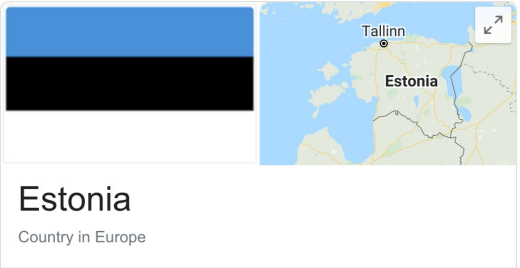
DeepMind

AI Research at Scale - Opportunities on the Road Ahead

Dan Belov

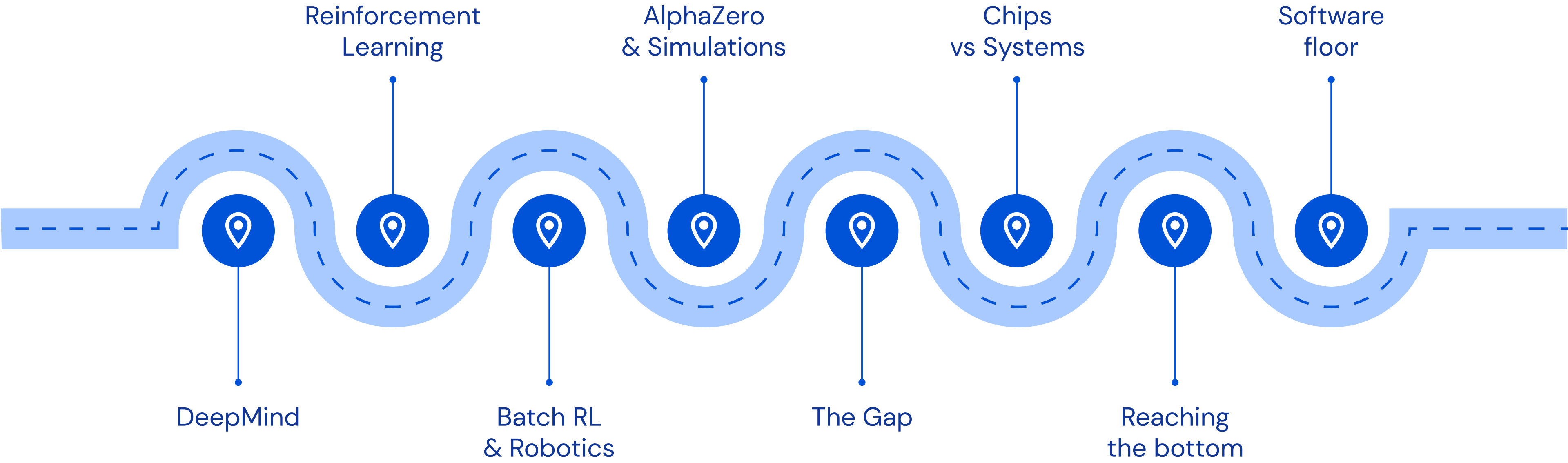
Digression: Who am I?

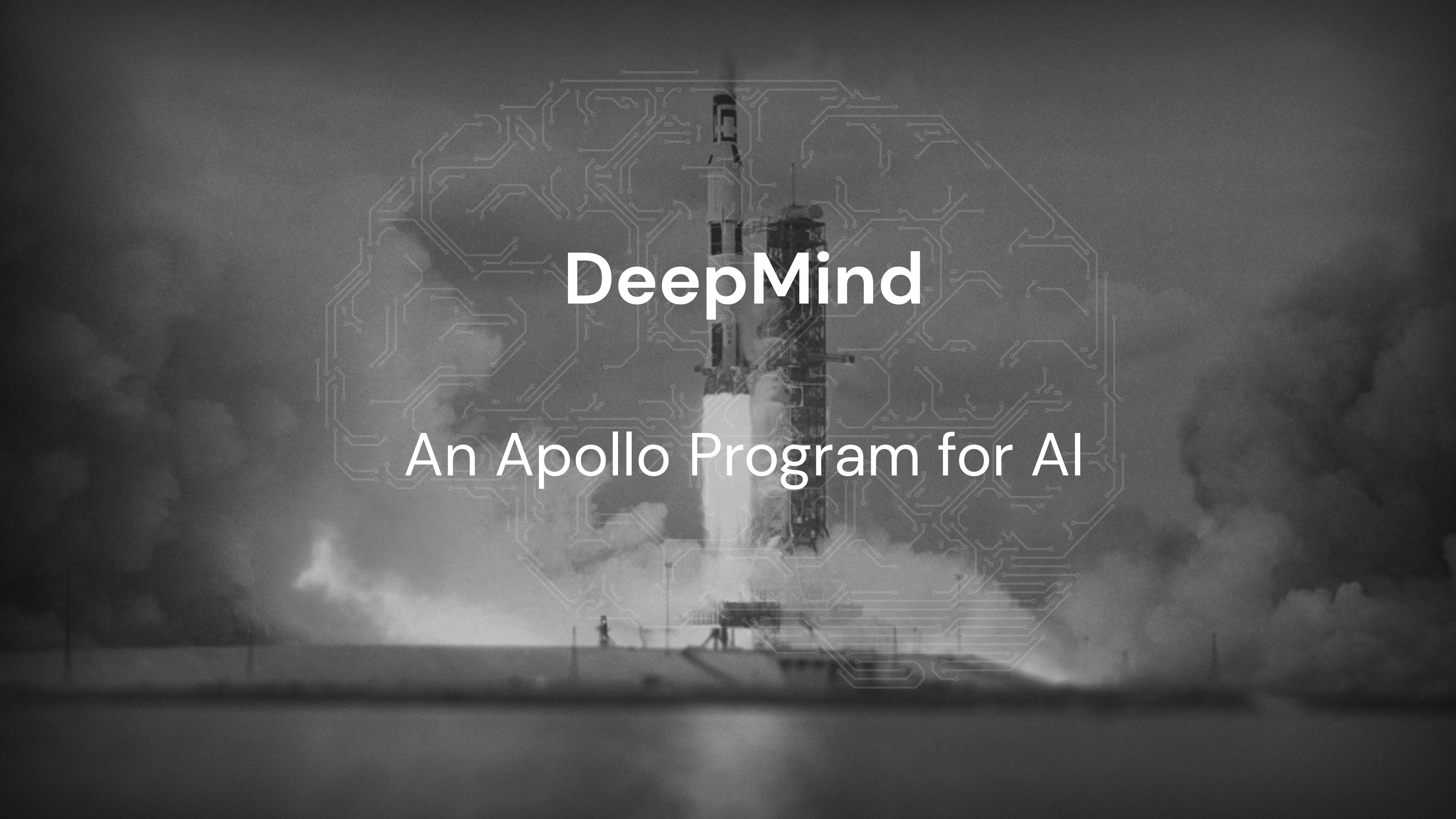


Evan-Amos / Public domain

http://www.computinghistory.org.uk/det/11264/SGI-Origin-2000/

Roadmap


DeepMind
Reinforcement Learning
Batch RL & Robotics
AlphaZero & Simulations
The Gap
Chips vs Systems
Reaching the bottom
Software floor


DeepMind
An Apollo Program for AI

OUR MISSION

01
Solve intelligence

02
Use it to solve everything else

Neuroscience Inspired Artificial Intelligence

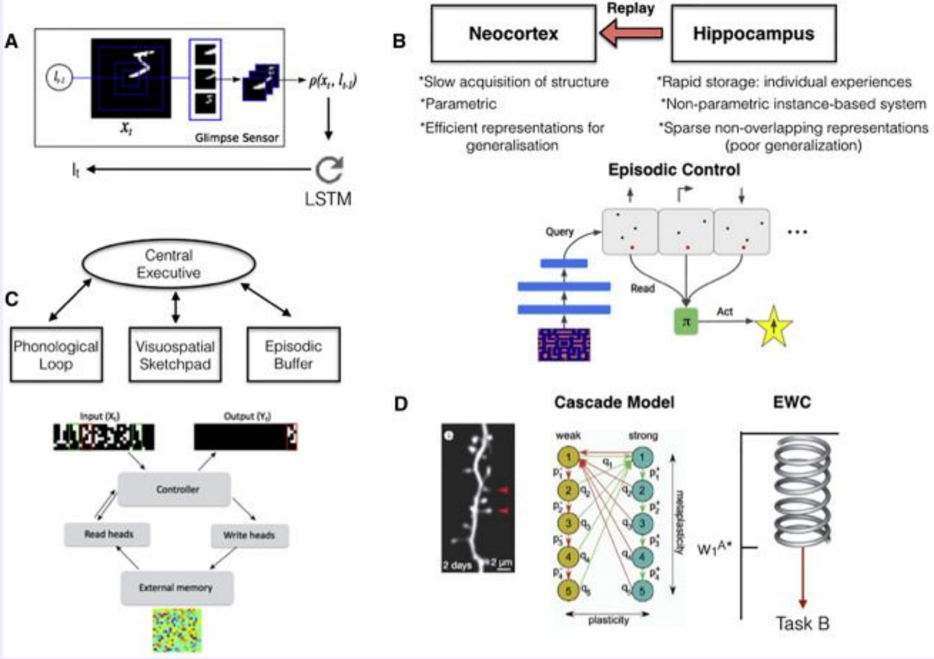
Neuroscience has made fundamental contributions to advancing AI research.

Past contributions have rarely involved a simple transfer of full-fledged solutions

Rather, neuroscience has typically been useful in a subtler way

Neuroscience-Inspired Artificial Intelligence

Demis Hassabis,[1,2,*] Dharshan Kumaran,[1,3] Christopher Summerfield,[1,4] and Matthew Botvinick[1,2]
[1]DeepMind, 5 New Street Square, London, UK
[2]Gatsby Computational Neuroscience Unit, 25 Howland Street, London, UK
[3]Institute of Cognitive Neuroscience, University College London, 17 Queen Square, London, UK
[4]Department of Experimental Psychology, University of Oxford, Oxford, UK
*Correspondence: dhcontact@google.com
http://dx.doi.org/10.1016/j.neuron.2017.06.011



Virtual environments accelerate AI but carry complexity

1

Microcosms of the real world

Games are a proving ground for real-world situations

2

Stimulate intelligence

By presenting a diverse set of challenges

3

Good to test in simulations

Efficient, run thousands in parallel, faster than real time

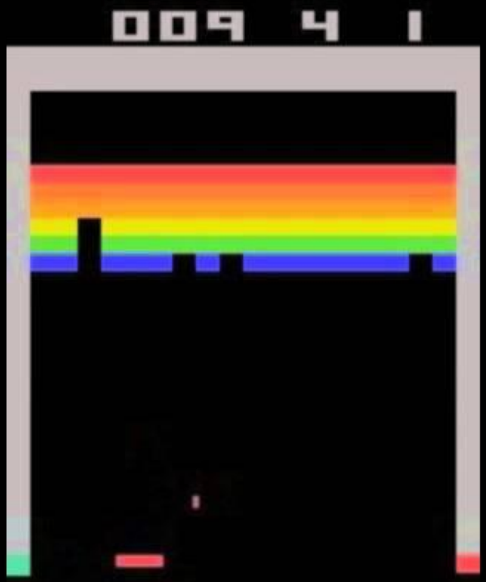
4

Measure progress and performance

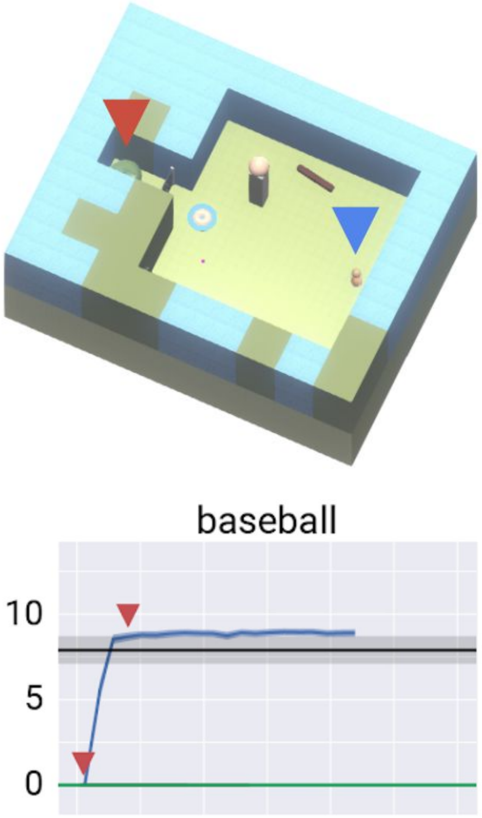
Measure progress and compare against human performance

Virtual environments are rich substrates to progress AI Research. But their increasing complexity demands commensurate technology infrastructure

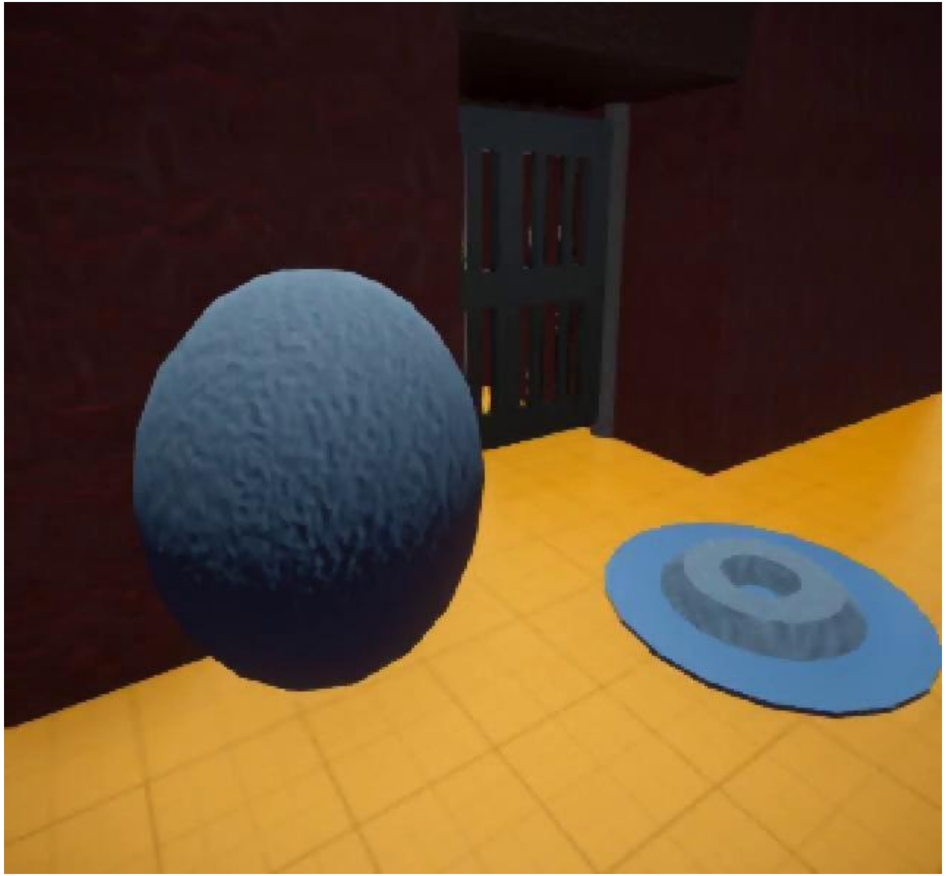
Atari with deep RL



Complex planning in 3D environments



R2D3
Tom Paine, Caglar Gulcehre et al
2019



How does this work?

Machine learning

1 ML is about inferring knowledge from observations or experiences, subject to the physical laws of the world.

2 ML is also about using this knowledge to guide observation and hypothesis testing.

3 ML is about creating new knowledge, using the present knowledge, to solve a large diversity of novel problems.

ML Magic

- Largest possible network with few general inductive biases.
- Massive curated dataset. Clear goal.
- Best existing communication, memory and computation infrastructure.

Supervised Deep Learning

Inferring knowledge
from observations.

Given pairs {inputs, outputs}
can now learn pretty much
any function, subject to:

- enough data
- enough compute
- representative samples

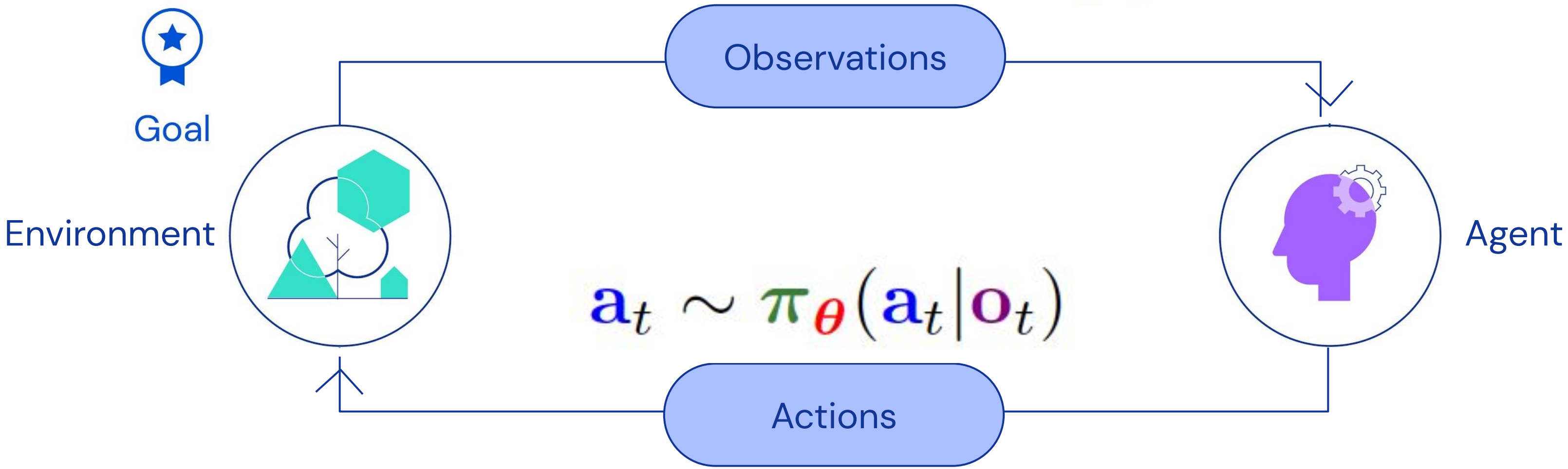
Iron law of Deep Learning:

More is More.

Reinforcement learning

$$\mathbf{o}_{t+1} \sim P(\mathbf{o}_{t+1}|\mathbf{a}_t, \mathbf{o}_t)$$

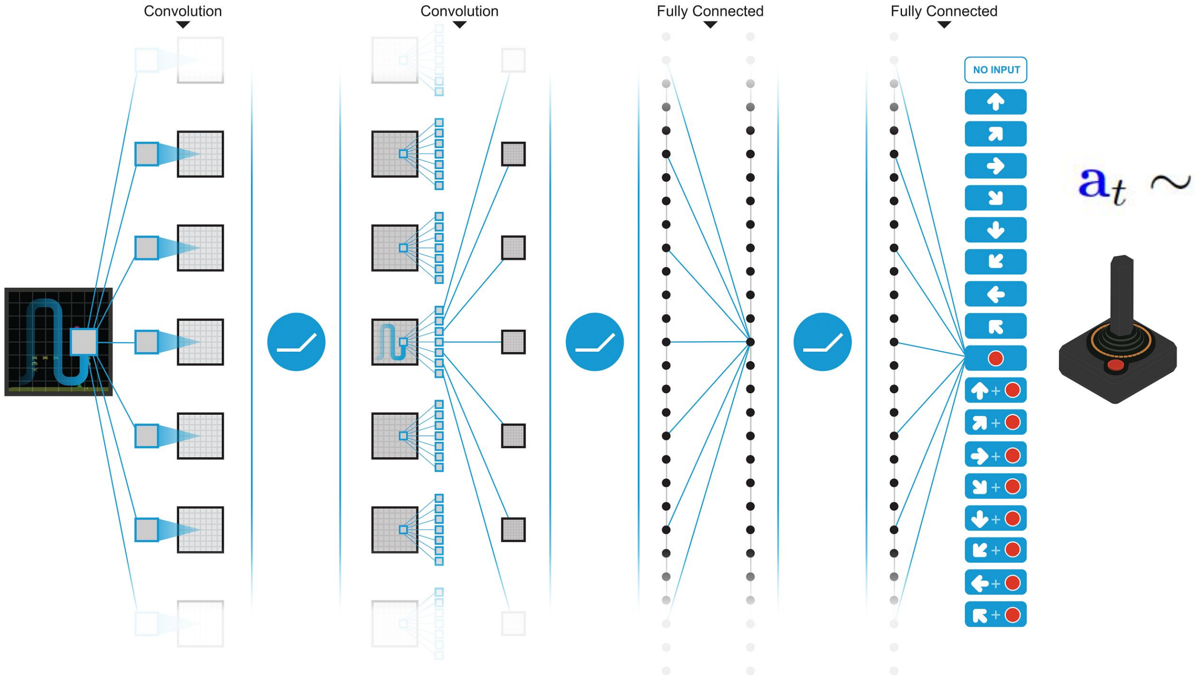
$$\mathbf{r}_t = \mathbf{r}(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$$



$$\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$$

Making good decisions by learning from experience

Deep RL uses deep neural nets as policies



$$a_t \sim \pi_\theta(a_t|o_t)$$

Making good decisions by learning from experience

Value functions and optimal value functions

The value of being in a state and following a deterministic policy $\mathbf{a}_t = \pi(\mathbf{o}_t)$

$$\mathbf{V}^{\pi}(\mathbf{o}_0) = \mathbb{E}_{\tau}\left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}(\mathbf{o}_t, \pi(\mathbf{o}_t), \mathbf{o}_{t+1}) \middle| \mathbf{o}_0\right]$$

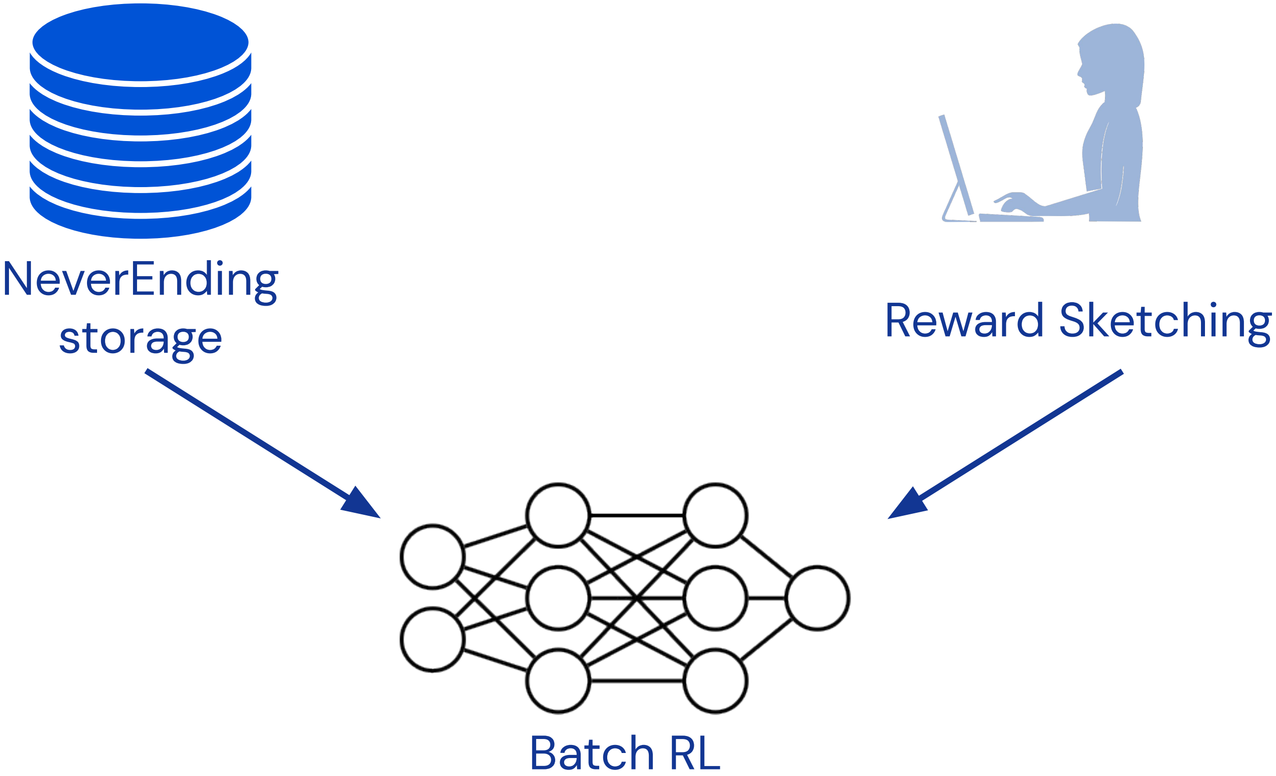
The optimal value function

$$\mathbf{V}^{\star}(\mathbf{o}_0) = \max_{\pi} \mathbb{E}_{\tau}\left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}(\mathbf{o}_t, \pi(\mathbf{o}_t), \mathbf{o}_{t+1}) \middle| \mathbf{o}_0\right]$$

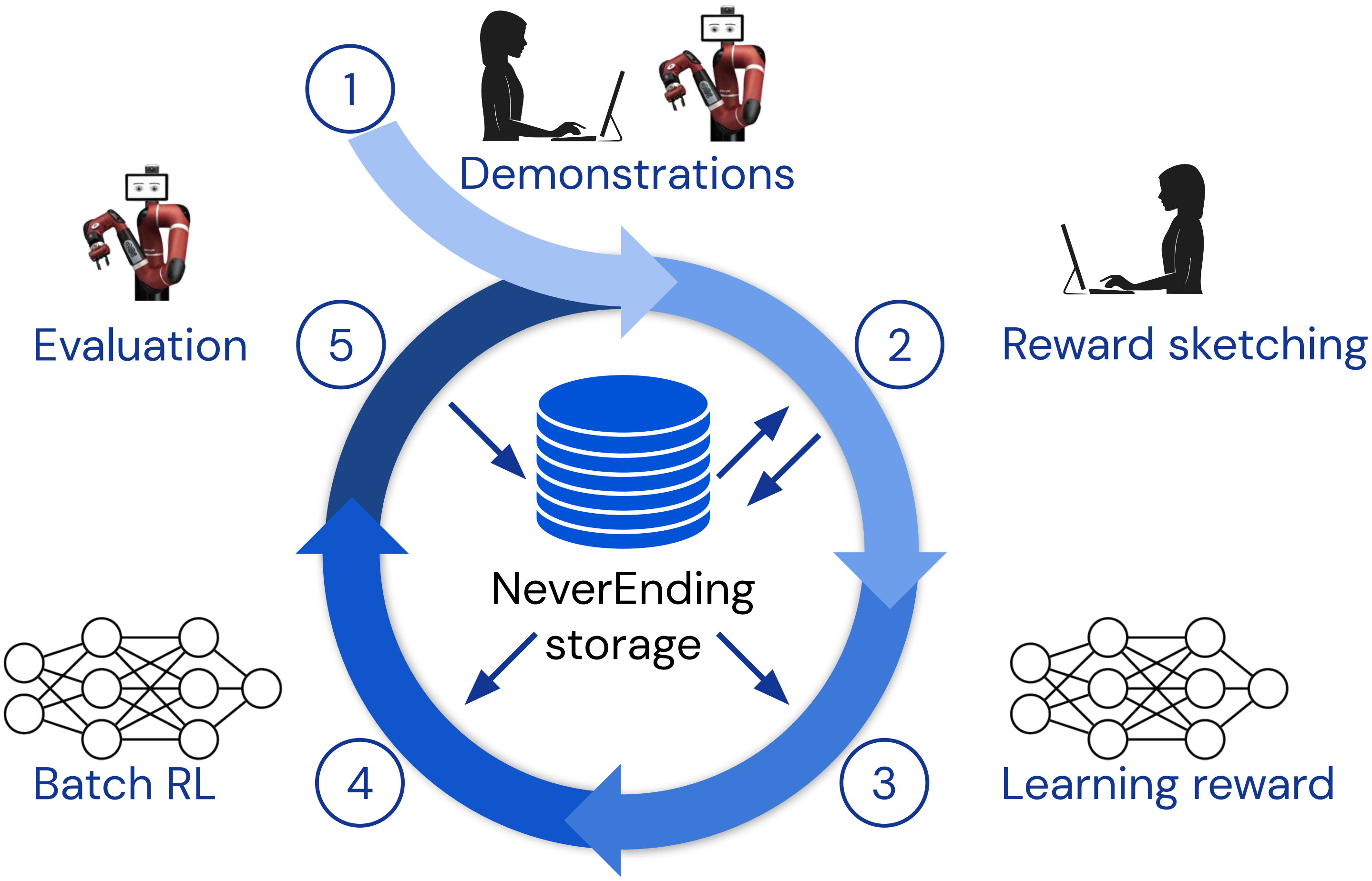
DeepMind

Example: Data Driven Robotics

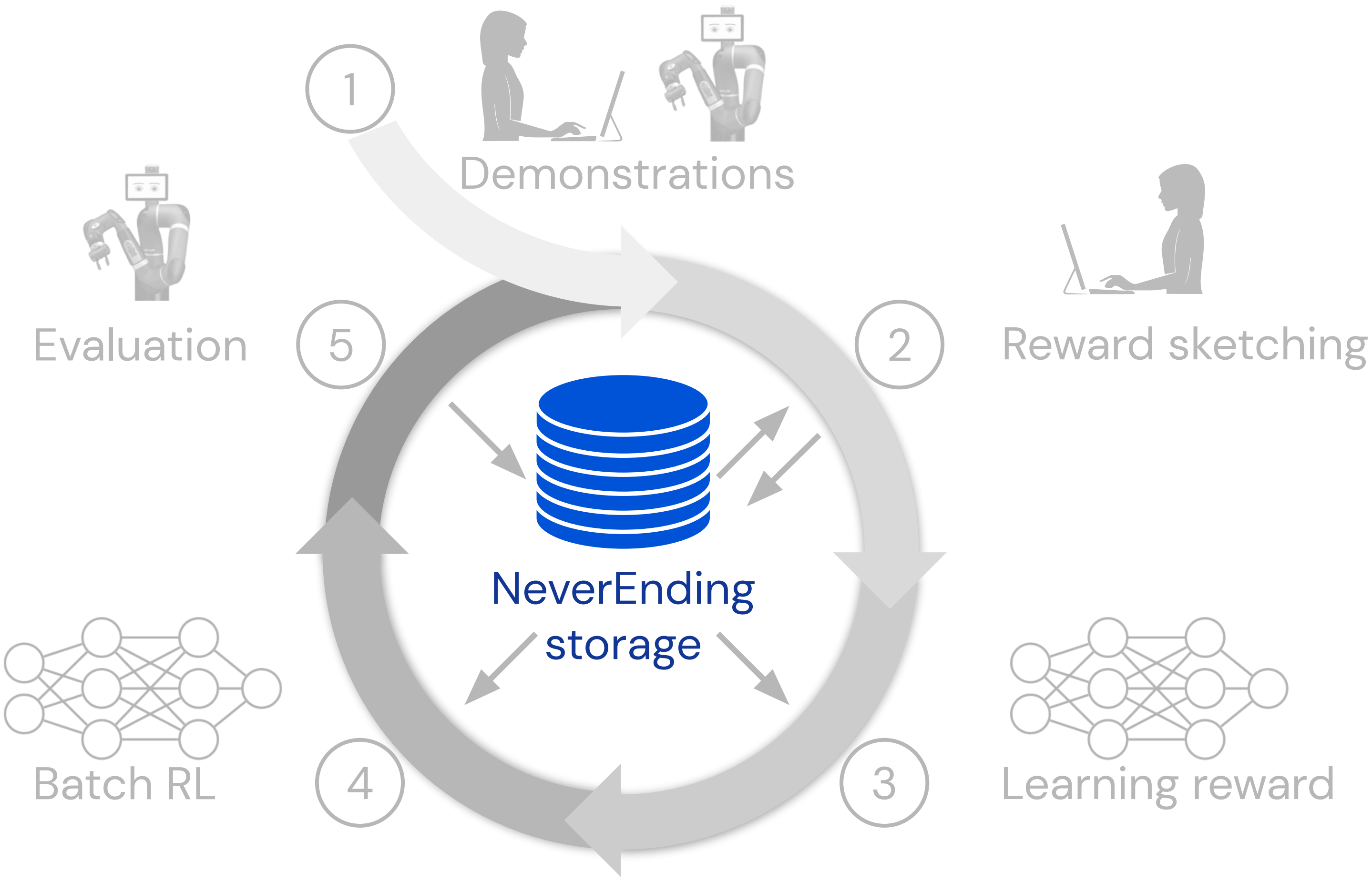
A framework for data-driven robotics



(Serkan Cabi et al., 2019)

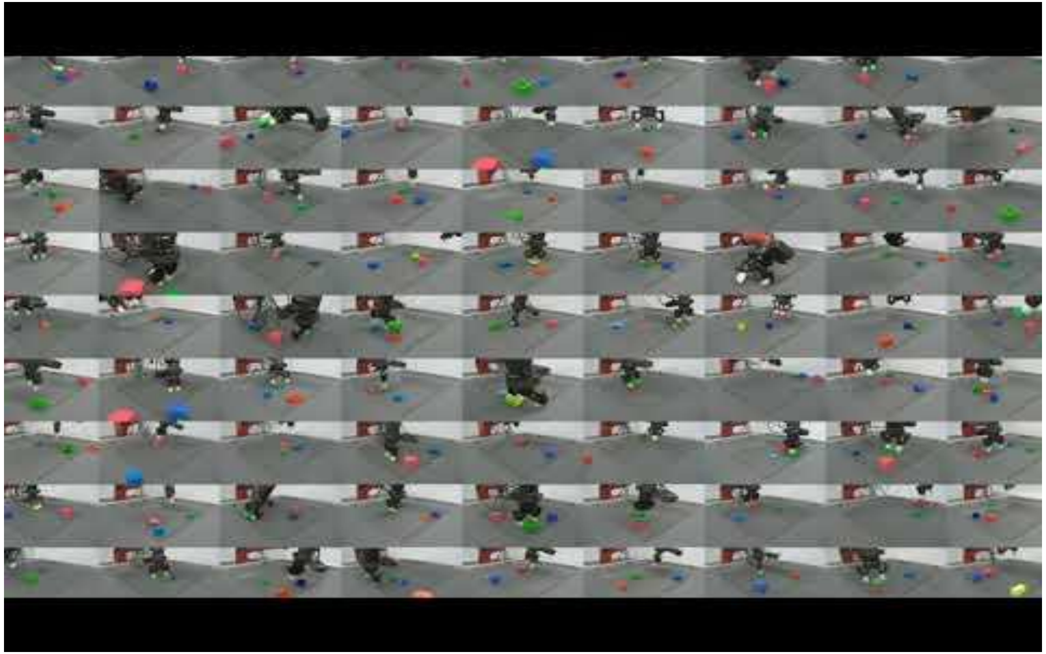


(Serkan Cabi et al., 2019)



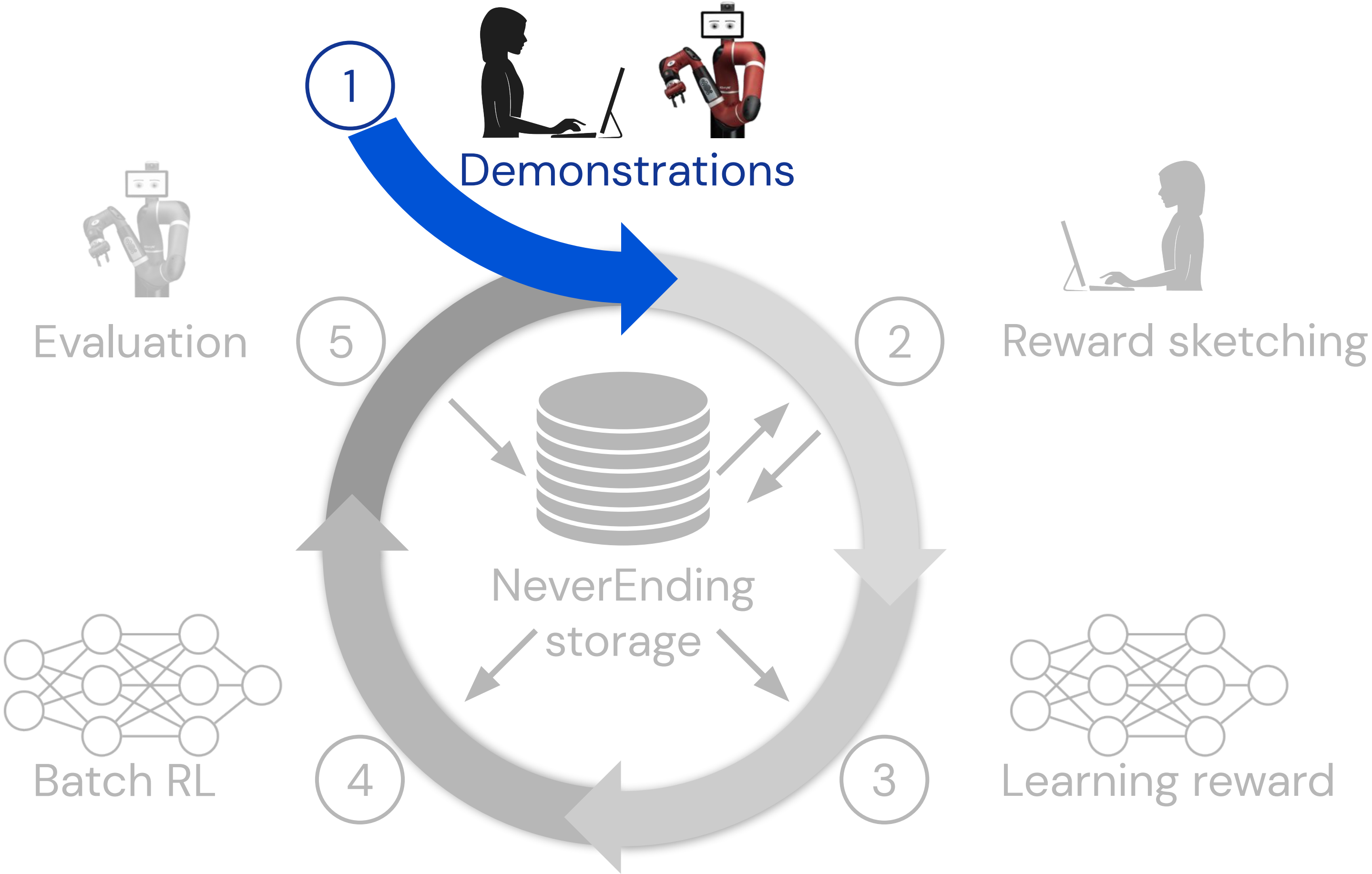
(Serkan Cabi et al., 2019)

NeverEnding storage



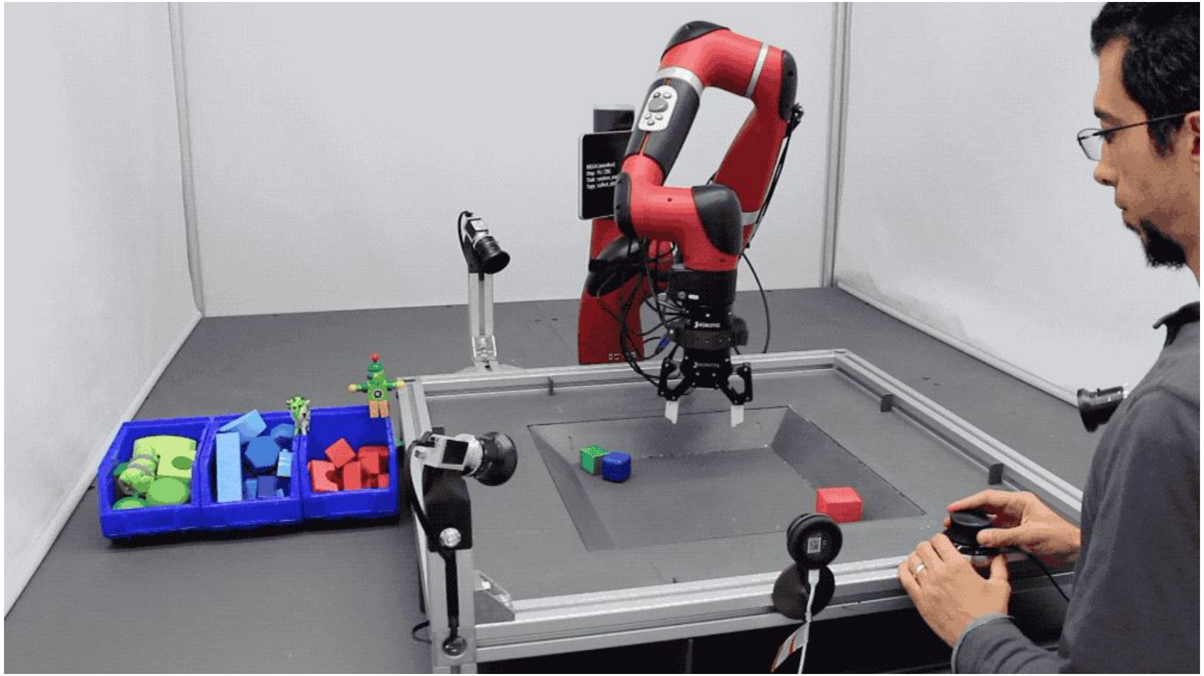
Contains:

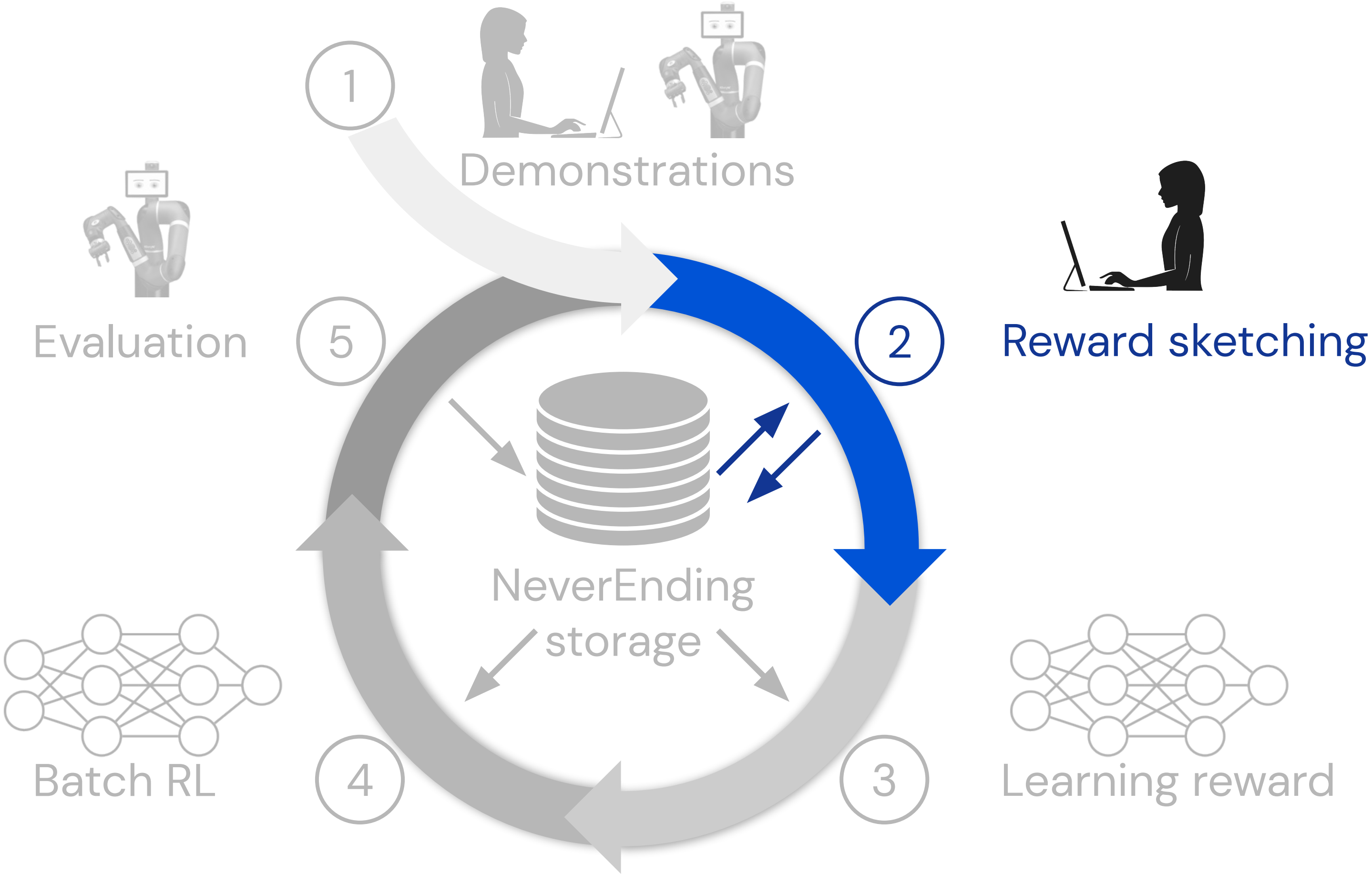
- Different tasks
- Demonstrations
- Agents
- Random policies
- Failed experiments etc.



(Serkan Cabi et al 2019)

Step 1: Demonstrations

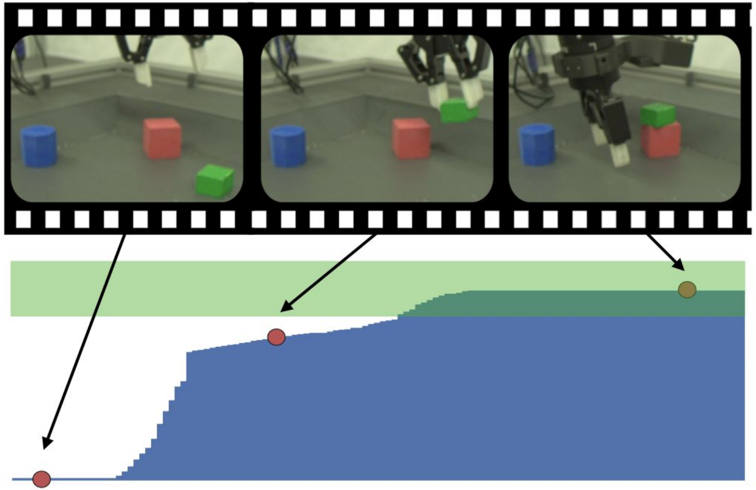


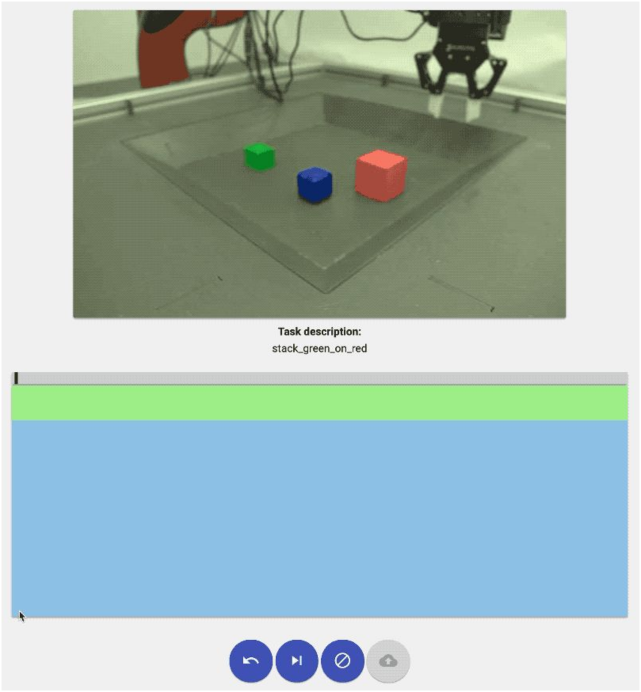


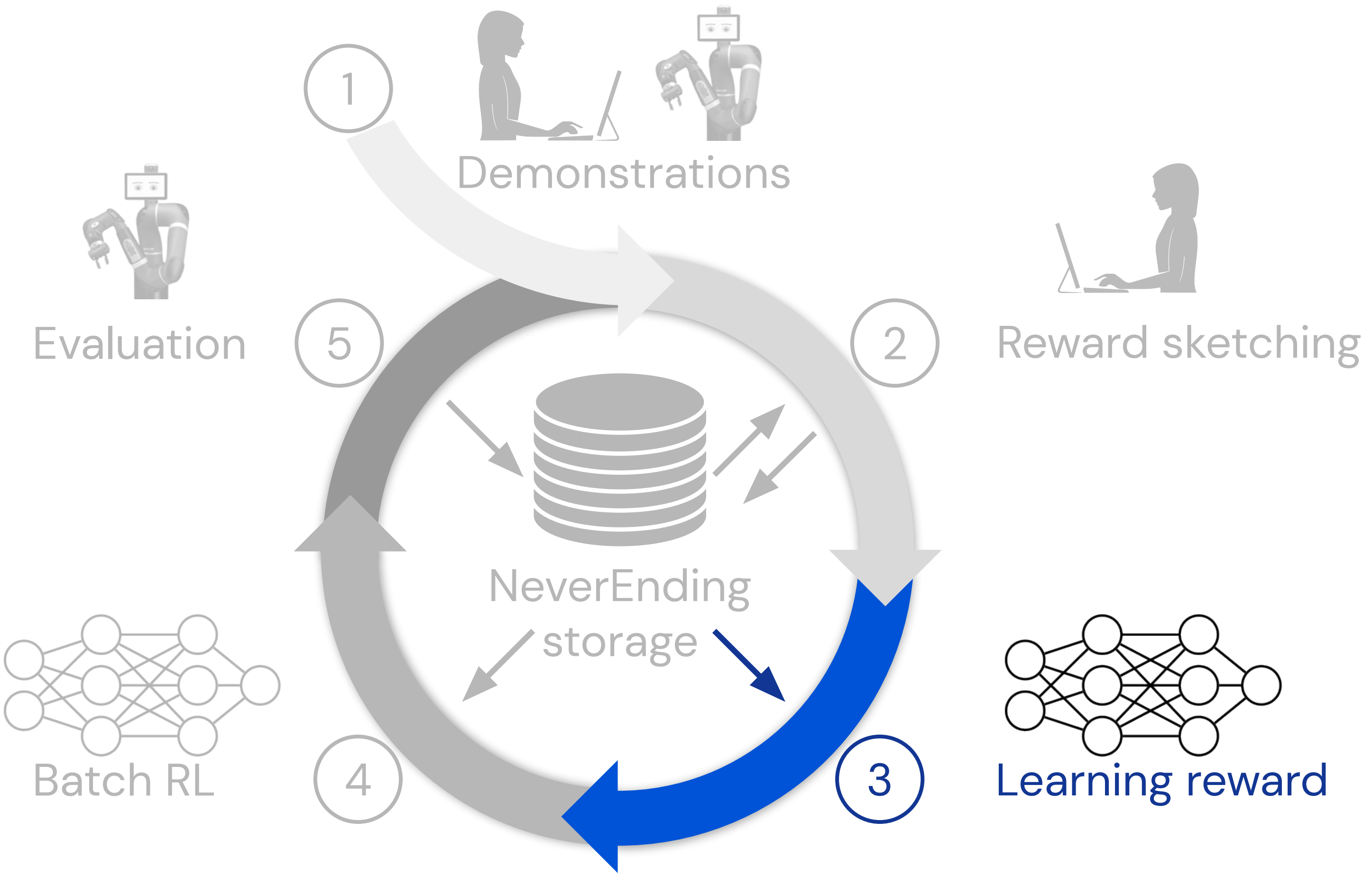
(Serkan Cabi et al., 2019)

Step 2: reward sketching

Task: stack green on red

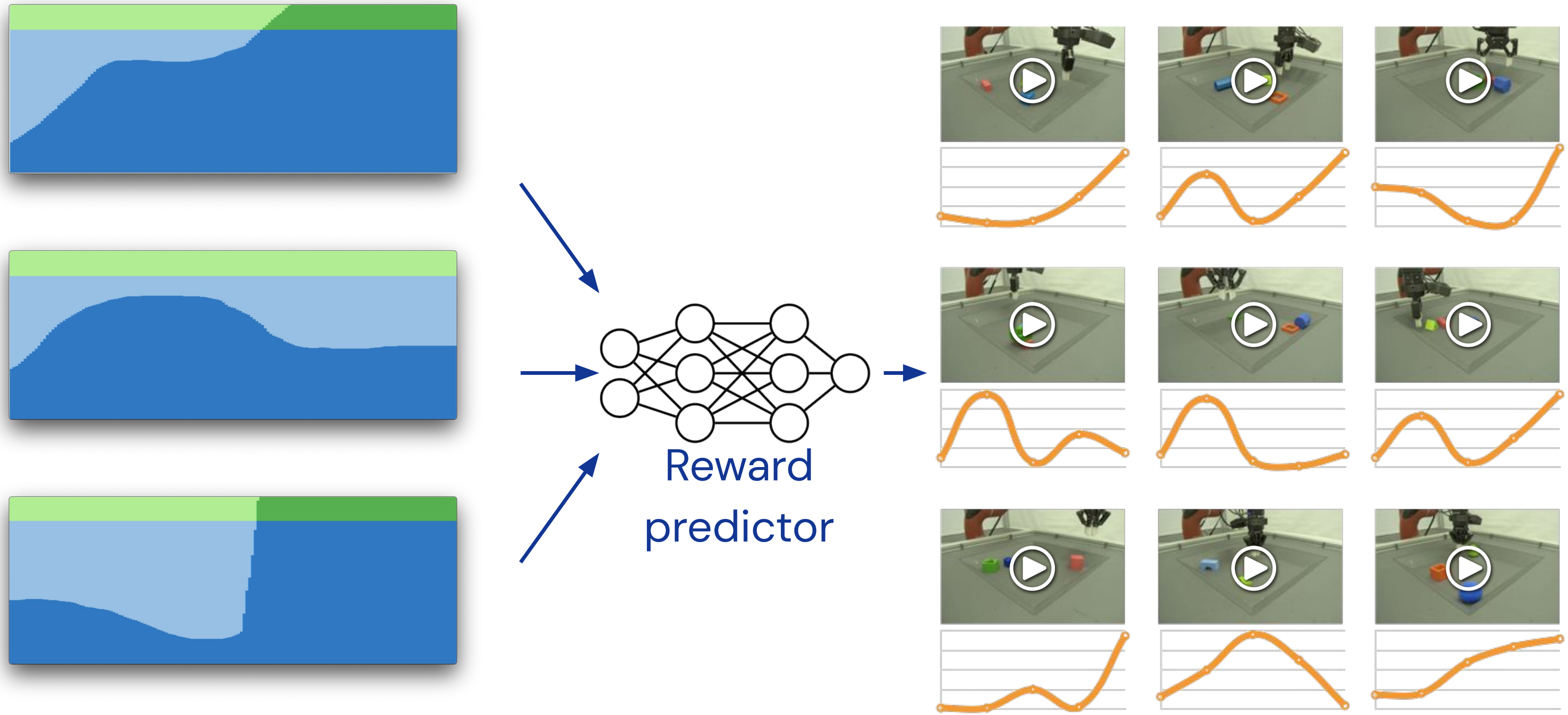


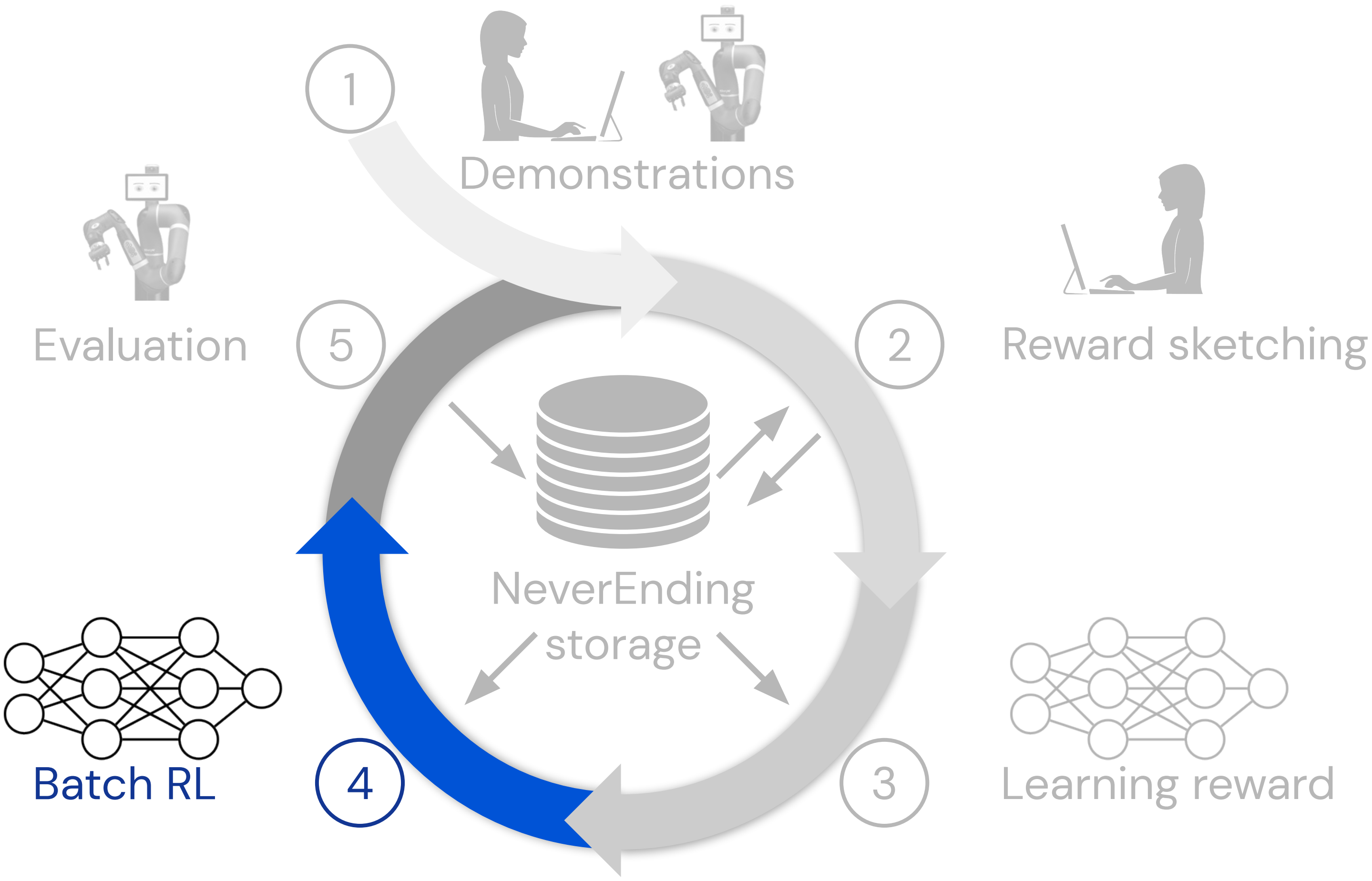




Serkan Cabi et al 2019

Learning reward



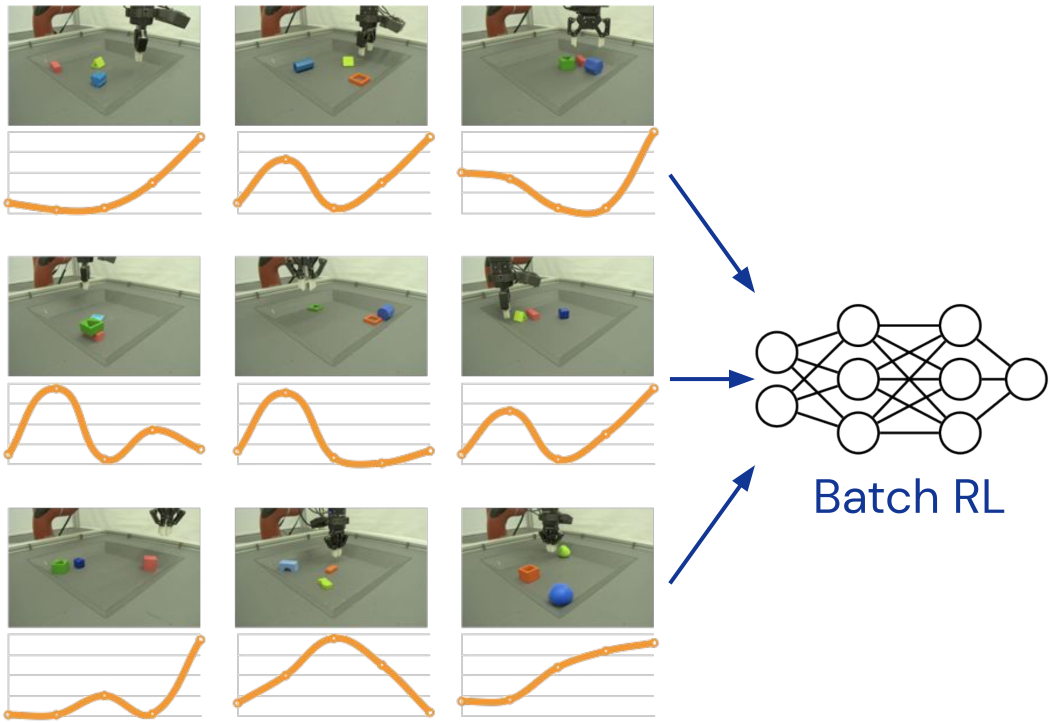


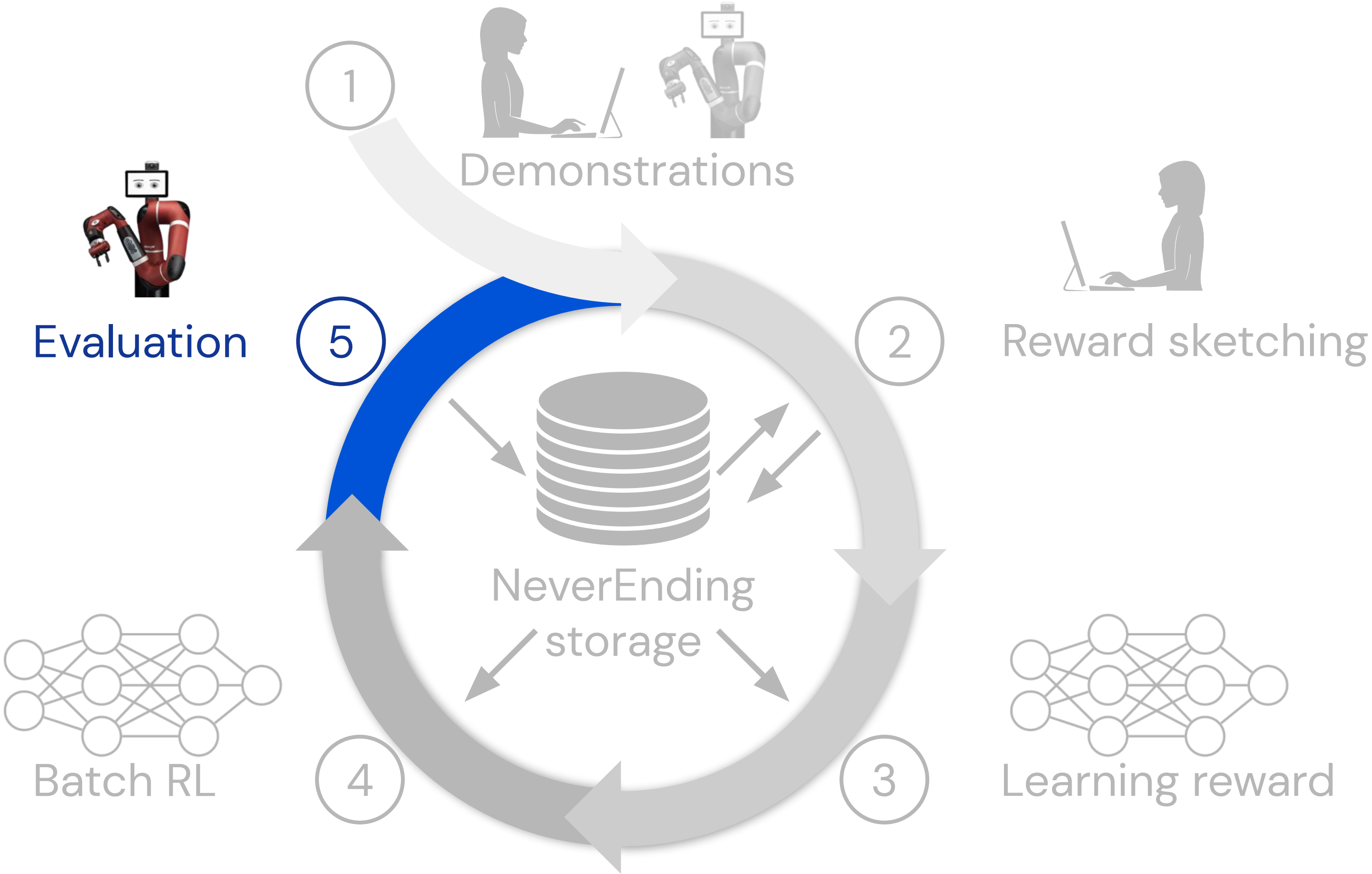
Serkan Cabi et al 2019

Batch RL

Agent:

- D4PG
- Recurrent state
- Uses demonstrations
- 25% task specific in each batch
- Diverse data is crucial!

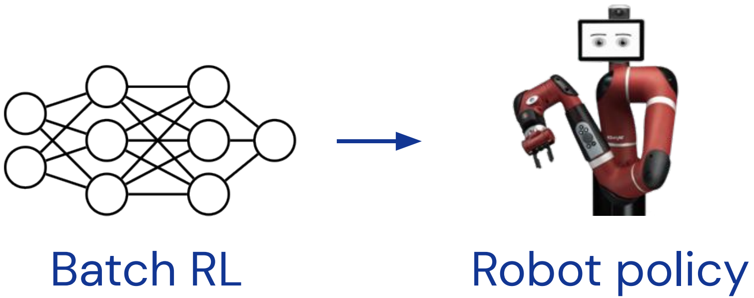




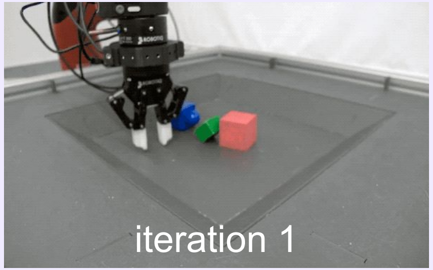
(Serkan Cabi et al., 2019)

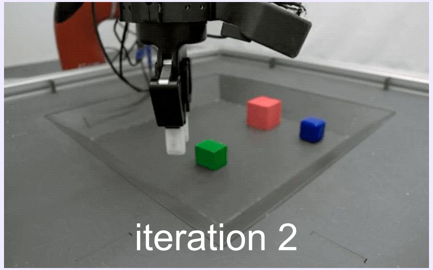
Evaluation

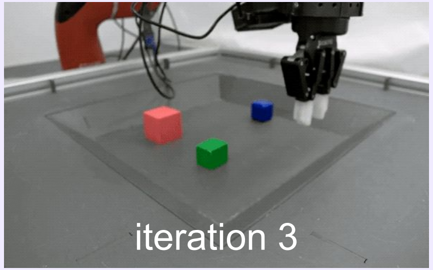
- Execute policy on the robot
- Record episodes to NeverEnding storage

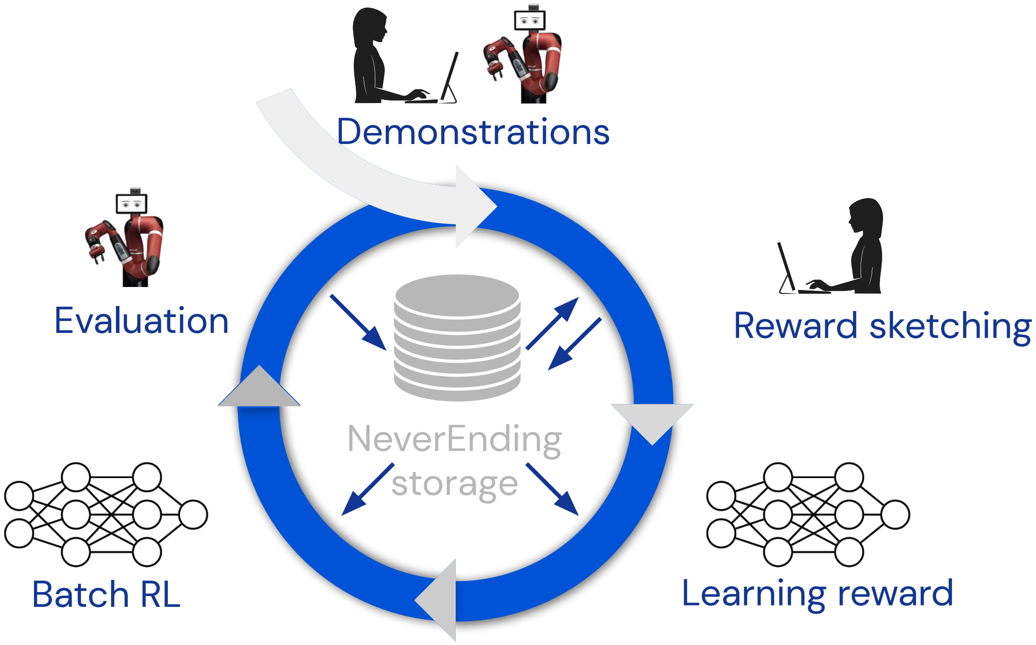


Iterative improvement

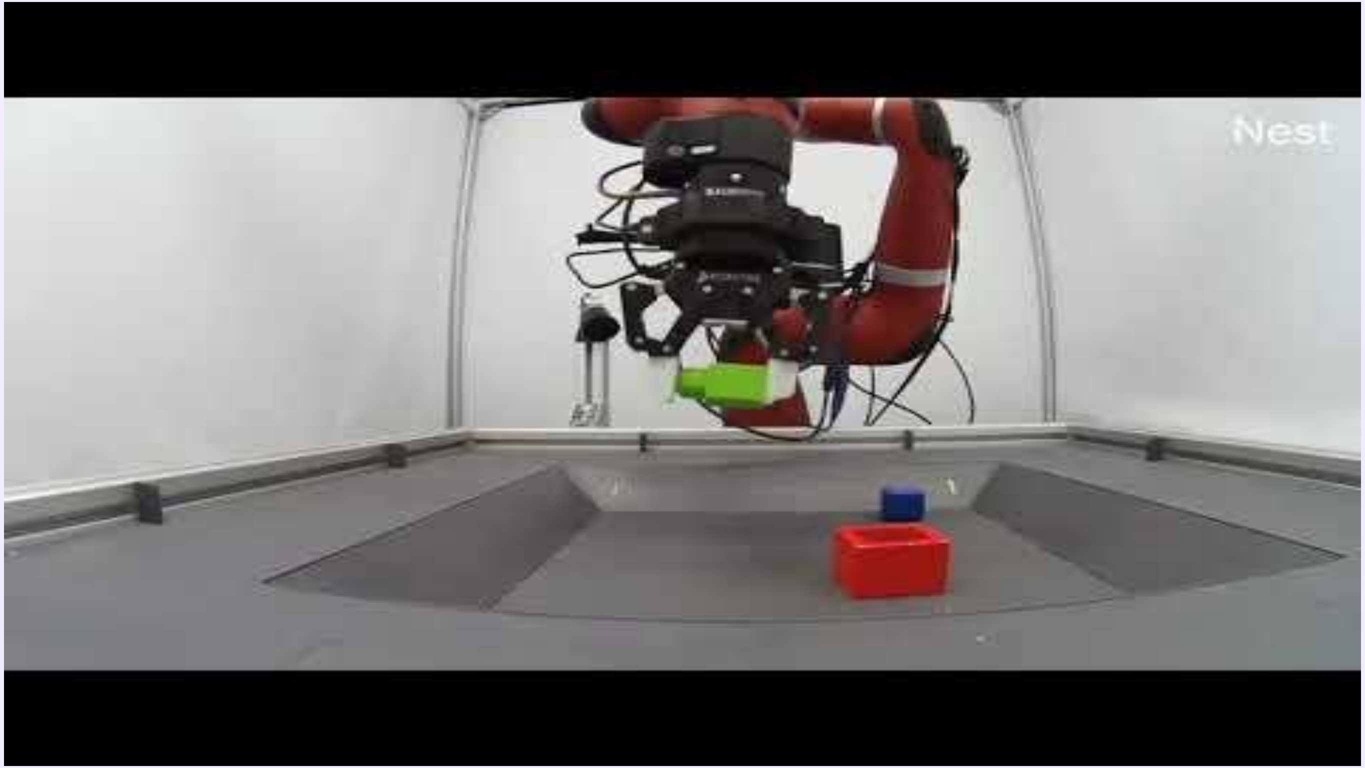








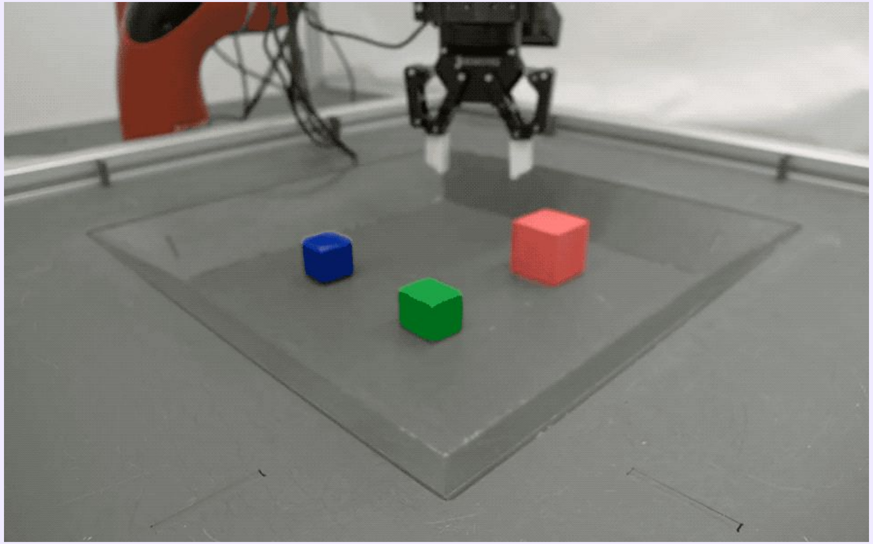
Adversarial robustness

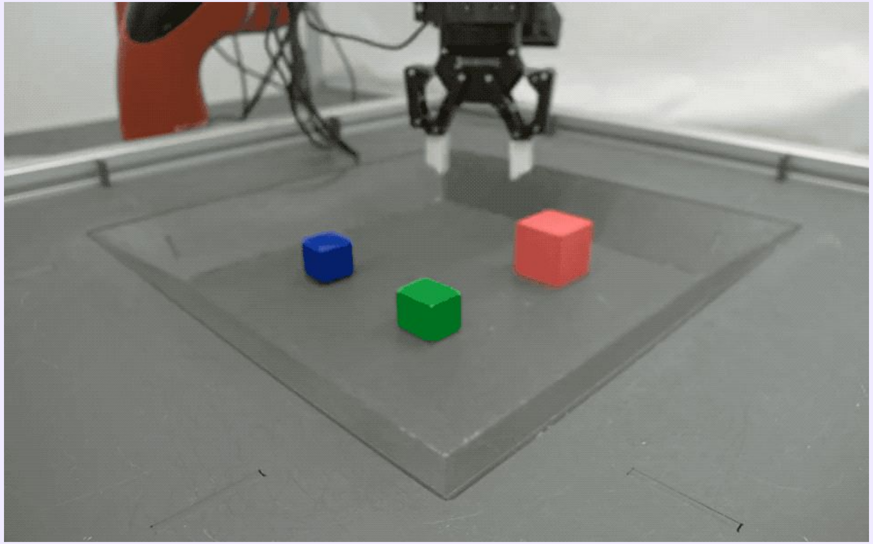


Better than human teleoperators

Human 

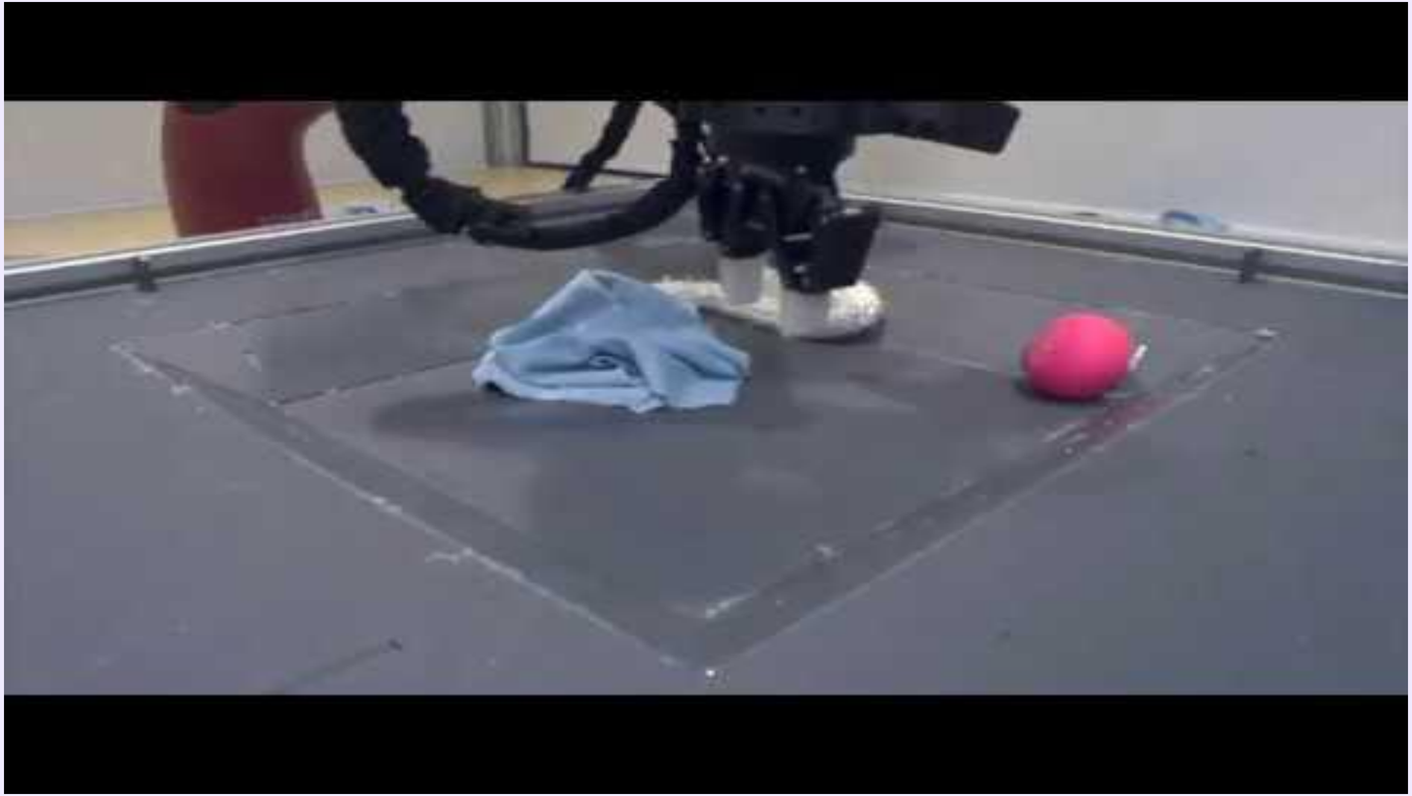
Agent 





3x speed

Handles non-scriptable objects

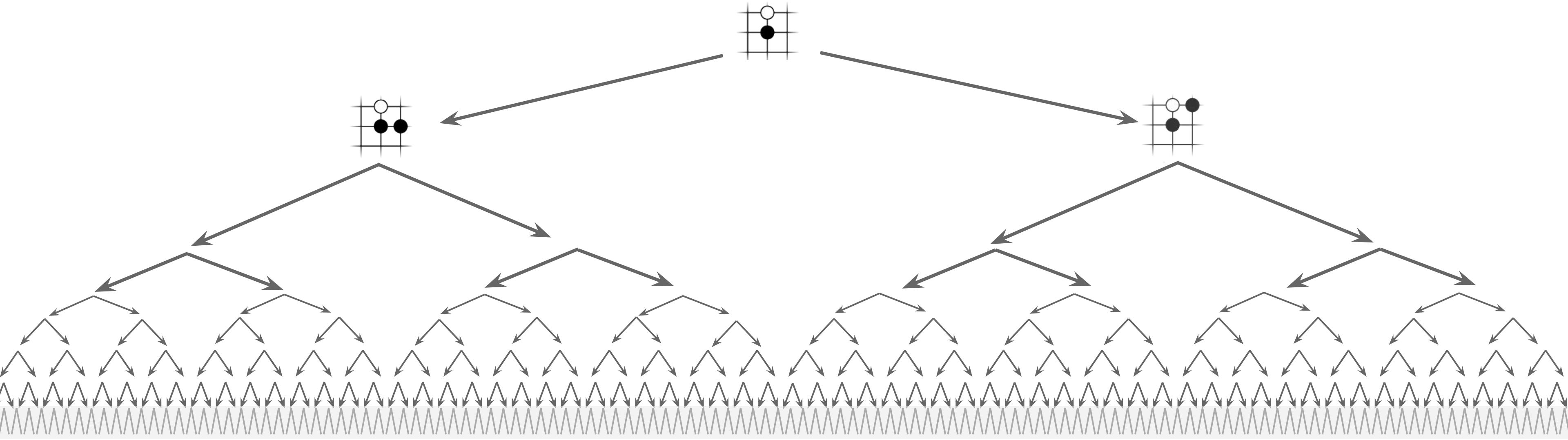


DeepMind

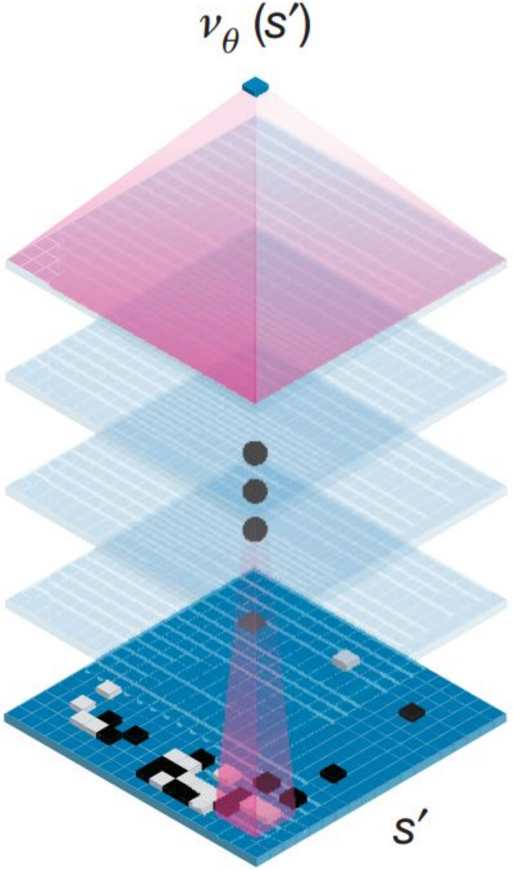
Example: From AlphaZero to GNNs

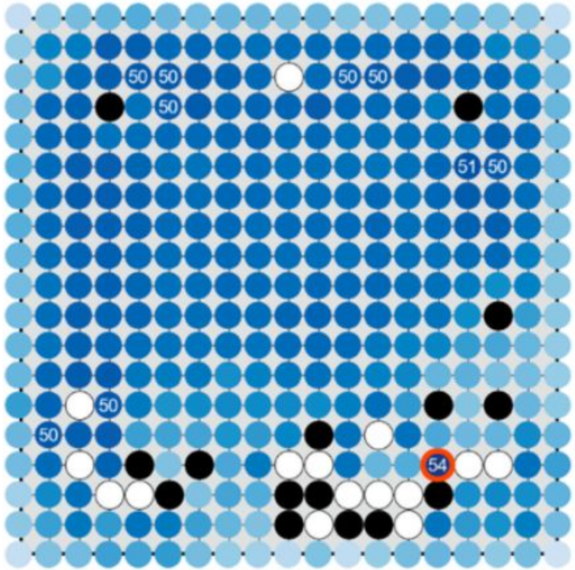

Google De
Mind
alle
Match
2016.3.15

Exhaustive search

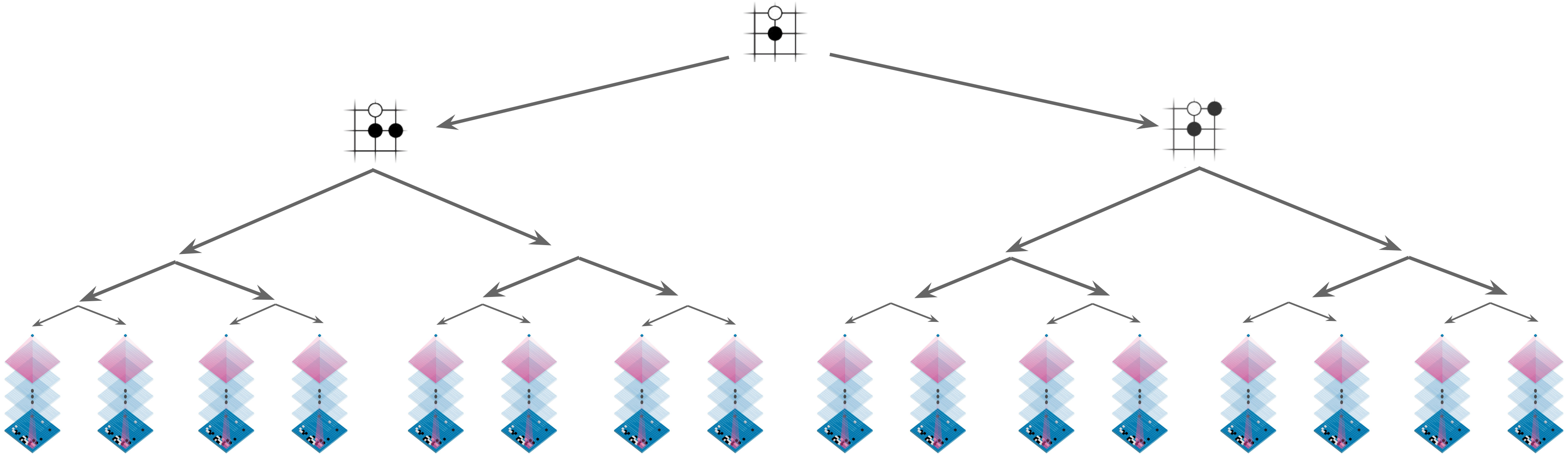


Value network

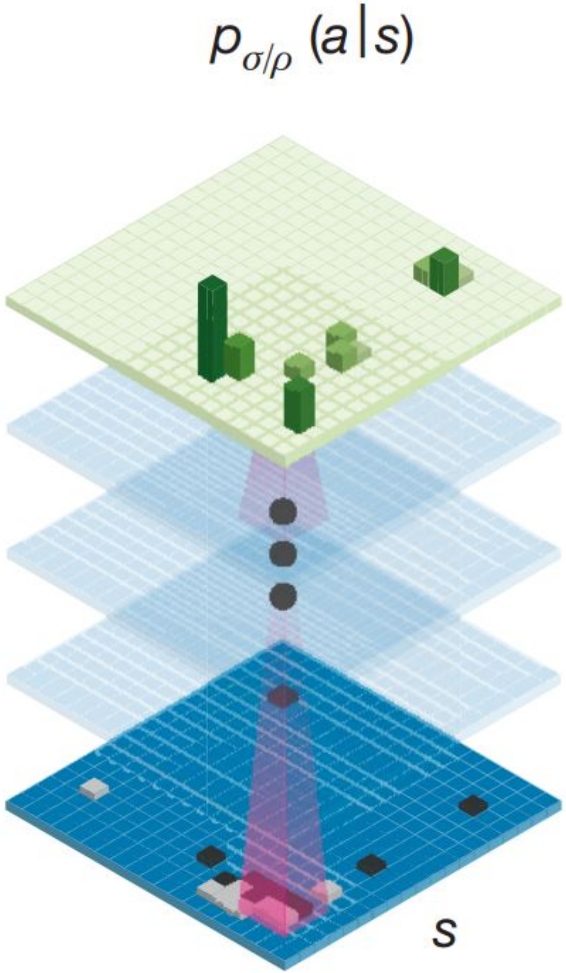


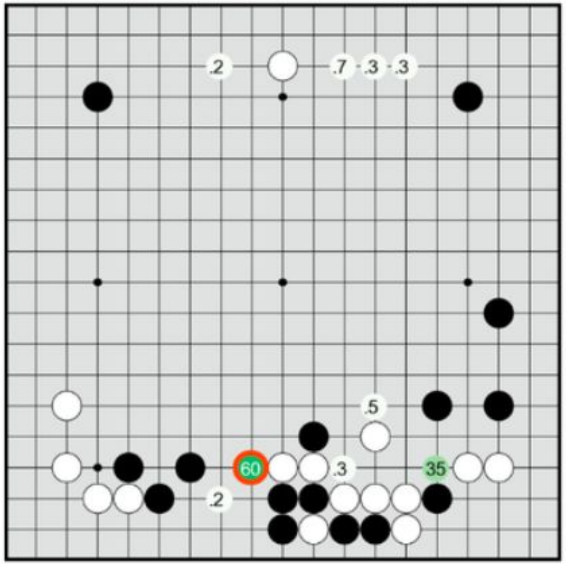


Reduce depth with value network

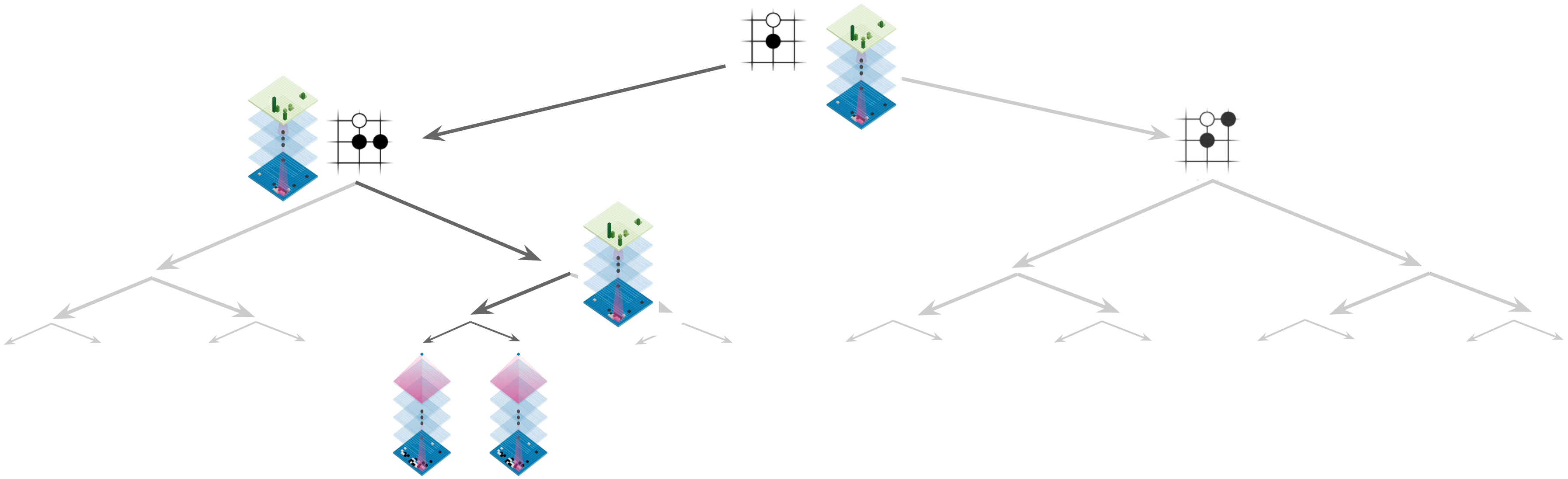


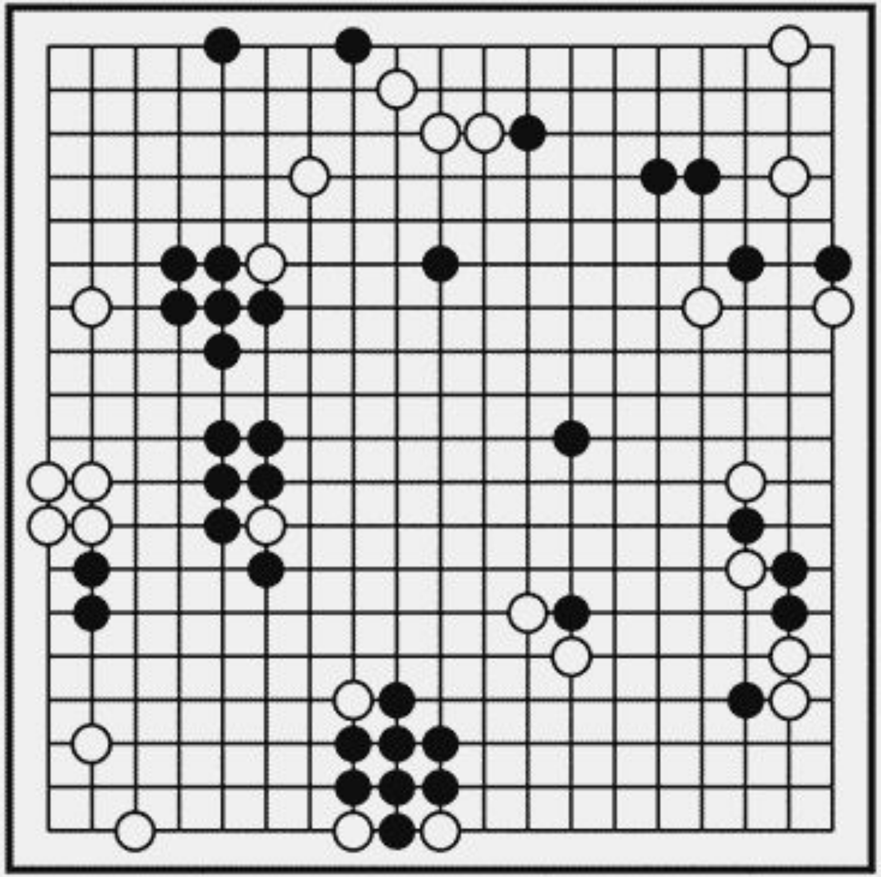
Policy network



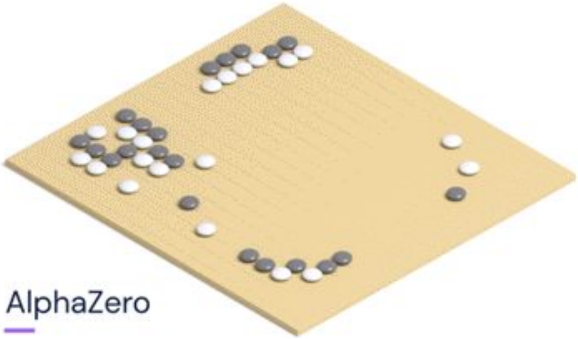


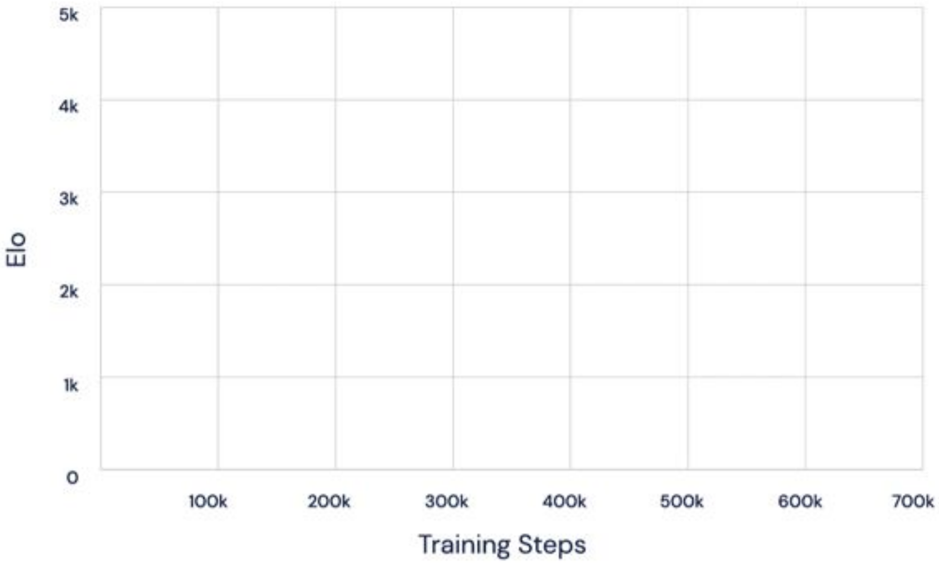
Reduce breadth with policy network





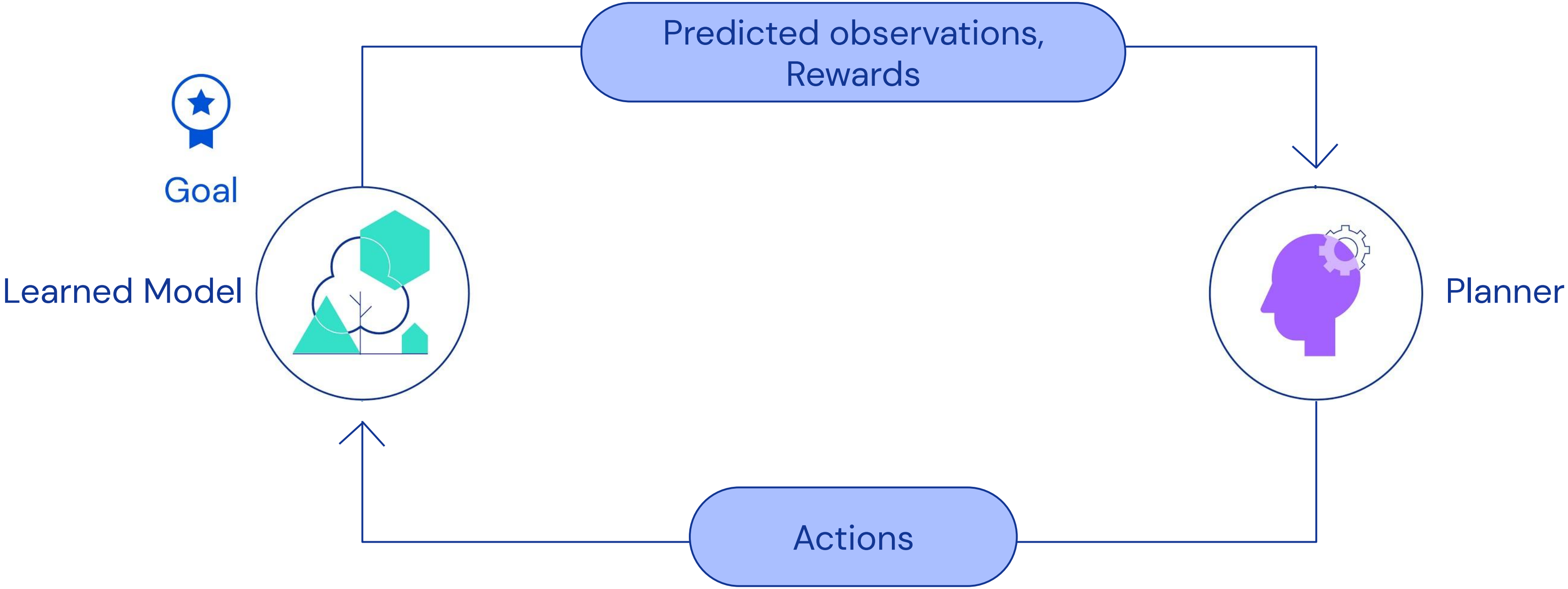
AlphaZero: Learning without Human Knowledge





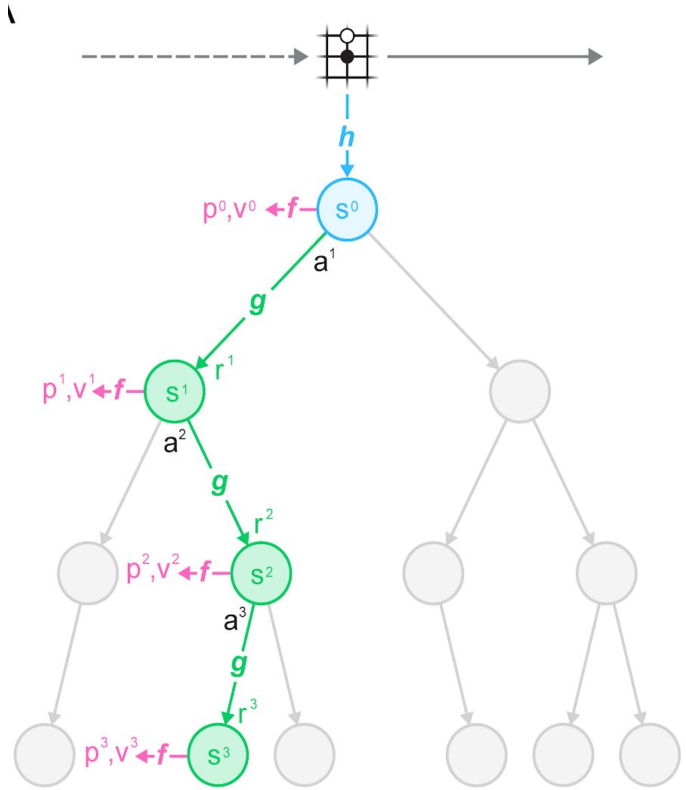
Model Based Reinforcement learning

Making good decisions by learning about the world from experience and through imagination

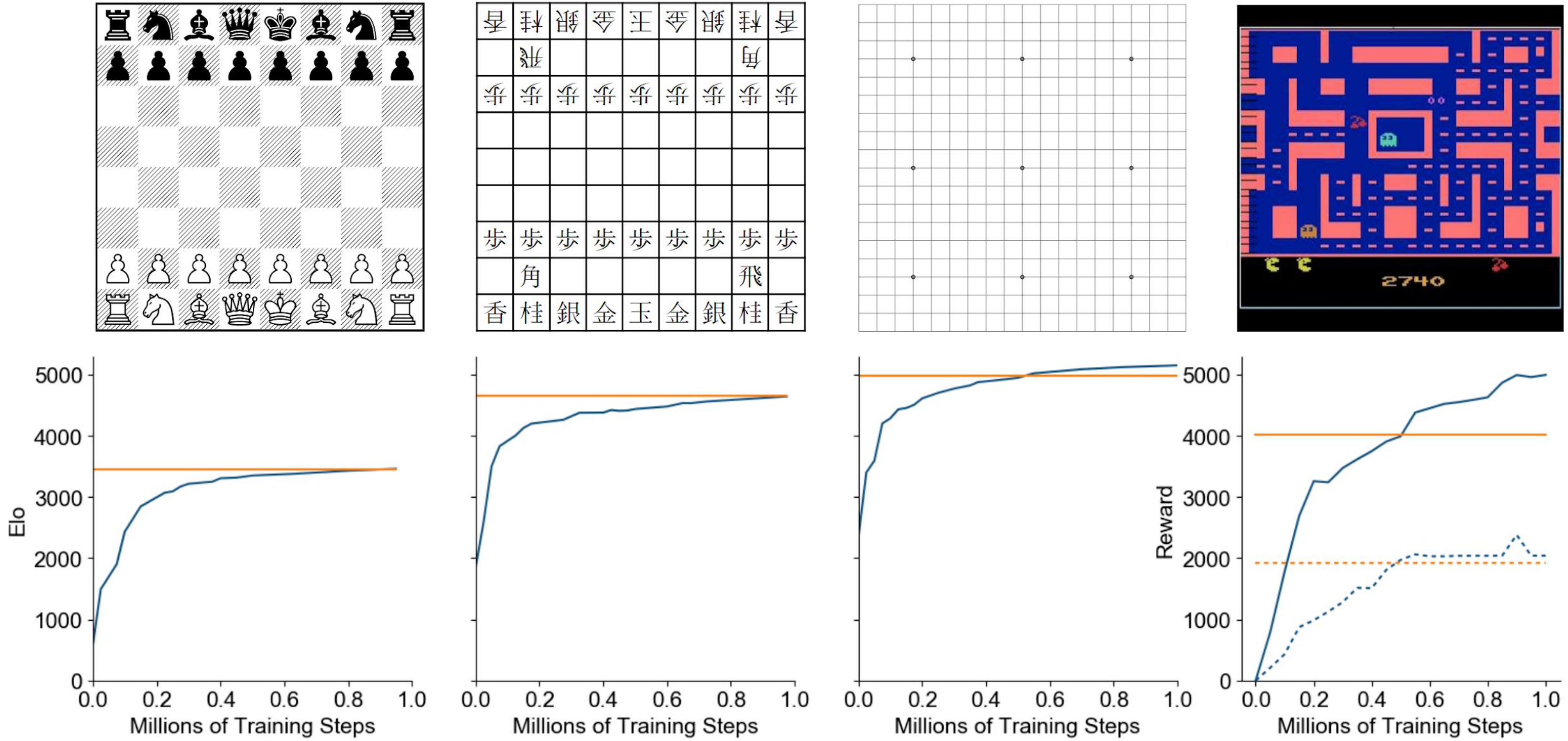


MuZero: Planning with a Learned Model

representation	$s^0 = h_\theta(o_1, \ldots, o_t)$
prediction	$\mathbf{p}^k, v^k = f_\theta(s^k)$
dynamics	$r^k, s^k = g_\theta(s^{k-1}, a^k)$



Chess **Shogi** **Go** **Atari**

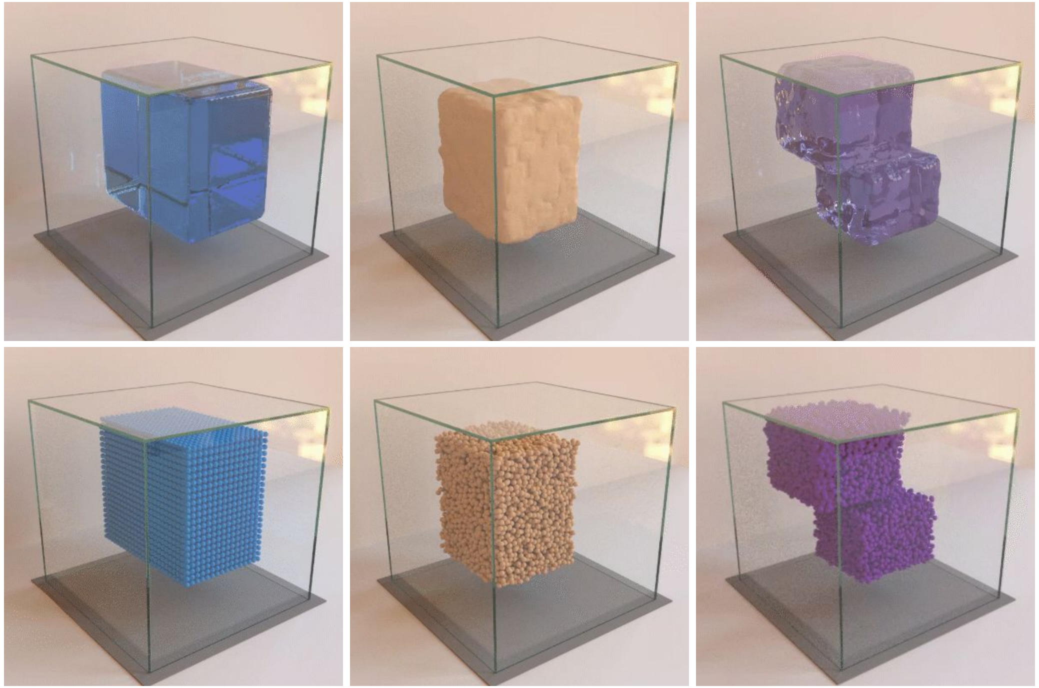


Learning Models of the World

Graph Neural Networks

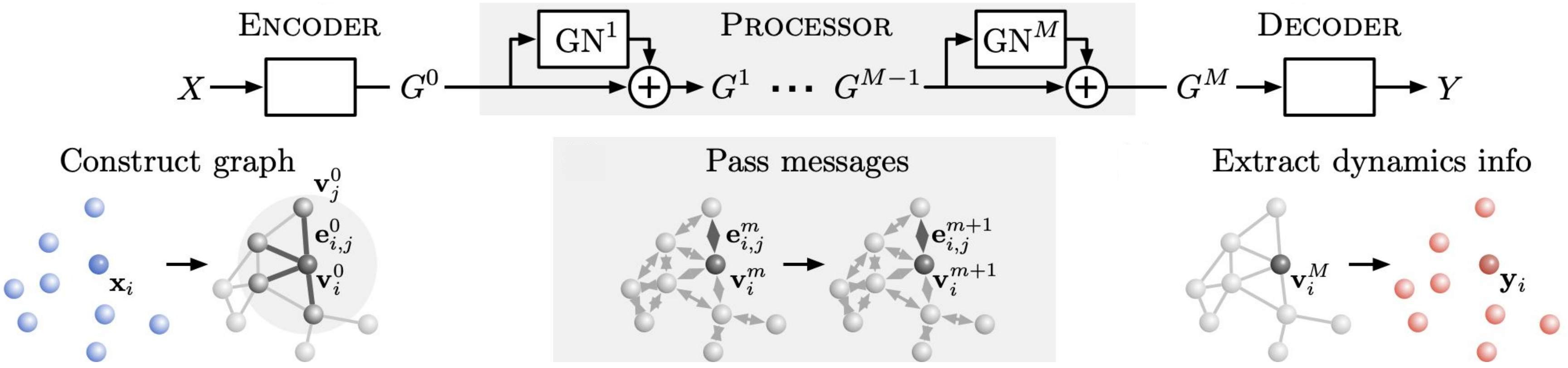
A general approach
to learning simulations.

Single GraphNets
architecture with single set
of hyperparameters



Learning to Simulate Complex Physics with Graph Networks
https://arxiv.org/abs/2002.09405

Model Framework



Encoder

- Input features:
  - Position
  - Previous 5 velocities
  - Particle type
- Embed features with MLPs
- Construct neighbourhood graph

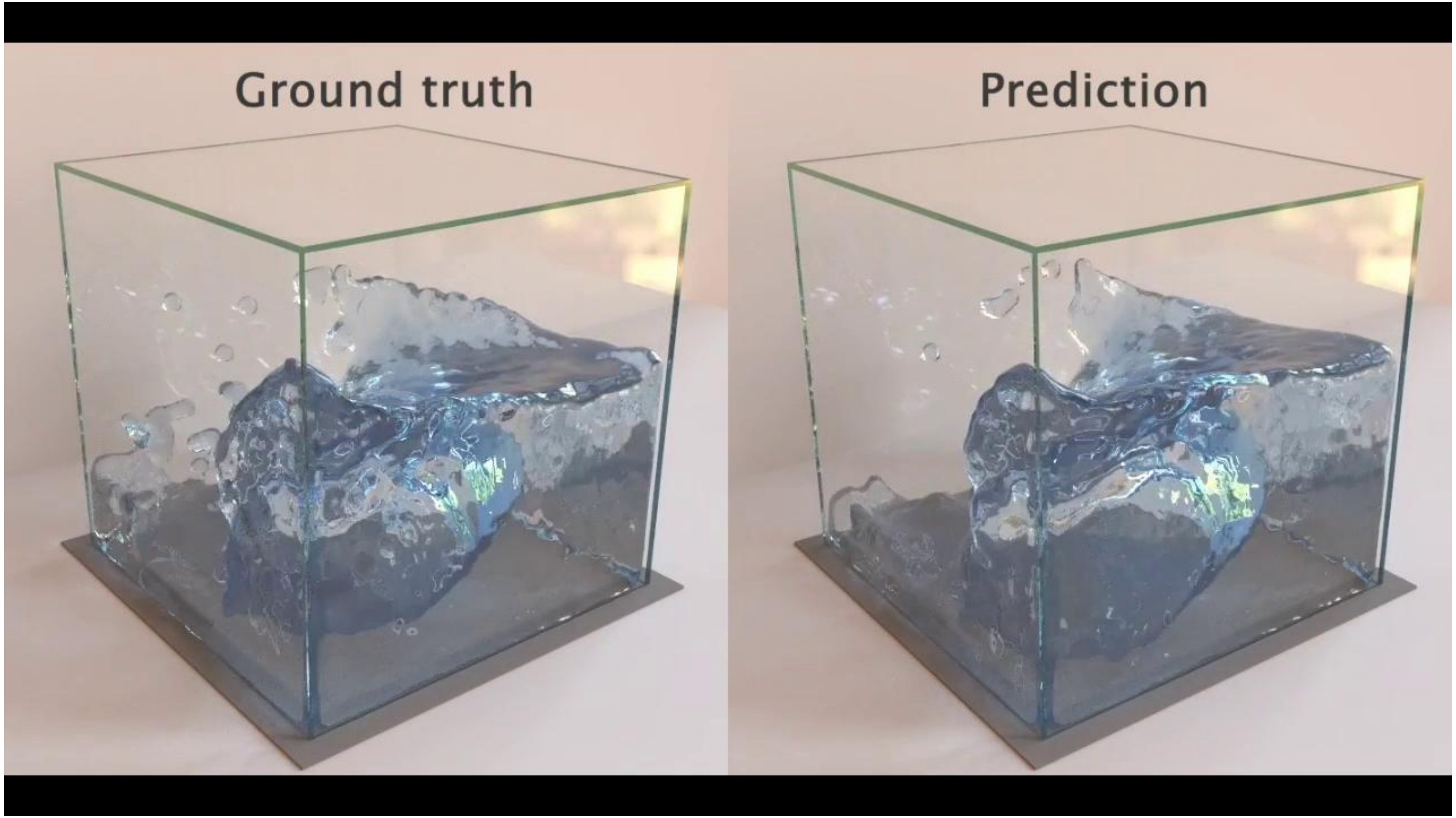
Processor

- Message-passing steps (many)
  - Edge function: MLP
  - Node function: MLP

Decoder

- Extract acceleration
- Feed into Euler integrator

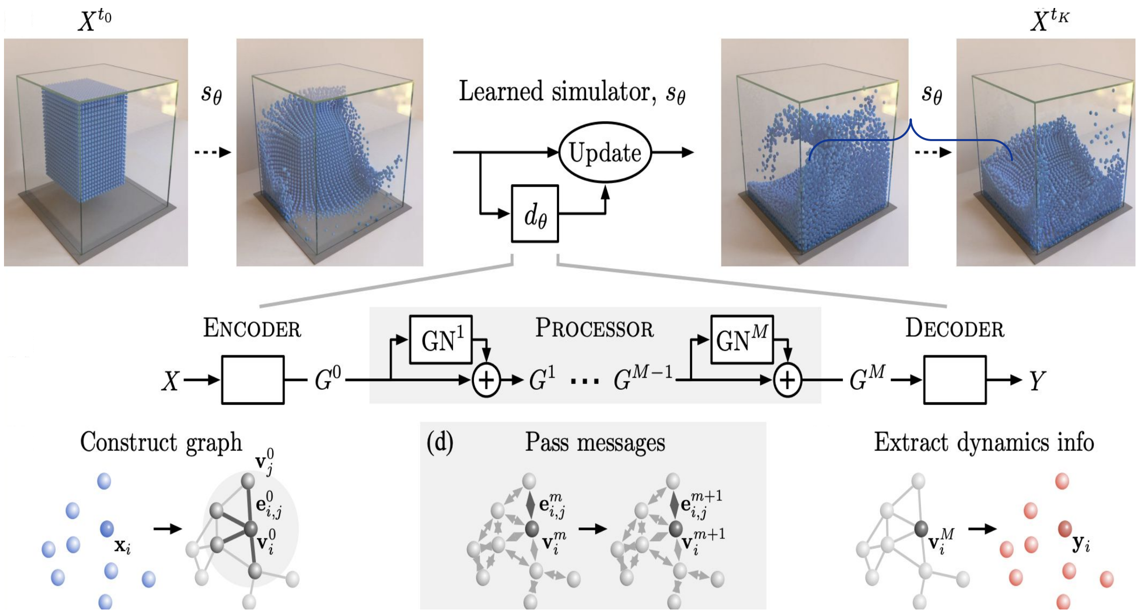
Results: Rollout over 1000 steps (initialized from timestep=0)

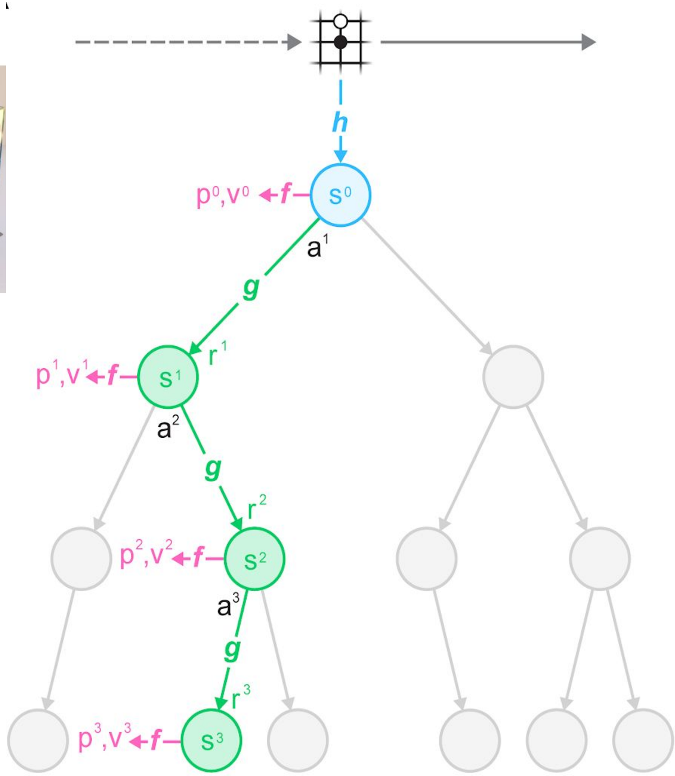


We need huge amounts of compute.

But why?

Algorithmic Scaling





The Bitter Lesson

"

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.

Rich Sutton

http://www.incompleteideas.net/IncIdeas/BitterLesson.html

Unreasonable effectiveness of computational scaling

- Larger networks generalize better
- Larger networks train faster
- More Search and Planning is better

Algorithms catch up (eventually) but the gap is large.

Algorithmic efficiency gains aren't enough

Demonstrating **44x less compute** required to achieve AlexNet*–level performance over 7 years.

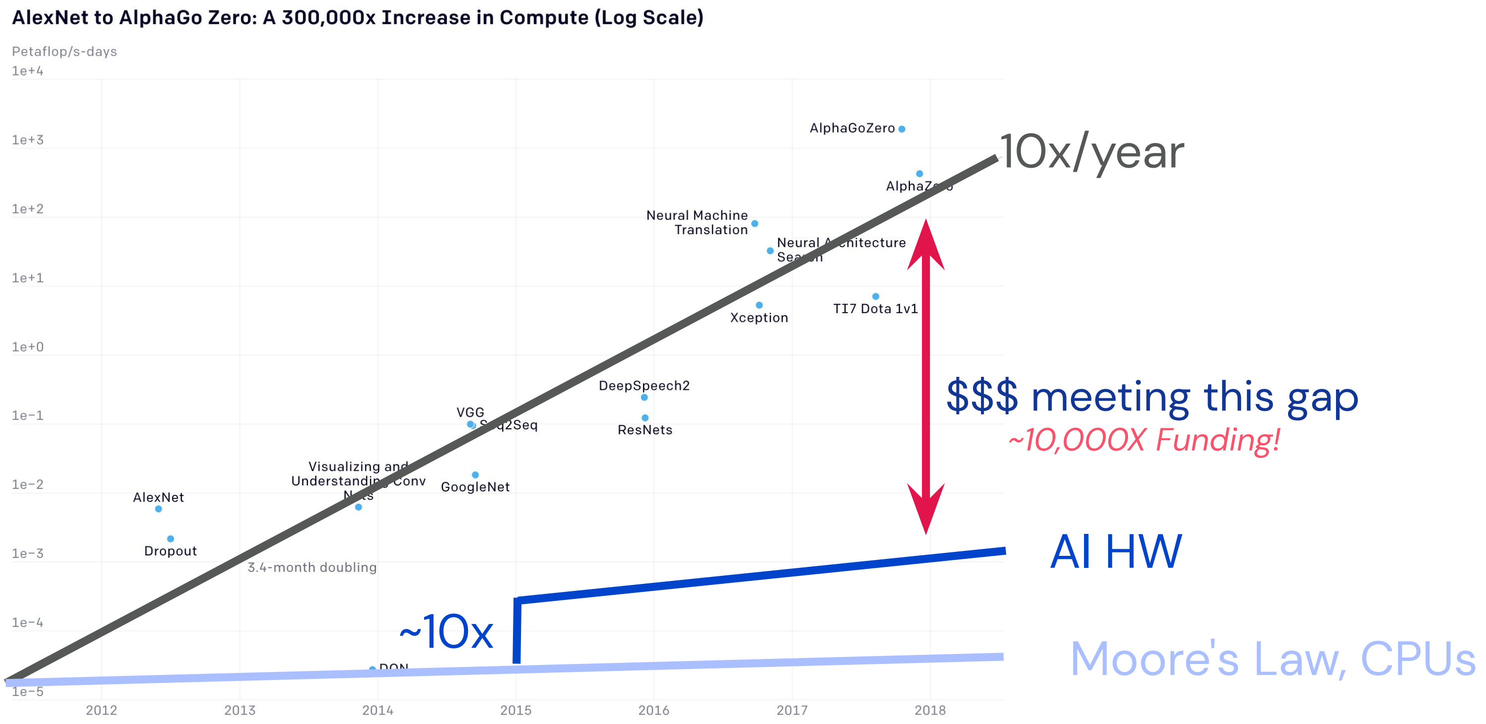
This is huge. But it's not enough

https://arxiv.org/abs/2005.04305

During the same period, compute demand increased >300,000x

https://openai.com/blog/ai-and-compute/

How have we met the demand gap?



https://openai.com/blog/ai-and-compute/

If growth holds then this isn't sustainable long-term

Absolute limit is far off

- >£1tn/yr in HW budget
- Build out will continue
- Sufficient economic incentive
- Great progress still with small models in production

Impacts may be seen soon

- Potential inaccessibility of certain research to certain groups
- Inability to progress certain types of research
- Lack of investment in understanding auxiliary impact

HW Enabler of AI Research

- Increasingly harder problems
- Research peeking into future of products
- Diversity of HW catalyzes new research ideas

DeepMind

Three Opportunities on The Road Ahead

1

Systems >> Chips

Research At Scale Is About People

- Hundreds of Researchers
- Thousands of Experiments
- Millions of Program/Meta parameters
- Infinite Backlog
- 99% Throwaway!

- Researchers
- Experiments
- Populations
- HyperParameters
- Controllers
- Distributed Systems / Data
- Math
- Compilers
- Runtimes / VMs
- Networks / Systems
- ML HW

Research Process as a System

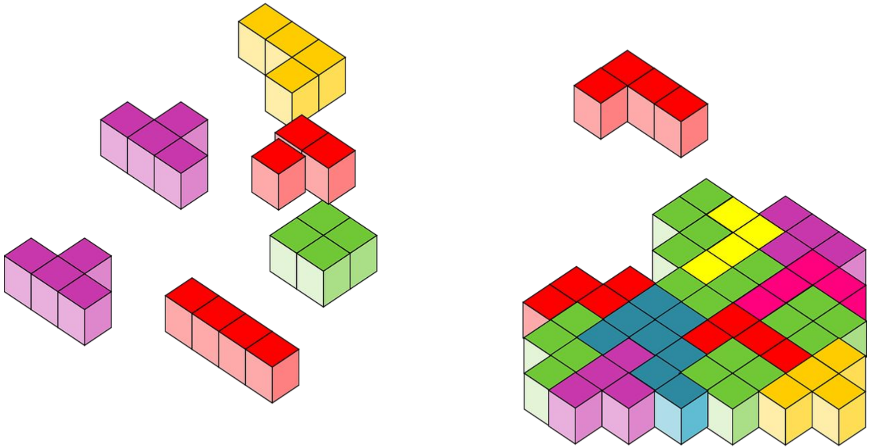
Optimize for Throughput in the whole system

Experiment Manager:

- complete view of past, present, future
- total control of scheduling and placement

Results:

- 2X improvement in overall Utilization



Sustained Performance / TCO

- Maximize Utility:

  performed work / (cost of equipment + energy)

  Most optimized research experiments: 70%

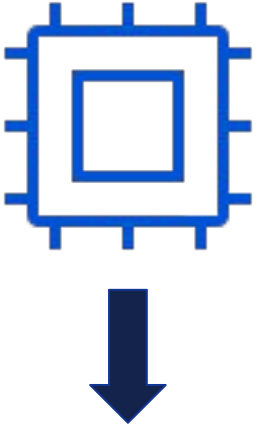
- Average research fleet wide peak FLOPs Utilization: **20%**
- How could this be?

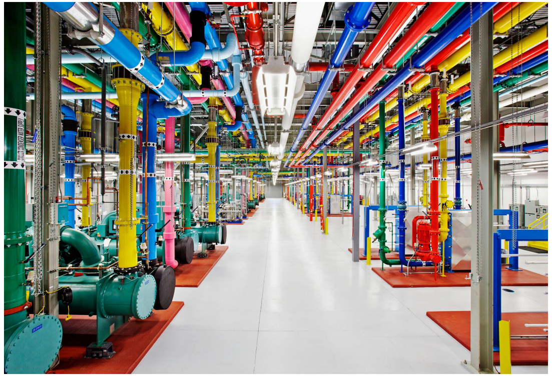
Sustained perf/$ is not Peak Device perf/$

- Single Device Performance/Cost a weak predictor of overall Perf/$
- Possible reasons for low Utilization:
  - Startup time
  - Memory bound
  - I/O starvation
  - Compiler issues
  - Contention
  - Runtime
  - Amdahl's Monster

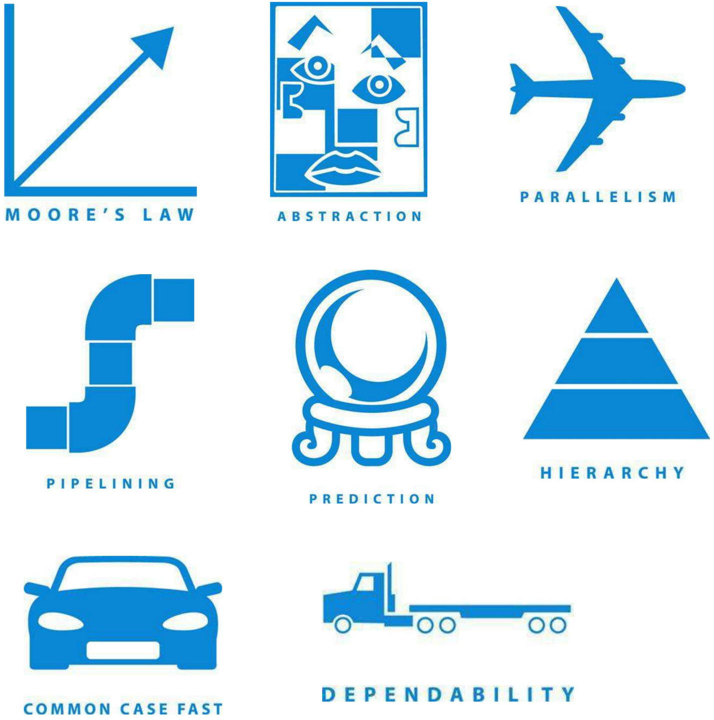
What to optimize for?

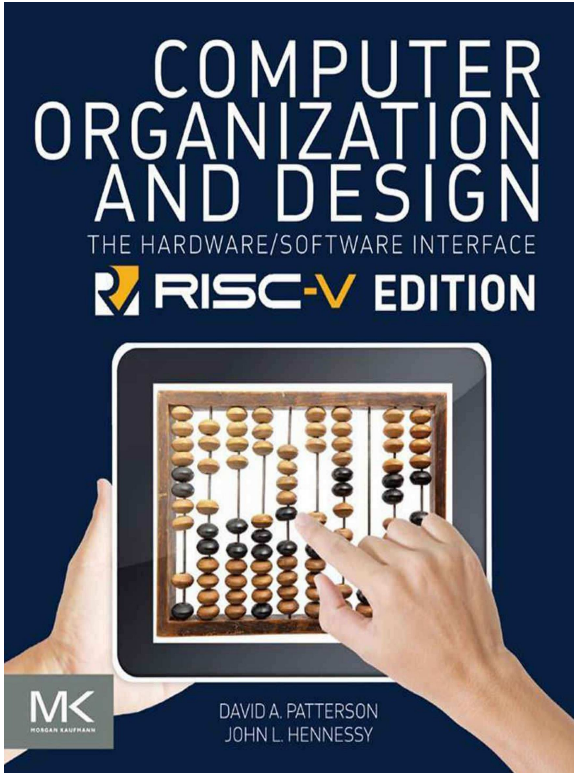
- Utilization is independent of Chip parameters
- Chip performance does not matter
- What matters is sustained system perf/$
- More opportunities to design top down from DataCenter scale





Which Great Ideas for Warehouse scale Computer?





2

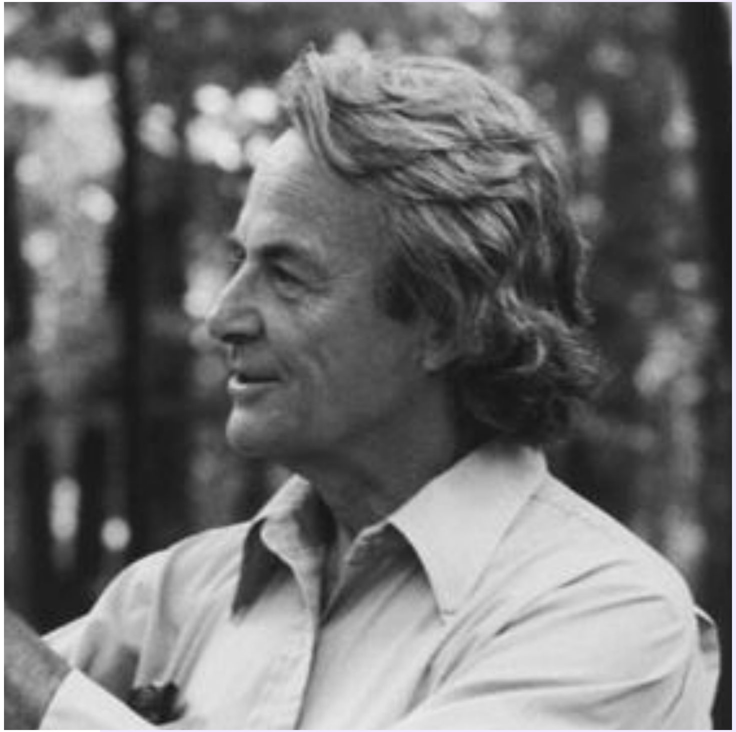
What to do when there isn't plenty of room at the bottom?

(Not so) Plenty of Room at the Bottom

"

I don't know how to do this on a small scale in a practical way, but I do know that computing machines are very large; the fill rooms. Why can't we make them very small, make them of little wires, little elements – and by little I mean little. For instance, the wires should be 10 or 100 atoms in diameter, and the circuits should be a few thousand angstroms across.

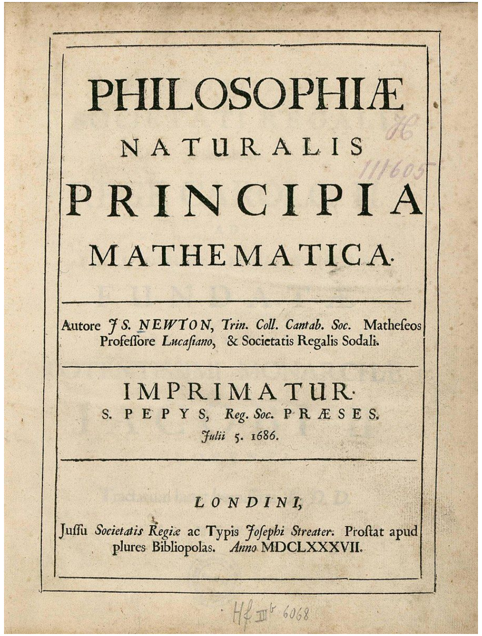
Richard Feynman, 1959



What does a warehouse scale computer look like when we've reached The Bottom?

What actually matters?

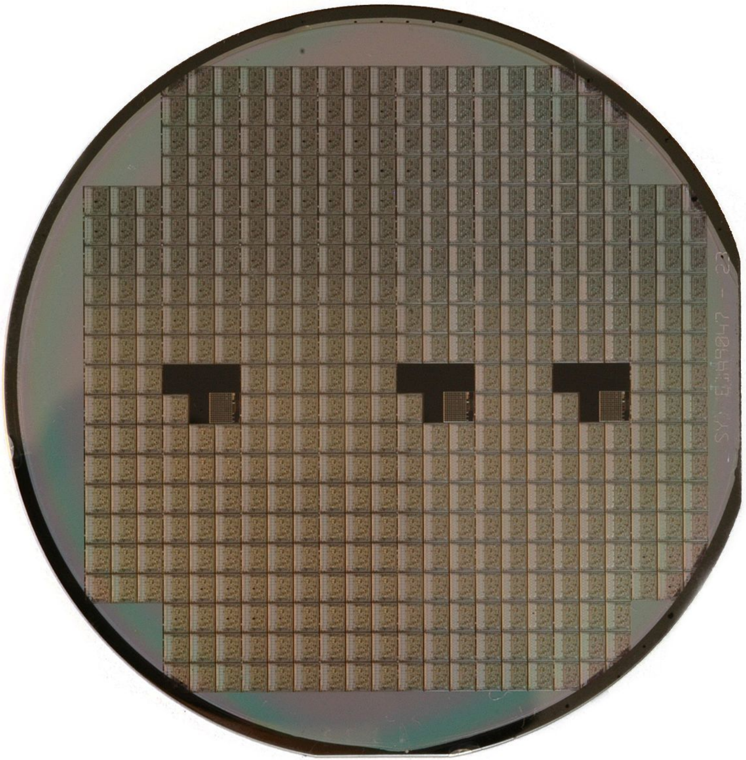
- From first principles:
  - Si
  - Power
  - Software



How far away are we from reaching the bottom?

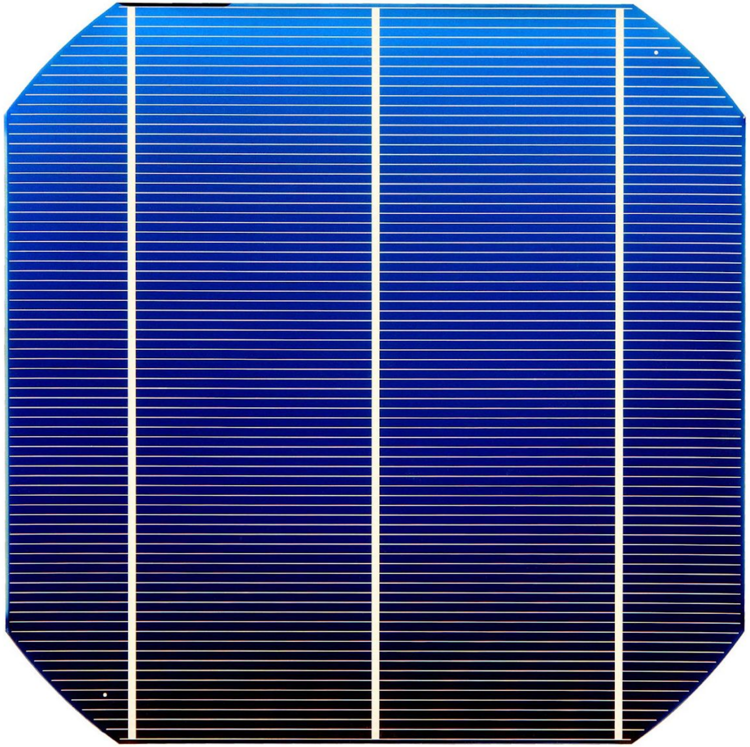
True cost of operations

- What if the WSC was built just out of pure Si?
  - 1000X improvement in perf/$



True cost of power

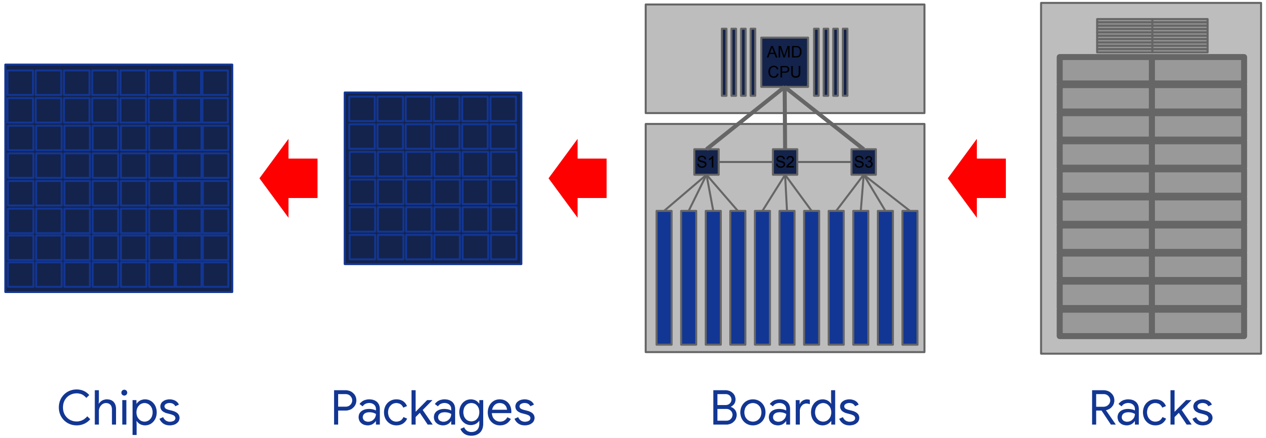
- Surprise: power is still Si (until fusion?)
- True cost of power:
    - 1000X more power/$ from pure Si + Sun



WhistlingBird https://commons.wikimedia.org/wiki/File:Silicon_solar_cell_(PERC)_front_and_back.jpg

50 year trends

- Components disappear into Si
- Make up volume with more computers



How do we bridge the 1000X gap?

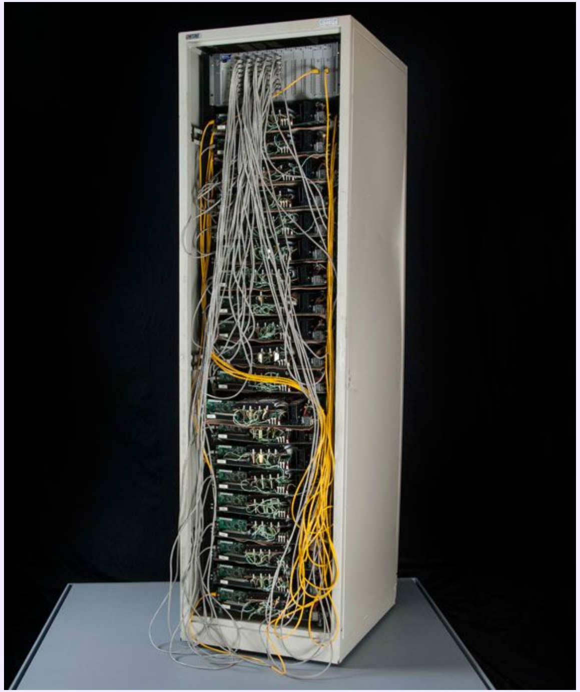
- Move everything into Si (even if it costs more)
- Only add components to the system that are required
- Exploit non-linearities in system properties
  - Number of distinct components
  - Rate of defects per area
  - 3D stacking
  - Massless cooling
  - Automation

Paradigm Shift

- Producing an electric car is not just replacing the combustion engine with an electric motor!
- New rules for system design?

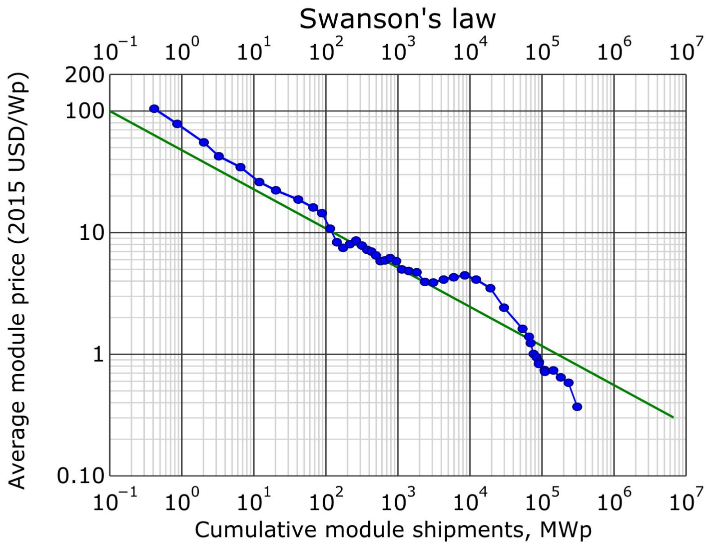
OLD	NEW
Big complex chips	Smaller simple chips
Bespoke, expensive	Few components, all in Si
Few	Huge quantities
Fancy, reliable	Cheap, less reliable
Control decisions in HW	Control decision in software

You think this is impossible?



https://sites.cs.ucsb.edu/~rich/class/cs293b-cloud/papers/Brewer_podc_keynote_2000.pdf

Simple designs are easy to scale up



Solar volume 1000X larger now than compute Silicon!

3

Where is the Software floor?

Back to the 90's

Cluster-Based Scalable Network Services

Armando Fox Steven D. Gribble Yatin Chawathe Eric A. Brewer Paul Gauthier
University of California at Berkeley Inktomi Corporation

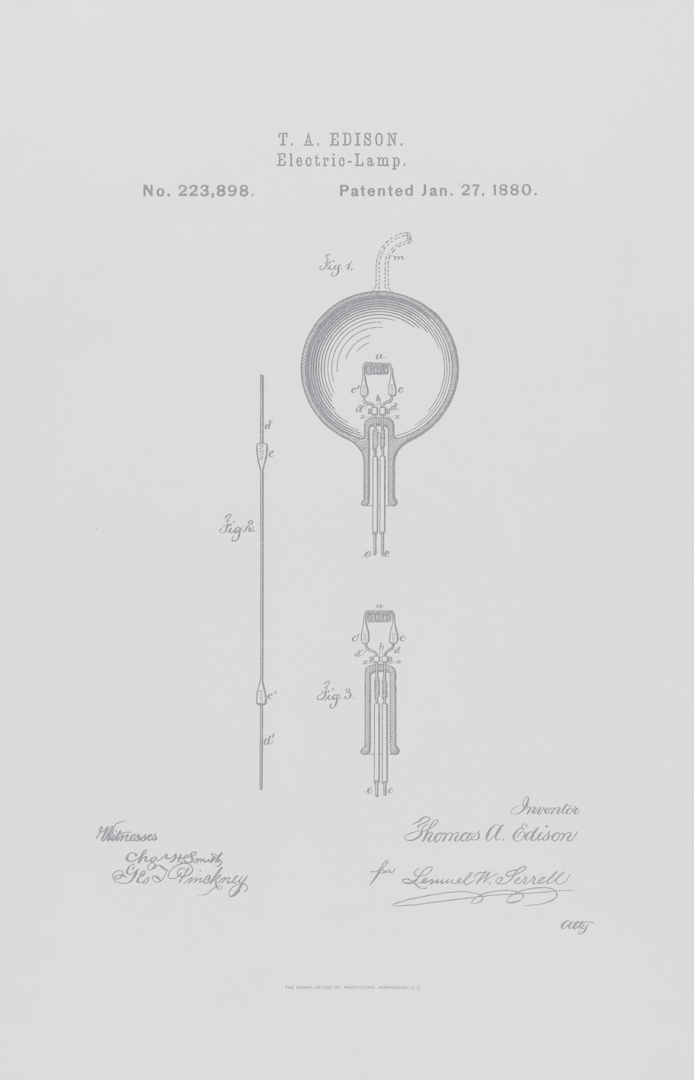
- By 2000 we learned how to run disaggregated services over thousands of machines.
- Machine Learning systems also run over thousands of Devices.
- Our software infrastructure stack had to increase in complexity to accommodate.

- Researchers
- Experiments
- Populations
- HyperParameters
- Controllers
- Distributed Systems / Data
- Math
- Compilers
- Runtimes / VMs
- Networks / Systems
- ML HW

Where do performance/$ gains come from?

- Big gains come from Paradigm Shifts
    - CPU -> GPU -> TPU
- Process Node improvements
  are a wave we all ride on.
- **Put other way:**

  Major performance improvements
  can only come at the expense
  of creating significantly different systems.



Where do performance/$ gains come from?

- Gains from Paradigm Shifts are eroded due to complexity and inflexibility of the infrastructructure stack.

Is there a solution?

What does diversity of HW matter?

- Diversity of HW gives rise to different research directions and patterns of thought but is too expensive to sustain due to high per-platform software overheads.

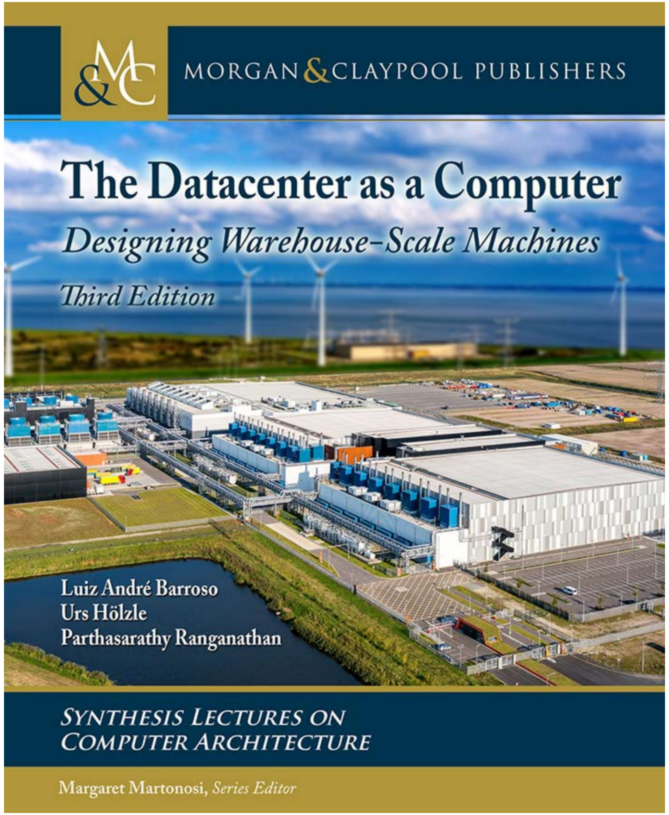
Is there a solution?

This time it's different?

- Properties of AI Programs:
  - Highly regular
  - Constrained by dataflow
  - Constrained by arithmetic
  - Low amount of data dependent control flow
  - >µs timescales for basic operations
- Coarseness & Timescales allow for distributed programs with software control.

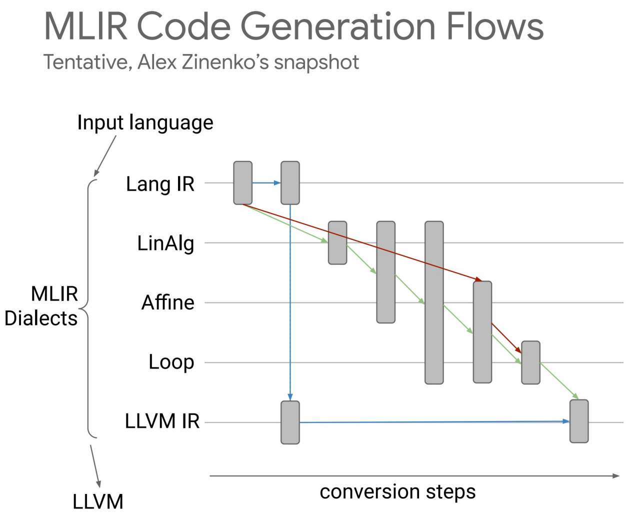
This time it's different?

- Warehouse scale programs on Warehouse scale computers?
- What does this mean for compilers and runtimes?
- Opportunity to redo 70s, 80s and 90s research in systems and PL?



This time it is different.

- Optimization during compilation is about **choices.**
- We are very good at making choices now:
  - Large scale RL
  - Generative models
- **What will this buy us?**



This time it is different.

- Radical slimming down of code base:
  - Multi-Purpose solutions
  - List of choices
  - Engine to decide which ones to make
- Greater adaptability. Easier to take advantage of paradigm shifts.
- Leading to greater research velocity.

Deep Learning for compilers … the time has come?

2010: Speech recognition

AUDIO → Deep Net → TEXT

2012: Computer vision

PIXELS → Deep Net → LABELS

2014: Machine translation

TEXT → Deep Net → TEXT

2020: Compilers?

IR + TRACE → Deep RL Agent → ISA

slide from V. Vanhoucke

DeepMind

Conclusions

This time it could *really be* different!

New epoch in computing is bringing in new opportunities.

1

AI Research is a very exciting area for systems design

2

It is time start focusing on whole systems

3

It is time to exploit economies of scale for HW

4

It is time to embrace advances in ML for the design of software infrastructure