Plan of the course

Contents: Reinforcement learning (6 weeks) and advanced deep learning (6 weeks). Principles, techniques and applications

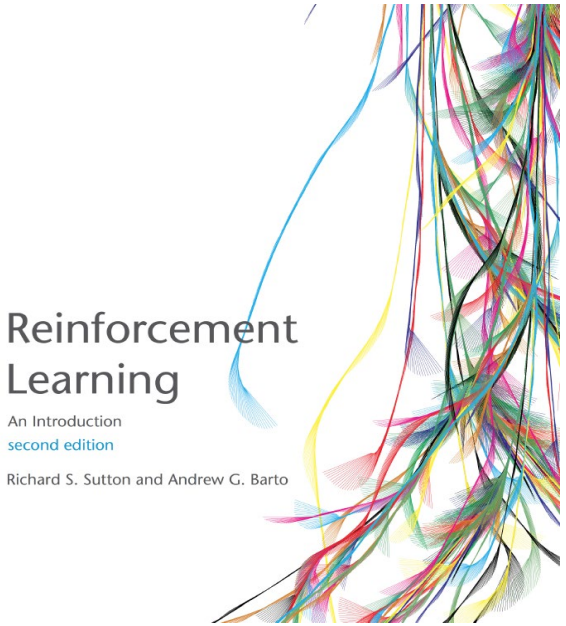
Lecturers: Marga Cabrera, Xavier Giró, Josep Vidal

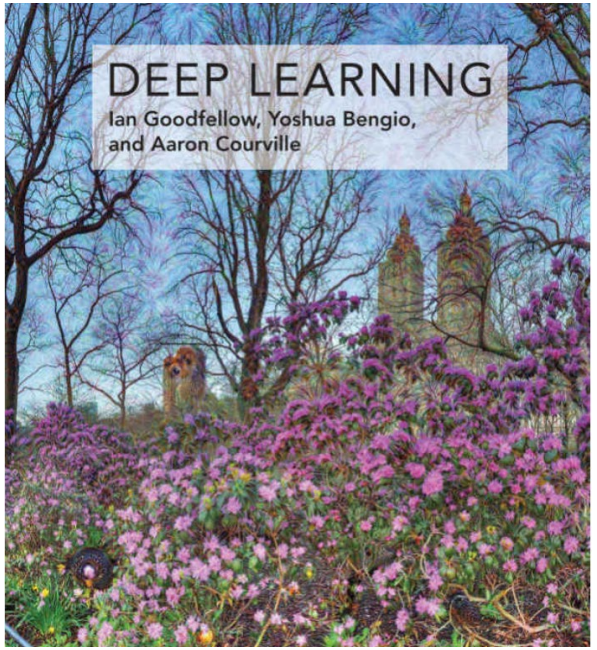
Materials: available in Atenea

Lab: Bring your own laptop, will use Python

Grading: Two exams (50%) + labs (30%) + online quizzes (20%)

Bibliography:
R. Sutton, A. Barto, *Reinforcement Learning: An Introduction*
I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*





DEEP AND REINFORCEMENT LEARNING

UPC TelecomBCN Barcelona (3rd edition). Autumn 2020.

Instructors:

Margarita Cabrera

Víctor Campos

Xavier Giró-i-Nieto

Juan José Nieto

Josep Vidal

Guests:

Ferran Alet (MIT)

Carlos Florensa (Covariant)

Ignasi Clavera (Berkeley)

Jordi Torres (BSC)

Oriol Vinyals (Deepmind)

Onsite courses

- Advanced Deep Learning and Reinforcement Learning. University College London and Deepmind. 2019.
- David Silver, "University College London (UCL) Course on RL". University College London (2015).
- Emma Brunskill, "CS234 Reinforcement Learning". Stanford University (2020).
- Benjamin Van Roy, "MS&E338 Reinforcement Learning". Stanford University (2020).
- Sergey Levine et al. "CS285 Deep Reinforcement Learning". UC Berkeley (2019).
- Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas "Deep Unsupervised Learning". UC Berkeley 2020.

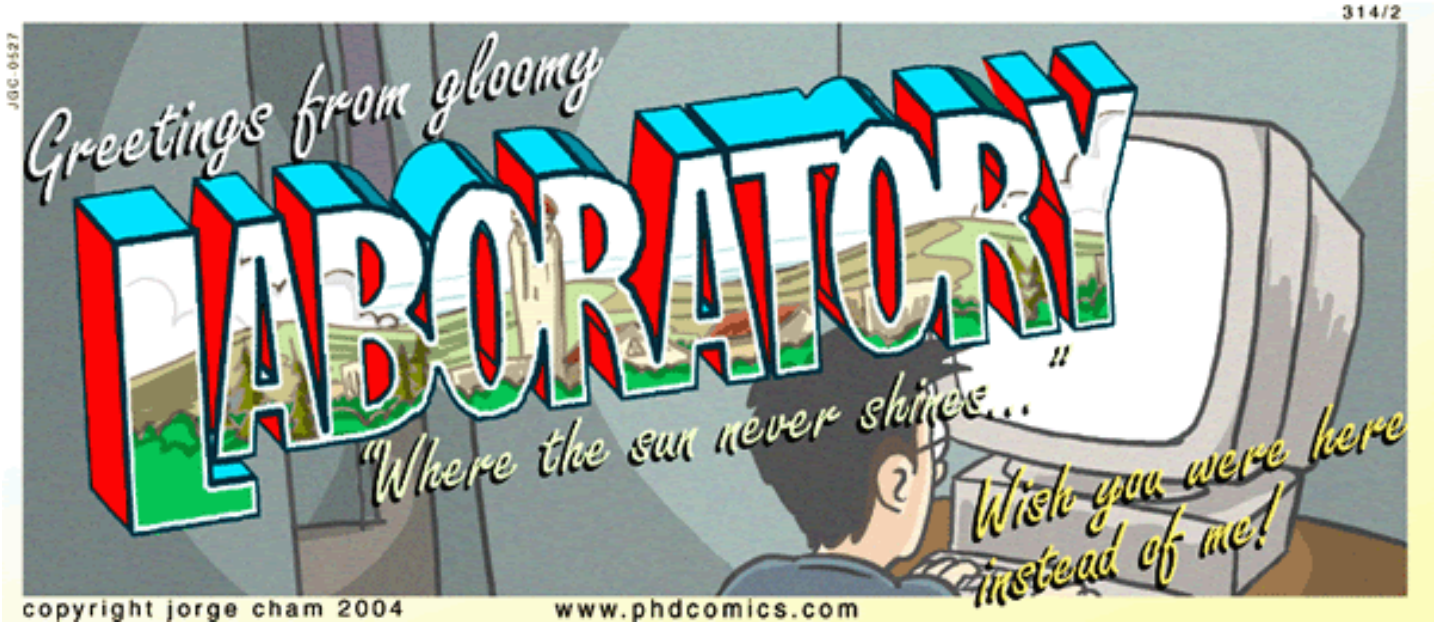
Online courses

- Josh Achiam, "Spinning Up in Reinforcement Learning". OpenAI (2020).
- Martha White, Adam White, Reinforcement Learning specialization. University of Alberta.

Others

- Jordi Torres, Deep Reinforcement Learning Explained. Towards Data Science 2020.
- Lilian Weng, Blog Posts on Reinforcement Learning

Microsoft: https://courses.edx.org/courses/course-v1:Microsoft+DAT257x+1T2020a/course/



Python tutorial: https://docs.python.org/3/tutorial/index.html
Intro to Colab Notebooks: https://colab.research.google.com/notebooks/intro.ipynb?hl=en

Cautionary measures

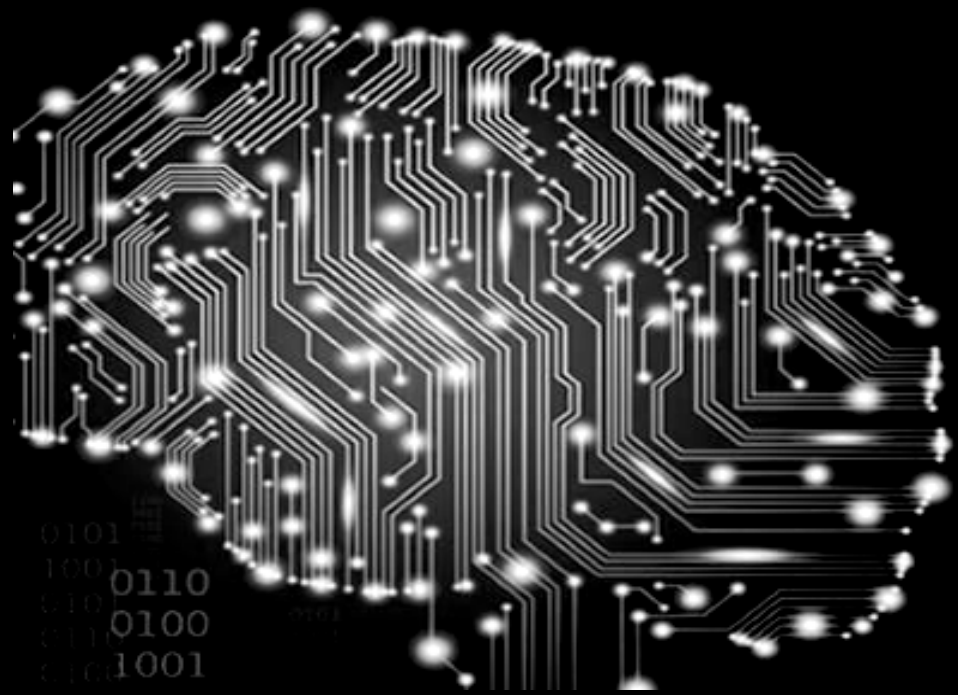
Beyond the use of face masks and hydrogels…

Keep the same seat during the day (and if possible during the semester) and fill out this form

https://forms.gle/e3htJNZi8ZGQScLj7

once a day if you do not change classrooms, and each time you change. The purpose is to have a record of attendees.

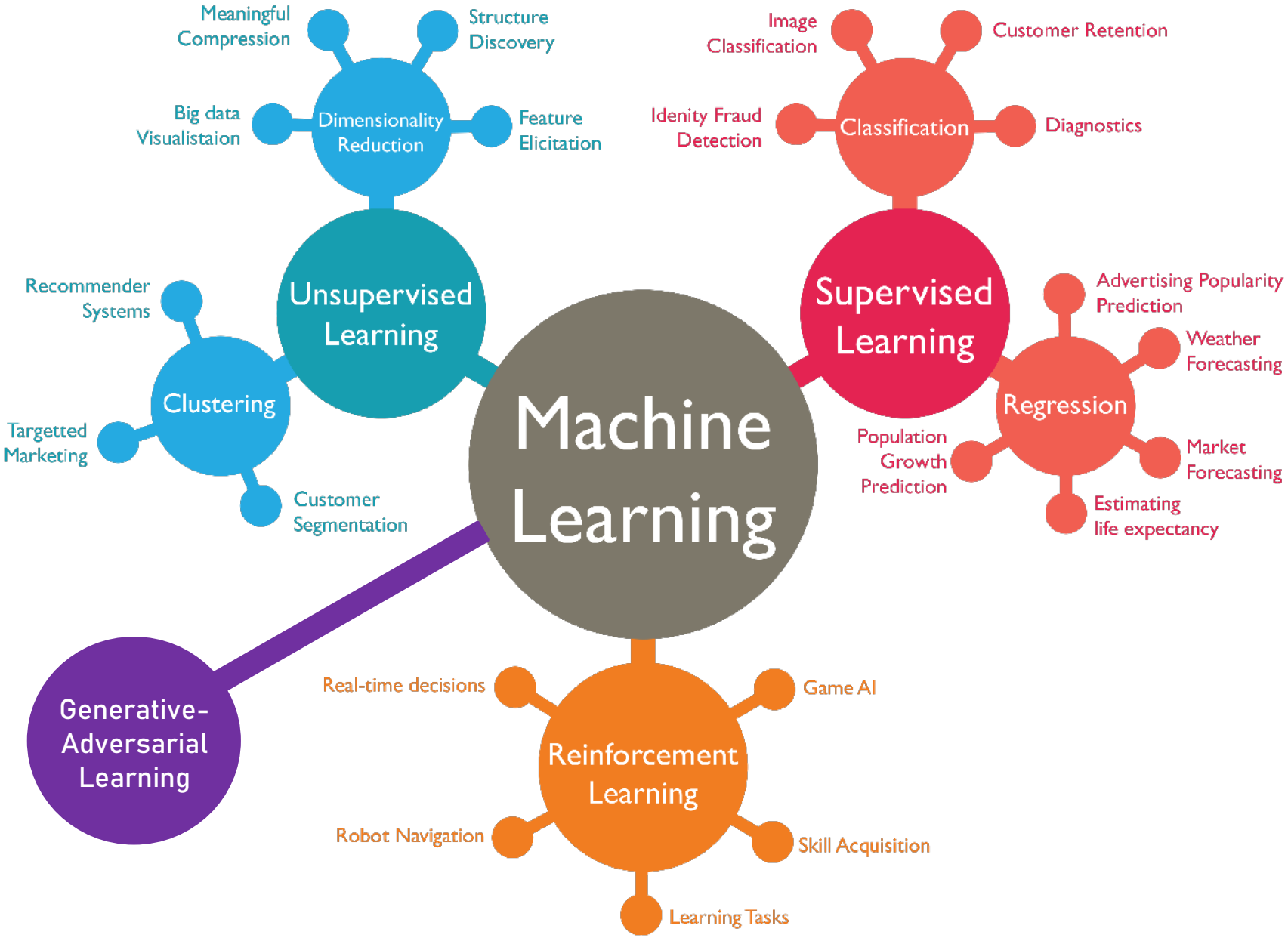
1. Introduction to Reinforcement Learning



Why do we need machines taking decisions?

Fundamental causes of human irrational behavior

- Evolutionary causes: fear, shyness, conformism, shame,…
- Physiological causes: when we learn, some neurons connect but some other existing connections weaken
- Mental sloth
- Unability to manage many variables simultaneously
- Unability to handle basic concepts of probability calculus
- Selfish bias for fear of losing self-steem


Machine Learning
Unsupervised Learning
Meaningful Compression
Structure Discovery
Dimensionality Reduction
Big data Visualistaion
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Image Classification
Customer Retention
Classification
Idenity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating life expectancy
Population Growth Prediction
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks
Generative-Adversarial Learning

A taxonomy of machine learning…

We want to build a machine that learns how to take decisions out of sensorial data (feature vectors) $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \ldots$

There are three main approaches:

1. **Supervised learning**: the learning rule is provided with the desired outputs y_1, y_2, y_3, y_4 , … for each given input. The objective is generating the right output when fed with a new input.

2. **Unsupervised learning**: a model is built for the values of $\mathbf{x}$ that can be used to explain some phenomenon, to understand the structure of data, to predict, to communicate, etc.

3. **Reinforcement learning**: we do not have reference answers. The machine takes some actions a_1, a_2, … affecting the environment, and gets some reward (or punishment) r_1, r_2, … The objective is learning to act in a way that long term rewards are maximized.

Supervised learning

Data base

- Feature vectors (observations): $\mathbf{x}_i \quad i = 1, \ldots, N$
- Each belonging to one of the c classes (nature states): y_i

Objective

Design a procedure that learns to associate $\mathbf{x}_i$ and y_i through the function $h_{\boldsymbol{\theta}}$

$$\hat{y}_j = h_{\boldsymbol{\theta}}(\mathbf{x}_j)$$

that depends on some parameters $\boldsymbol{\theta}$, using the data base and optimising some loss function

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L\left(y, h_{\boldsymbol{\theta}}(\mathbf{x})\right)$$

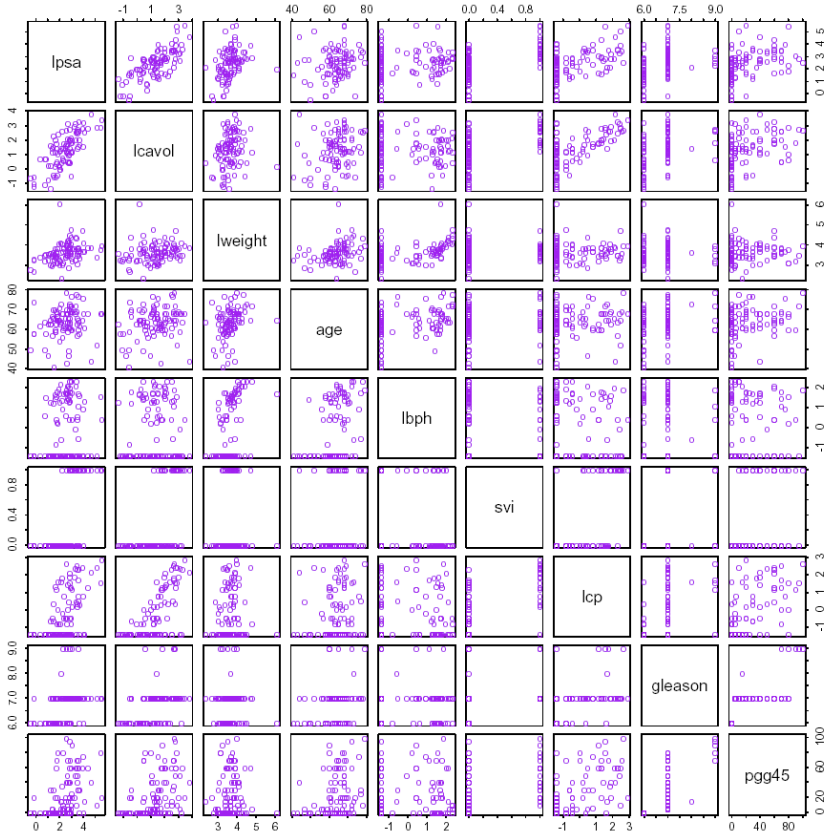
Example 1.1. Diagnose of prostate cancer

Build a data base of clinical histories $\mathbf{x}_i$ containing relevant information

- lpsa
- log cancer volume (lcavol)
- log prostate weight (lweight)
- age
- log benign prostatic hyperplasia (lbph)
- seminal vesicle invasion (svi)
- Gleason score (gleason)
- percent of Gleason scores 4 or 5 (pgg45)

of healthy and sick patients (y_i), and build a machine able to predict prostate cancer condition

$$\hat{y}_j = h_{\boldsymbol{\theta}}(\mathbf{x}_j)$$



A taxonomy of machine learning…

We want to build a machine that learns how to take decisions out of sensorial data (feature vectors) $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \ldots$

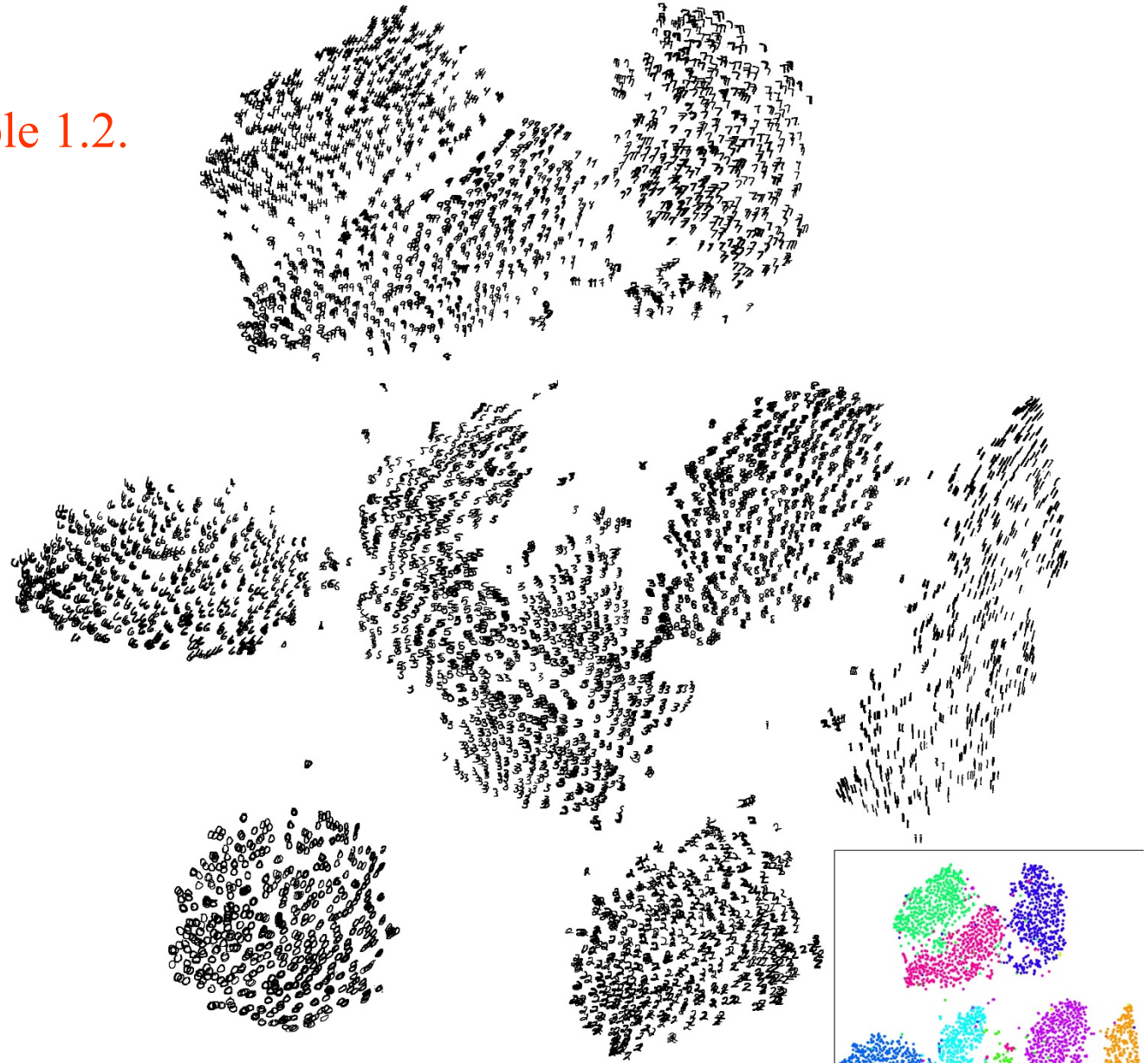
There are three main approaches:

1. **Supervised learning**: the learning rule is provided with the desired outputs y_1, y_2, y_3, y_4, … for each given input. The objective is generating the right output when fed with a new input.

2. **Unsupervised learning**: a model is built for the values of $\mathbf{x}$ that can be used to explain some phenomenon, to understand the structure of data, to predict, to communicate, etc.

3. **Reinforcement learning**: we do not have reference answers. The machine takes some actions a_1, a_2, … affecting the environment, and gets some reward (or punishment) r_1, r_2, … The objective is learning to act in a way that long term rewards are maximized.

Example 1.2.



https://lvdmaaten.github.io/tsne/examples/mnist_tsne.mov

A taxonomy of machine learning…

We want to build a machine that learns how to take decisions out of sensorial data (feature vectors) $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \ldots$

There are three main approaches:

1. **Supervised learning**: the learning rule is provided with the desired outputs y_1, y_2, y_3, y_4, … for each given input. The objective is generating the right output when fed with a new input.

2. **Unsupervised learning**: a model is built for the values of $\mathbf{x}$ that can be used to explain some phenomenon, to understand the structure of data, to predict, to communicate, etc.

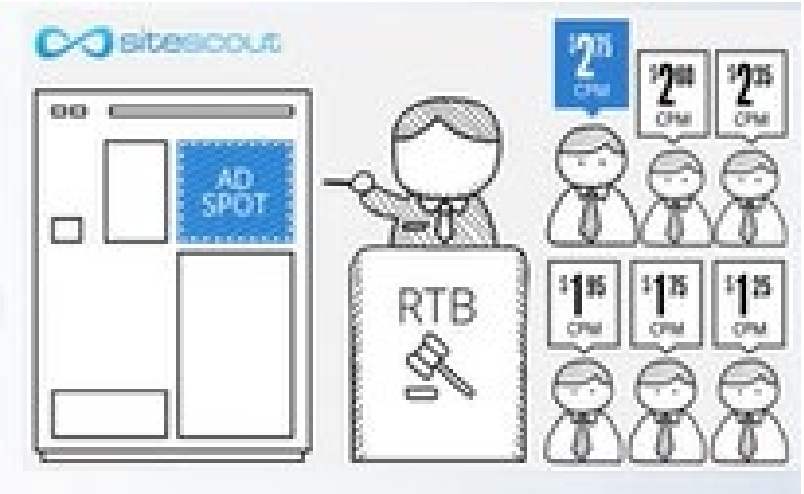
3. **Reinforcement learning**: we do not have reference answers. The machine takes some actions $a_1, a_2, \ldots$ affecting the environment, and gets some reward (or punishment) $r_1, r_2, \ldots$ The objective is learning to act in a way that long term rewards are maximized.

Example 1.3. Manage advertisements in a website (YouTube, Facebook,…)

Each user visiting the page is shown a banner that he/she may either click or not.

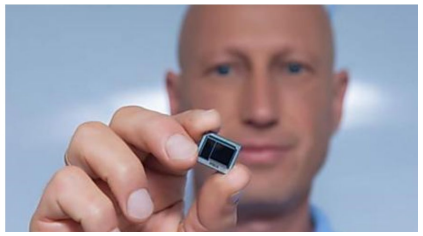
On the other side we have banner ad providers: companies that will pay us if users click on their banners.

Pick the banner we want to show,… but we cannot use supervised learning because we have no data set nor previous experience.



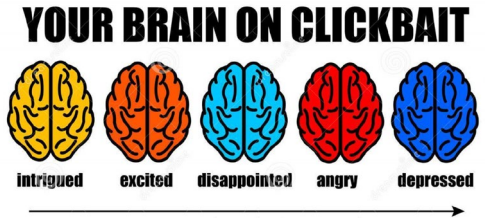
How to solve it?

A general idea: initialize with a naive solution and try to learn from trial and error.



If your computer has an Internet connection, this is something you should to today

Issue 1. Introduce optimality: maximize benefit right now vs. make users happy so that they return.



Issue 2. Follow the current strategy or discover better ones?

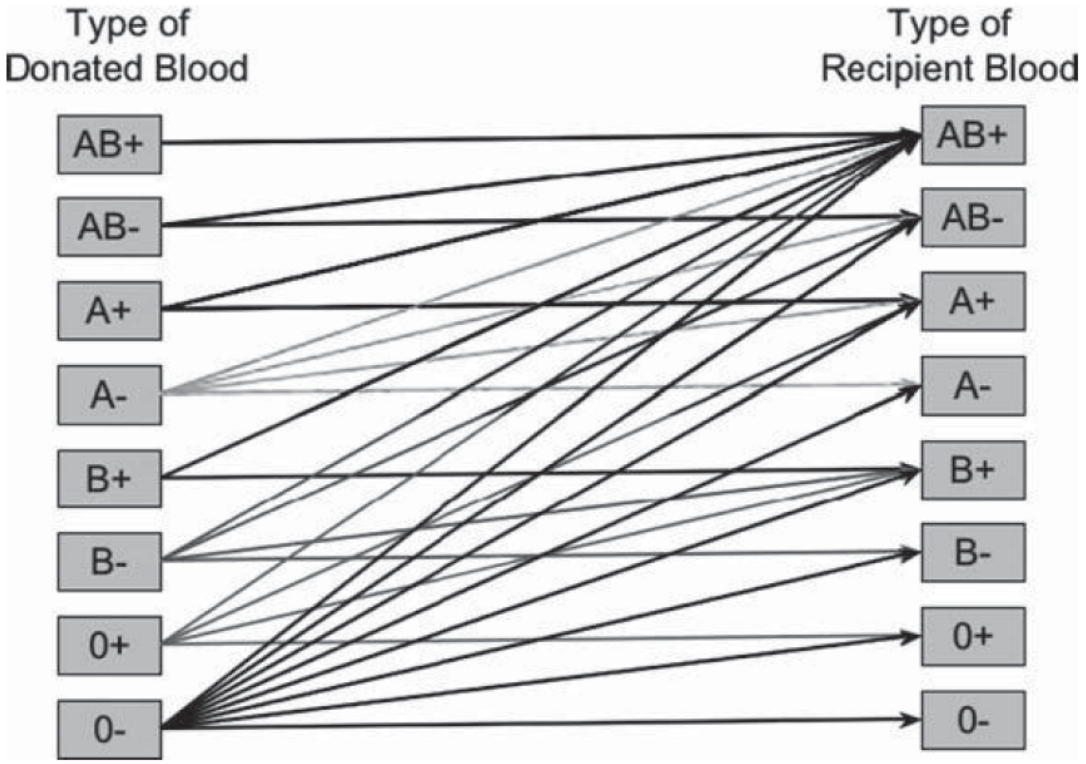
Comparison…

Supervised learning	Reinforcement learning
Learning to approximate reference answers	Learning optimal strategy by interacting with environment, using trial and error
Needs correct answers	Needs feedback on agent's own actions
Model does not affect the input data	Agent can affect its own observations
	Training can be very costly, unless we have computer models we can rely on

Many applications of reinforcement learning

Some examples of RL applications (model based)

- Selling assets
- Control a power station
- Managing blood inventories

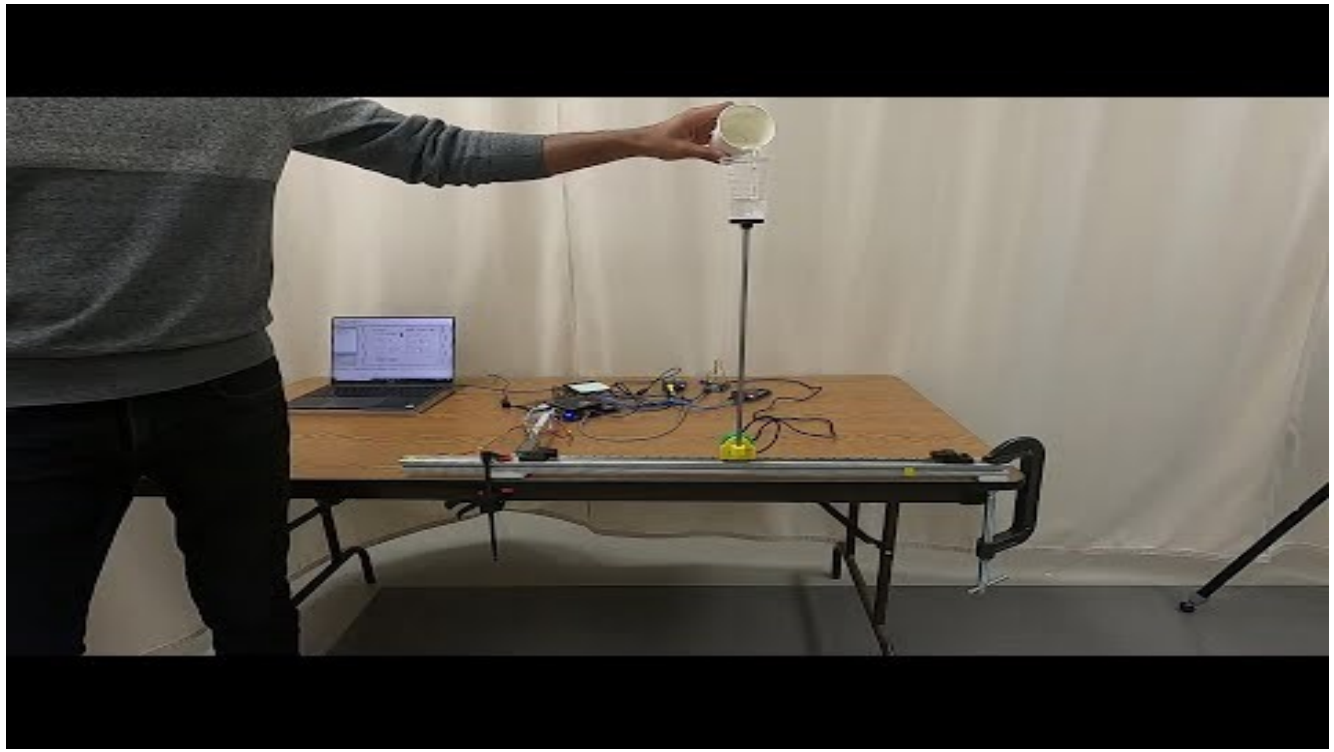


Some examples of RL applications (model based)

- Aircraft landing scheduling
- Management of railroad resources (boxcars, locomotives, employees)
- Schedule of mid-air refueling in air operations

Some examples of RL applications (model-free)

- Inverted pendulum disturbance rejection



Some examples of RL applications

- Make a humanoid robot walk
- Recommendation system for online store
- Self-driving vehicles (cars, helicopters)

Wayve, "The first example of deep reinforcement learning on-board an autonomous car" (2018)

Some examples of RL applications

- Make a humanoid robot walk
- Recommendation system for online store
- Self-driving vehicles (cars, helicopters)

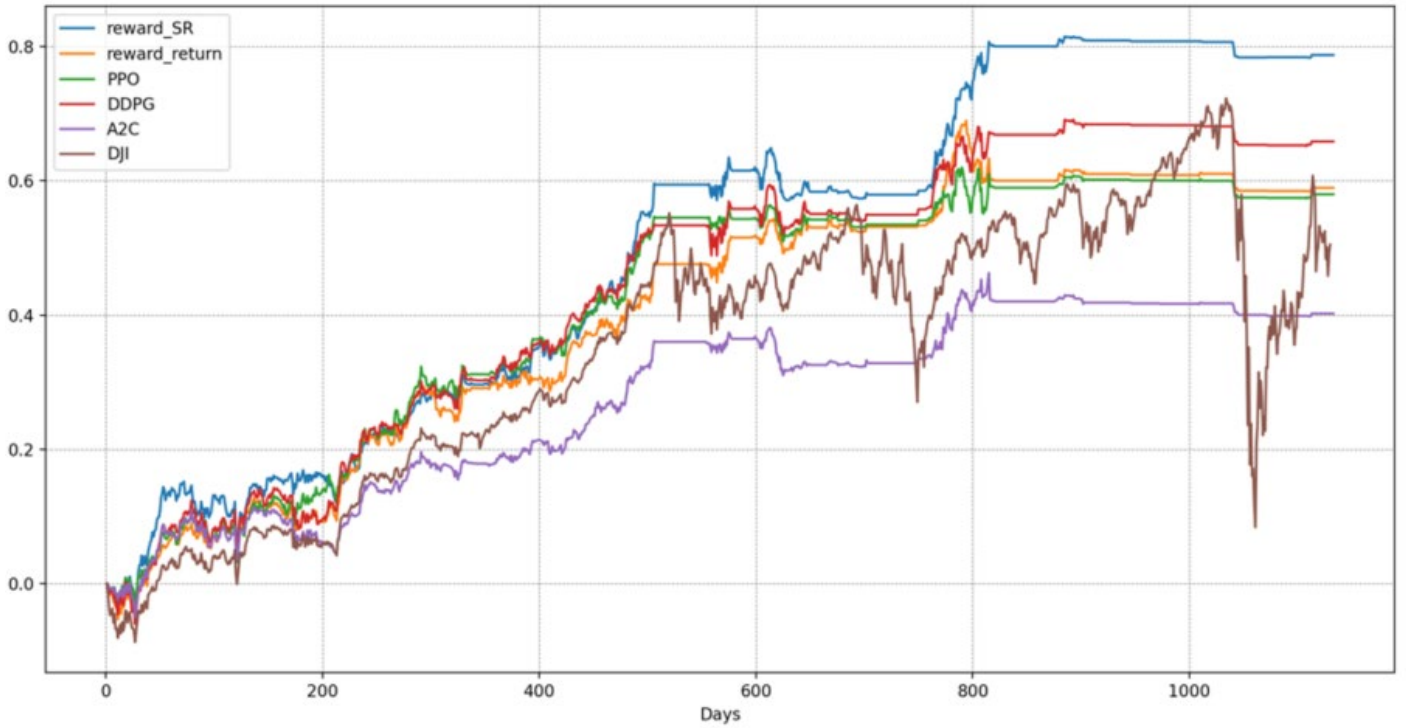
Some examples of RL applications

- Play Atari games better than humans
- Conversation systems: learn to make user happy
- Quantitative finance, portfolio management
- Deep learning: optimizing non-differentiable loss, finding optimal architecture

V. Mnih, et al. “Playing Atari with Deep Reinforcement Learning”, https://arxiv.org/abs/1312.5602

Some examples of RL applications

- Play Atari games better than humans
- Conversation systems: learn to make user happy
- Quantitative finance, portfolio management
- Deep learning: optimizing non-differentiable loss, finding optimal architecture



Some examples of RL applications

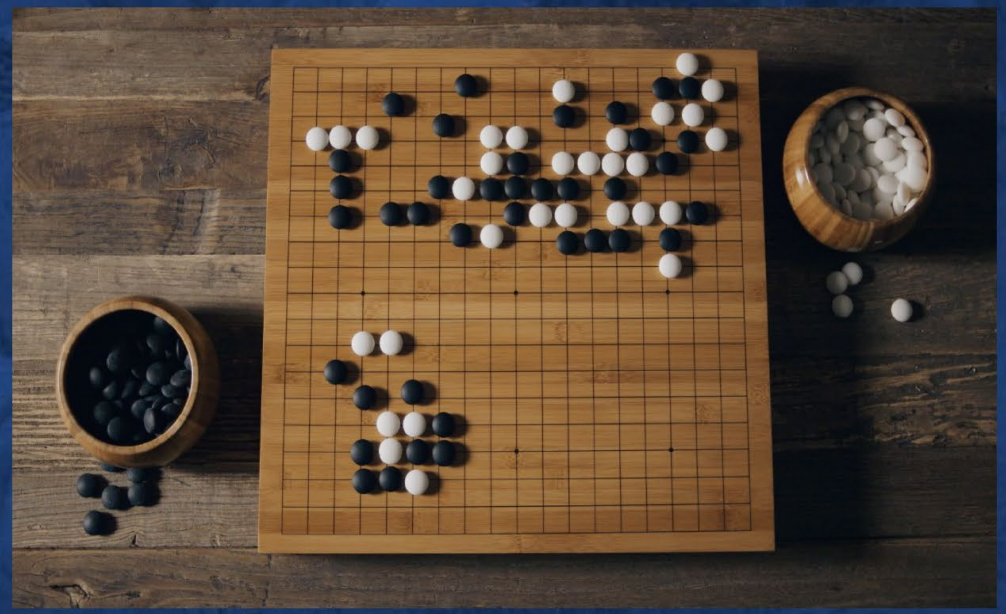
- Find pages relevant to queries
- Defeat the world champion at Backgammon, Go, Chess

Why is Go hard for computers to play?

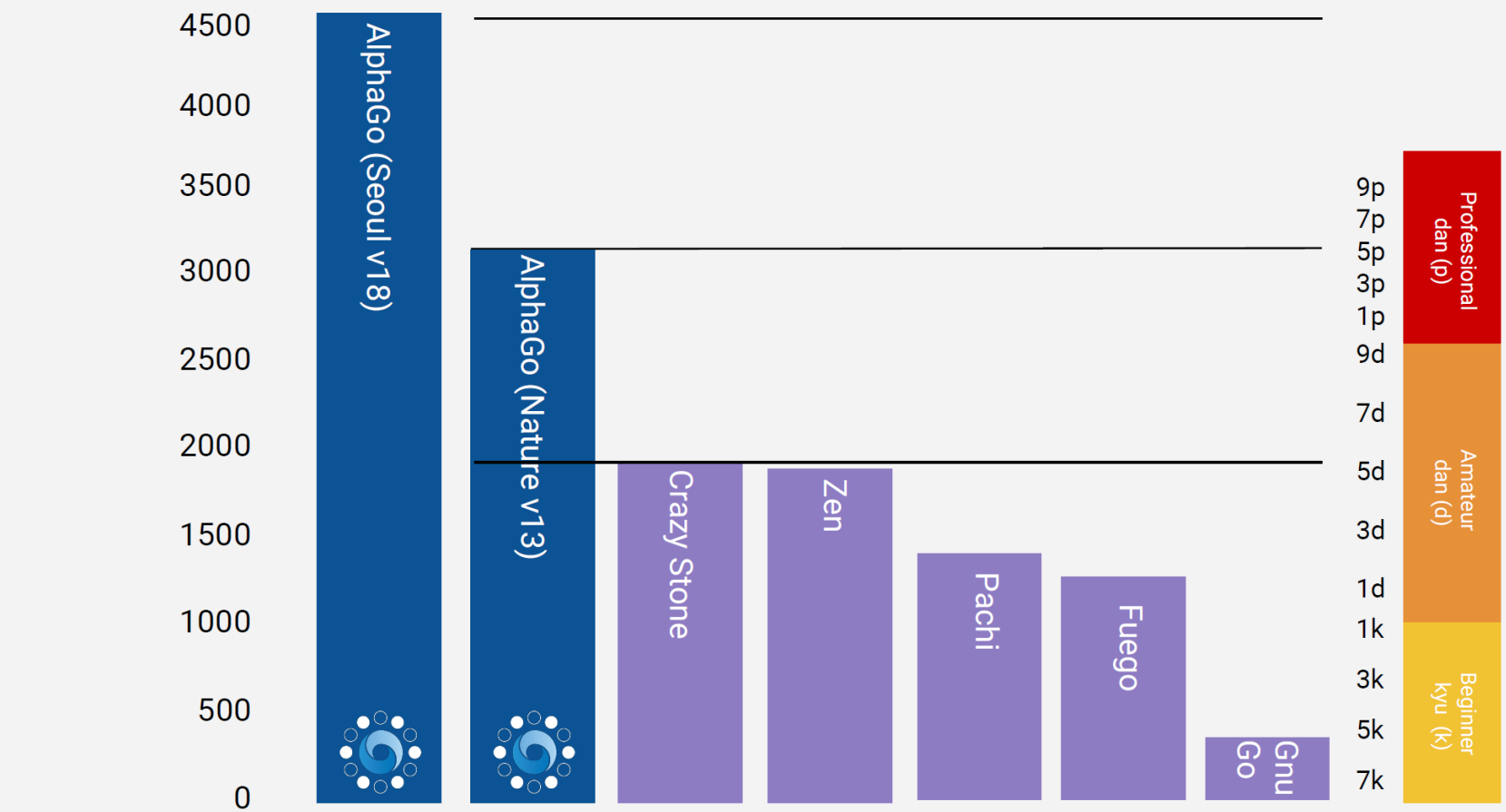
Game tree complexity = b^d

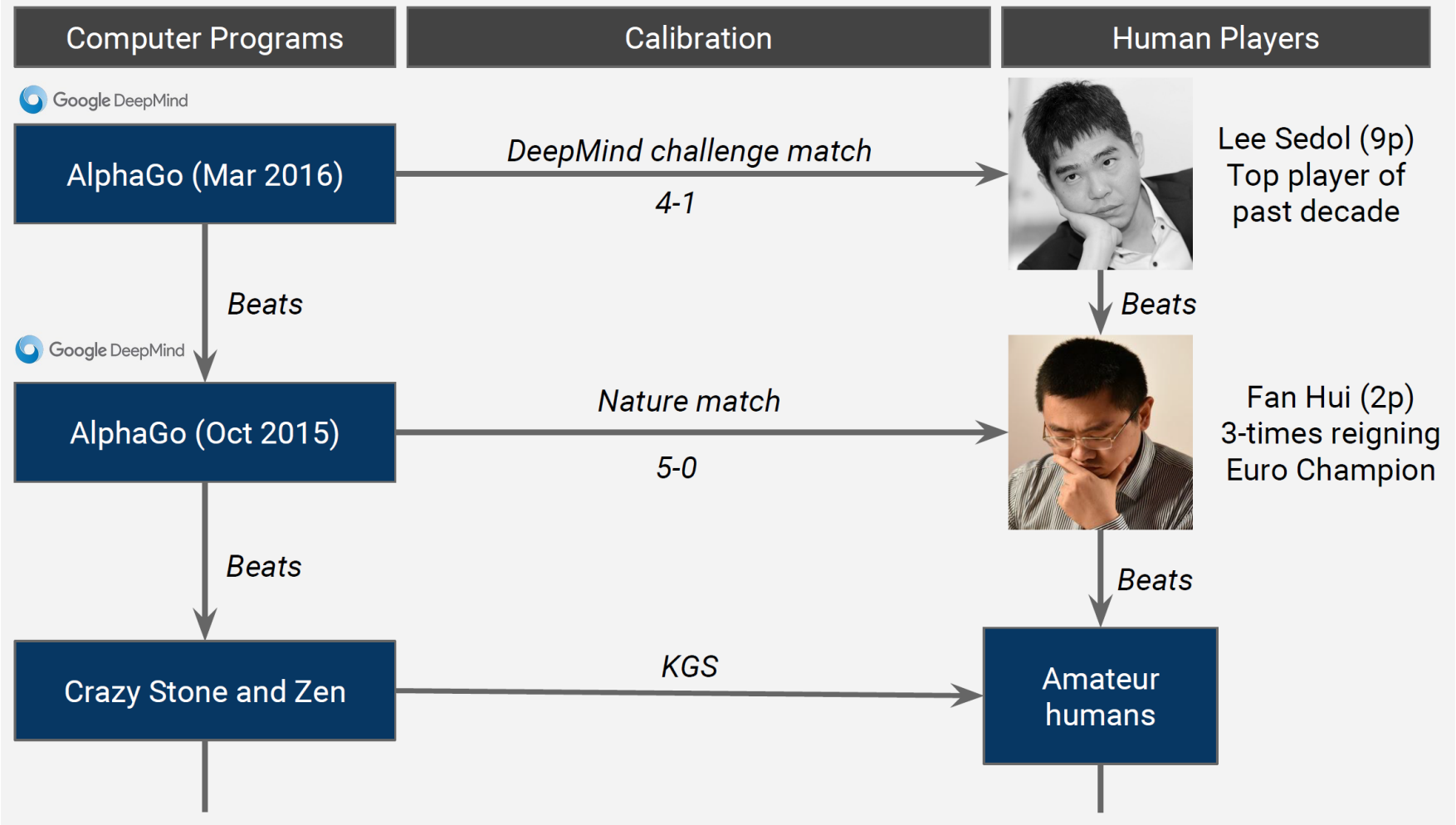
Brute force search intractable:

1. Search space is huge
2. “Impossible” for computers to evaluate who is winning

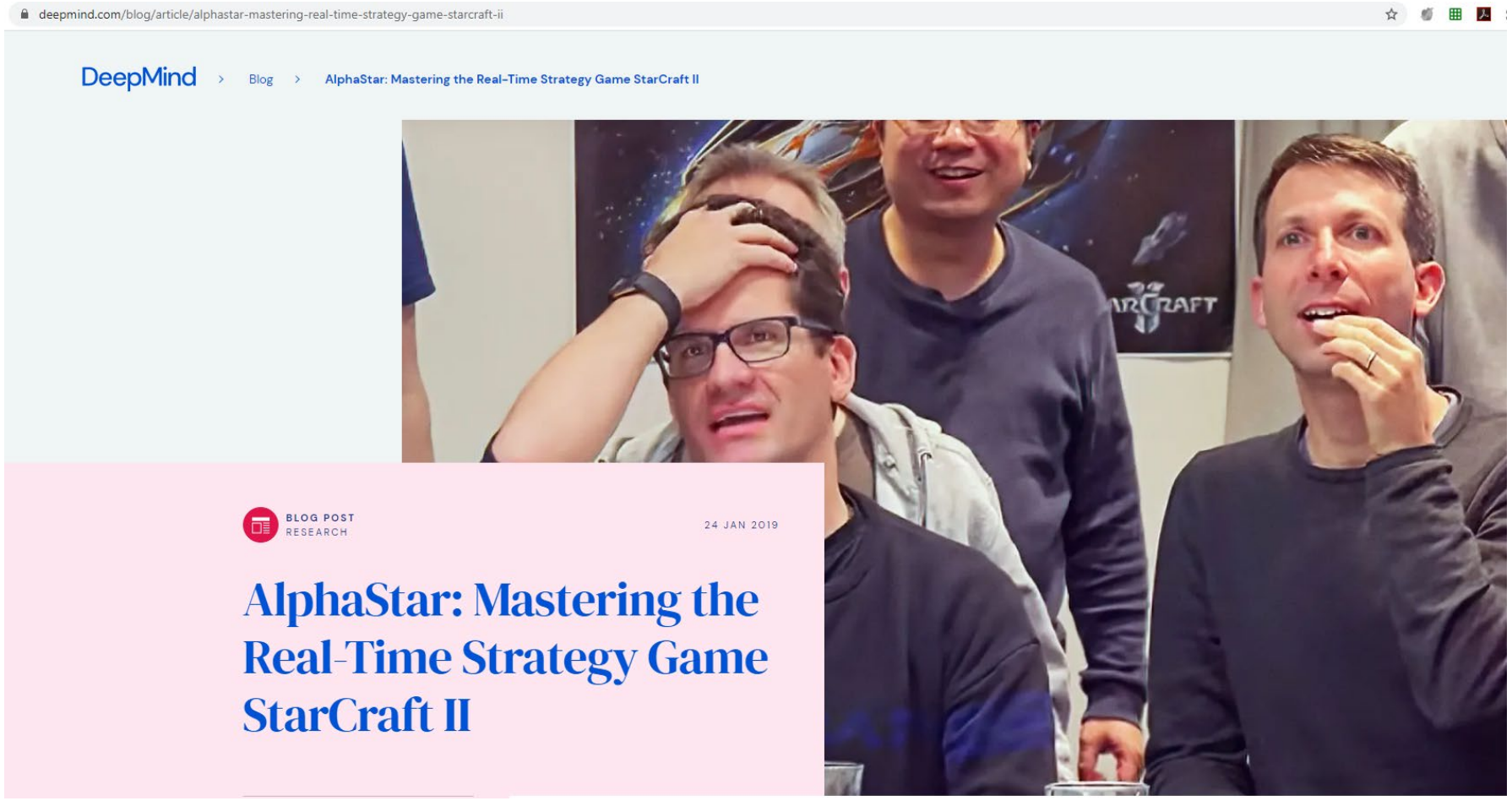


Evaluating AlphaGo against computers

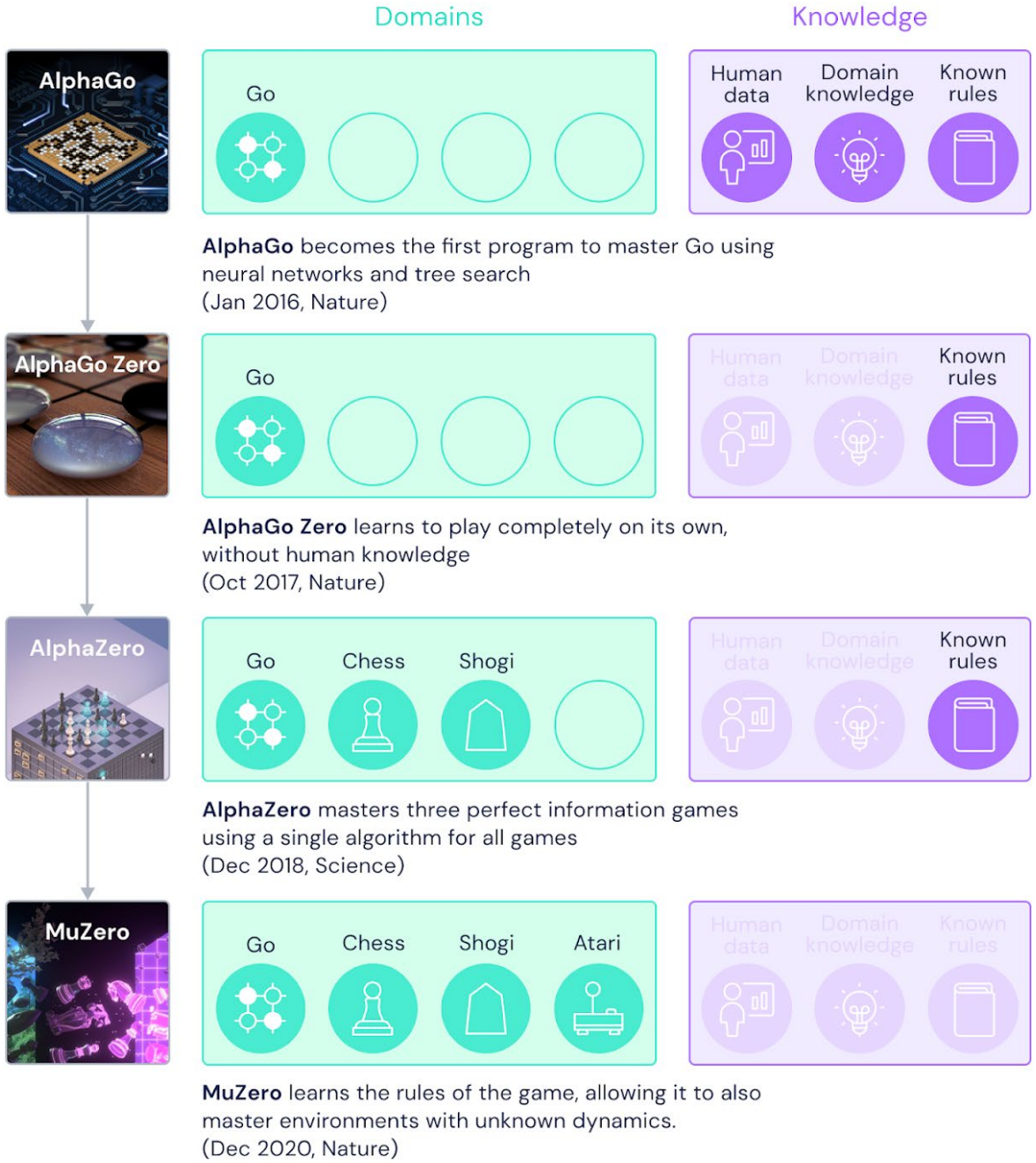




DeepMind was bought by Google for >$500 millions in 2014
The story behind it by Greg Kohs, "AlphaGo" (2017)
AlphaGo Zero defeated 100-0 the AlphaGo Lee versión in 2018

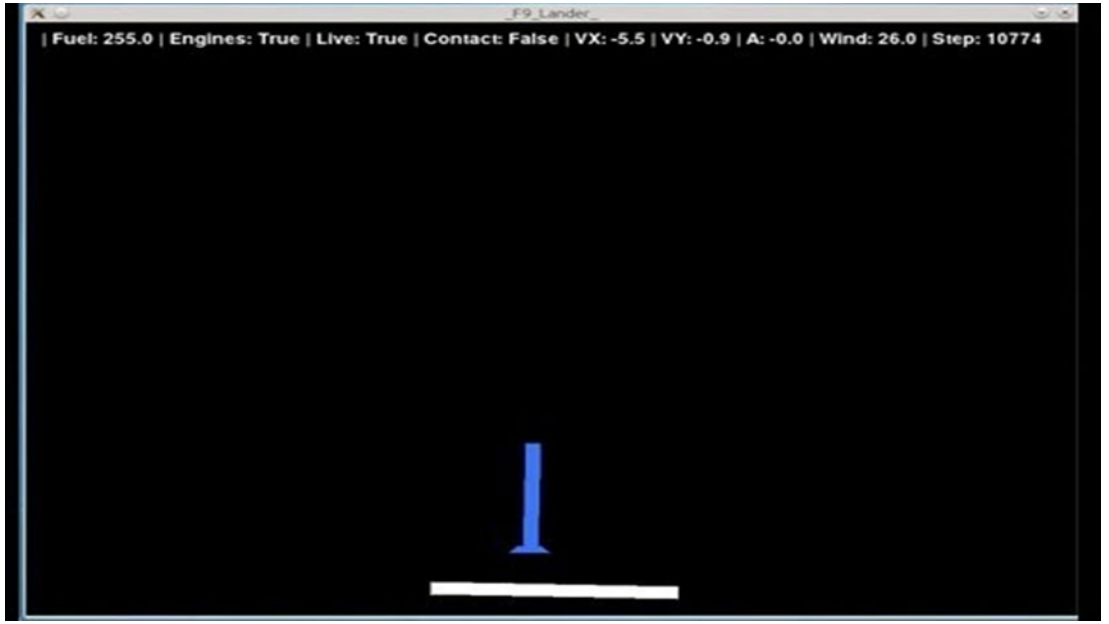


[MuZero: Mastering Go, chess, shogi and Atari without rules | DeepMind]


Domains
Knowledge
AlphaGo
Go
Human data
Domain knowledge
Known rules
AlphaGo becomes the first program to master Go using neural networks and tree search
(Jan 2016, Nature)
AlphaGo Zero
Go
Known rules
AlphaGo Zero learns to play completely on its own, without human knowledge
(Oct 2017, Nature)
AlphaZero
Go
Chess
Shogi
Known rules
AlphaZero masters three perfect information games using a single algorithm for all games
(Dec 2018, Science)
MuZero
Go
Chess
Shogi
Atari
MuZero learns the rules of the game, allowing it to also master environments with unknown dynamics.
(Dec 2020, Nature)

Some examples of RL applications

- SpaceX Falcon 9 landing with RL – Policy Proximal Optimization



https://towardsdatascience.com/spacex-falcon-9-landing-with-rl-7dde2374eb71

Some examples of RL applications

- Automatic piloting in aircrafts

An AI just beat a human F-16 pilot in a dogfight (Aug 2020)

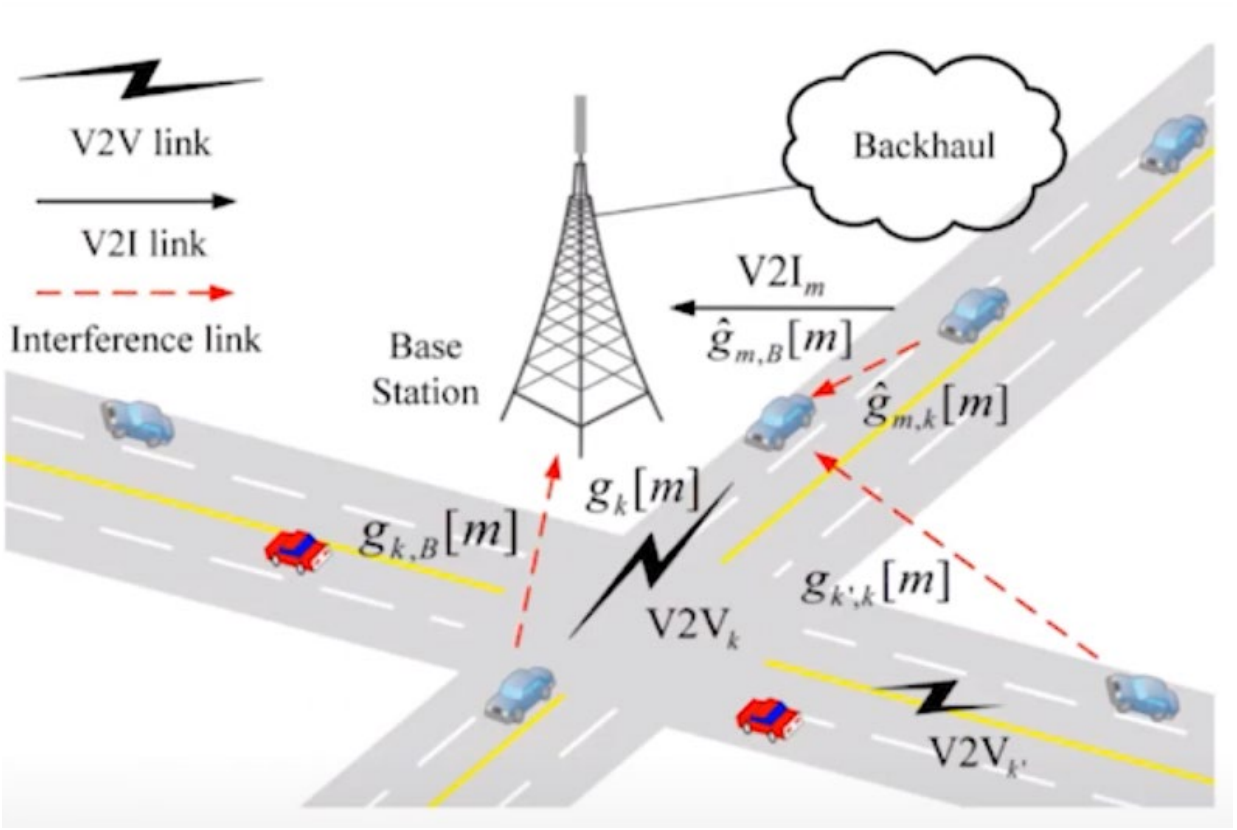
Some examples of RL applications

- Tax policy making https://blog.einstein.ai/the-ai-economist/

Stephan Zheng, Alexander Trott, Sunil Srinivasa, Nikhil Naik, Melvin Gruesbeck, David C. Parkes, and Richard Socher. "The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies." arXiv preprint arXiv:2004.13332 (2020).

Some examples of RL applications

- Resource allocation in V2X communications for 5G

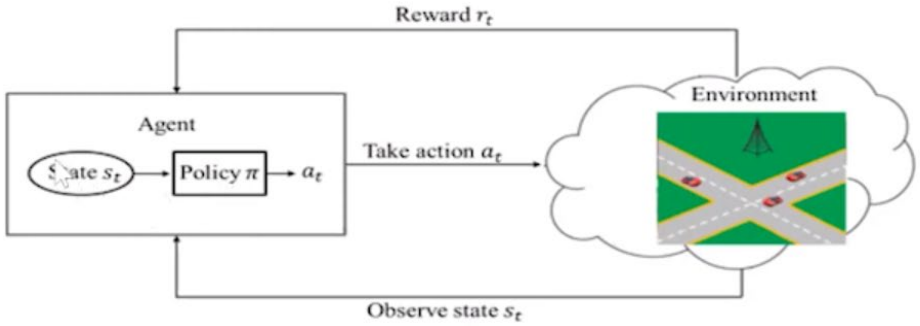


Some examples of RL applications

- Resource allocation in V2X communications for 5G

- **Agent:** Every V2V link
  - State s_t: V2V channel, interference, V2I channel, neighbors,…
  - Action a_t: sub-channel selection and transmission power level
- **Environment:**
  - Everything outside the V2V link: wireless channels + other V2V links.
  - Receiving action a_t , sending reward r_t to agents, transiting to new state s_{t+1}
- **Reward:** V2I capacity, V2V capacity, & latency constraint

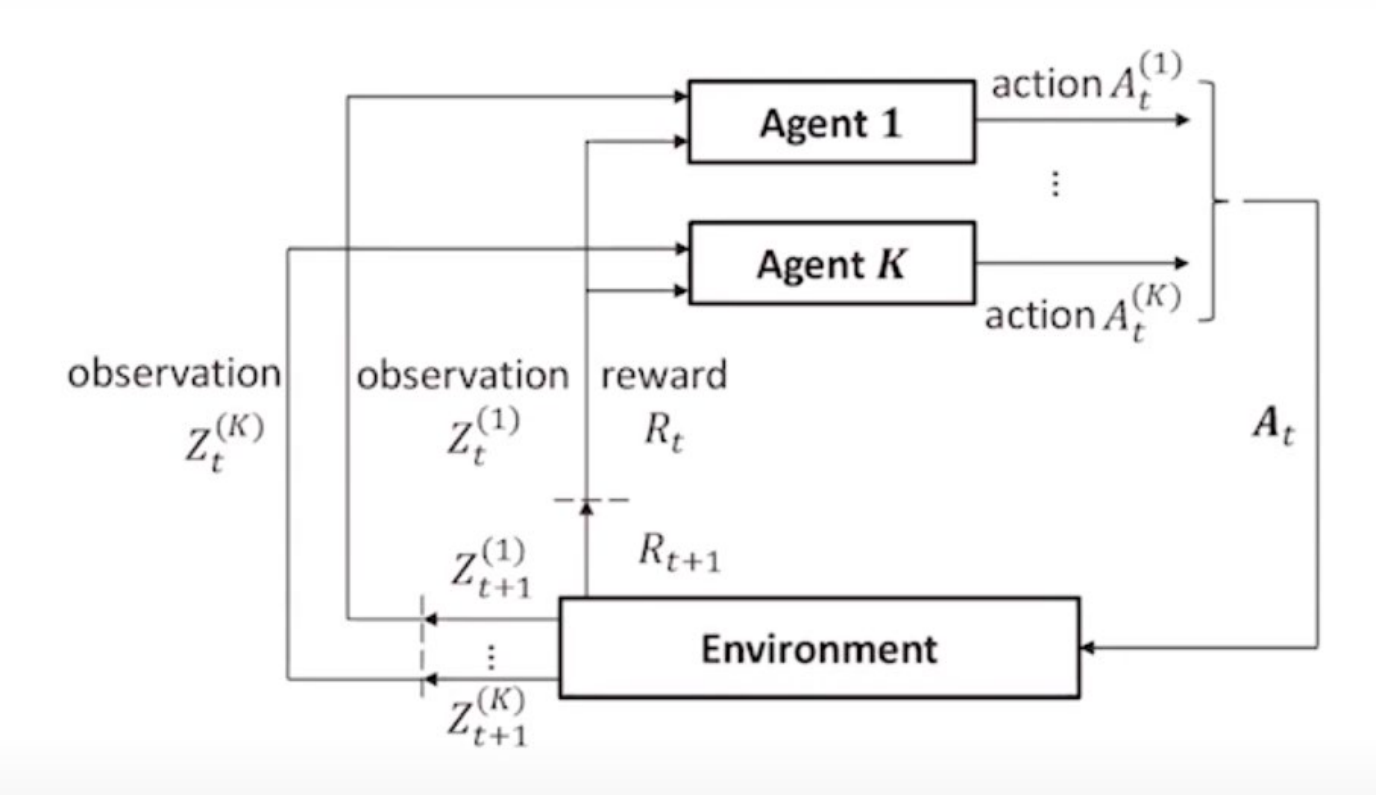
$$r_t = \lambda_c \sum_{m \in \mathcal{M}} C_m^c + \lambda_d \sum_{k \in \mathcal{K}} C_k^d - \lambda_p (T_0 - U_t)$$



H. Ye, G. Y. Li, B. H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications", *IEEE Trans. Veh. Tech.*, vol 68, no 4, April 2019

Some examples of RL applications

- Resource allocation in V2X communications for 5G: a multi-agent problem?



L. Liang, H. Ye, G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning", *IEEE J. Sel. Areas Commun.,* vol 37, no 10, Oct. 2019

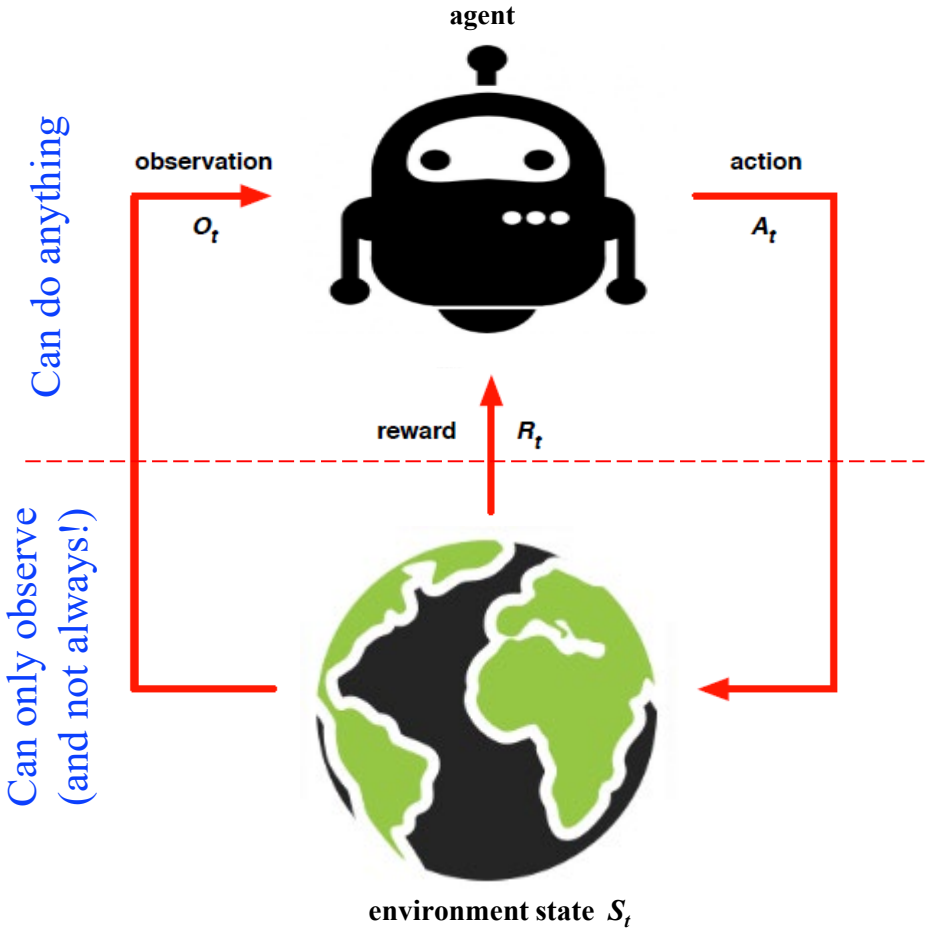
Reward is enough?

DeepMind scientist hypothesise that intelligence and its associated abilities will emerge by sticking to a simple but powerful principle: **reward maximization**.

D. Silver, S. Singh, D. Precup, R. S. Sutton, "Reward is enough", *Artificial Intelligence*, Volume 299, 2021 (https://www.sciencedirect.com/science/article/pii/S0004370221000862)

Problem statement

Goal-directed agent interacting with an uncertain environment



At each step t, the agent:

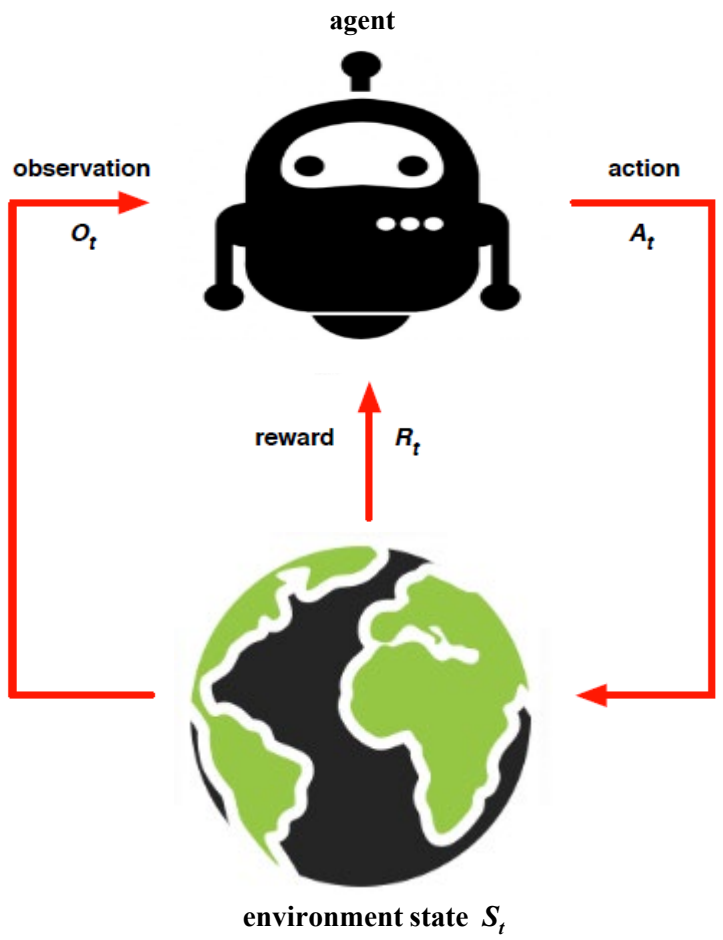
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

t increases at each environment step

Back to example 1.3. Manage advertisements in a website (YouTube, Facebook,…)



Action A_t : show a certain banner

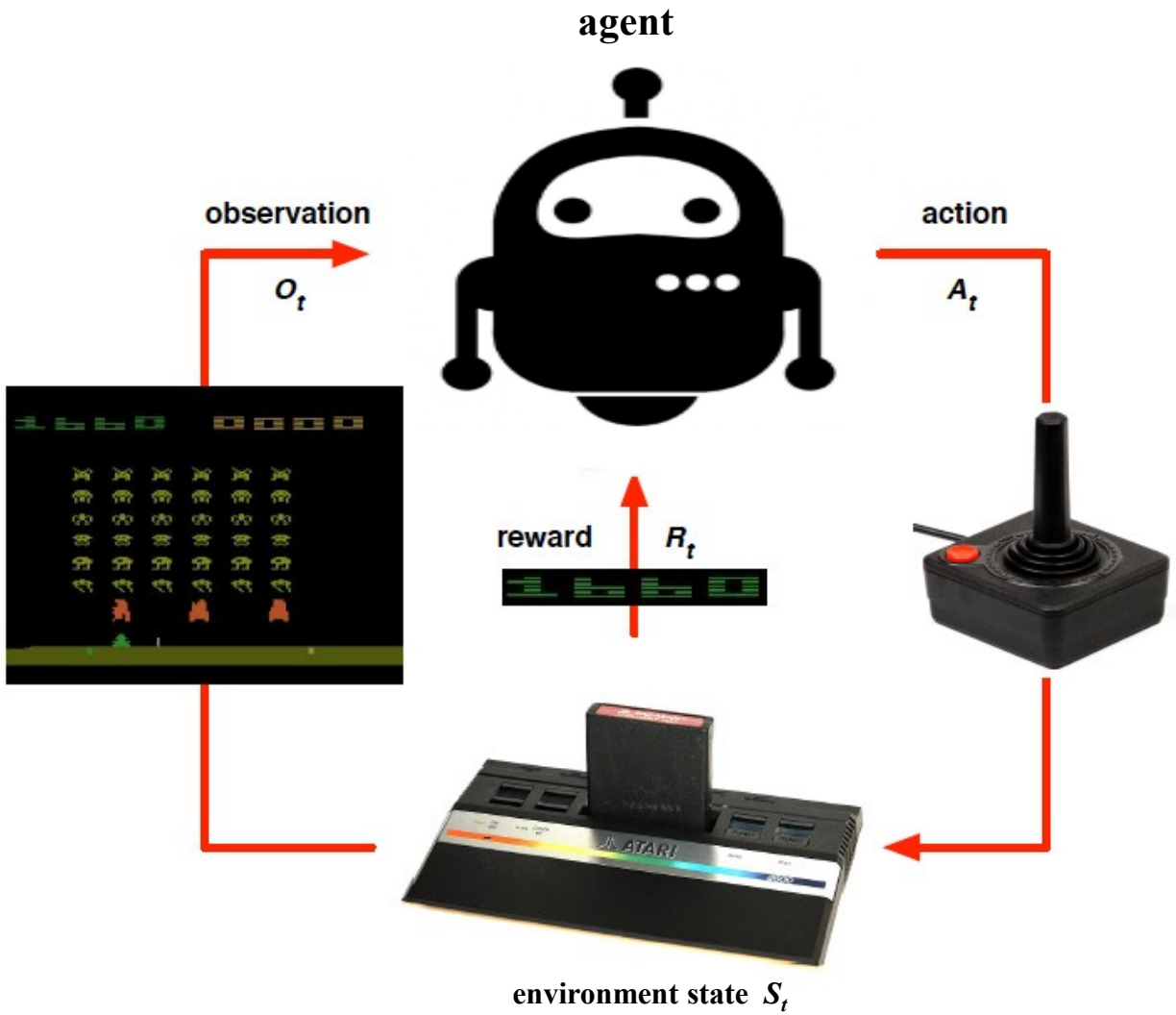
Observation O_t : number of clicks by users

State S_t : equal to O_t in a fully observable environment. Maybe discrete or continous.

Reward R_t : reward from advertising companies (maybe a delayed signal)

Can actions modify the environment? For example, showing "clickbait" banners will increase click rates, but after a while no one will trust you!

For the Atari gamer, identify all elements of an MDP…

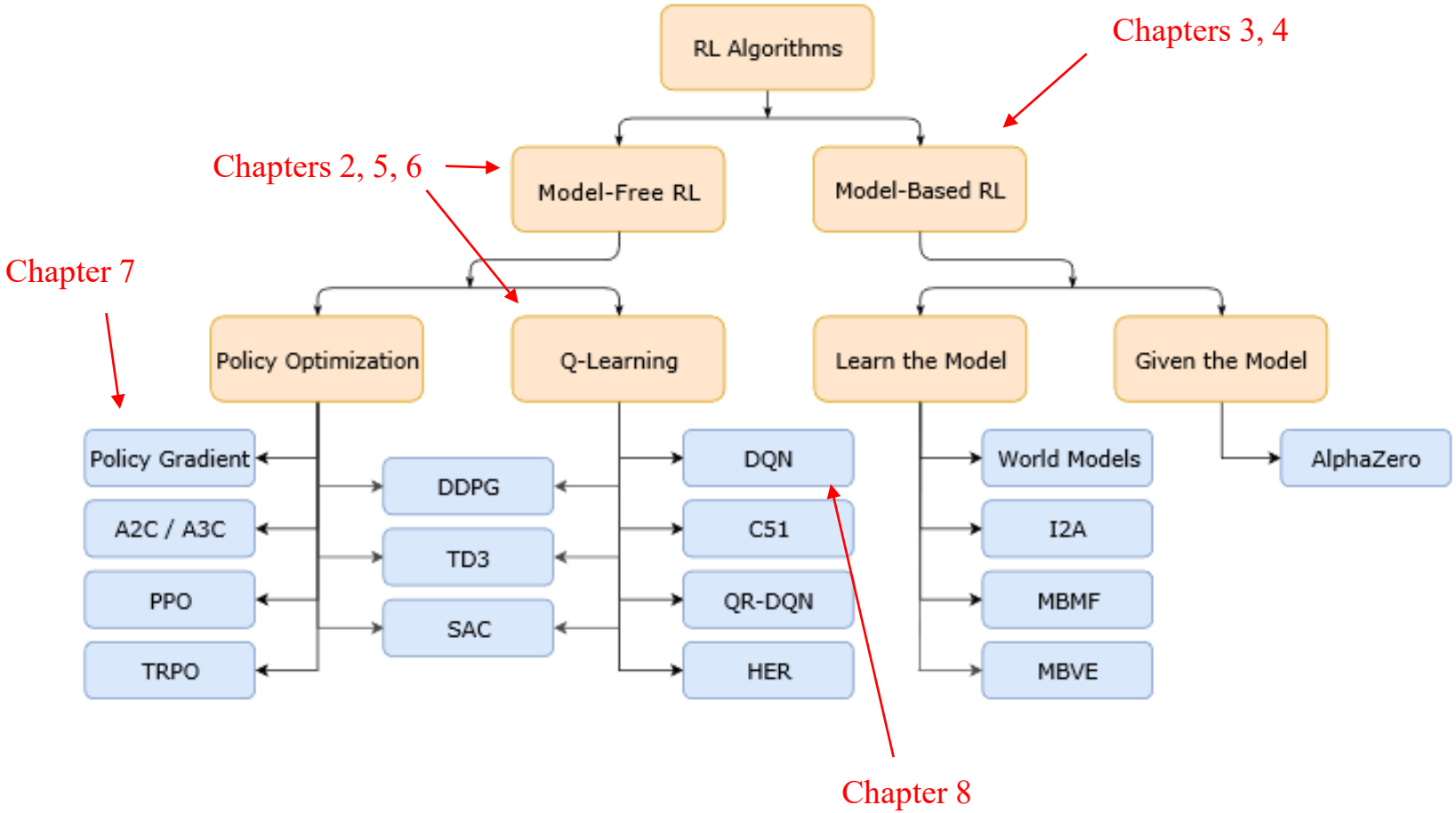


A thermostat controller, identify all elements of an MDP…

Key concepts in short…

- Two paradigms:
  - Model-based: environment behaviour is known
  - Model-free RL: there is no prior knowledge of the world
- Trade-off between exploration and exploitation:
  - Trial-and-error search
  - Optimize reward
- In RL there is no supervisor, only a reward signal.
- Feedback is possibly delayed, not instantaneous.
- Decisions are taken sequentially (time really matters).
- Agent's actions affect the environment, and hence the feedback it receives.

A taxonomy of RL



From [OpenAI Spinning Up]