S9865 GTC 2019
Solving Logistics Problems with Deep RL

Karim Beguir & Alexandre Laterre

DEEP RL IN ACTION: ROUTE OPTIMIZATION IN NYC

OBJECTIVE:

HIGH PERFORMANCE ROUTING ENGINE

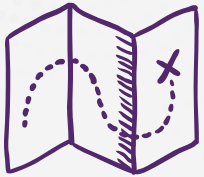
REAL TIME<30 SECS

DATA:

NYC

100s TAXIs

1000s REQUESTS

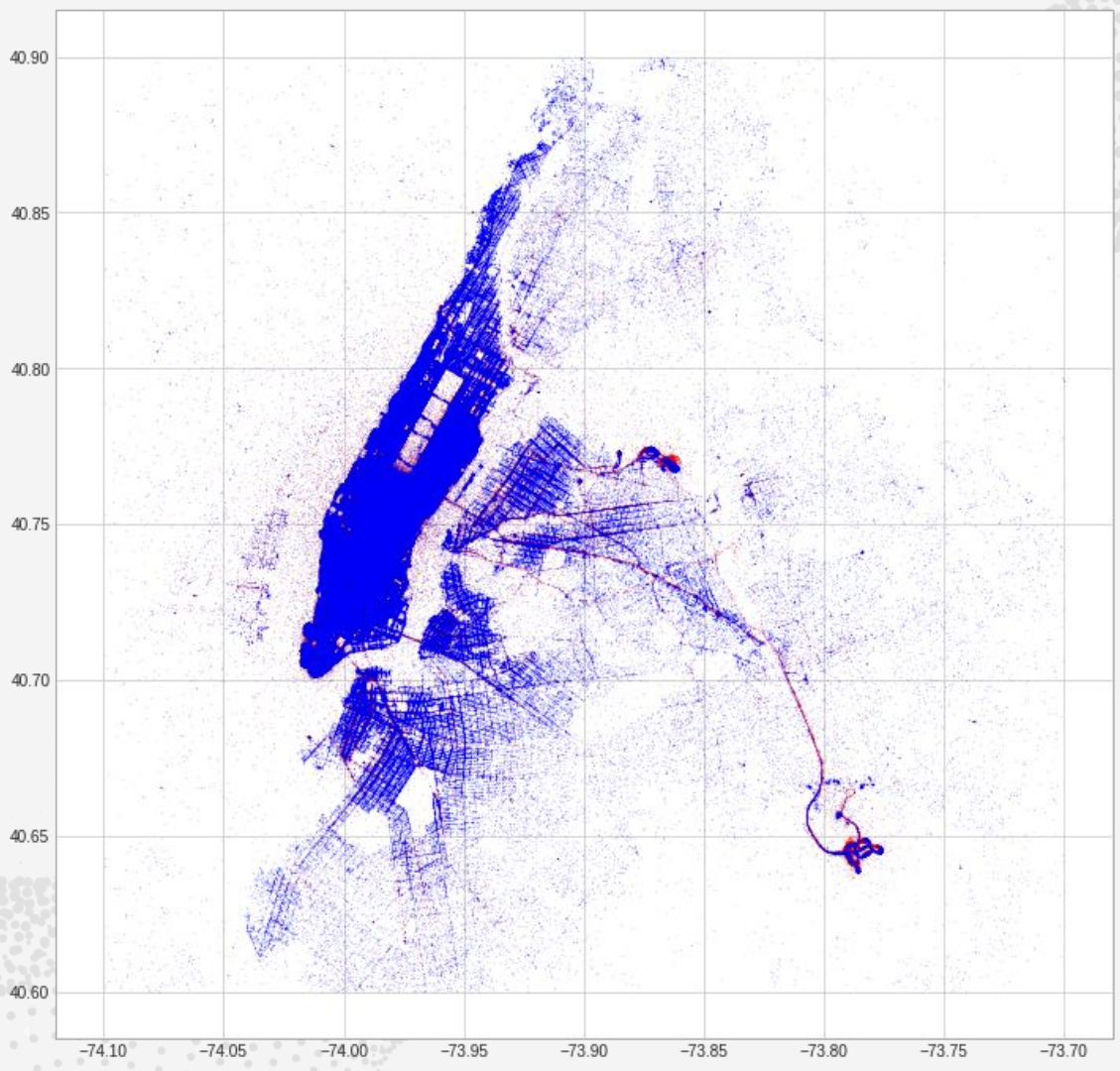


OPTIMIZE:

FLEET USAGE

& DEMAND PREDICTION


Speed
LIVE DEMO
Sunday, 01 March 2015 00:00:00



ROUTING:

< 10

SECONDS

EFFICIENCY:

> 3%

IMPROVEMENT

InstaDeep™

WHY IS THIS IMPORTANT?
DEEP RL VS. CLASSIC APPROACHES

Optimization Solvers

- ✔ Guaranteed optimality
- ✖ Poor scalability for large instances

Heuristics

- ✔ Fast and scalable
- ✖ Tailor-made for a specific problem
- ✖ No guarantee of optimality

Monte-Carlo Search Algorithms

- ✔ Fast and scalable
- ✔ No specific heuristic or evaluation function
- ✖ No guarantee of short-term optimality

Common Flaw: No Learning!

- **Do not** leverage past computations.
- **Unable** to generalize to **unseen** instances
- Might have to be **redesigned** if constraints **change**

UNDER THE AI HOOD:

“AI = Reinforcement Learning + Deep Learning” = Deep RL

David Silver (AlphaZero’s inventor) Google DeepMind

UNDER THE AI HOOD

MIT Technology Review

Reinforcement Learning

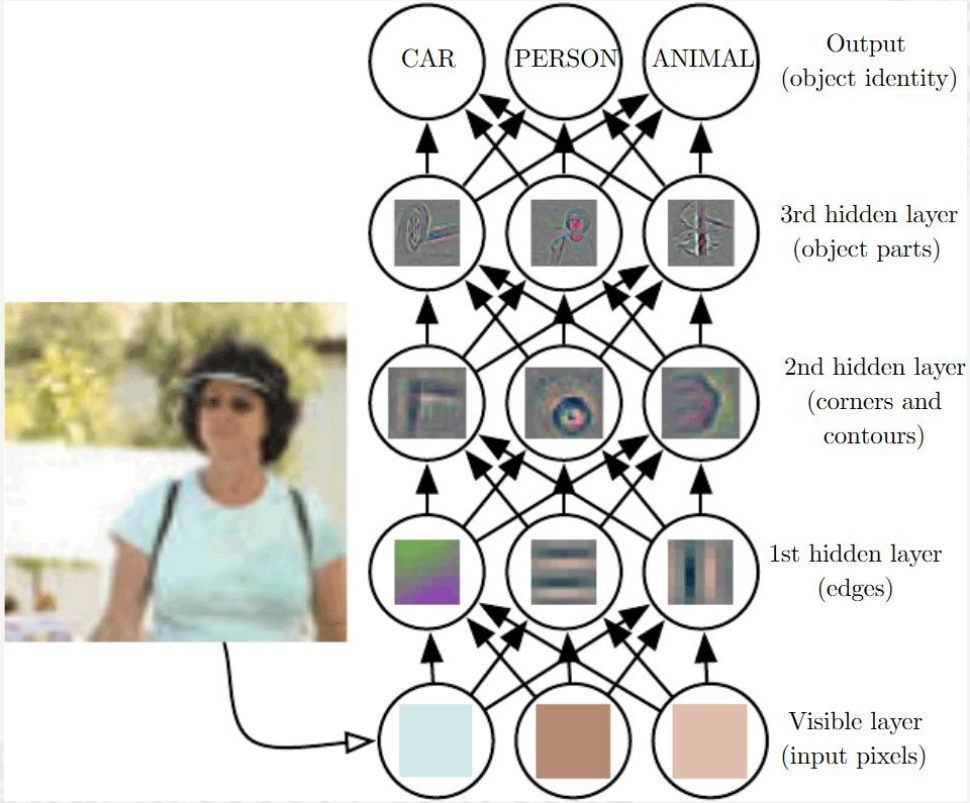
By experimenting, computers are figuring out how to do things that no programmer could teach them.

Deep Learning

With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart.

(MIT Tech Review)

Both in MIT TR 10 Breakthrough Technologies (2017 and 2013)

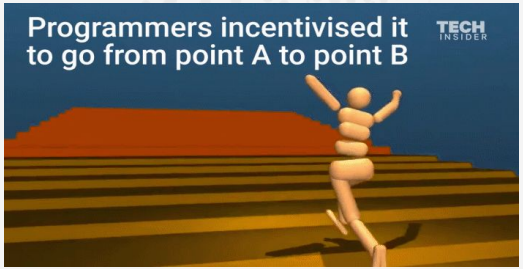


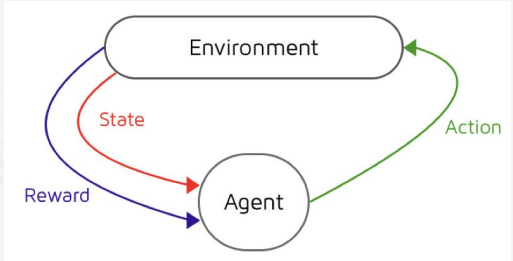
DEEP LEARNING : PATTERN RECOGNITION

Neural Nets (NN) “**think fast**”

- **Quick** (straight function call)
- **Adaptive** (universality)
- **Intuitive** (generalization)

REINFORCEMENT LEARNING (RL)



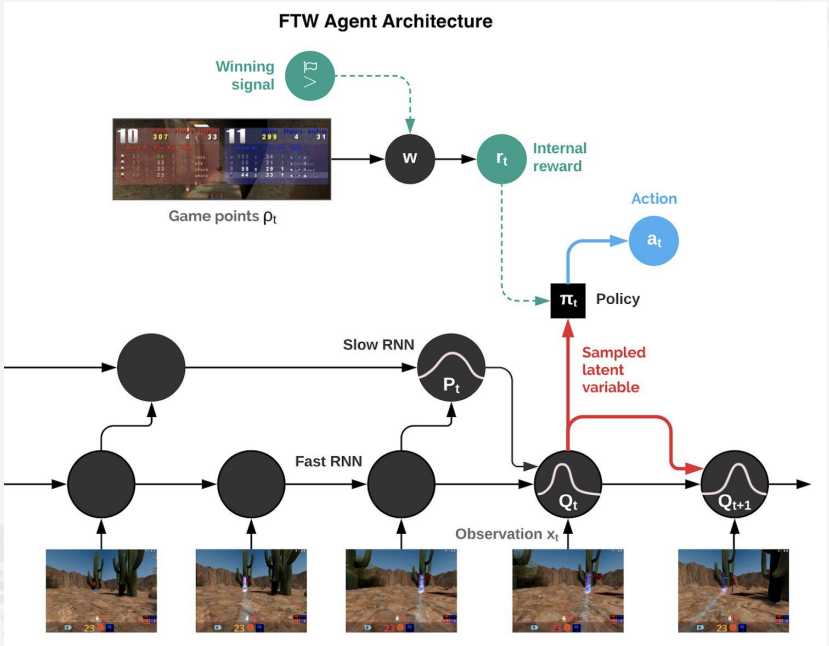


RL is the science of **trial and error**: learn from experience what works (positive reward) and what doesn't (negative reward)

LEARNING END TO END

DeepMind CTF: learn from **pixels** & **sparse rewards**

Deep RL for navigation: learn from **pixels** only (Mar 2019)



LEARNING AT SCALE

OpenAI Five: ~**1,000** actions to choose from (Chess ~35, Go ~250)

State space: **20,000** Dimensions (Chess **70**D, Go **400**D)

	OPENAI 1V1 BOT	OPENAI FIVE
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)

Compute budget: **190 petaflops/s-days** to reach 99.95% playing rank

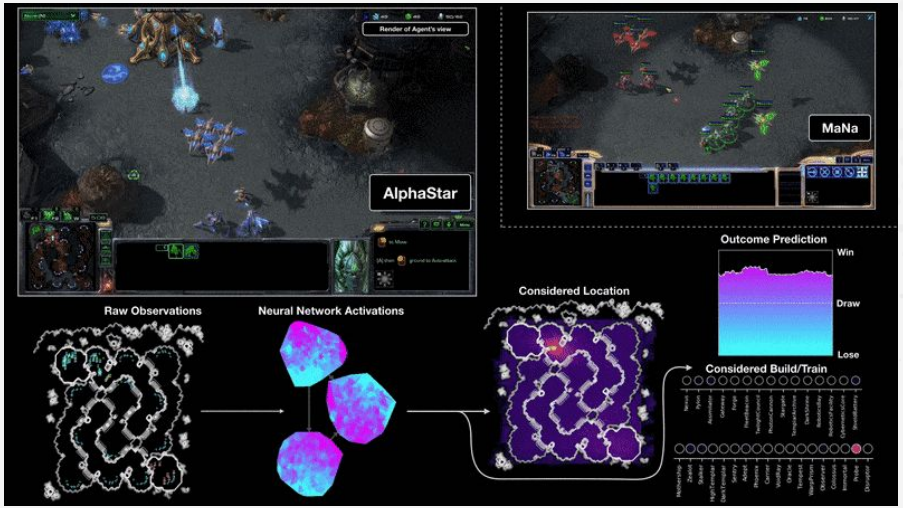
EMBRACING COMPLEXITY

AlphaStar: latest breakthrough in an exponential AI compute world.

Combinatorial Explosion: 10^1685 solutions vs. 10^170 for the game of Go

Game	Board Size	State-Space Complexity	Year defeated
Tic Tac Toe	9	10^{3}	1952*
Connect 4	42	10^{13}	1995*
Backgammon	28	10^{20}	1979
Chess	64	10^{47}	1997
Go (19x19)	361	10^{170}	2015
Heads up NL Holdem	N/A	10^{180}	2017
StarCraft II	N/A	10^{1685}	???

@LIV_BOEREE



Compute budget: **multi-million USD**

Case Study:
Deep RL for Bin Packing

InstaDeep™

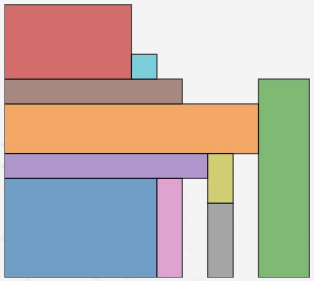
The Bin Packing Problem

Packing a set of items into fixed-size bins while minimizing a cost function e.g. number of bins required, surface of the packing

Constraints include:

- Prevent items from overlapping
- Gravity and the need for a physical support
- Limited rotations
- Packing stability and weight distribution

Formulation as a Markov Decision Process



STATE

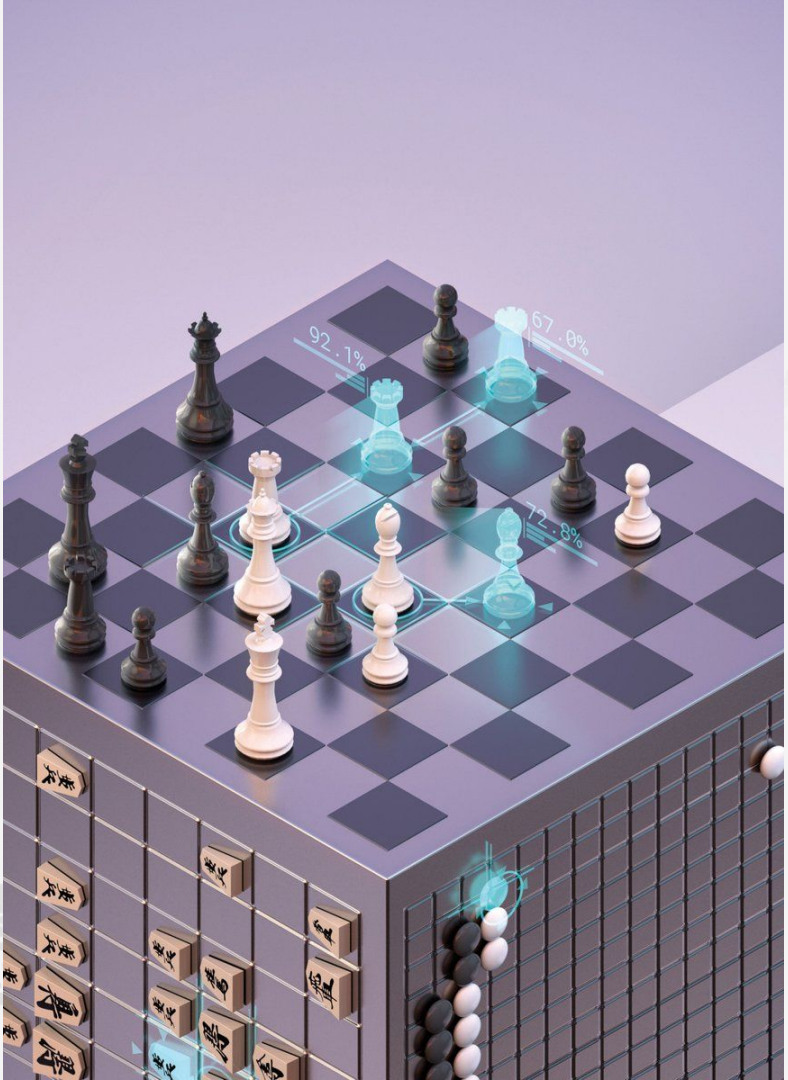
- Placement of the items in the bin
- the shape of the remaining items

ACTION

- Item to place
- Coordinate (x,y,z)
- Rotation to perform

REWARD

A function of the density and stability of the packing



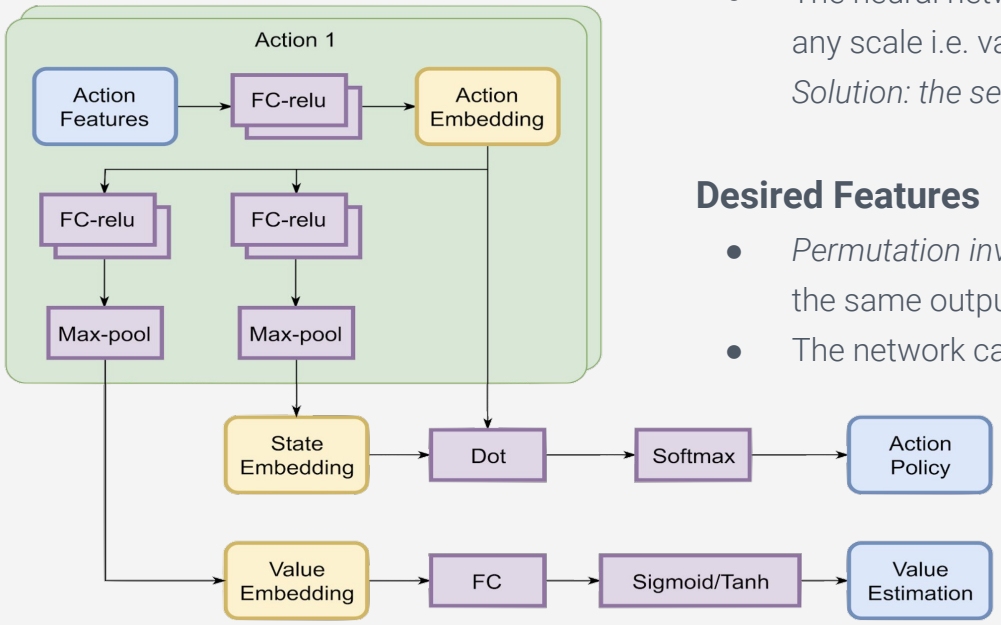
How To Create Your AlphaZero-like Algorithm

1. **Policy-Value Neural Network**
   Provide the learned intelligence of the algorithm

2. **Planning Algorithm**
   Play the role of powerful policy improvement operator

3. **Adversarial learning Process**
   A self-play mechanism to incentivize learning

Neural Network Architecture



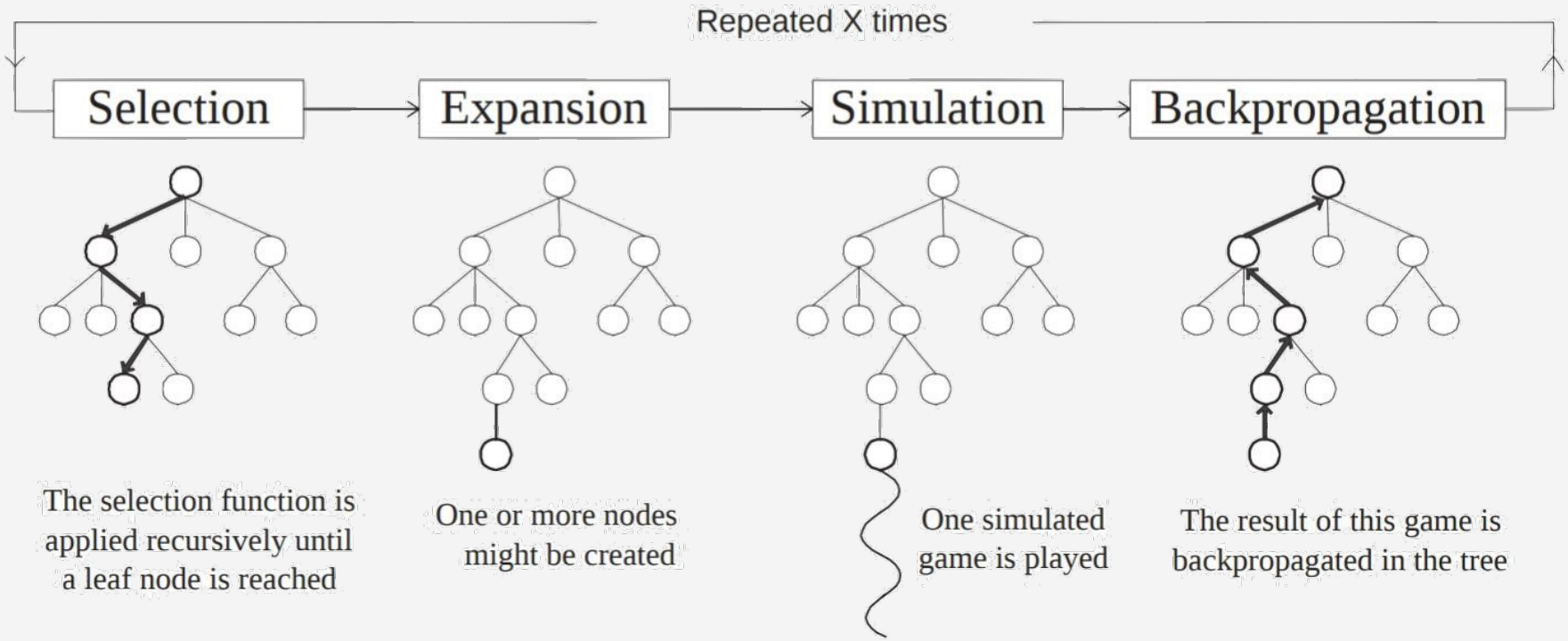
Requirement

- The neural network with a single set of weights can solve problems of any scale i.e. variable number of items and of bin size.
  Solution: the set of the feasible actions is the input of the network.

Desired Features

- *Permutation invariant*, i.e. any permutation of the input set results in the same output permutation.
- The network can process inputs of any size.

Monte Carlo Tree Search

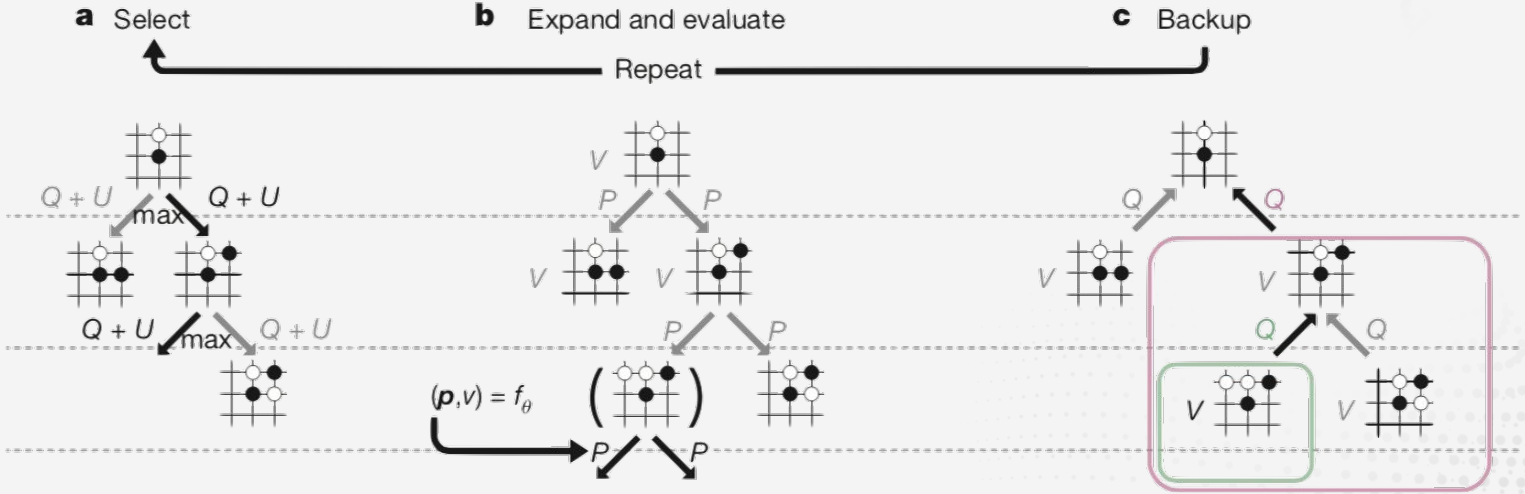


Policy-Value Monte Carlo Tree Search

Use a trained neural network to guide the search.

The goal is twofold:

- Limit the breadth of the search by focusing on the most likely moves (network policy)
- Limit the depth of the search by using an estimation of the future return (network value)



InstaDeep™

Adversarial Learning

A Key Element for Success

AlphaZero is trained using **self-play** which pushes the algorithm to improve as it seeks at beating itself. Single-player games lack the natural incentive provided by Zero-sum games and self-play.

Solution: Introducing an adversarial mechanism

e.g.

Ranked-Reward reproduces the benefits of self-play for single-player MDP by reshaping the rewards of a single agent according to its relative performance over recent games.

Ranked Reward: Enabling Self-Play Reinforcement Learning for Combinatorial Optimization

Alexandre Laterre a.laterre@instadeep.com
Yunguan Fu y.fu@instadeep.com
Mohamed Khalil Jabri mk.jabri@instadeep.com
Alain-Sam Cohen as.cohen@instadeep.com
David Kas d.kas@instadeep.com
Karl Hajjar k.hajjar@instadeep.com
Hui Chen h.chen@instadeep.com
Torbjørn S. Dahl t.dahl@instadeep.com
Amine Kerkeni ak@instadeep.com
Karim Beguir kb@instadeep.com

Abstract

Adversarial self-play in two-player games has delivered impressive results when used with reinforcement learning algorithms that combine deep neural networks and tree search. Algorithms like AlphaZero and Expert Iteration learn *tabula-rasa*, producing highly informative training data on the fly. However, the self-play training strategy is not directly applicable to single-player games. Recently,

Published at NeurIPS 2018

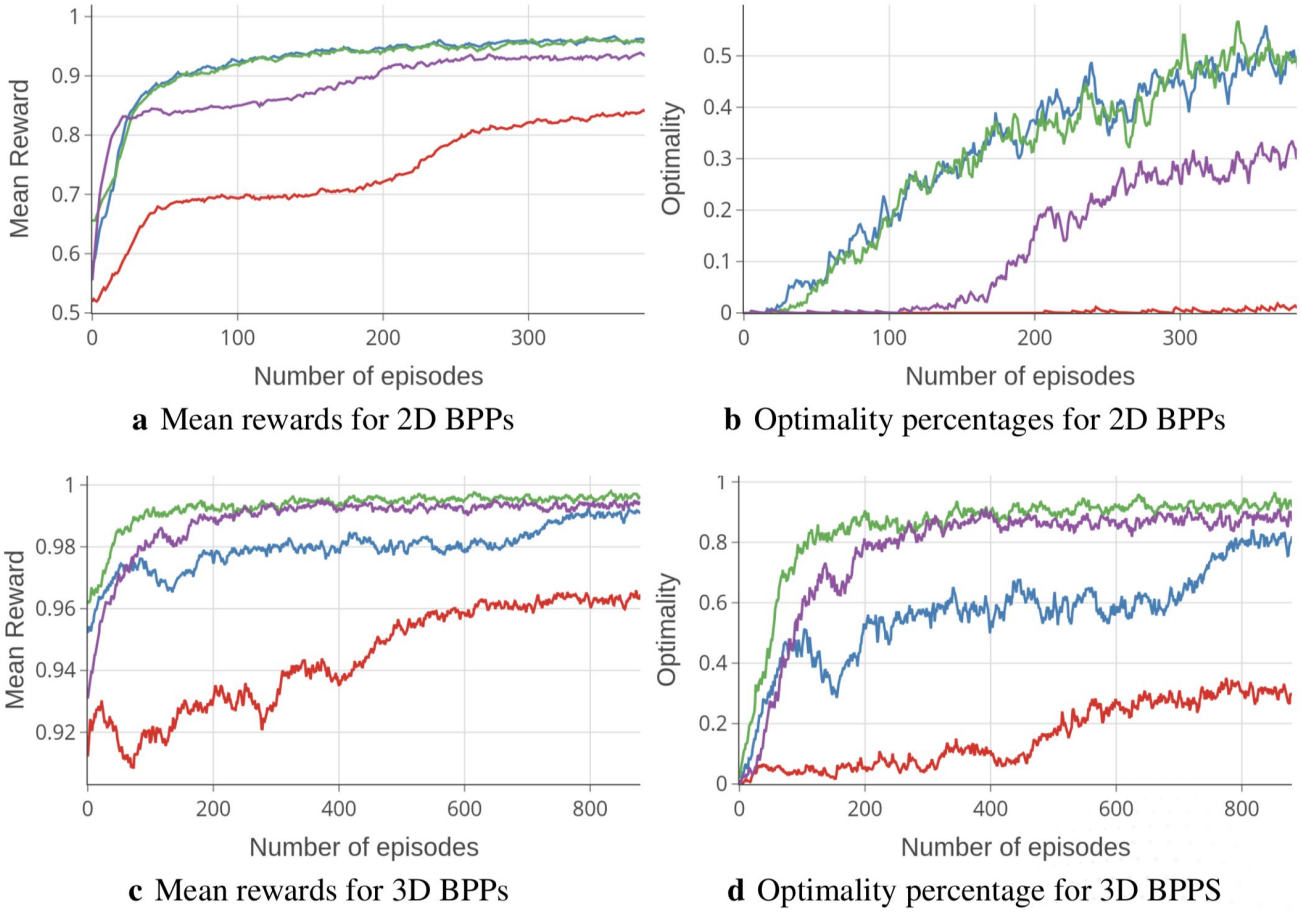
Deep Reinforcement Learning Workshop

Ranked Reward Algorithm

The R2 mechanism compares each of the agent's solutions to its recent performance so that no matter how good it gets, it will have to surpass itself to get a positive reward.

- A fixed-size buffer *B* to record recent agent's returns
- Percentile α used to calculate the baseline value R_{α}
- Reshaping mechanism:

$$z = \begin{cases} 1 & \text{if } G > R_\alpha \text{ or } G = 1 \\ 0 & \text{if } G = R_\alpha \\ -1 & \text{if } G < R_\alpha \end{cases}$$



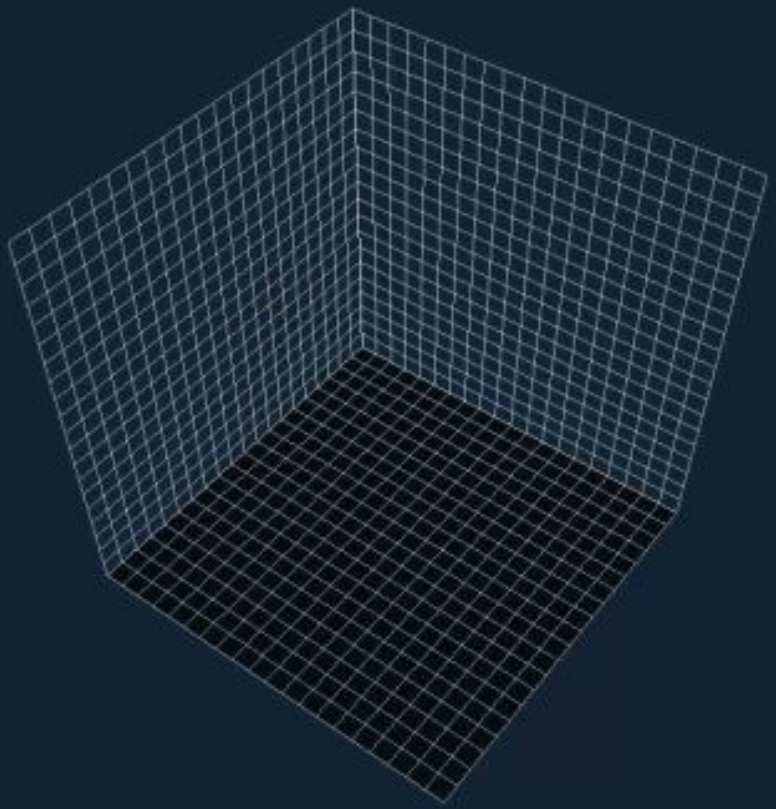
a Mean rewards for 2D BPPs

b Optimality percentages for 2D BPPs

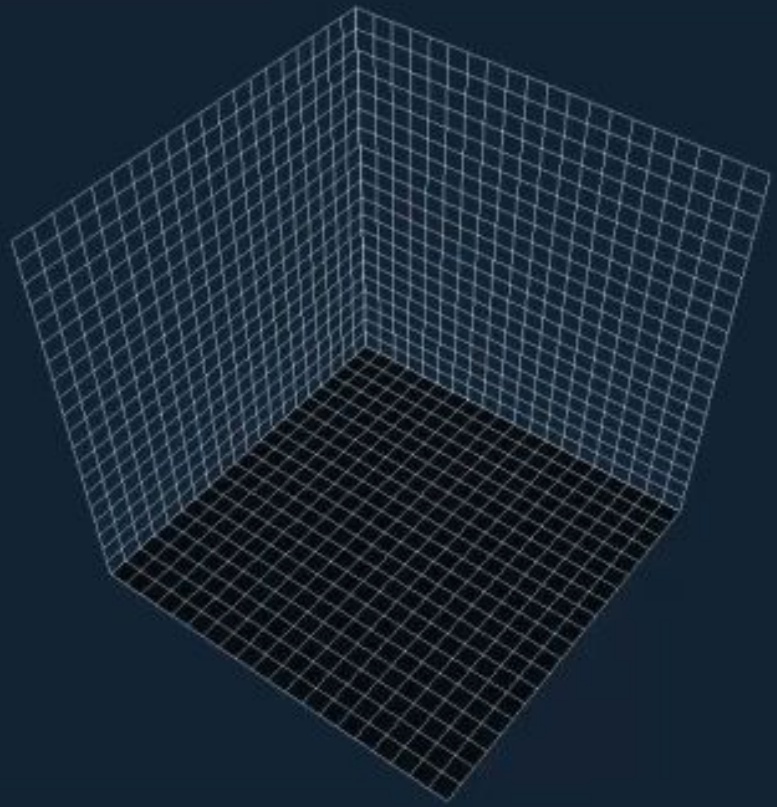
c Mean rewards for 3D BPPs

d Optimality percentage for 3D BPPS

Figure 1: Mean rewards and optimality percentages of R2 on 2D and 3D bin packing problems with percentile of 50 (blue), 75 (green), 90 (purple) and Rank-Free (red).

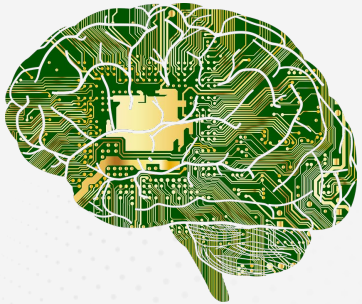


Human: 60%



AI: 100%

As a result, we obtain ...

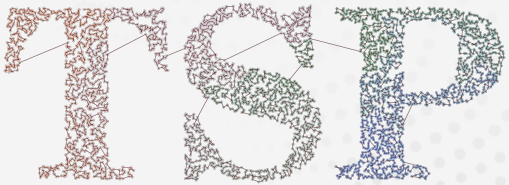


Efficient Optimization Solver

R2 Agent is able to beat classic IP solvers, advanced heuristics, etc.

Knowledge transferable to larger problem instances

The intelligence developed to master small instances of the problem are transferable such that the agent is able to solve larger problem instances as if it had been trained on them.



Approach applicable to many NP-Hard problems

Travelling Salesman Problem, Graph Coloring Problem, etc.

CONCLUSION

- Logistics optimization is dominated by **classical methods** that **do not learn** and **use heuristics**.
- Deep RL can learn to **solve logistics** problems despite **combinatorial explosion.** It can learn end-to-end, and deploy GPU computing at scale to comprehend complexity.
- InstaDeep has built a **Deep RL decision making platform** for logistics.
- Powered by **GPUs**, Deep RL techniques have started **disrupting logistics**.

hello@instadeep.com

@instadeepai

REFERENCES

Ronald Williams. Simple statistical gradient following algorithms for connectionnist reinforcement learning, 1992.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, pages 3104–3112, 2014.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In Advances in Neural Information Processing Systems, pp. 2692–2700, 2015

Mnih,Volodymyr,Kavukcuoglu,Koray,Silver,David,Rusu,AndreiA,Veness,Joel,Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, et al. Human-level control through deep reinforcement learning. Nature, 2015.

Bello, Irwan, Pham, Hieu, Le, Quoc V, Norouzi, Mohammad, and Bengio, Samy. Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv:1611.09940, 2016.

Learning Combinatorial Optimization Algorithms over Graphs. Dai,Hanjun , Khalil, Elias, Zhang, Yuyu, Dilkina, Bistra, and Song, Le. College of Computing, Georgia Institute of Technology 2017.

Silver, David, Schrittwieser, Julian, Simonyan, Karen, Antonoglou, Ioannis, Huang, Aja, Guez, Arthur, Hubert, Thomas, Baker, Lucas, et al. Mastering the game of go without human knowledge. Nature, 550:354–359, 2017a.

Thinking Fast and Slow with Deep Learning and Tree Search. Thomas Anthony, Zheng Tian, and David Barber. UCL 2017

Thank You