate from the optimization algorithm, or have a language to talk about model complexity in learning. This course will be largely presented through the lens of statistical AI.
- Stepping back, the modern world of AI is like New York City—it is a melting pot that has drawn from many different fields ranging from statistics, algorithms, neuroscience, optimization, economics, etc. And it is the symbiosis between these fields and their application to important real-world problems that makes working in AI so rewarding.

Further reading

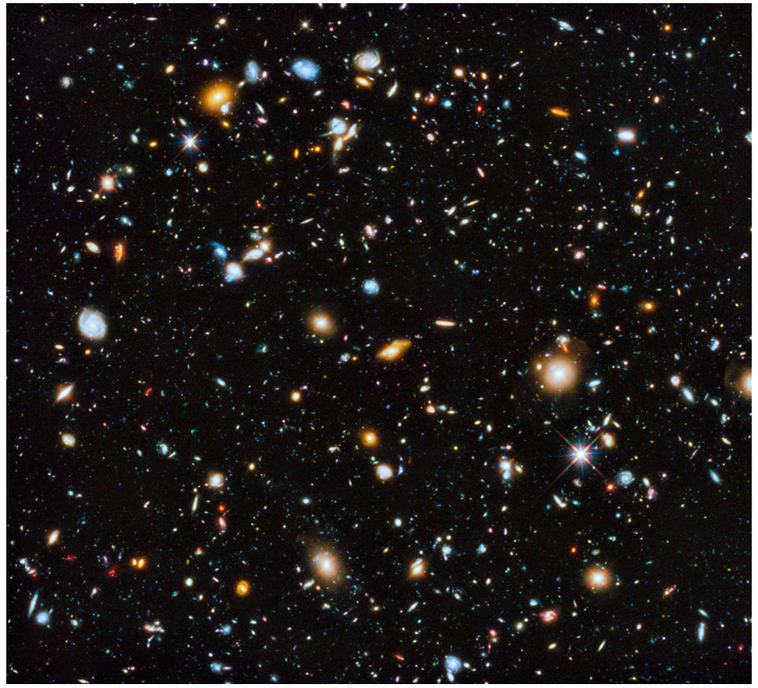
Wikipedia article: https://en.wikipedia.org/wiki/History_of_artificial_intelligence

Encyclopedia of Philosophy article: https://plato.stanford.edu/entries/artificial-intelligence

Turing's Computing Machinery and Intelligence: https://www.csee.umbc.edu/courses/471/papers/turing.pdf

History and Philosophy of Neural Networks: https://research.gold.ac.uk/10846/1/Bishop-2014.pdf

General: AI today



- If there were one word to describe the state of AI today, it would be "surreal." It's hard to imagine that just ten years ago, AI was very much still an academic endeavor, and now countries are forming national strategies around AI.

Prior to 2012, AI results closely tracked Moore's Law, with compute doubling every two years. Post-2012, compute has been **doubling every 3.4 months**.

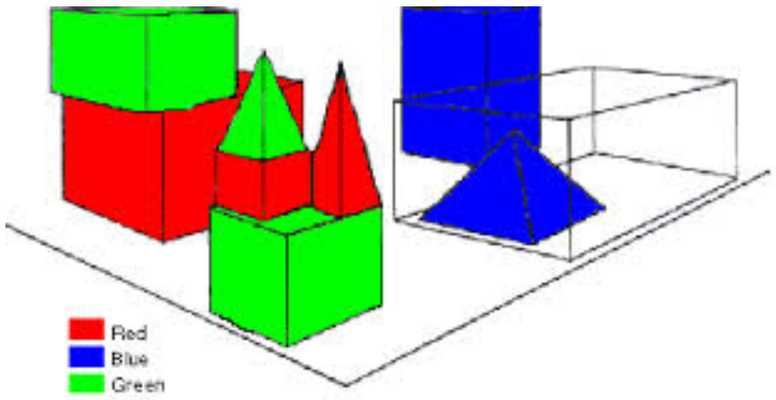
In 2019, the largest AI conference, **NeurIPS**, *expects 13,500 attendees, up 41% over 2018 and over* **800% relative to 2012**.

In the US, the share of **jobs** *in AI-related topics increased from* **0.26% of total jobs posted in 2010 to 1.32% in October 2019**.

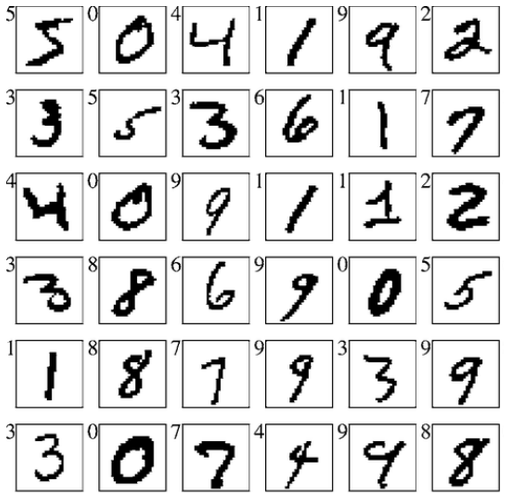
- The AI index is a project hosted out of Stanford which aims to track the state of AI in a data-driven way.
- Each year, it releases a report summarizing not just research progress, but also conference participation, impact on the economy, education, etc.
- Here are some quotes from the report: compute (mostly GPUs) has been doubling every 3.4 months.
- NeurIPS, the flagship machine learning conference has grown by 8x over the last 8 years.
- And the number of AI jobs has increased by 5x.

In vitro

SHRDLU

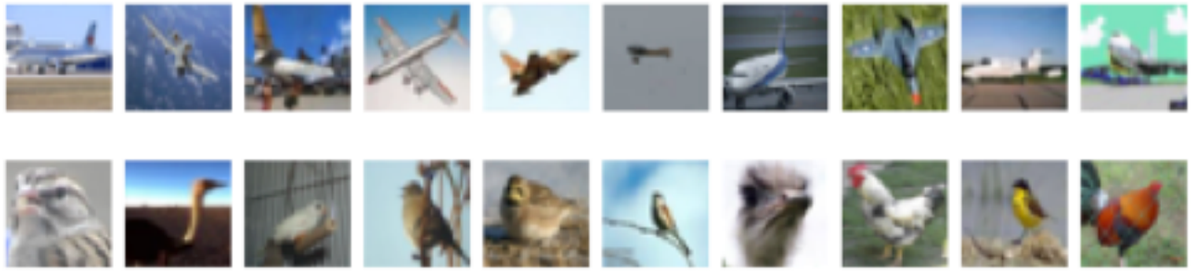


MNIST



Atari

CIFAR-10



- One of the biggest changes we've witnessed in AI is the transition from the lab to the real-world.
- For a long time, AI was limited to relatively artificial environments and datasets, which was (and is) still useful to spur the development of new methods.
- But now we are seeing much more real-world deployment of AI in ways that have a direct impact on people's lives.
- It's important to note that AI, like any technology, is an amplifier. It makes what is good better, and it makes what is bad worse. We need to be aware of both sides.

Prospects

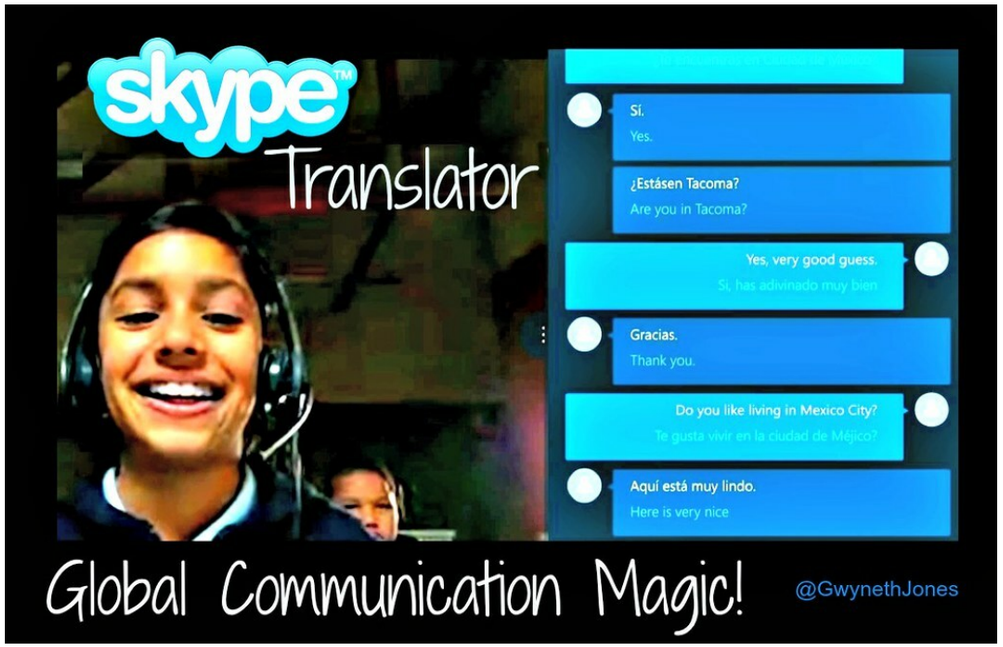
- Let us start with the positive side.

Virtual assistants

- In the last decade, speech recognition and question answering have improved remarkably.
- You can now talk to your favorite virtual assistant and expect some basic (though obviously not perfect) level of language understanding. And children are growing up thinking that talking to computers is normal.
- Search engines such as Google have already demonstrated the enabling power that comes with having the world's information at one's fingertips.
- Providing this through a natural language interface makes it more efficient and natural, and could have the potential to be especially be useful for people who might not have the means to use a computer (if they were designed for that population).

Machine translation

Input sentence:	*Translation (PBMT):*	*Translation (GNMT):*	*Translation (human):*
李克強此行將啟動中加總理年度對話機制，與加拿大總理杜魯多舉行兩國總理首次年度對話。	Li Keqiang premier added this line to start the annual dialogue mechanism with the Canadian Prime Minister Trudeau two prime ministers held its first annual session.	Li Keqiang will start the annual dialogue mechanism with Prime Minister Trudeau of Canada and hold the first annual dialogue between the two premiers.	Li Keqiang will initiate the annual dialogue mechanism between premiers of China and Canada during this visit, and hold the first annual dialogue with Premier Trudeau of Canada.



- Language barriers pose significant challenges to immigrants, travellers, businesses, and minority subcommunities, both in terms of connecting with others but also access to valuable information.
- Machine translation aims to overcome these barriers.
- Machine translation has made huge strides since the 1960s, and while it is not perfect, it is good enough for someone to get the basic gist of a document written in another language and to even communicate with another person speaking another language in real-time.

Autonomous driving

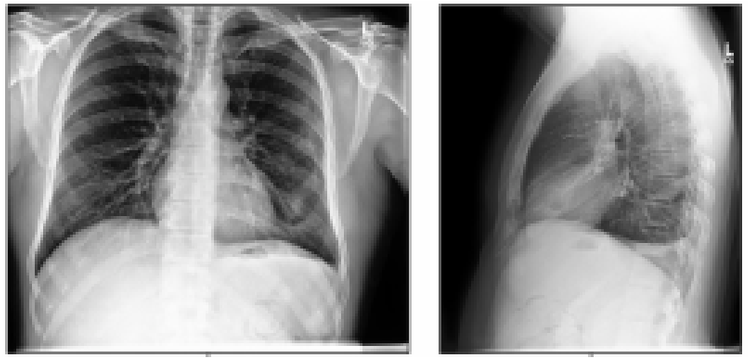
- Autonomous vehicles have the potential to one day significantly reduce accidents and congestion on our roads.
- A prerequisite is that the car doesn't hit anything, and one of the ways that it figures out what's in front is using a camera.
- Computer vision has made swift progress towards recognizing and localizing objects in relatively unstructured scenes, but there is still some headroom to achieve the needed reliability.

Visual assistive technology

- This example is the Seeing AI app from Microsoft Research, which narrates whatever the camera is pointed at.
- This visual assistive technology could be a game-changer for the visually impaired.
- Conversely, auto-captioning technology, which turns sound into sight, is potentially also quite useful for the hearing-impaired.

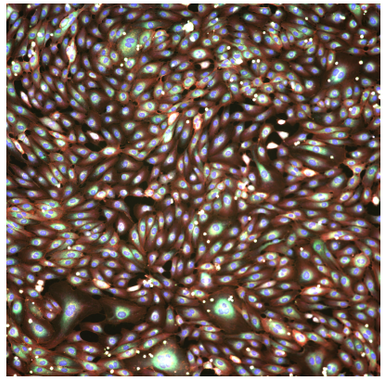
Healthcare

Chest radiology



Diabetic retinopathy

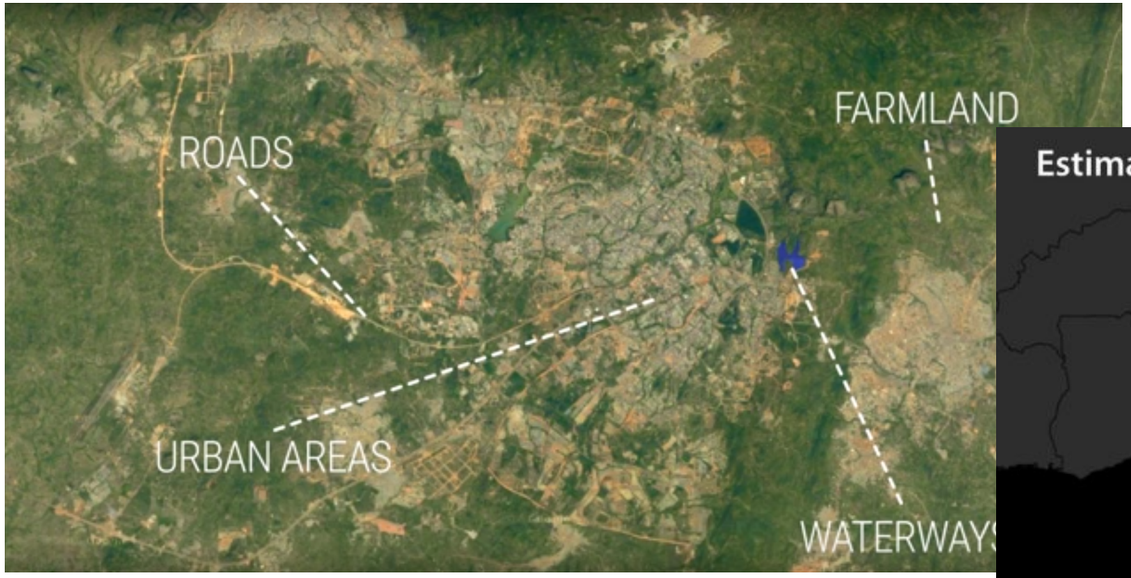
Drug screening for COVID-19

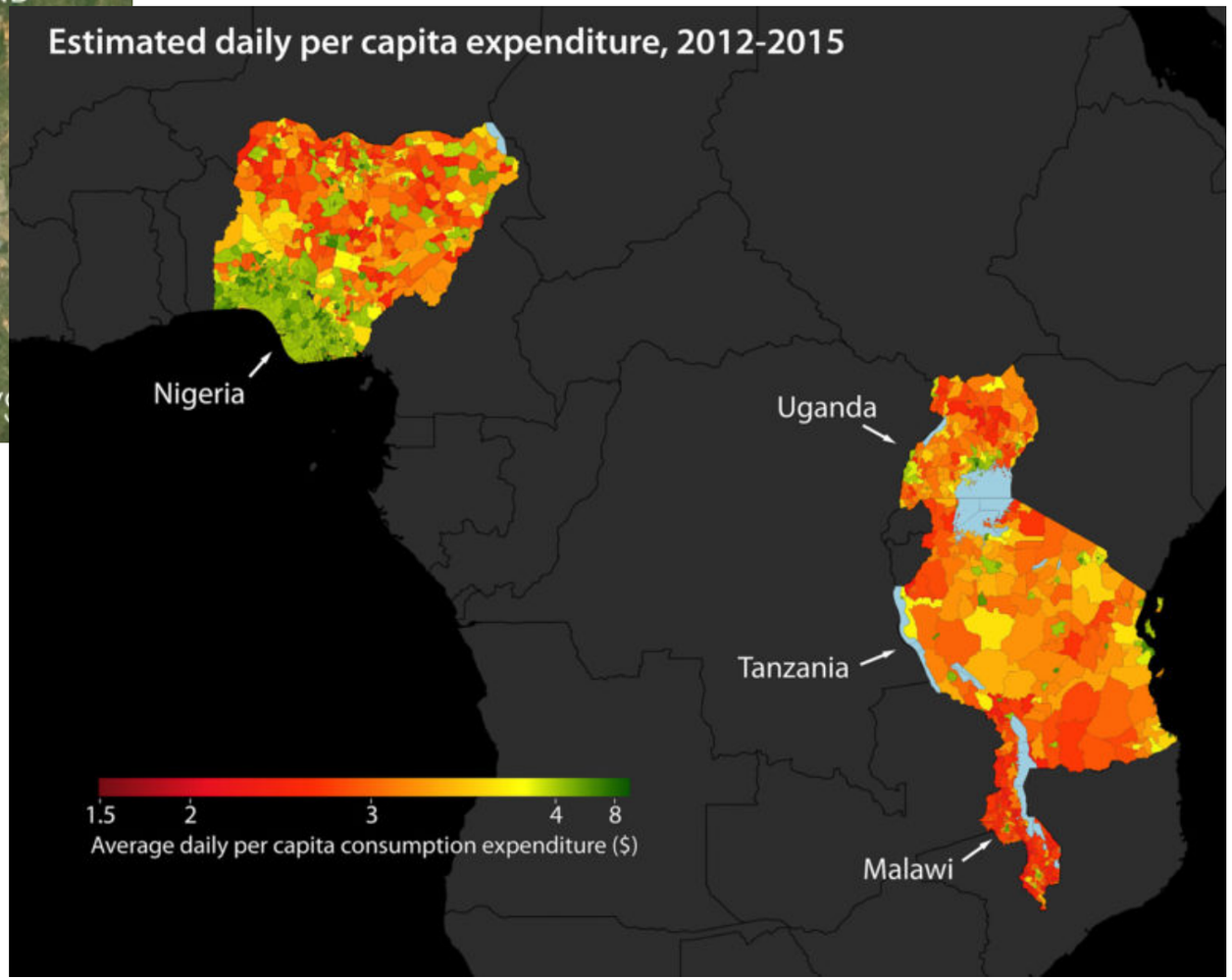


- AI for healthcare is also an area of growing importance, both for diagnosis and for therapeutic development, especially in areas in the world with a shortage of clinical specialists.
- One example is interpreting chest x-rays for detecting diseases such as pneumonia and collapsed lung.
- Another is diagnosing diabetic retinopathy, which causes blindness in diabetic patients.
- Finally, there's a recent dataset with experiments showing how COVID-19 infected cells respond to certain drugs, with the hope that one can find drugs that can treat late-stage COVID-19.

[Jean et al. 2016]

Poverty mapping



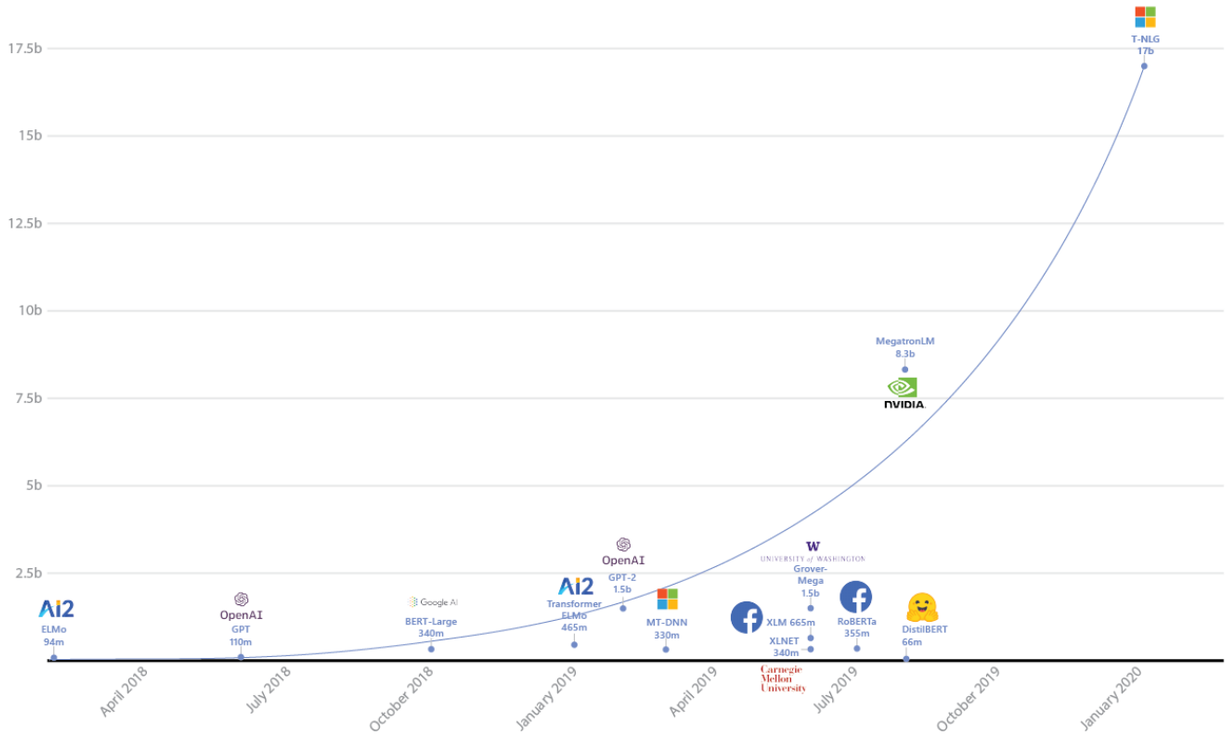


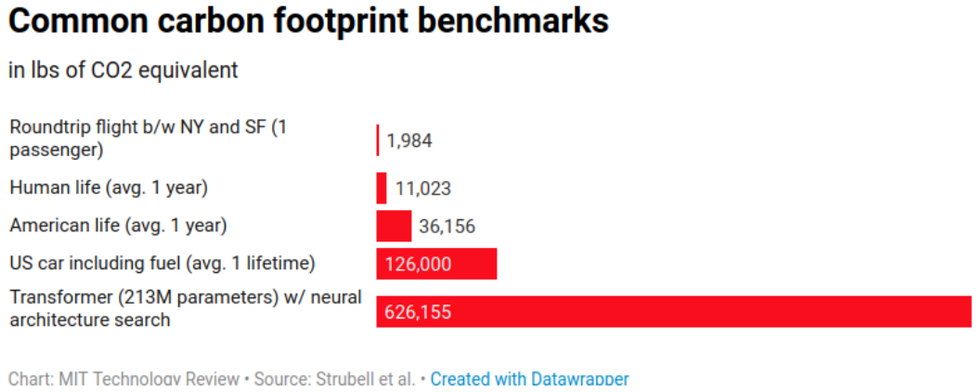
- At a more societal level, it is well-known that poverty is a huge problem in the world, with more than 700 million people living in extreme poverty according to the World Bank.
- But even identifying the areas in greatest need is challenging due to the difficulty of obtaining reliable survey data.
- Some work has shown that satellite images (which are readily available) can be used to predict various wealth indicators based on the types of roofs or presence of roads or night lights.
- This information could be informative for governments and NGOs to take proper action and monitor progress.

Risks

- This all sounds great, so what's the catch?

Energy consumption





GPT-3 (released May 2020) from OpenAI has **175 billion** parameters

- First, there is the cost of training high-performing models.
- If we look at NLP, there has been a trend of training more and more insanely large language models, which are now used to power everything from virtual assistants to machine translation.
- In April 2018, models had about 100 million parameters.
- BERT, which made a big splash, had 340 million parameters.
- In January 2020, Microsoft released a 17 billion parameter model.
- In May 2020, OpenAI released GPT-3, which was ten times larger still, which, if placed on the chart, would be way off the screen.
- There was a paper that examined the environmental impact of training deep learning models. A 213 million parameter model with neural architecture search was about 5 times the CO2 emissions during the lifetime of a US car. I'll let you speculate about the energy consumption of GPT-3.
- Perhaps not surprisingly, there is a lot of research into improving efficiency without sacrificing accuracy.

Privacy

- Machine learning algorithms assume by default that all its data is sitting in one place and fully-accessible.
- But our mobile devices generate a wealth of information, and we might not want to be sending that all up to some big Internet company.
- Recently there's been work in privacy-preserving machine learning, which allows data and learning to happen on devices in a decentralized way, and only transmit limited statistics to a central server.

Security

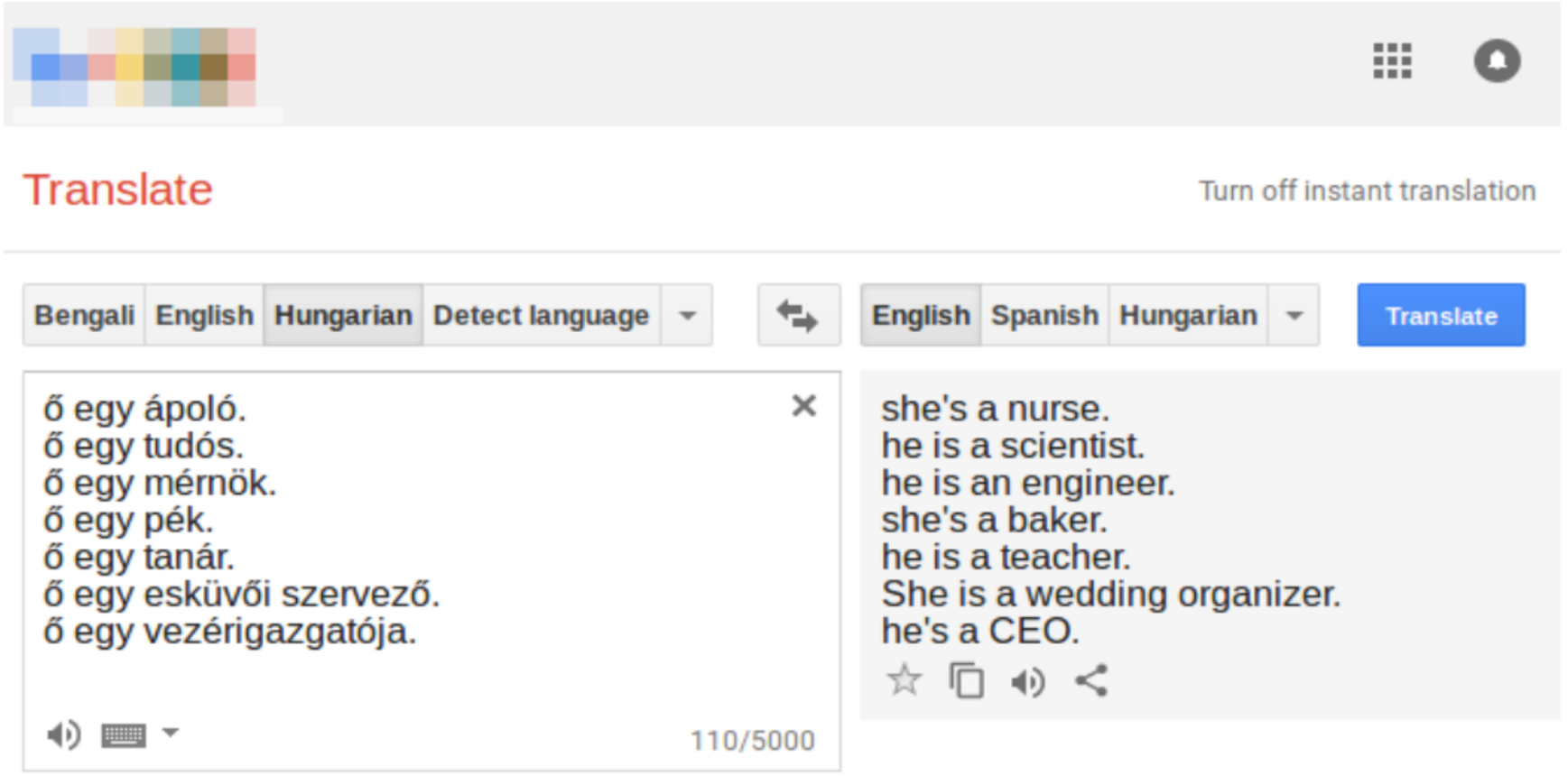
[Evtimov+ 2017]

[Sharif+ 2016]

- In high-stakes applications such as autonomous driving and authentication (face ID), models need to not only be accurate but need to be robust against **attackers**.
- Researchers have shown how to generate **adversarial examples** to fool systems.
- For example, you can put stickers on a stop sign to trick a computer vision system into mis-classifying it as a speed limit sign.
- You can also purchase special glasses that fool a system into thinking that you're a celebrity.
- Guarding against these attackers is a wide open problem.

[Prates+ 2018]

Bias



- A less spectacular but maybe more pernicious problem is bias.
- In this example, if you take Hungarian, in which "he" and "she" are not differentiated, and translate into English, the model has to hallucinate gender from context.
- This experiment reveals the various gender stereotypes that the model harbors.
- Even though machine learning algorithms are based on mathematical principles, a trained model latches on to statistics in the training data. And the training data comes from society, so any biases in society are reflected in the data and propagated (and even amplified) into the predictions.

Fairness

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

- The stakes are increased when machine learning models are used to make decisions that impact someone's life in a serious way (e.g., bail, loans, hiring).
- There was one instance where Northpointe created a software system called COMPAS to predict recidivism risk, i.e., whether they will commit a crime again.
- ProPublica, a nonprofit organization that does investigative journalism, found that given that an individual didn't commit a crime, Black people were were predicted to have higher risk, and therefore, unfair.
- Northpointe rebutted, noting that given a risk score of 7, the probability of commiting a crime was the same for both Black and White people, and therefore it was fair.
- These just happen to be two different notions of fairness (equalized odds versus predictive rate parity). Unfortunately, there is no "right answer," and it has even been shown mathematically that it is impossible for a imperfect classifier to satisfy three notions of fairness simultaneously (Kleinberg et al., 2016).
- This indicates that making progress on fairness is not just a technical challenge, but one that requires social and policy considerations.
- There are many other examples. Just to point out another one: researchers have found racial bias in health algorithms, which falsely concludes that Black patients are healthier than White patients simply because more money is spent on Black patients [Obermeyer+ 2019].

Generating fake content

In order to get something done, maybe we need to think less. Seems counter-intuitive, but I believe sometimes our thoughts can get in the way of the creative process. We can work better at times when we "tune out" the external world and focus on what's in front of us.

I've been thinking about this lately, so I thought it would be good to write an article about it.

So what exactly does this mean? Well, for starters, let's start with some definitions.

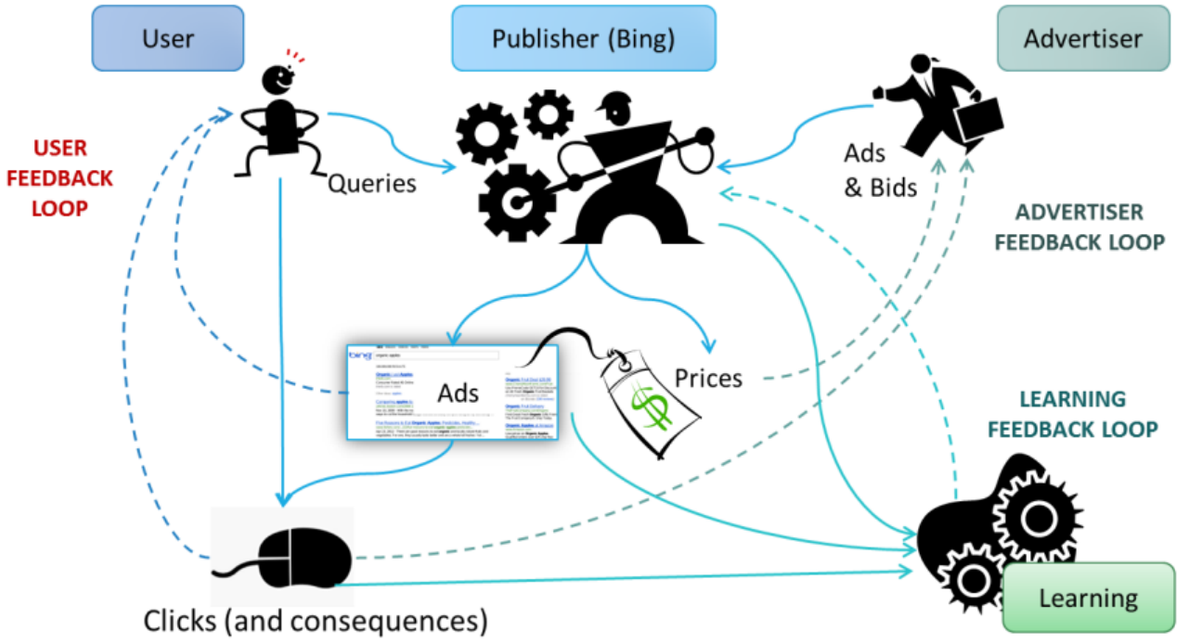
Definition #1: Creative Thinking (CT) is the act of using your mind to come up with ideas that are not already present in reality or that have never existed before. It is a form of mental activity that involves the use of logic and reason.

Definition #2: Over-Thinking (OT) is the act of trying to come up with ideas that have already been thought through by someone else. OT usually results in ideas that are impractical, impossible, or even stupid.

- Deep learning has enabled us to generate surprisingly realistic content.
- On the left, is an Obama deepfake, a video that shows Obama saying things he never did.
- On the right is part of a blog post that a student used GPT-3 to generate, which ended up ranked number one on Hacker News for a while.
- It is clear that we have already lost the ability to tell the difference between real and fake content. Furthermore, given the ease and scale that fake content can be generated, bad actors spreading disinformation is a big threat to our society.

Feedback loops in learning

[Leon Bottou]



- Finally, AI systems are deployed in a dynamic environment in which the system makes predictions (search results, recommendations, ads), users take action (e.g., clicks).
- These actions are recorded as data used to retrain the system, often to reinforce these actions.
- This introduces a feedback which usually amplifies or polarizes the initial signal, leading to unstable behavior.
- An open research challenge is to design learning algorithms whose dynamics are stable.

Prospects and risks of AI

AI technology is an amplifier

🙂 Can reduce accessibility barriers and improve efficiency

☹️ Can amplify bias, security risks, centralize power

Can build it $\neq$ should build it

- To conclude, it is worth stressing that AI, or technology in general, is an amplifier.
- We've seen that it can reduce accessibility barriers and improve the lives of the less fortunate.
- On the other hand, there are biases, new security threats, and a danger of centralization of power due to AI as well. We've only skimmed the surface here; there is much more to be said on this topic.
- As you proceed through this course, I would urge you to keep these issues in mind. If we are not careful, we could end up doing more harm than good. Just because you can build it doesn't mean you should.
- Figuring out the right way to reap the benefits and mitigate the risks will also require having a deep technical understanding, especially to develop novel solutions, which is what this course seeks to provide.