DATA SCIENCE
SUMMIT ML EDITION

Everything is a game

use cases for Reinforcement Learning

Filip Wójcik, Ph.D.
Senior Data Scientist, UE Wrocław & Objectivity

filip.wojcik@ue.wroc.pl
https://filip-wojcik.com/en
https://github.com/maddataanalyst/relex

www.ml.dssconf.pl 21st-22nd of June 2022 Warsaw + Online

Organizer: AcademicPartners FUNDACJA

Agenda

01 **RL intro**

What you should know about reinforcement learning?

02 **RL problems**

What are key aspects of RL problems?

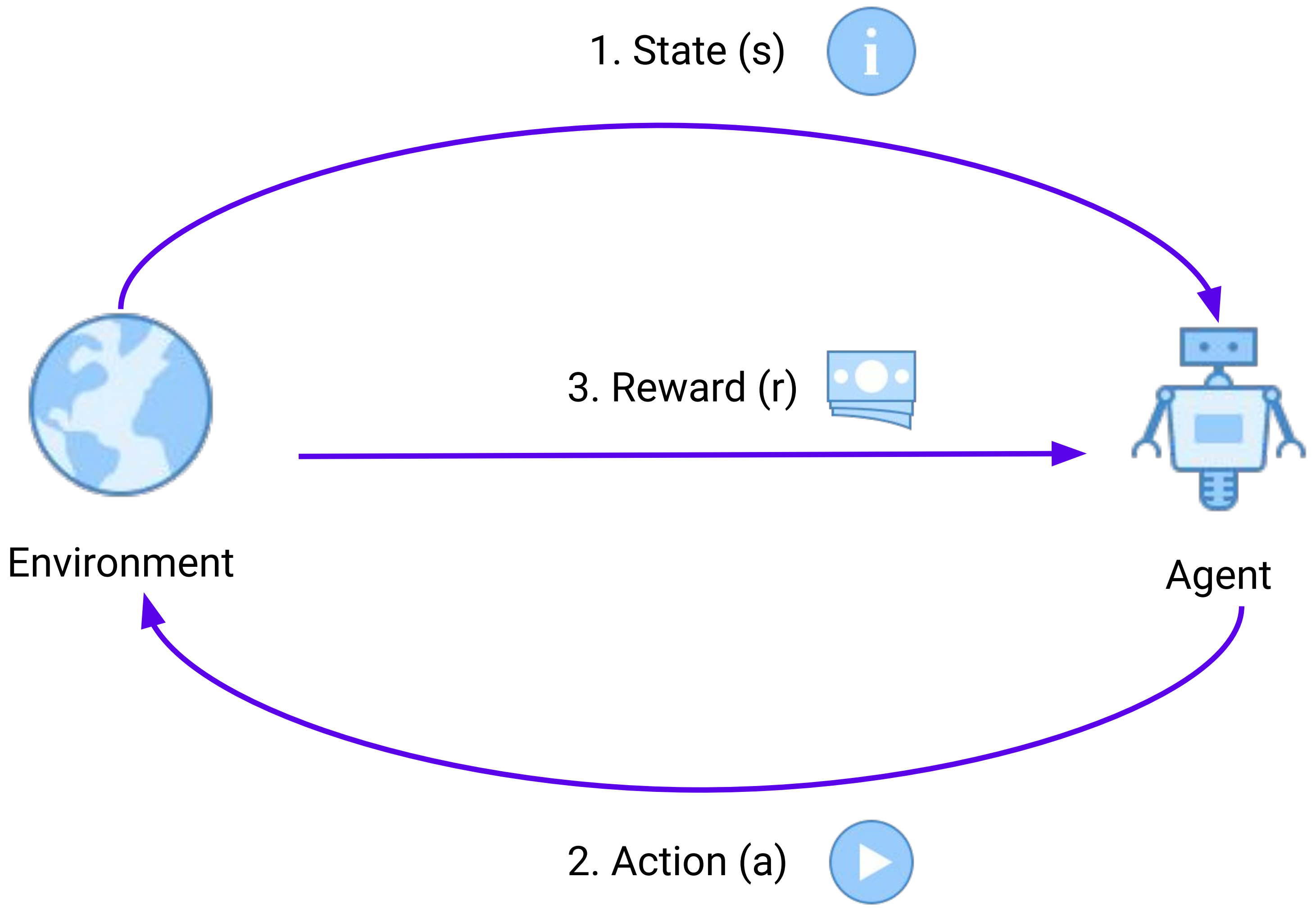
03 **Case studies**

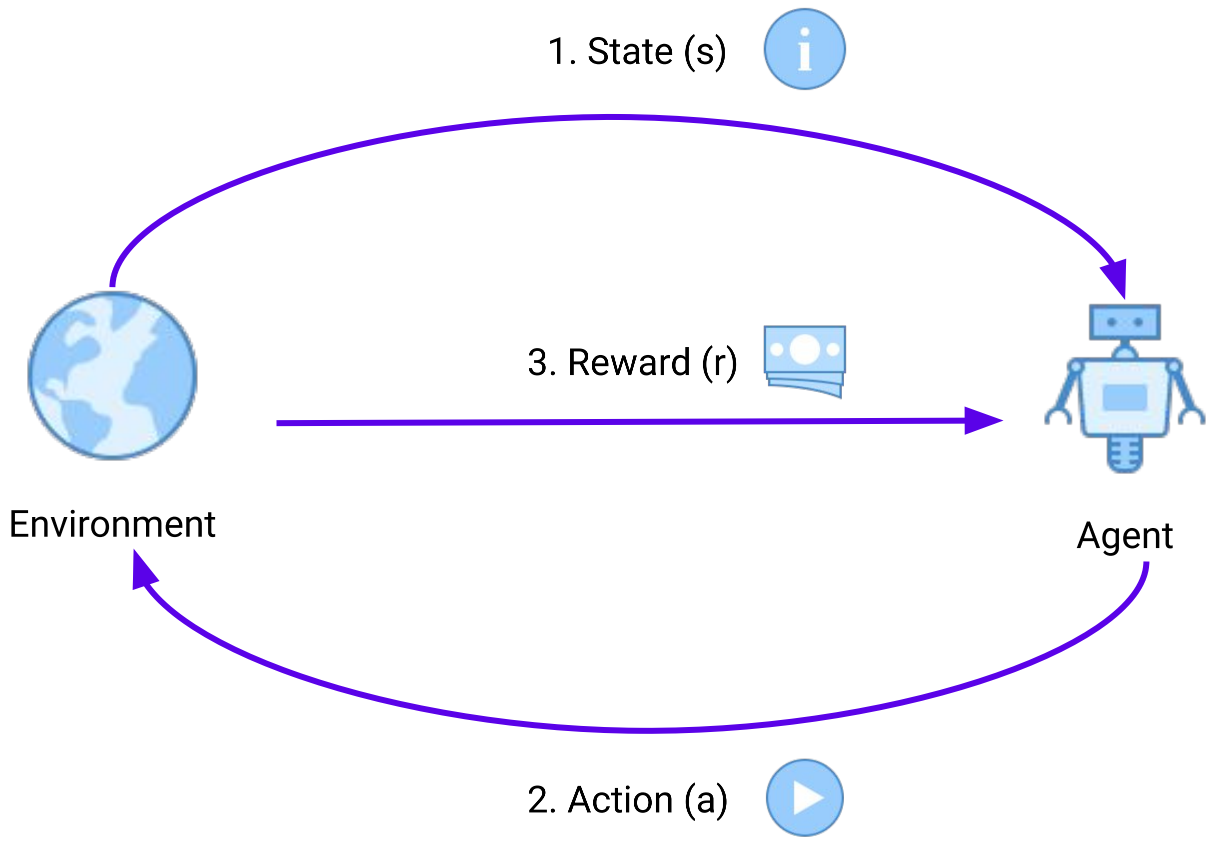
Three different cases, of how you can use RL to solve problems beyond playing games.

RL Intro

What it is all about?

Algebra vector created by pikisuperstar - www.freepik.com


1. State (s)
3. Reward (r)
Environment
Agent
2. Action (a)



Key terms

Trajectory - a set of state-action-rewards

$$\tau \doteq \langle (s_t,\ a_t,\ r_t),\ (s_{t+1},\ a_{t+1},\ r_{t+1}),\ \ldots,\ (s_{t+k},\ a_{t+k}, r_{t+k}) \rangle$$

Policy- a function (of any type), that allows agent to choose actions in a given state

$$\pi(a|s,\theta)$$

(Total) Return - accumulated, total returns from the whole episode (in episodic tasks). Can be discounted.

$$G \doteq r_t + r_{t+1} + \ldots + r_{t+k}$$

$$G \doteq r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} \ldots + \gamma^k r_{t+k} = \sum_{k=0}^{T} \gamma^k r_{t+k}$$

for a given state *s*

what is *the expected future reward*

if we *follow the policy π*

$$V_\pi(s_t) \doteq \mathrm{E}_\pi[G|s_t]$$

$$V_\pi(s_t) \doteq \mathrm{E}_\pi\left[\sum_{k=0}^{T} \gamma^k\, r_{t+k+1}|s_t\right]$$

$$V_\pi(s_t) \doteq \sum_a \pi(a_t|s_t) \sum_{s'r} p(s',r|s,a)\left[r + \gamma \mathrm{E}_\pi\left[G_{t+1}|s'\right]\right]$$

$$V_\pi(s_t) \doteq \sum_a \pi(a_t|s_t) \sum_{s'r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

State Values V(S)

$$\gamma = 0.9$$

Start			
S1, r=0 $V_\pi(s_1) = r_1 + \gamma V_\pi(s_2) \approx 0.65$			
S2, r=0 $V_\pi(s_2) = r_2 + \gamma V_\pi(s_3) \approx 0.73$	S3, r=0 $V_\pi(s_3) = r_3 + \gamma V(s_4) = 0.81$	S4, r=0 $V_\pi(s_4) = r_4 + \gamma V(s_5) = 0.9$	
		S5, r=0 $V_\pi(s_5) = r_{end} = 1$	End, r=1

π(start) = ↓ ↓ → ↓ →

FATALITY

Other important equations

Action-value Q(s,a):

if we execute action *a* at state *s*

and then follow a policy *π*

the expected reward is based on current reward and

state-value

$$Q_\pi(s,\, a\,) \doteq E[r_t + \gamma v_\pi(s_{t+1}) \,|\, s = s_t, a = a_t]$$

$$Q_\pi(s,\, a\,) \doteq \textstyle\sum_{s',r'} p(s', r \,|\, s, a)[r + \gamma v_\pi(s')]$$

Temporal-difference error TD:

if we don't want to wait until the end of the episode

we can estimate the current state value

using the predicted next state value and current reward

if we *follow the policy π*

$$\text{TD error} = r_t - \gamma V_\pi(s_{t+1}) - V_\pi(s_t)$$

RL Problems

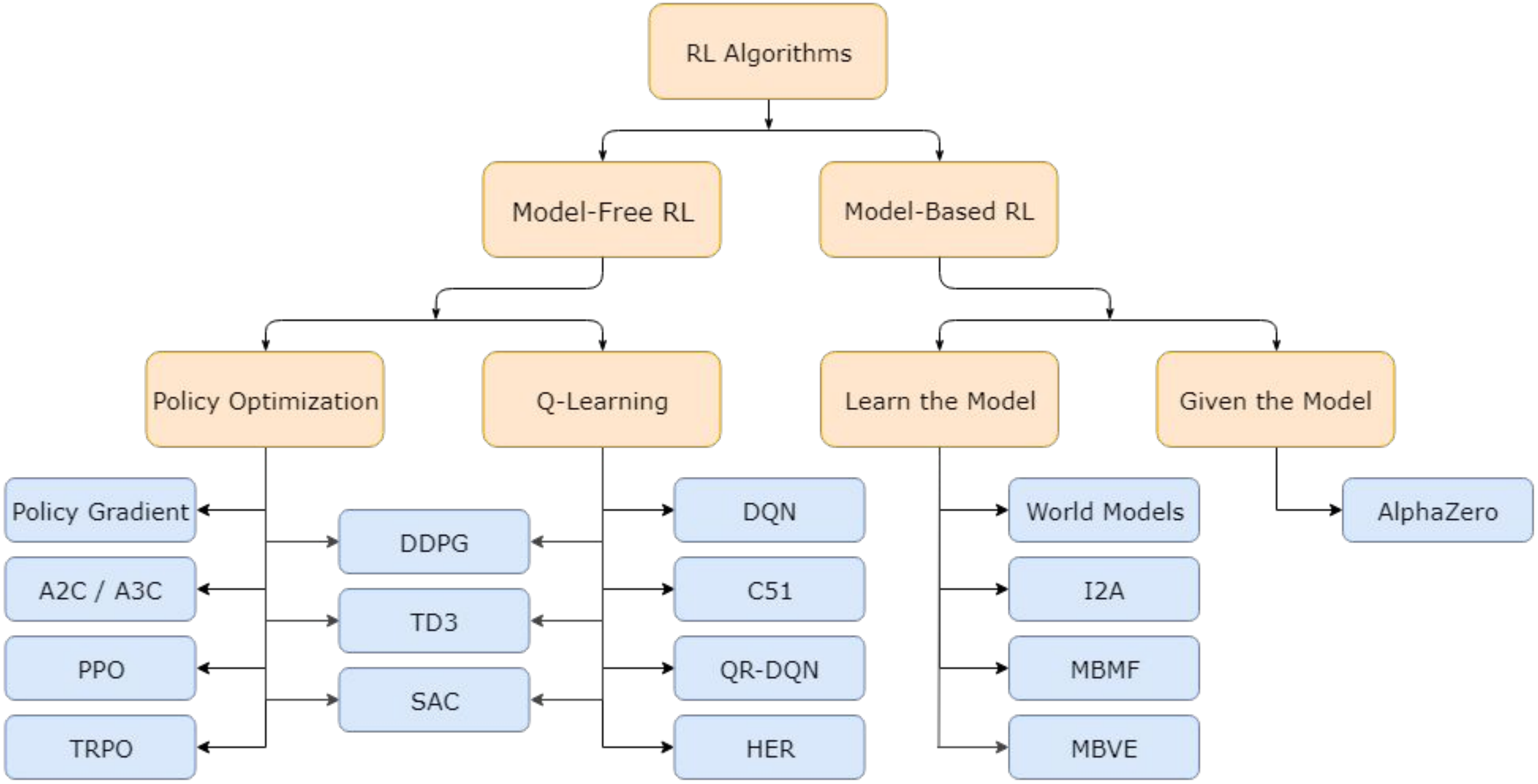
How RL differs from ML?

Teamwork people vector created by pch.vector - www.freepik.com

RL vs ML

	RL	ML
Learning type	**Trial & error - interaction** 1. Agent interacts with the environment 2. Data collected via action-reaction process.	**Supervised or unsupervised** 1. Data is delivered as-is. 2. Data delivered by the training process/oracle.
Time of learning	**Sequential** 1. Agent performs some steps, and waits for the result. 2. Consequences might come in time	**One-shot/batch** 1. Data is delivered all at once or in parts 2. Answers (if any) are given in training set.
Rewards	**Evaluative** 1. Single reward means nothing. 2. It needs to be evaluated in many trials and compared.	**Driven by loss/error** 1. Typically a loss or error function is given. 2. Minimized by chosen learning method
Learning completeness	**Sampled** 1. Via interaction the agent can access parts of the env. space. 2. It is not guaranteed that the whole space will be explored.	**Exhaustive** 1. Typically, the algorithm learns the domain of features on training data. 2. Then applied the knowledge (e.g. scaling) on test data.

To model or not to model?



https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

Case studies

Let’s see some action…

but not in games.

Business landing vector created by pikisuperstar - www.freepik.com

Case 1: Stocks

Stock trading with RL

Case 1 | Stock trading - intro

Research objective

Verify if **RL** algorithm trained in a simulated stock environment (single symbol) can achieve profit in challenging market conditions (2020+).

Algorithms

PPO trained on time-windowed features.
Benchmarks:

- **Random agent**
- **Simple Moving Average Crossover Strategy**

Purpose of the study

1. Demonstration / PoC
2. Preparation for a research paper (multiple stocks, portfolio management)

Case 1 | Stock trading- experimental design

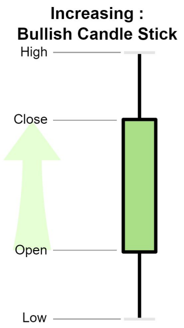
01 **Single symbol -** mostly ETFs were tested

02 **Time-based split:**

2016-2020 - learning

2020- - UNLEASH HELL!!!

03 **Random prices-** in both phases: prices randomly selected from High-Low range



04 **State & actions:**

State: OHLC moving average + std.

Action: buy/sell/hold (all)

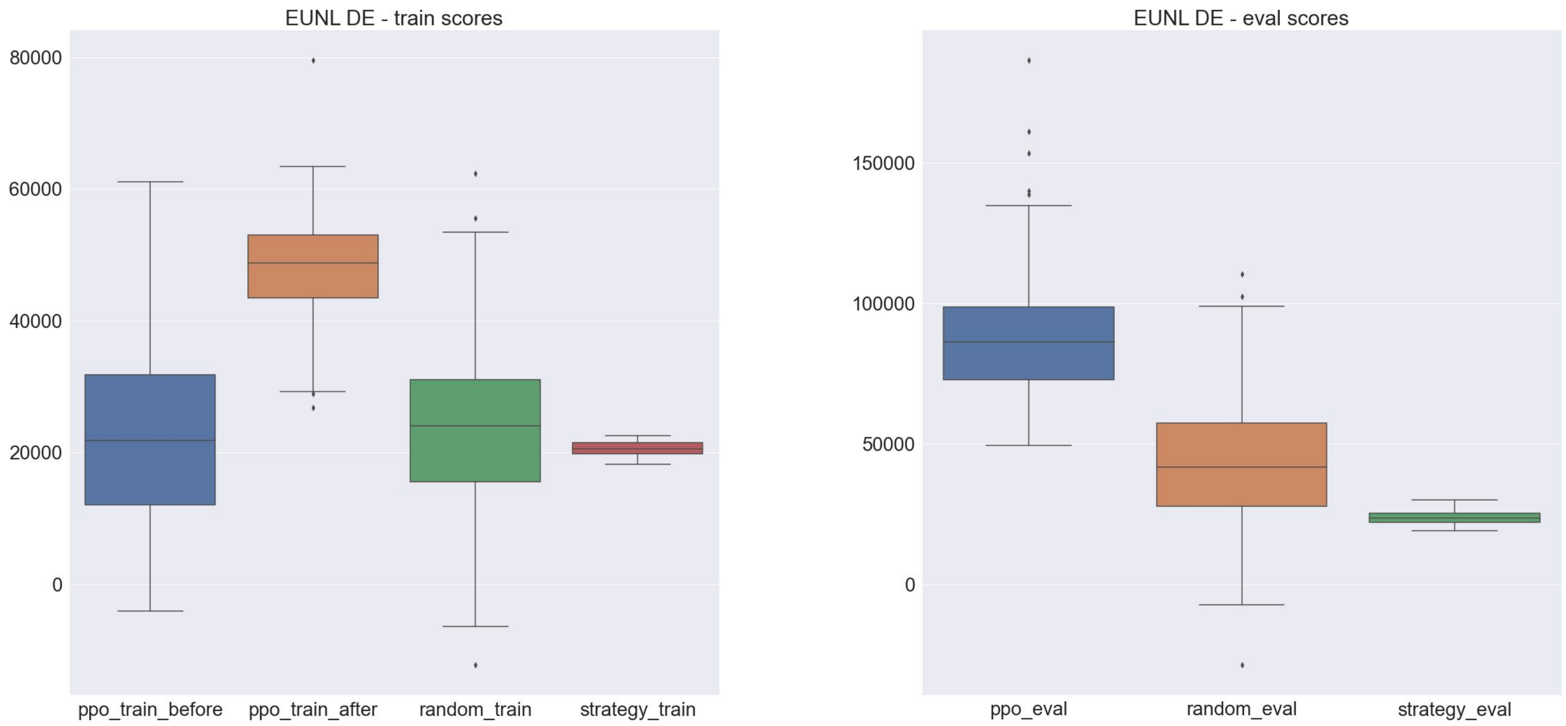
Reward: PnL up to date / final PnL

Case 1

Stock trading- results

Testing:

1. 100 iterations on 2016-2020 period
2. Non-parametric Kruskal test
3. Post-hoc pairwise tests:
   a. non-parametric
   b. Holm p-value correction
   c. $\alpha = 0.05$



	Source	ddof1	H	p-unc
Kruskal	model	2	199.46917	0.0

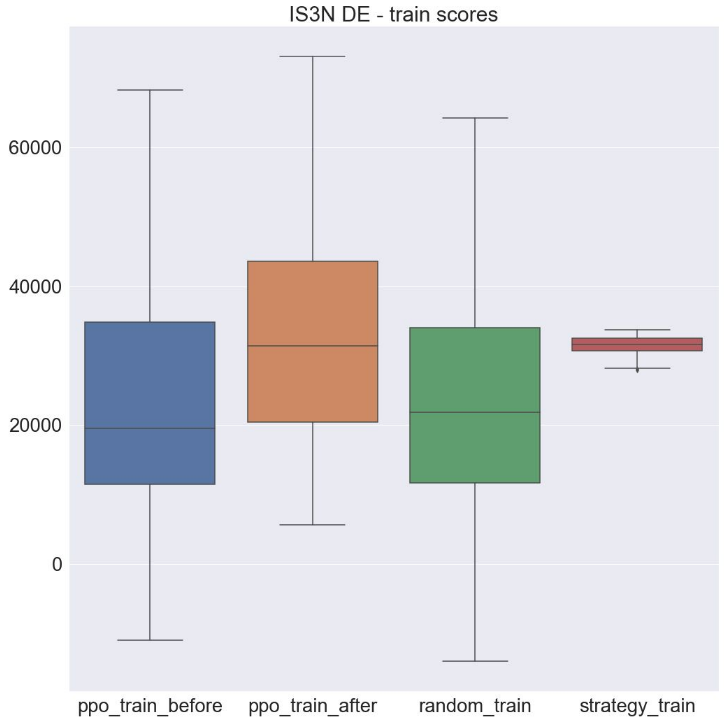
	Contrast	A	B	Paired	Parametric	U-val	alternative	p-unc	p-corr	p-adjust	hedges
0	model	ppo_eval	random_eval	False	False	9230.0	two-sided	0.0	0.0	holm	1.89465
1	model	ppo_eval	strategy_eval	False	False	10000.0	two-sided	0.0	0.0	holm	3.88764
2	model	random_eval	strategy_eval	False	False	7955.0	two-sided	0.0	0.0	holm	1.06278

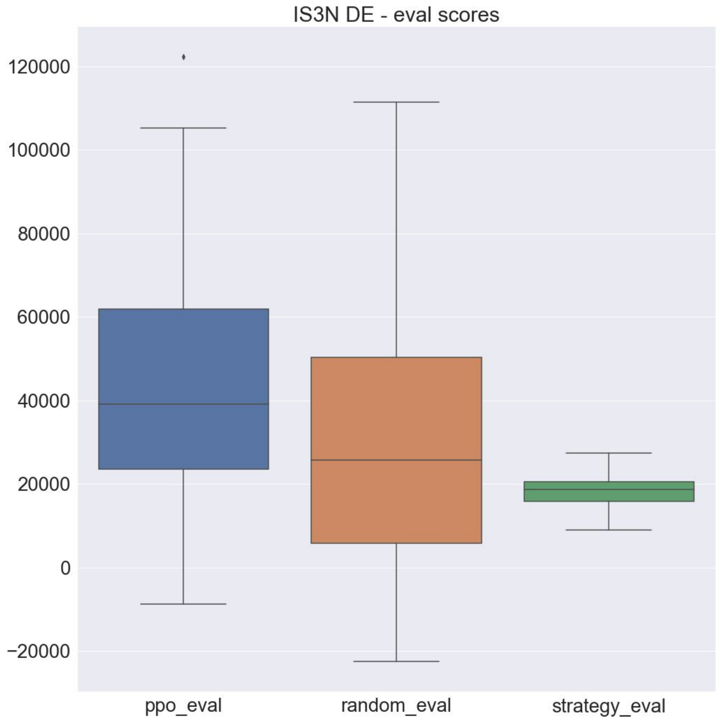
Case 1

Stock trading- results

Testing:

1. 100 iterations on 2016-2020 period
2. Non-parametric Kruskal test
3. Post-hoc pairwise tests:
   a. non-parametric
   b. Holm p-value correction
   c. α = 0.05





	Source	ddof1	H	p-unc
Kruskal	model	2	41.80215	0.0

	Contrast	A	B	Paired	Parametric	U-val	alternative	p-unc	p-corr	p-adjust	hedges
0	model	ppo_eval	random_eval	False	False	6352.0	two-sided	0.00096	0.00192	holm	0.44644
1	model	ppo_eval	strategy_eval	False	False	7855.0	two-sided	0.00000	0.00000	holm	1.25408
2	model	random_eval	strategy_eval	False	False	5815.0	two-sided	0.04658	0.04658	holm	0.51262

Case 1 | Stock trading - materials & tools

Libraries & tools

1. **FinRL** - a library for testing various RL algorithms in stock trading envs: https://github.com/AI4Finance-Foundation/FinRL
2. **YFinance** - an unofficial library for fetching stock prices from Yahoo!. https://github.com/ranaroussi/yfinance

Publications

1. Huang, H., Gao, T., Gui, Y., Guo, J., & Zhang, P. (2022). **Stock Trading Optimization through Model-based Reinforcement Learning with Resistance Support Relative Strength**. arXiv preprint arXiv:2205.15056.
2. Amirhossein Saeidi, S., Fallah, F., Barmaki, S., & Farbeh, H. (2022). **A Novel Neuromorphic Processors Realization of Spiking Deep Reinforcement Learning for Portfolio Management**. arXiv e-prints, arXiv-2203.
3. Liao, S. L., Lin, S. K., Kuang, X. J., & Chen, T. **Portfolio Allocation with Dynamic Risk Preference Via Reinforcement Learning: Evidence from the Taiwan 50 Index**. Available at SSRN 4008759.
4. Sadriu, L. (2022). **Deep Reinforcement Learning Approach to Portfolio Optimization**.
5. Ștefan-Constantin, R. A. D. U., ANGHEL, L. C., & ERMIȘ, I. S. **Are Reinforcement Learning Based Algorithms a Viable Alternative to Traditional Wealth Management Strategies?**. STRATEGICA, 453.
6. Zhou, P., Tang, J., & Li, Y. (2022, April). **Research on investment strategies of stock market based on sentiment indicators and deep reinforcement learning**. In International Conference on Statistics, Applied Mathematics, and Computing Science (CSAMCS 2021) (Vol. 12163, pp. 1151-1156). SPIE.

Case 2: Resource management

Resource allocation with RL

Case 2 | Resource management - intro

Research objective

Verify if RL algorithm can perform **constrained resource allocation** in **highly stochastic environment.**

Algorithms

PPO, AC, VPG, DQN trained on custom "resource management" simulator.
Benchmarks:

- **Random agent**
- **Linear optimization agent**

Purpose of the study

Research paper (in review):

Wójcik, F. (2022). **Utilization of deep reinforcement learning for discrete resource allocation problem in project management - a simulation experiment**. *Business Informatics, Wroclaw University of Economics*.

Case 2

Resource management - experimental design

01

"Company management":

1. Limited resources and $ on start.
2. Goal: earn money, don't go default.
3. Default if:
    a. no more money available;
    b. all resources are lost.

02

Multiple projects to choose - each project has different:

1. Chance of success.
2. Payout per resource
3. If agent stays idle - pay upkeep cost ("bench" in outsourcing corpo).

03

Three levels of difficulty - easy/medium/hard differing by P(success) on each project.

1. On each iteration, projects behave in a non-deterministic ways.
2. Randomly generated levels… projects :)

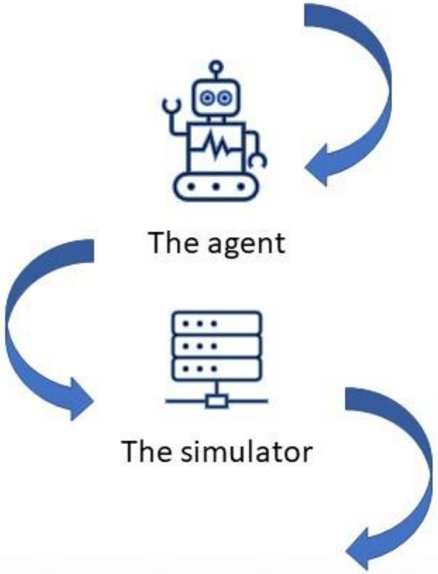
04

State & actions:

1. **State**: project demands + company state
2. **Action**:
    a. resource allocation: proj 1, 2, both
    b. do nothing
    c. free resources (to avoid upkeep cost)

Case 2 | Resource management - experimental design

Time	Demand 1	Pay 1	P(success) 1	Demand 2	Pay 2	P(success) 2	Resouces	Balance
1	120	3	0.5	100	5	0.4	105	100



Action:
Allocate 50% in proj. 1 & 50% proj. 2

Result:
Proj. 1 – **failed (50 res.)**
Proj. 2- **success (50 res.)**
Reward = $50 \times 5 + (55 \times (-1)) = 195$

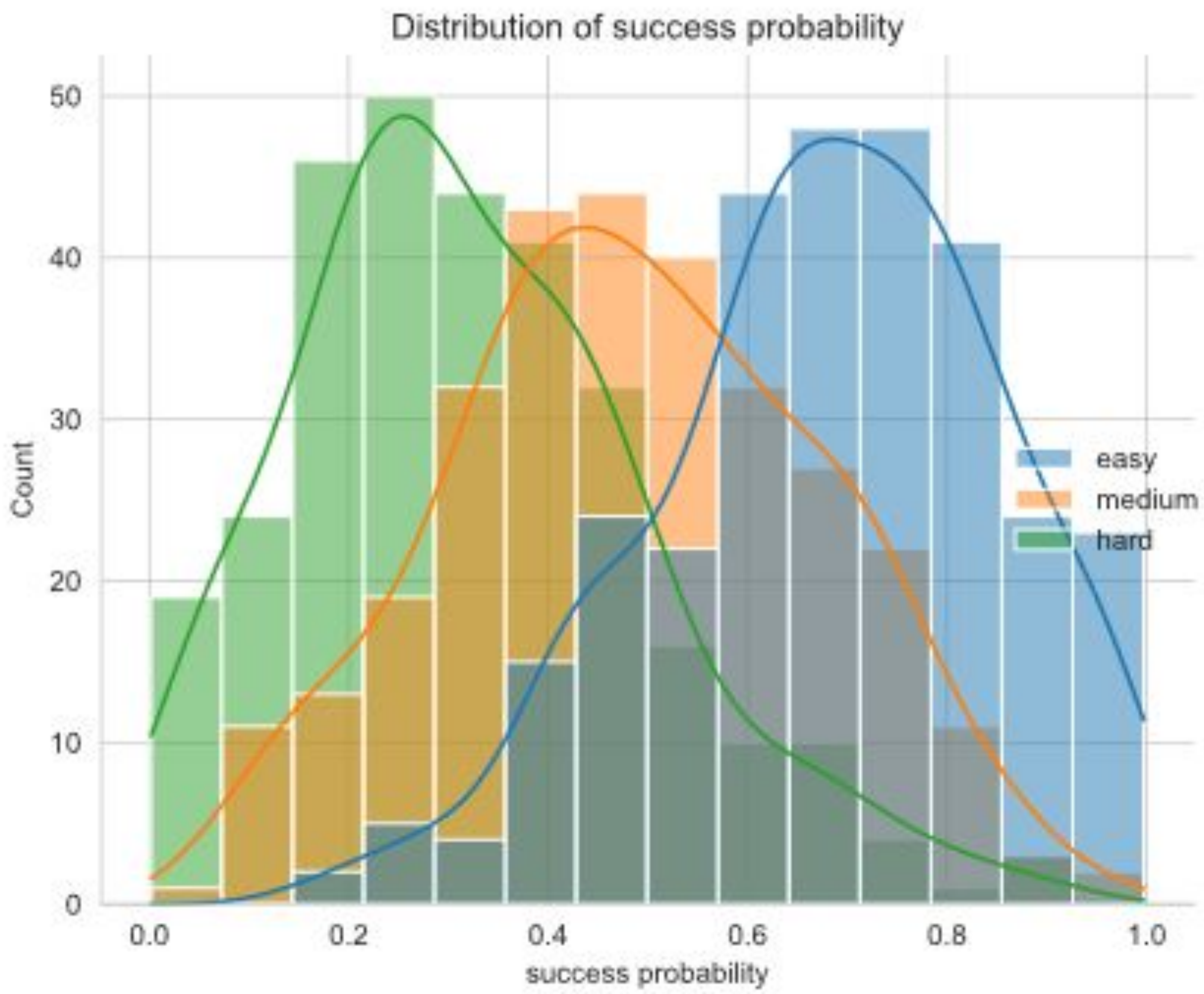
Time	Demand 1	Pay 1	P(success) 1	Demand 2	Pay 2	P(success) 2	Resouces	Balance
1	120	3	0.5	100	5	0.4	105	100
2	10	100	0.8	150	1	0.2	100	295

State & actions:

1. **State**: project demands + company state
2. **Action**: resource allocation: proj 1, 2, both or do nothing

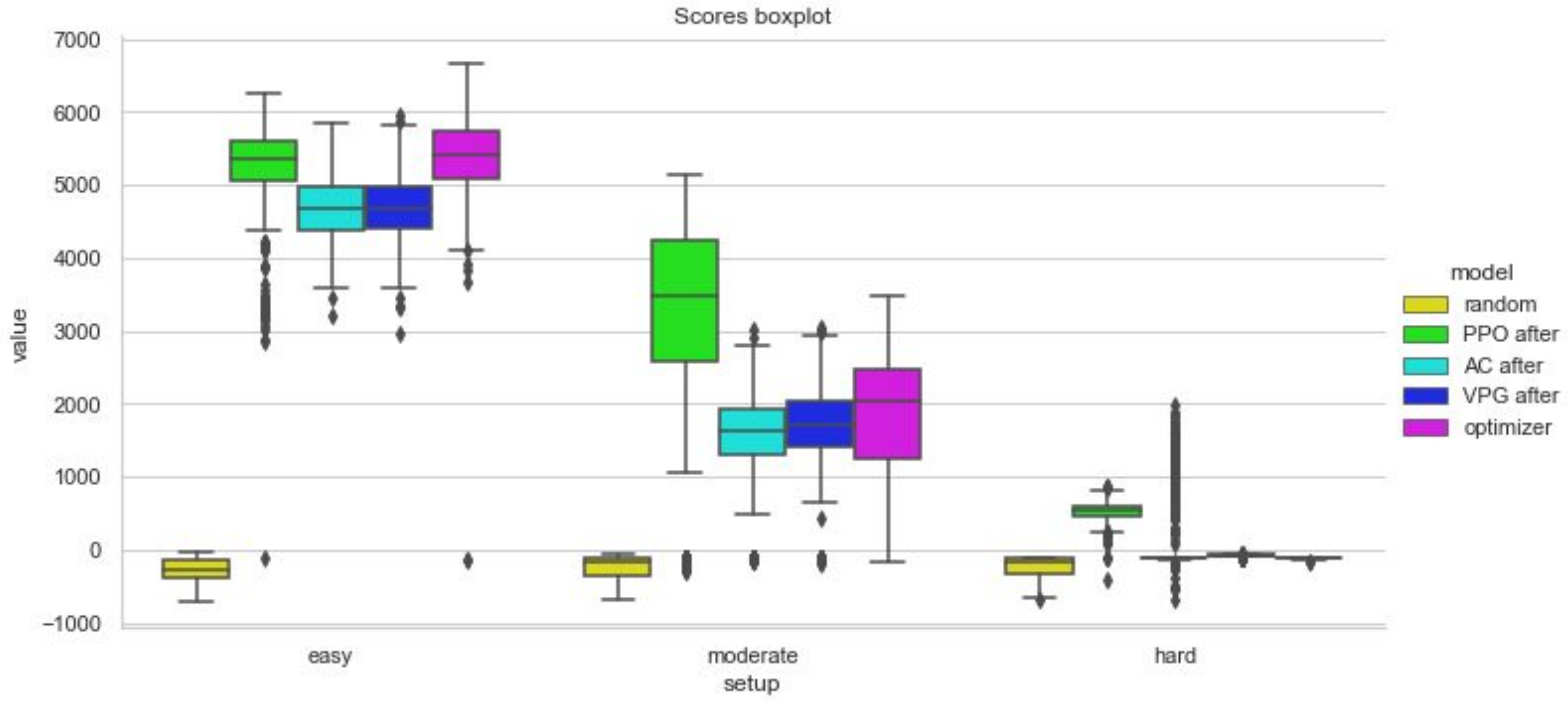
Resource management - experimental design

Three levels of difficulty - easy/medium/hard differing by P(success) on each project.



Case 2

Resource management - results



Case 2 | Resource management - results

Scores per env

Setup\Agent	**Random**	**Optimization**	**VPG**	**AC**	**PPO**
Easy	-290.89 -270.731 (181.360)	**5400.996** **5423.375** (619.269)	4681.178 4680.816 (437.656)	4685.794 4690.119 (425.214)	5221.025 5367.668 (653.343)
Medium	-266.279 -171.603 (187.549)	1696.866 2050.942 (1081.587)	1649.711 1721.975 (653.539)	1524.030 1640.094 (668.603)	**3145.002** **3471.829** (1426.586)
Hard	-268.207 -171 (185.856)	-115.573 -113.268 (12.073)	-73.364 -72.109 (9.932)	137.743 -107.305 (531.494)	**529.445** **532.573** (134.387)

Case 2 | Resource management - results

Testing:

1. 1000 iterations on random environment.
2. Non-parametric (Kruskal) testing with post-hoc pairwise comparisons.
3. Checking the effect size to capture non-trivial significance.

Welch Anova	Ddof 1	4	Ddof 2	1080.87	F stat p-val	2581.87 ≪ 0.001
Model A	**Model B**	**Score diff**	**Diff se**	**Statistic**	**P-val**	**Eff. size**
AC	**PPO**	-1620.97	70.46	-23.01	≪ 0.001	-1.45***
AC	VPG	-125.68	41.81	-3.01	0.02	-0.19
AC	optimizer	-172.84	56.87	-3.04	0.02	-0.19
AC	random	1790.31	31.05	57.65	≪ 0.001	3.64***
PPO	VPG	1495.29	70.17	21.31	≪ 0.001	1.35***
PPO	optimizer	1448.14	80.06	18.09	≪ 0.001	1.14***
PPO	random	3411.28	64.35	53.01	≪ 0.001	3.35***
VPG	optimizer	-47.16	56.51	-0.83	0.9	-0.05
VPG	random	1915.99	30.41	63.01	≪ 0.001	3.98***
optimizer	random	1963.14	49.09	39.99	≪ 0.001	2.53***

Welch Anova	Ddof 1	4	Ddof 2	1136.26	F stat p-val	3694.87 ≪ 0.001
Model A	**Model B**	**Score diff**	**Diff se**	**Statistic**	**P-val**	**Eff. size**
AC	**PPO**	-391.7	24.52	-15.98	≪ 0.001	-1.01***
AC	VPG	211.11	23.77	8.88	≪ 0.001	0.56**
AC	optimizer	253.32	23.78	10.65	≪ 0.001	0.67**
AC	random	405.95	25.18	16.12	≪ 0.001	1.02***
PPO	VPG	602.81	6.03	100.03	≪ 0.001	6.32***
PPO	optimizer	645.02	6.03	106.89	≪ 0.001	6.76***
PPO	random	797.65	10.26	77.77	≪ 0.001	4.91***
VPG	optimizer	42.21	0.7	60.37	≪ 0.001	3.82***
VPG	random	194.84	8.32	23.41	≪ 0.001	1.48***
optimizer	random	152.63	8.33	18.33	≪ 0.001	1.16***

Case 2 | **Resource management - materials & tools**

Libraries & tools

1. **Cvxopt**- a library for performing linear optimization tasks: https://cvxopt.org/
2. **PuLp**- another brilliant resource allocation library https://coin-or.github.io/pulp/

Publications

1. Xu, Y., Zhao, Z., Cheng, P., Chen, Z., Ding, M., Vucetic, B., & Li, Y. (2021). **Constrained Reinforcement Learning for Resource Allocation in Network Slicing**. IEEE Communications Letters, 25(5), 1554–1558. https://doi.org/10.1109/LCOMM.2021.3053612
2. Yan, Y., Chow, A. H. F., Ho, C. P., Kuo, Y.-H., Wu, Q., & Ying, C. (2021). **Reinforcement Learning for Logistics and Supply Chain Management: Methodologies, State of the Art, and Future Opportunities**. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3935816
3. Ye, H., Li, G. Y., & Juang, B.-H. F. (2019). **Deep Reinforcement Learning Based Resource Allocation for V2V Communications**. IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, 68(4), 3163. https://doi.org/10.1109/TVT.2019.2897134
4. Ye, K., Shi, X., Li, H., & Shi, N. (2014). **Resource allocation problem in port project portfolio management. Proceedings - 2014 7th International Joint Conference on Computational Sciences and Optimization**, CSO 2014, 159–162. https://doi.org/10.1109/CSO.2014.36
5. Yu, L., Zhang, C., Jiang, J., Yang, H., & Shang, H. (2021). **Reinforcement learning approach for resource allocation in humanitarian logistics**. Expert Systems with Applications, 173. https://doi.org/10.1016/j.eswa.2021.114663
6. Yuan, Y., Li, H., & Ji, L. (2021). **Application of Deep Reinforcement Learning Algorithm in Uncertain Logistics Transportation Scheduling.** Computational Intelligence and Neuroscience, 2021. https://doi.org/10.1155/2021/5672227
7. Zuo, J., & Joe-Wong, C. (2021). **Combinatorial multi-armed bandits for resource allocation**. 2021 55th Annual Conference on Information Sciences and Systems, CISS 2021. https://doi.org/10.1109/CISS50987.2021.9400228

Case 3: Task Scheduling

Production line optimization with RL

Production process vector created by jcomp - www.freepik.com

Case 3 | Task Scheduling - intro

Research objective

Verify if RL algorithm can optimize **production line organization** to minimize product assembly time,idle periods and collisions.
Verify if RL can perform constrained task scheduling.

Algorithms

PPO, AC, VPG, DQN trained on custom “production line” simulator.
Benchmarks:

- **Random agent**
- **“Available choice” agent -** agent that chooses minimal waiting time product.

Purpose of the study

Commercial implementation for **Objectivity Co Uk.**

1. **Rafał Sokołowski - Lead Developer**
2. Julia Orłowska - Team Leader
3. Marcel Falkiewicz, Ph.D. - consultation & design
4. Filip Wójcik, Ph.D. - consultation, design & development

Objectivity

Case 3 | Task scheduling - experimental design

01

Simulated production line:

1. Multiple product types on the line.
2. Multiple assembly stations.
3. Each product type is processed differently

02

The goal - minimize waiting time on the line, process all waiting products/product types as fast as possible.

03

1. **Scalable environment -** products and stations can be added.
2. **Customizable reward/penalty system**.
3. Human-friendly visualization :)

04

State & actions:

1. **State**:
   a. number of available products of each type;
   b. product type waiting time on each station.
2. **Action**: product type to put next on the line.
3. Rewards:
   a. **positive** - for each product leaving the line;
   b. **negative -** idle time.

Case 3 | Task scheduling - experimental design

Summary statistics: total time, total wait time

```
Time: 17 | Cumulative Reward: 0.40
Waited: 1 | Reward: 0.00
```

Current situation:

1. Which product (P#) is where
2. How much time left on each assembly station

```
Position    :  | P0 | P1 | -- | P0 | -- |
Time        :  | 6  | 2  | -- | 5  | -- |
```

Number of products of each type.

```
P0: 3
P1: 1
P2: 0
```

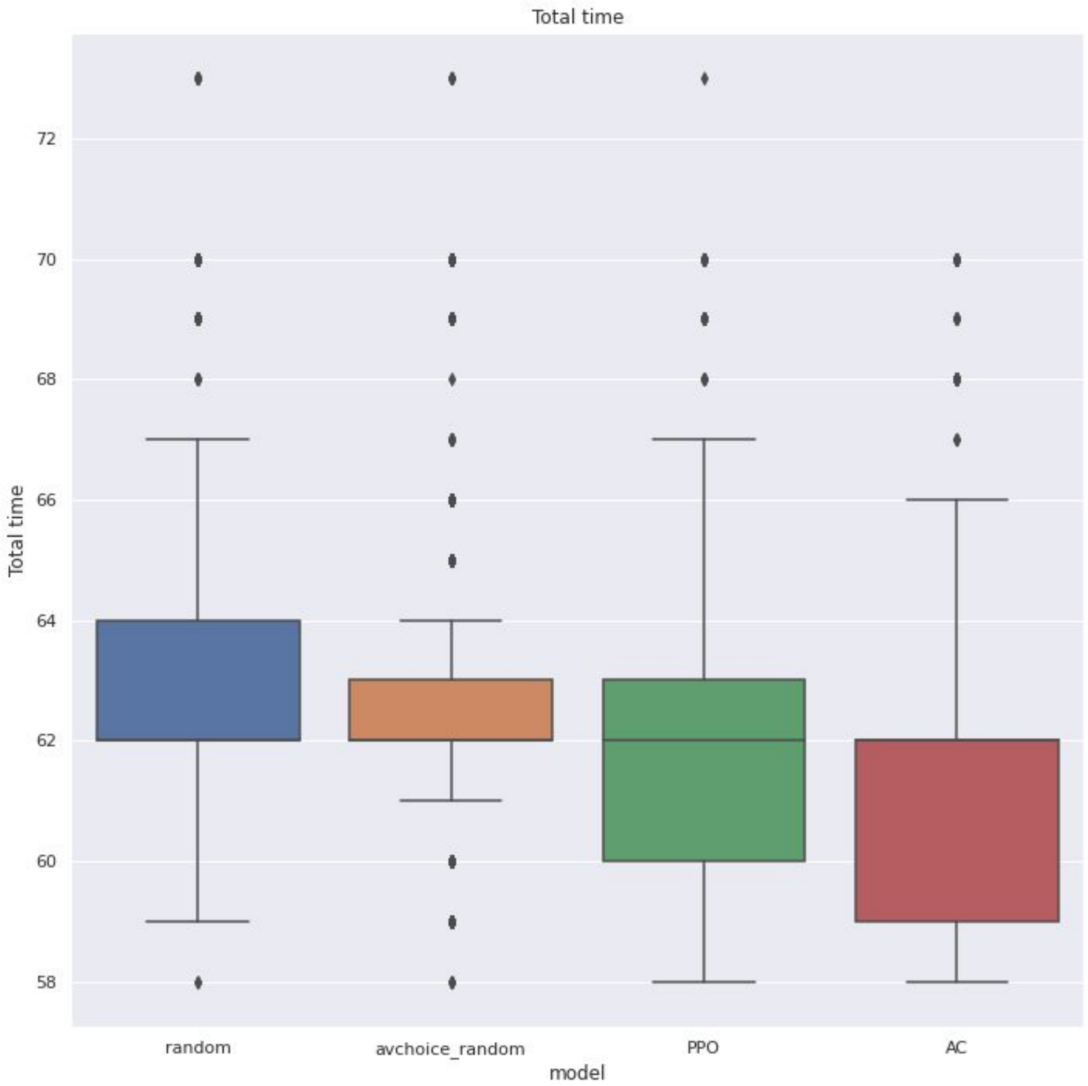
```
Time Matrix
[7 1 3 5 6]
[3 3 2 5 0]
[6 2 5 0 4]
```

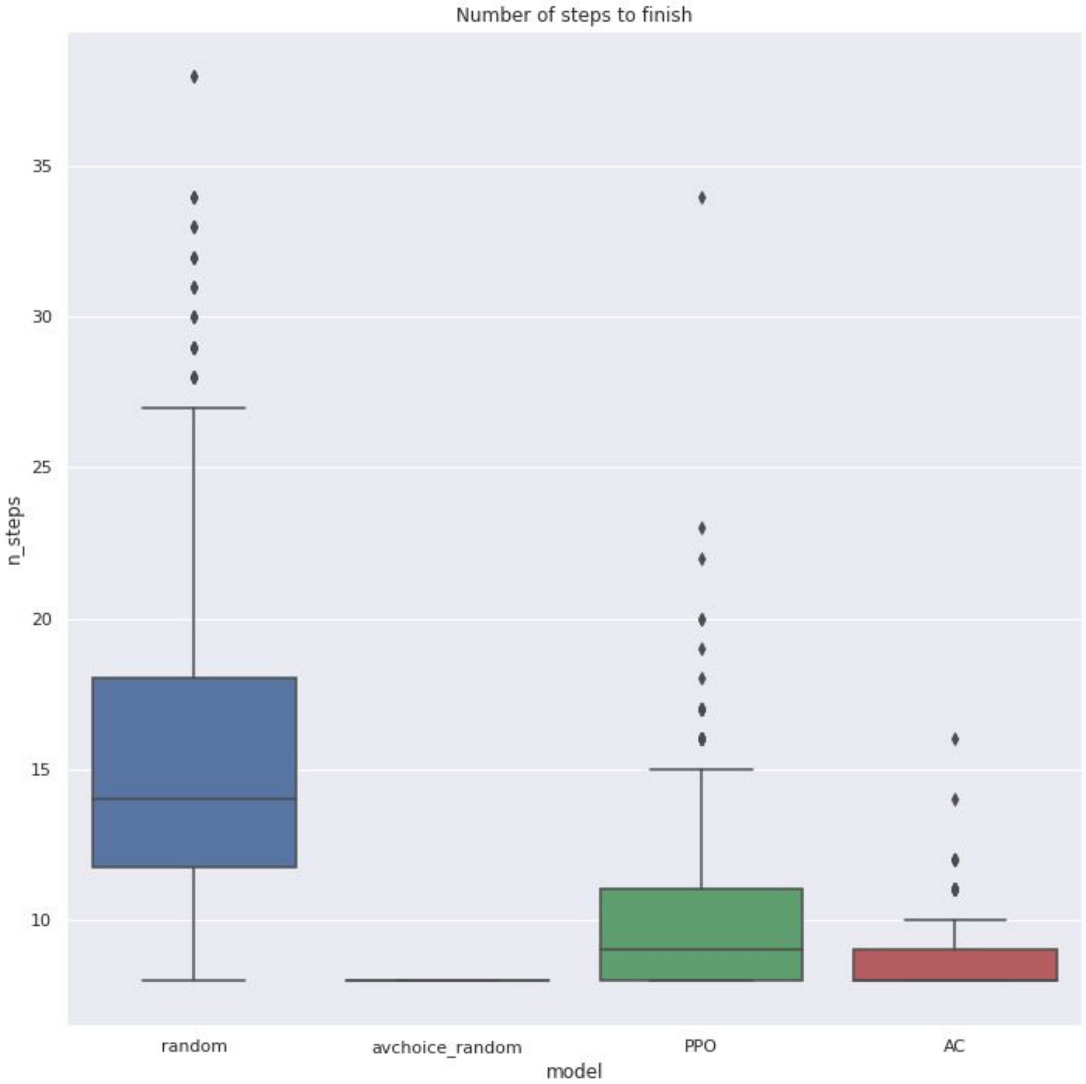
“Time cost” matrix: how much time each product type spends on each assembly station.

Case 3 | Task scheduling - results

```
Time: 0 | Cumulative Reward: 0.10

Waited: 0 | Reward: 0.10


Position    :  | P0 | -- | -- | -- | -- |

Time        :  | 7  | -- | -- | -- | -- |


                    Time Matrix

P0: 4               [7 1 3 5 6]

P1: 2               [3 3 2 5 0]

P2: 1               [6 2 5 0 4]
```

Case 3 | Task scheduling - results





Libraries & tools

1. **Stable Baselines 3**- ready-to use RL algorithms: https://stable-baselines3.readthedocs.io/
2. **Tensorflow Agents** - TF library for RL. Option to extend with own algorithms https://www.tensorflow.org/agents
3. **Ray** - production- ready RL library, that is extremely fast https://docs.ray.io/en/latest/rllib/index.html

Publications

1. Dong, T., Xue, F., Xiao, C., & Li, J. (2020). **Task scheduling based on deep reinforcement learning in a cloud manufacturing environment. Concurrency and Computation: Practice and Experience**, 32(11), e5654.
2. Hu, Z., Tu, J., & Li, B. (2019, July). **Spear: Optimized dependency-aware task scheduling with deep reinforcement learning**.
3. Shyalika, C., Silva, T., & Karunananda, A. (2020). **Reinforcement learning in dynamic task scheduling: A review.** SN Computer Science, 1(6), 1-17.
4. Wang, L., Pan, Z., & Wang, J. (2021). A **review of reinforcement learning based intelligent optimization for manufacturing scheduling**. Complex System Modeling and Simulation, 1(4), 257-270.
5. *Waubert de Puiseau, C., Meyes, R., & Meisen, T. (2022). **On reliability of reinforcement learning based production scheduling systems: a comparative survey**. Journal of Intelligent Manufacturing, 33(4), 911-927.

Summary | Cases suitable for RL modeling

Time-dependence

1. Problems that require sequential actions;
2. Time-dependent interactions.

No analytical solution

1. If analytical solution exists - probably optimization is better.
2. If the problem is complicated and looks like a simulation exercises - probably RL is a good choice.

Stochasticity (uncertainty)

1. Deterministic problems → classic optimization
2. Uncertainty → RL

Looks like… a turn-based strategy game :)

1. If the problem looks like a strategy game - definitely it can be modeled with RL.
2. Think about:
    a. Base building
    b. Army planning
    c. Resource gathering…
3. You can make creative analogies!

The code & research

The code: https://github.com/maddataanalyst/relex

RELEX - Reinforcement Learning Experiments

Intro

RELEX project was created with three ideas in mind: To teach myself Reinforcement Learning by implementing algorithms from scratch; To teach others who struggle to understand some detailed aspects of reinforcement learning algorithms; To create a space where I can experiment with innovative ideas and environments for research purposes.

Therefore, this is not a production-ready or deployment-ready library. But if you are looking for some (hopefully) easy-to-understand implementations of RL from scratch or some inspirations for study/research/paper - probably this is the right place :)

Website + blog: https://filip-wojcik.com/en

NeuraSYS Home Projects Talks Publications Recent posts Blog Contact

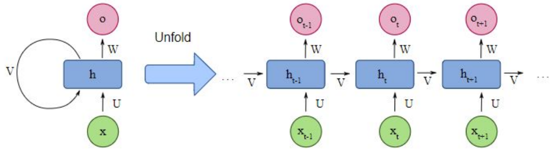
Search... /

← Posts

RNNS

Fifty Shades of Time: Time series & RNNs

RNNs



The research: https://www.researchgate.net/profile/Filip-Wojcik

Usage of deep neural networks as recommendation engines. Review of selected approaches

Conference Paper | Full-text available | March 2021 · Zagadnienia aktualnie poruszane prze..

Filip Wójcik

Add to project | Add supplementary resources

Beyond the barrier of mistrust – an overview of selected methods for explaining predictions of machine learning models.

Conference Paper | Full-text available | March 2021 · Zagadnienia aktualnie poruszane prze..

Filip Wójcik

Add to project | Add supplementary resources

Improvement of e-commerce recommendation systems with deep hybrid collaborative filtering with content: A case study

Article | Full-text available | November 2020 · Econometrics

Filip Wójcik · Michał Górnik

Add to project | Add supplementary resources

DATA SCIENCE SUMMIT ML EDITION

Thank you for watching!

Remember to rate the presentation and leave your questions in the section below.

AcademicPartners FUNDACJA