Deep Reinforcement Learning in foraging simulation

Paulo Bruno De Sousa Serafim

July, 2023

paulo.desousa@gssi.it

paulobruno.github.io

GRAN SASSO SCIENCE INSTITUTE

SCHOOL OF ADVANCED STUDIES

Scuola Universitaria Superiore

MDCC UFC

Deep Reinforcement Learning in foraging simulation

Reinforcement Learning?

Markov Decision Process

Definition [edit]

A Markov decision process is a 4-tuple (S, A, P_a, R_a), wh

- S is a set of states called the *state space*,
- A is a set of actions called the *action space* (alternatively, A_s is the set of actions available from state s),
- $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is the probability that action a in state s at time t will lead to state s' at time $t + 1$,
- $R_a(s, s')$ is the immediate reward (or expected immediate reward) received after transitioning from state s to state s', due to action a

The state and action spaces may be finite or infinite, for example the set of real numbers. Some processes with countably infinite state and action spaces can be reduced to ones with finite state and action spaces.[3]

A poli

actio

REINFORCEMENT LEARNING

Reinforcement learning [edit]

Main article: Reinforcement learning

Reinforcement learning uses MDPs where the probabilities or rewards are unknown.[11]

For this purpose it is useful to define a further function, which corresponds to taking the action a and then cor whatever policy one currently has):

$$Q(s, a) = \sum_{s'} P_a(s, s')(R_a(s, s') + \gamma V(s')).$$

lgorithms [edit]

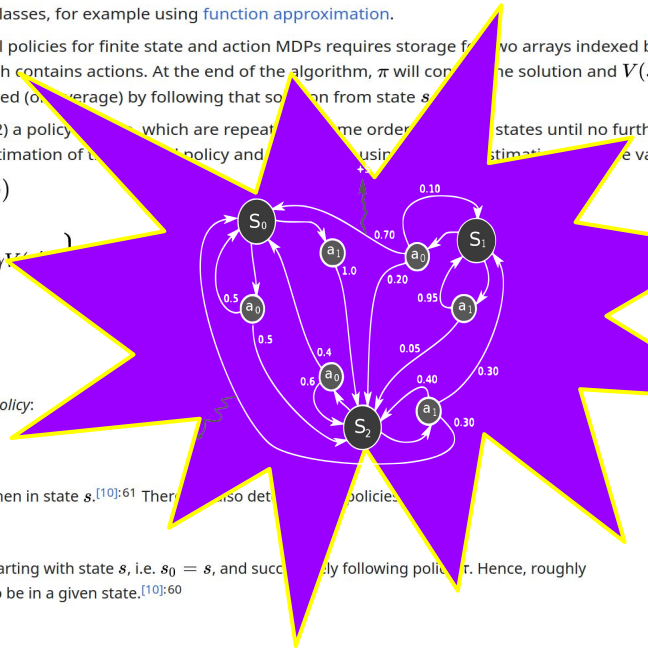
lutions for MDPs with finite state and action spaces may be found through a variety of methods such as dynamic programming. The a this section apply to MDPs with finite state and action spaces and explicitly given transition probabilities and reward functions, but the ncepts may be extended to handle other problem classes, for example using function approximation.

e standard family of algorithms to calculate optimal policies for finite state and action MDPs requires storage f vo arrays indexed b *ue* V, which contains real values, and *policy* π, which contains actions. At the end of the algorithm, π will con he solution and $V($ ntain the discounted sum of the rewards to be earned (o erage) by following that so on from state s

e algorithm has two steps, (1) a value update and (2) a policy which are repeat me order states until no furth anges take place. Both recursively update a new estimation of policy and usin e va

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s') \left(R_{\pi(s)}(s, s') + \gamma V(s')\right)$$

$$\pi(s) := \operatorname{argmax}_a \left\{ \sum_{s'} P_a(s, s') \left(R_a(s, s') + \gamma V(\right) \right\}$$



Policy [edit]

The agent's action selection is modeled as a map called *policy*:

$$\pi : A \times S \to [0, 1]$$
$$\pi(a, s) = \Pr(a_t = a \mid s_t = s)$$

The policy map gives the probability of taking action a when in state s.[10]:61 Ther so det olicie

State-value function [edit]

The value function $V_\pi(s)$ is defined as, *expected return* starting with state s, i.e. $s_0 = s$, and succ ly following poli r. Hence, roughly speaking, the value function estimates "how good" it is to be in a given state.[10]:60

$$V_\pi(s) = \mathrm{E}[R \mid s_0 = s] = \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s\right],$$

where the random variable R denotes the **return**, and is defined as the sum of future discounted rewards:

$$R = \sum_{t=0}^{\infty} \gamma^t r_t,$$

where r_t is the reward at step t, $\gamma \in [0, 1)$ is the discount-rate. Gamma is less than 1, so events in the distant future are weighted less than events in the immediate future.

Definition [edit]

A Markov decision process is a 4-tuple (S, A, P_a, R_a), wh

- S is a set of states called the *stat*
- A is a set of actions called the *ac* natively, A_s is the set of actions available from state s),
- $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s$ bability that action a in state s at time t will lead to state s
- $R_a(s, s')$ is the immediate reward (or ard) received after transitioning from state s to state

The state and action spaces may be finite or numbers. Some processes with countably i be reduced to ones with finite state and actio

A polic action

Markov Decision Process

REINFORCEM

Reinforcement learning

Main article: Reinforcem

Reinforcement learning uses

For this purpose it is useful to whatever policy one currently

$$Q(s, a) = \sum_{s'} P_a(s, s')(R_a$$

lgorithms [edit]

lutions for MDPs with finite state and action spaces may be found through a variety of methods such as dynamic programming. The a this section apply to MDPs with finite state and action spaces and explicitly given transition probabilities and reward functions, but the ncepts may be extended to handle other problem classes, for example using function approximation.

e standard family of algorithms to calculate optimal policies for finite state and action MDPs requires storage f vo arrays indexed b *ue* V, which contains real values, and *policy* π, which contains actions. At the end of the algorithm, π will con he solution and $V($ ntain the discounted sum of the rewards to be earned (o erage) by following that so on from state s

e algorithm has two steps, (1) a value update and (2) a policy which are repeat me order states until no furth anges take place. Both recursively update a new estimation of policy and usin e va

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s') \left(R_{\pi(s)}(s, s') + \gamma V(s')\right)$$

$$\pi(s) := \operatorname{argmax}_a \left\{ \sum_{s'} P_a(s$$

Policy [edit]

The agent's action

$\pi : A$

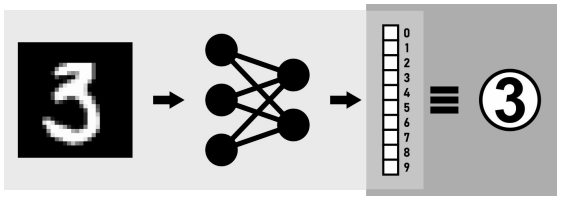
$V_\pi(s) = \mathrm{E}[R$

$\gamma^t r_t,$

is the re , so events in the distant future are weighted less than the immedi

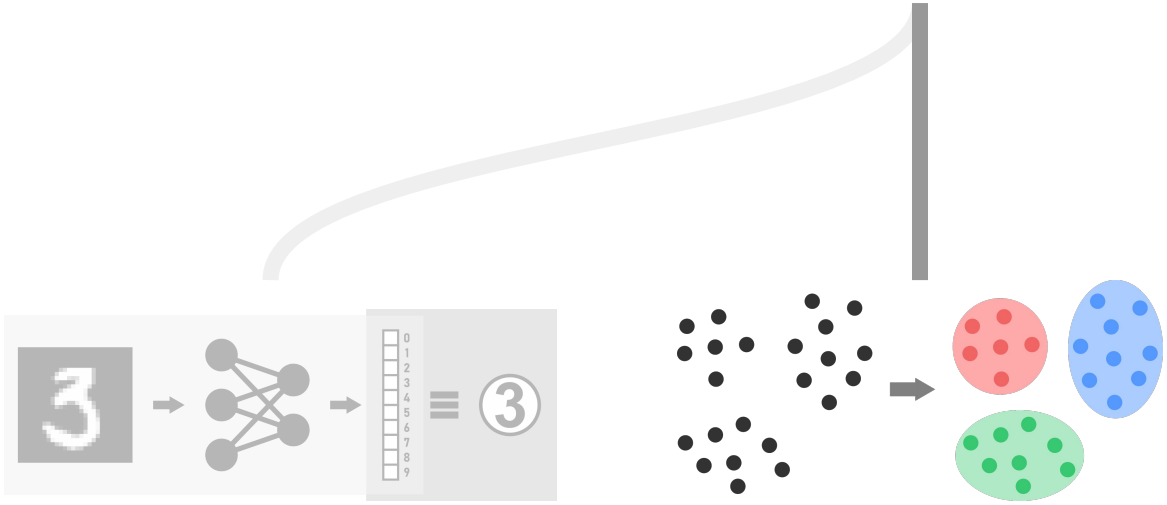
Machine Learning

Machine Learning



Supervised Learning

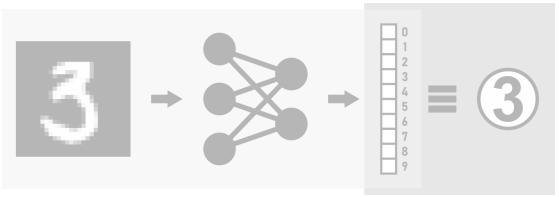
Machine Learning

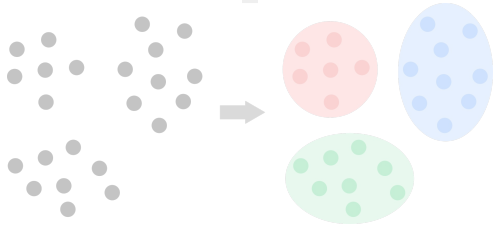


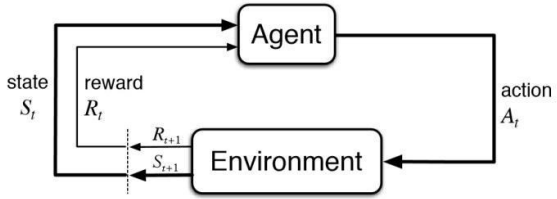
Supervised Learning

Unsupervised Learning

Machine Learning



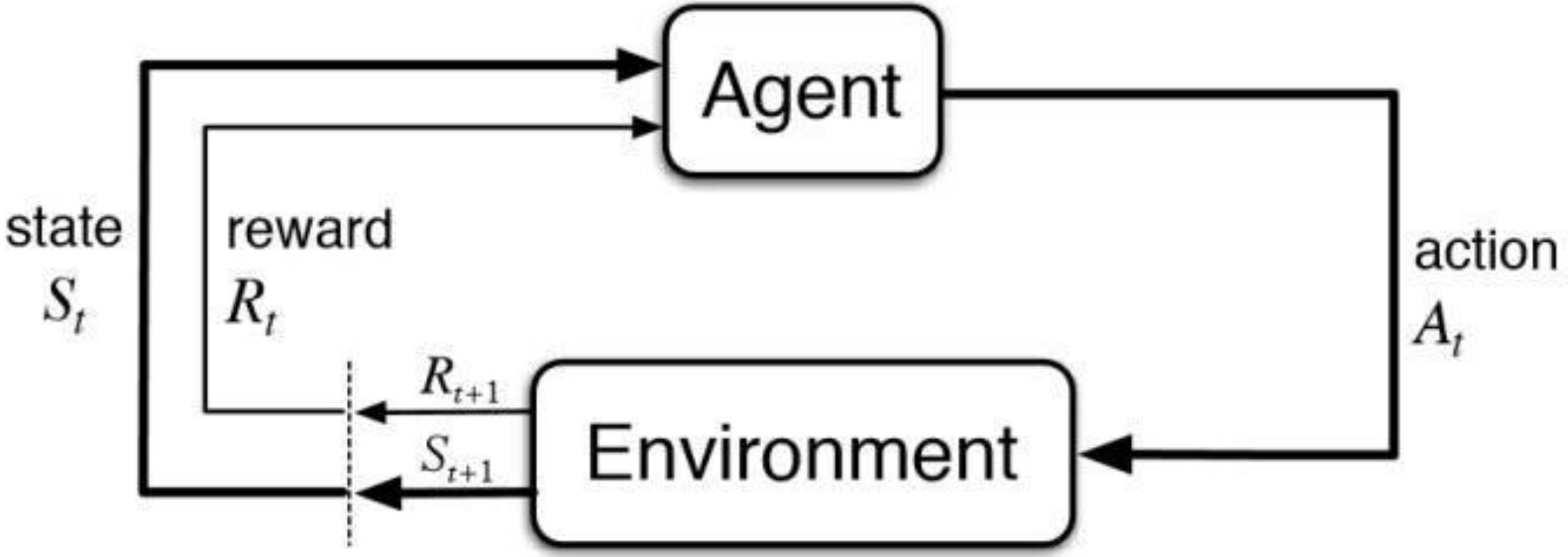




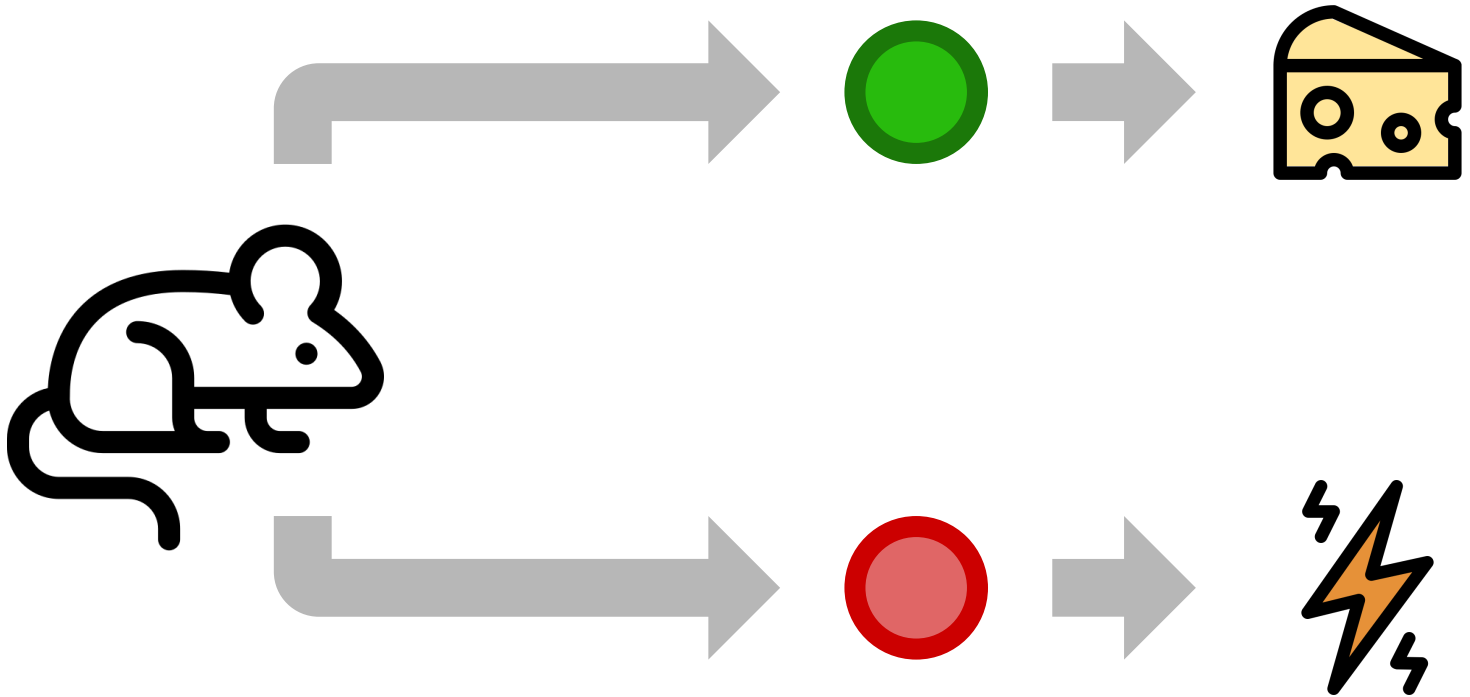
Supervised Learning

Unsupervised Learning

Reinforcement Learning

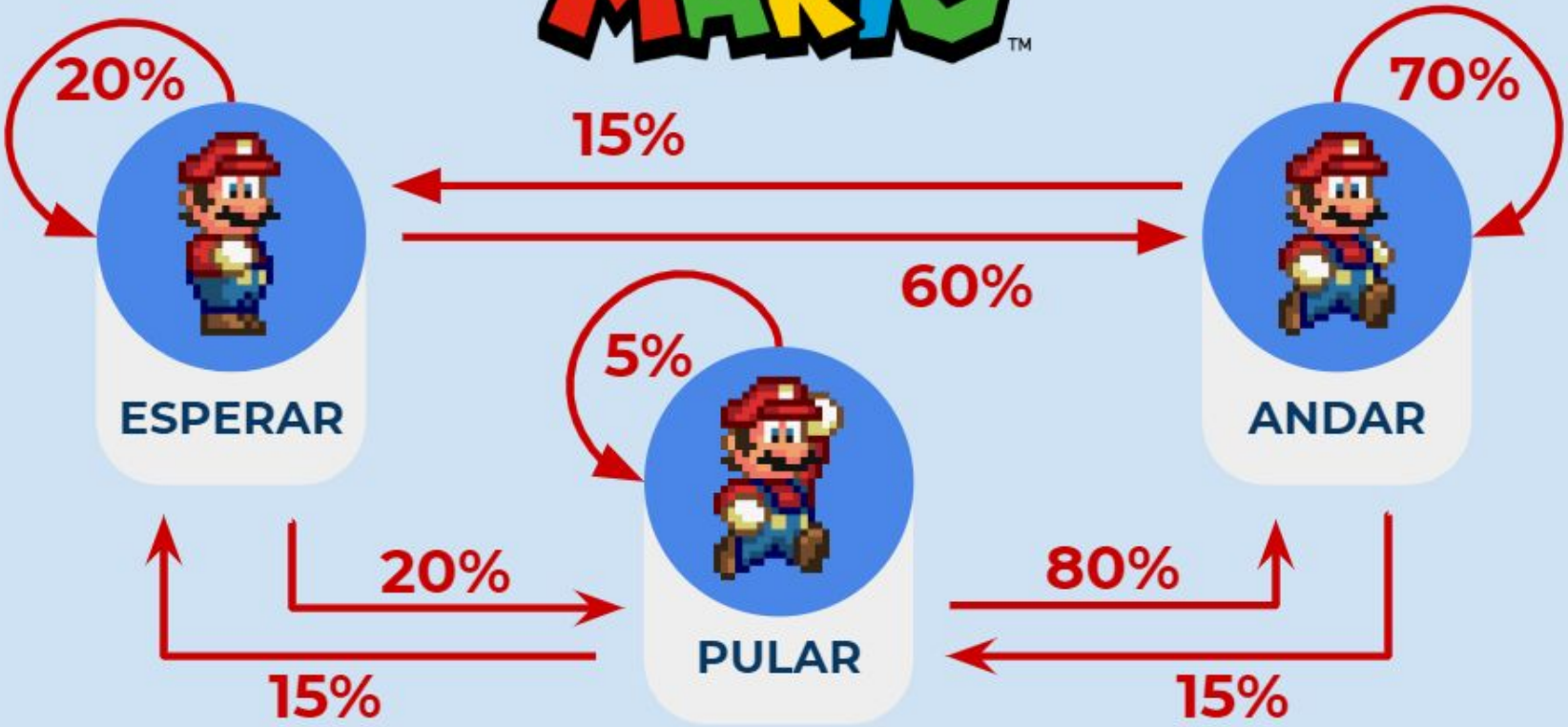


Sutton and Barto. Reinforcement Learning. 2018.



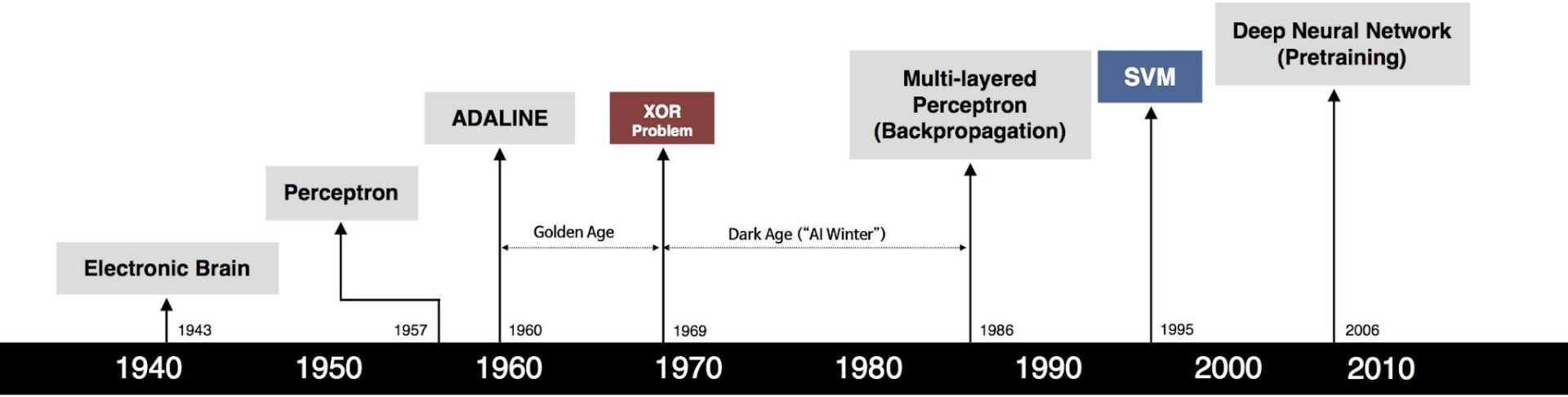
PROCESSO DE MARKOV

SUPER MARIO™



Deep ~~Reinforcement~~ Learning

Artificial Neural Networks Timeline



S. McCulloch – W. Pitts

F. Rosenblatt

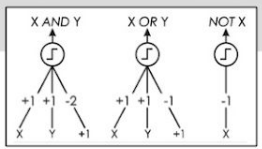
B. Widrow – M. Hoff

M. Minsky – S. Papert

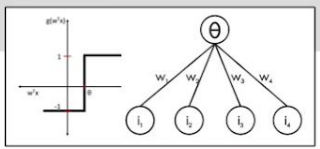
D. Rumelhart – G. Hinton – R. Wiliams

V. Vapnik – C. Cortes

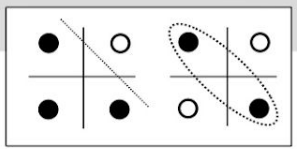
G. Hinton – S. Ruslan



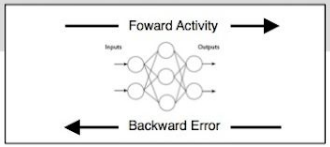
- Adjustable Weights
- Weights are not Learned



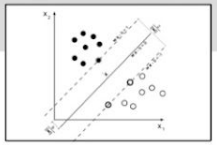
- Learnable Weights and Threshold



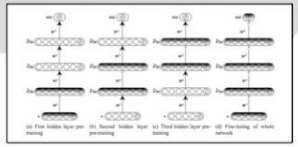
- XOR Problem



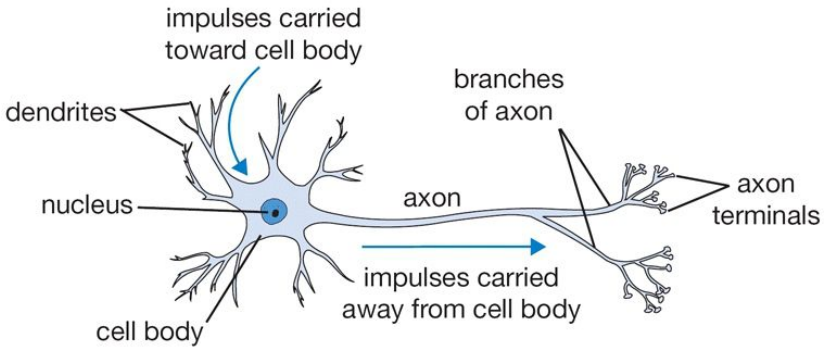
- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting

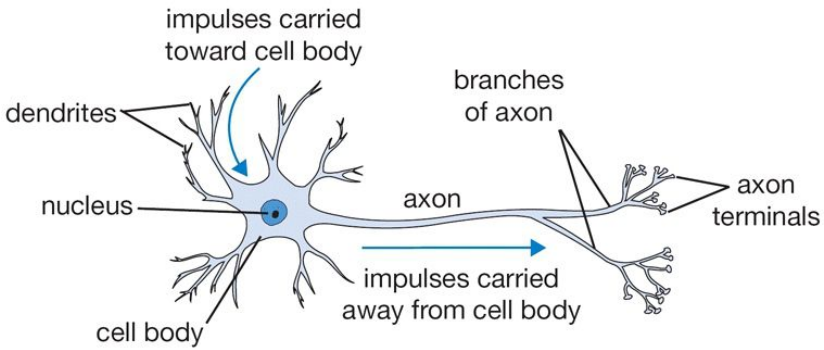


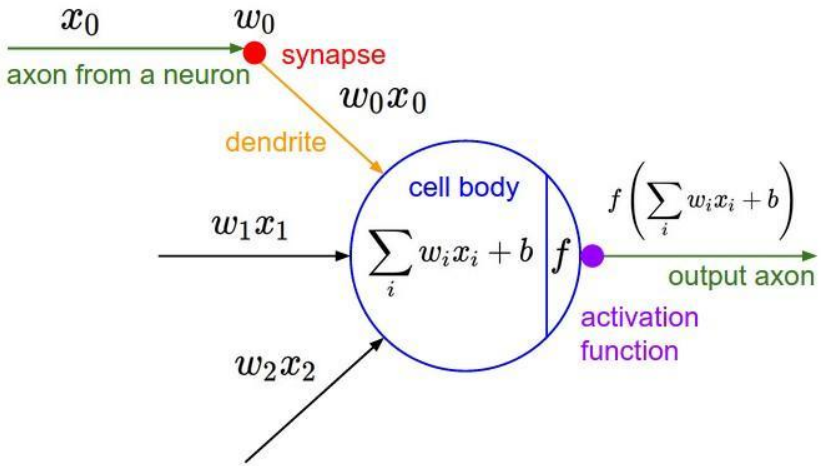
- Limitations of learning prior knowledge
- Kernel function: Human Intervention

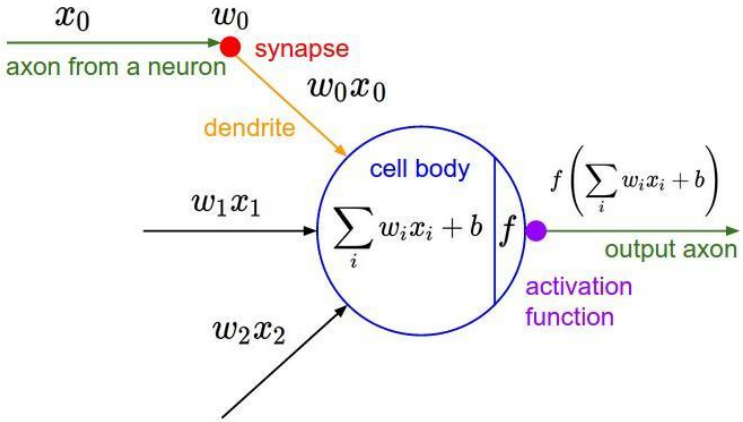


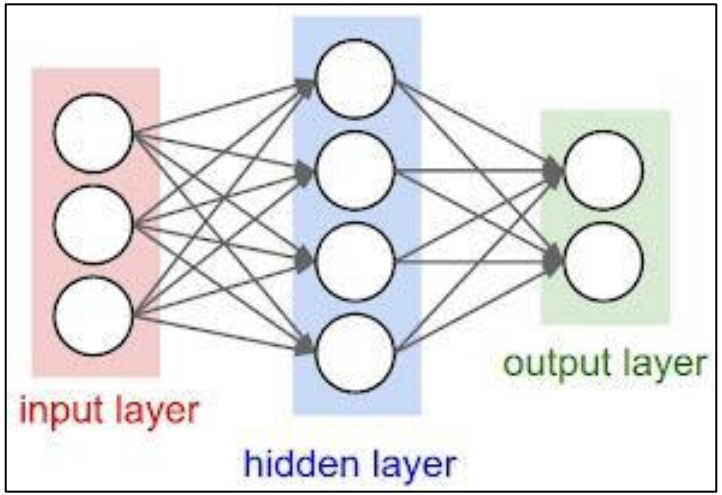
- Hierarchical feature Learning

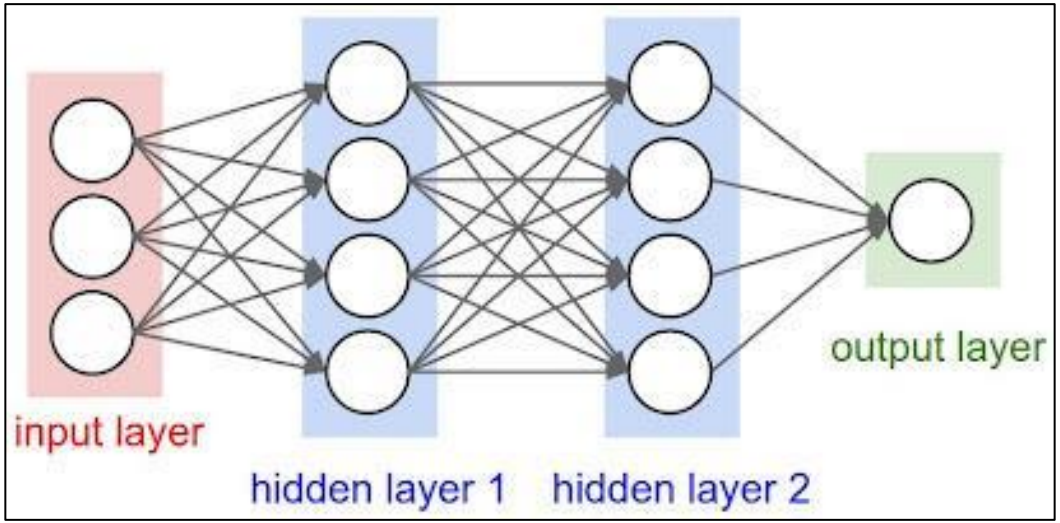

impulses carried
toward cell body
dendrites
branches
of axon
nucleus
axon
axon
terminals
impulses carried
away from cell body
cell body

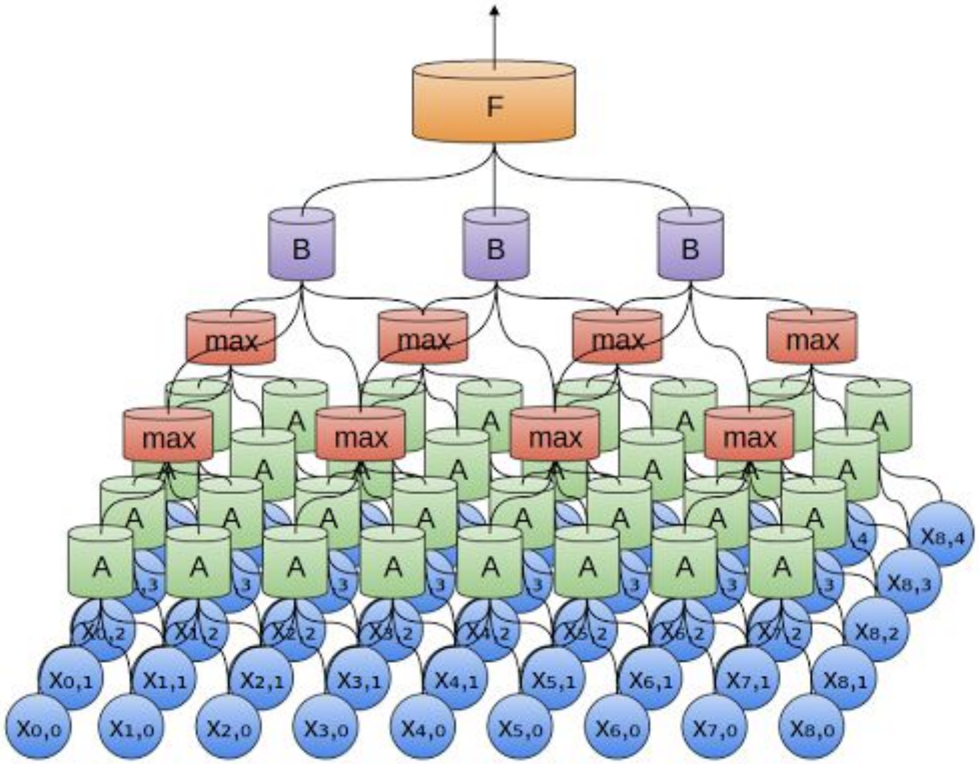

impulses carried
toward cell body
dendrites
nucleus
cell body
axon
branches
of axon
axon
terminals
impulses carried
away from cell body


x_0
w_0
axon from a neuron
synapse
$w_0 x_0$
dendrite
cell body
$w_1 x_1$
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
activation
function
$w_2 x_2$


x_0
w_0
axon from a neuron
synapse
$w_0 x_0$
dendrite
cell body
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
$w_1 x_1$
activation
function
$w_2 x_2$


input layer
hidden layer
output layer


input layer
hidden layer 1
hidden layer 2
output layer



F
B
B
B
max
max
max
max
max
max
max
max
A
X8,4
X8,3
X8,2
X0,1
X1,1
X2,1
X3,1
X4,1
X5,1
X6,1
X7,1
X8,1
X0,0
X1,0
X2,0
X3,0
X4,0
X5,0
X6,0
X7,0
X8,0

“Deep Learning”?

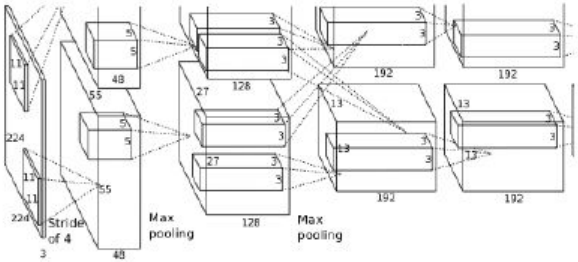
“Deep Learning”?

*“Historically, this was very important in overcoming the belief that these **deep neural networks** were **no good** and could **never be trained**.*

*A friend of mine sent a paper to the International Conference on **Machine Learning**, not that long ago, and the referee said it should not be accepted by ICML, because it was about **neural networks** and it was **not appropriate for ICML**.”*

Geoffrey Hinton - Recent Developments in Deep Learning, 2013
(https://youtu.be/vShMxxqtDDs?t=419)

Deep Learning “recipe”

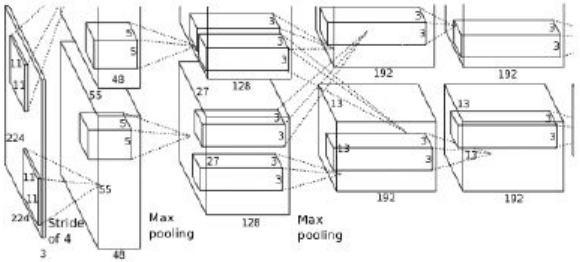


Deep Neural Networks

Deep Learning “recipe”

Big Data

+

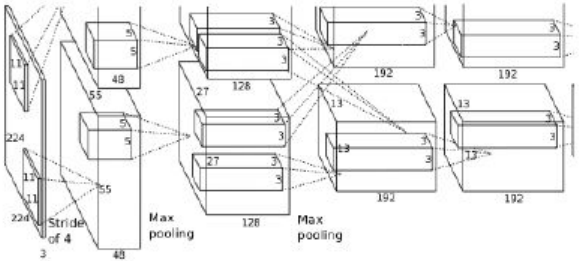


Deep Neural Networks

Deep Learning “recipe”

Big Data

+



Deep Neural Networks

+

Computational Power

Deep Neural Network (Pretraining)

SVM

Multi-layered Perceptron (Backpropagation)

XOR Problem

ADALINE

Perceptron

Electronic Brain

Golden Age

Dark Age ("AI Winter")

1943 1957 1960 1969 1986 1995 2006

1940 1950 1960 1970 1980 1990 2000 2010

S. McCulloch - W. Pitts

F. Rosenblatt

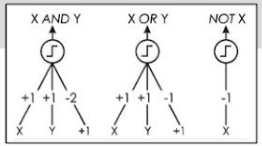
B. Widrow - M. Hoff

M. Minsky - S. Papert

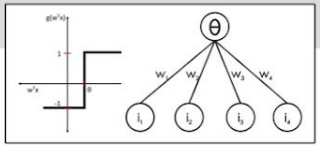
D. Rumelhart - G. Hinton - R. Wiliams

V. Vapnik - C. Cortes

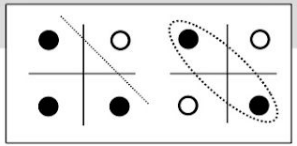
G. Hinton - S. Ruslan



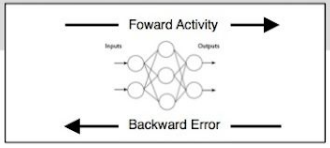
- Adjustable Weights
- Weights are not Learned



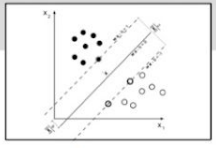
- Learnable Weights and Threshold



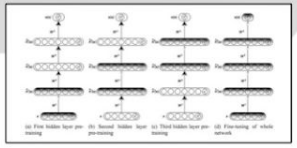
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



- Limitations of learning prior knowledge
- Kernel function: Human Intervention



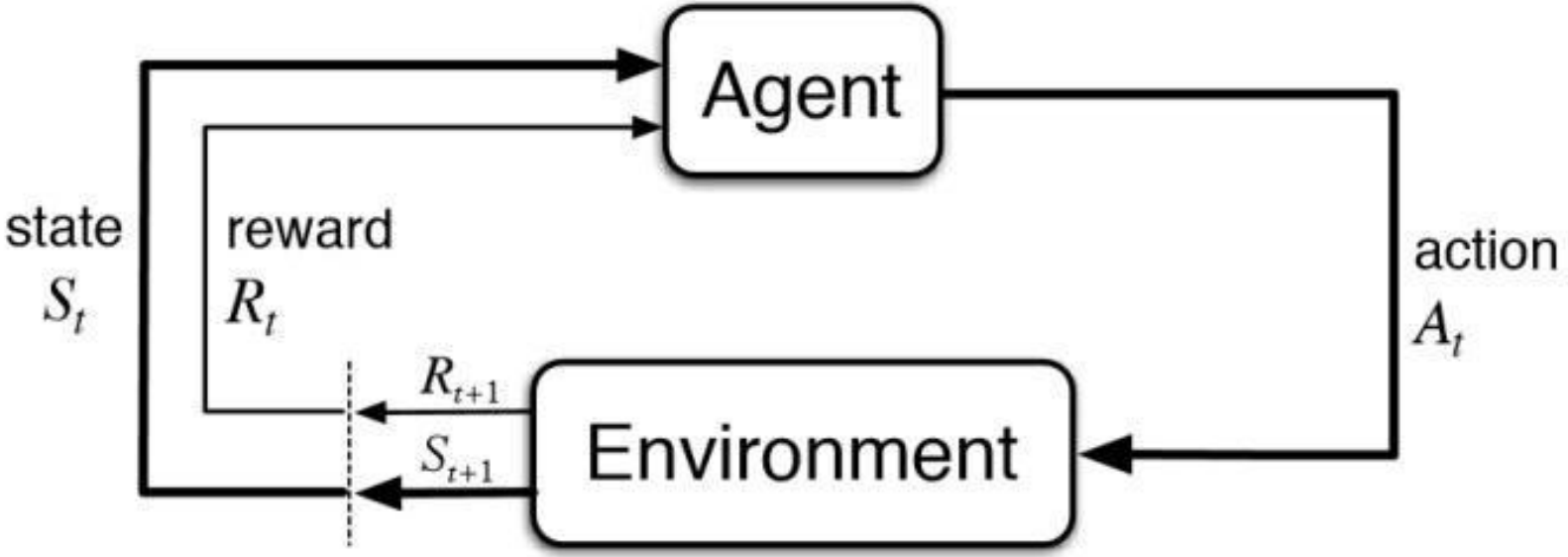
- Hierarchical feature Learning


Yann LeCun

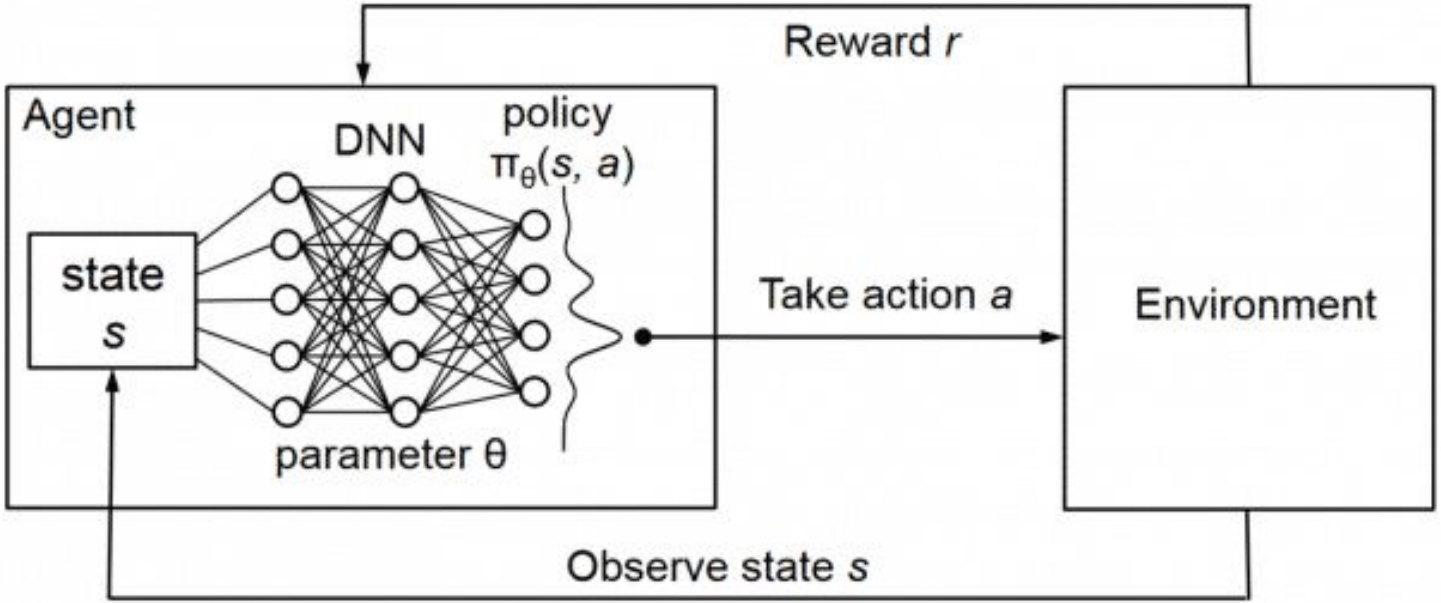

Geoffrey Hinton


Yoshua Bengio

Deep Reinforcement Learning?



Sutton and Barto. Reinforcement Learning. 2018.



http://people.csail.mit.edu/hongzi/content/publications/DeepRM-HotNets16.pdf

Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih **Koray Kavukcuoglu** **David Silver** **Alex Graves** **Ioannis Antonoglou**

Daan Wierstra **Martin Riedmiller**

DeepMind Technologies

{vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com

Abstract

We present the first deep learning model to successfully learn control policies directly from high-dimensional sensory input using reinforcement learning. The model is a convolutional neural network, trained with a variant of Q-learning, whose input is raw pixels and whose output is a value function estimating future rewards. We apply our method to seven Atari 2600 games from the Arcade Learning Environment, with no adjustment of the architecture or learning algorithm. We find that it outperforms all previous approaches on six of the games and surpasses a human expert on three of them.

LETTER

doi:10.1038/nature14236

Human-level control through deep reinforcement learning

Volodymyr Mnih[1]*, Koray Kavukcuoglu[1]*, David Silver[1]*, Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

The theory of reinforcement learning provides a normative account[1], deeply rooted in psychological[2] and neuroscientific[3] perspectives on animal behaviour, of how agents may optimize their control of an environment. To use reinforcement learning successfully in situations approaching real-world complexity, however, agents are confronted with a difficult task: they must derive efficient representations of the environment from high-dimensional sensory inputs, and use these to generalize past experience to new situations. Remarkably, humans and other animals seem to solve this problem through a harmonious combination of reinforcement learning and hierarchical sensory processing systems[4,5], the former evidenced by a wealth of neural data revealing notable parallels between the phasic signals emitted by dopaminergic neurons and temporal difference reinforcement learning algorithms[3]. While reinforcement learning agents have achieved some successes in a variety of domains[6–8], their applicability has previously been limited to domains in which useful features can be handcrafted, or to domains with fully observed, low-dimensional state spaces. Here we use recent advances in training deep neural networks[9–11] to

agent is to select actions in a fashion that maximizes cumulative future reward. More formally, we use a deep convolutional neural network to approximate the optimal action-value function

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \mid s_t = s,\, a_t = a,\, \pi\right],$$

which is the maximum sum of rewards r_t discounted by γ at each time-step t, achievable by a behaviour policy $\pi = P(a|s)$, after making an observation (s) and taking an action (a) (see Methods)[19].

Reinforcement learning is known to be unstable or even to diverge when a nonlinear function approximator such as a neural network is used to represent the action-value (also known as Q) function[20]. This instability has several causes: the correlations present in the sequence of observations, the fact that small updates to Q may significantly change the policy and therefore change the data distribution, and the correlations between the action-values (Q) and the target values $r + \gamma \max_{a'} Q(s', a')$. We address these instabilities with a novel variant of Q-learning, which uses two key ideas. First, we used a biologically inspired mechanism termed experience replay[21–23] that randomizes over the data, thereby

https://youtu.be/TmPfTpjtdgg

AlphaGo *4* vs *1* Lee Sedol

ALPHAGO

TRIBECA FILM FESTIVAL 2017

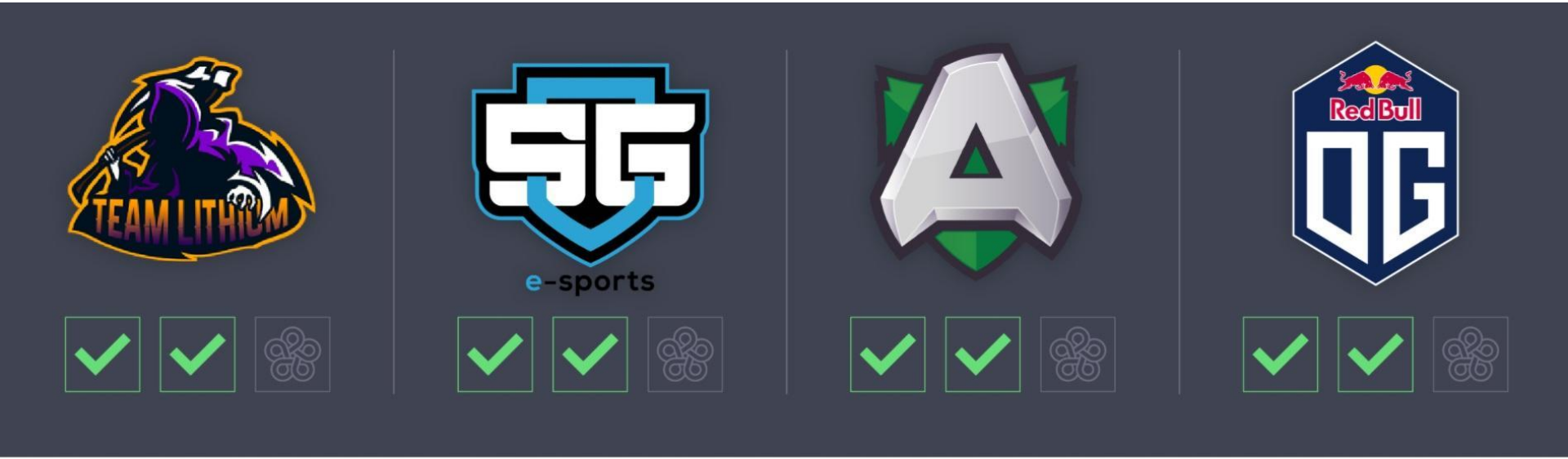
BFI LONDON FILM FESTIVAL OFFICIAL SELECTION 2017

WINNER TRAVERSE CITY FILM FESTIVAL 2017

OFFICIAL SELECTION MILL VALLEY FILM FESTIVAL 2017

Official Selection WARSAW FILM FESTIVAL

VIFF Official Selection 2017



OpenAI Five's record versus semi-pro team Lithium and pro teams SG esports, Alliance, and OG since our losses at The International.

APRIL 15, 2019 • 7 MINUTE READ

OpenAI Five Defeats Dota 2 World Champions

OpenAI Five is the first AI to beat the world champions in an esports game, having won two back-to-back games versus the world champion Dota 2 team, OG, at Finals this weekend. Both OpenAI Five and DeepMind's AlphaStar had previously beaten good pros privately but lost their live pro matches, making this also the first time an AI has beaten esports pros

BLOG POST
RESEARCH

24 JAN 2019

AlphaStar: Mastering the Real-Time Strategy Game StarCraft II

DeepMind | StarCraft II

DEMONSTRATION

GRZEGORZ 'MANA' KOMINCZ

Before the match:

"I'm hoping for a 5-0, not to lose any games, but I think the realistic goal would be 4-1 in my favor." [2]

BLIZZARD ENTERTAINMENT

DeepMind | StarCraft II

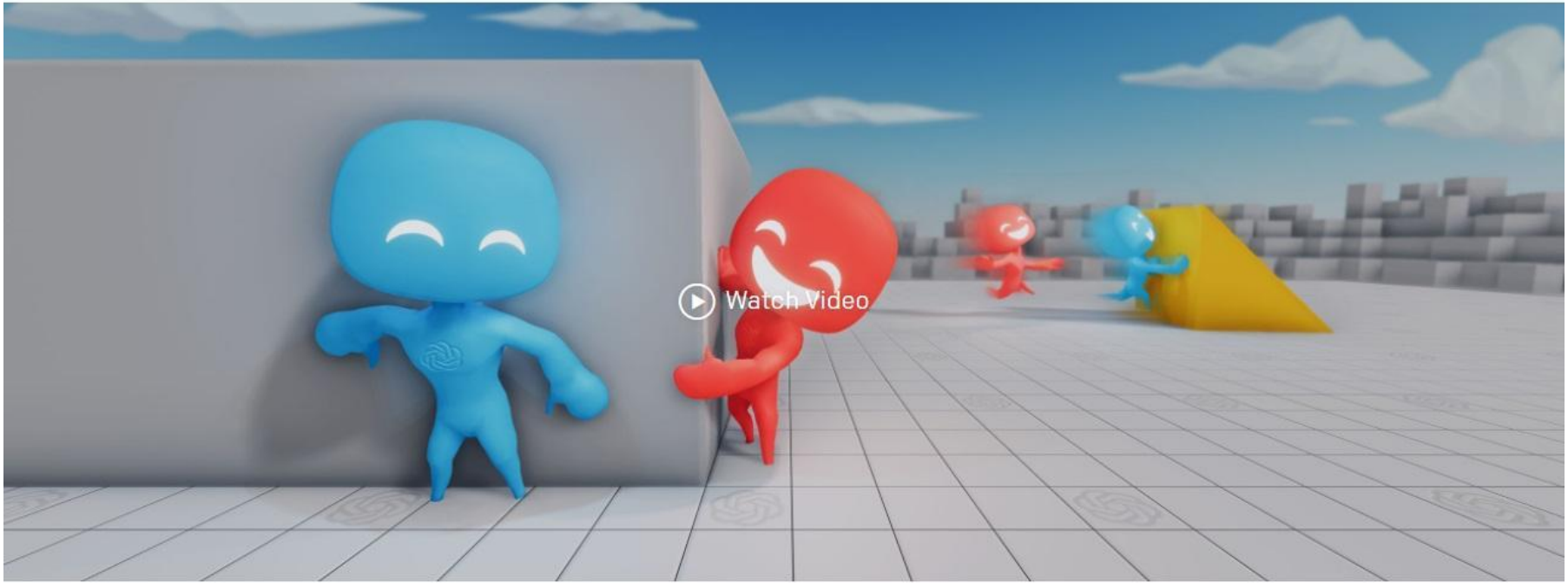
DEMONSTRATION

GRZEGORZ 'MANA' KOMINCZ

ROUND	REPLAY
1.	ALPHASTAR WINS
2.	ALPHASTAR WINS
3.	ALPHASTAR WINS
4.	ALPHASTAR WINS
5.	ALPHASTAR WINS

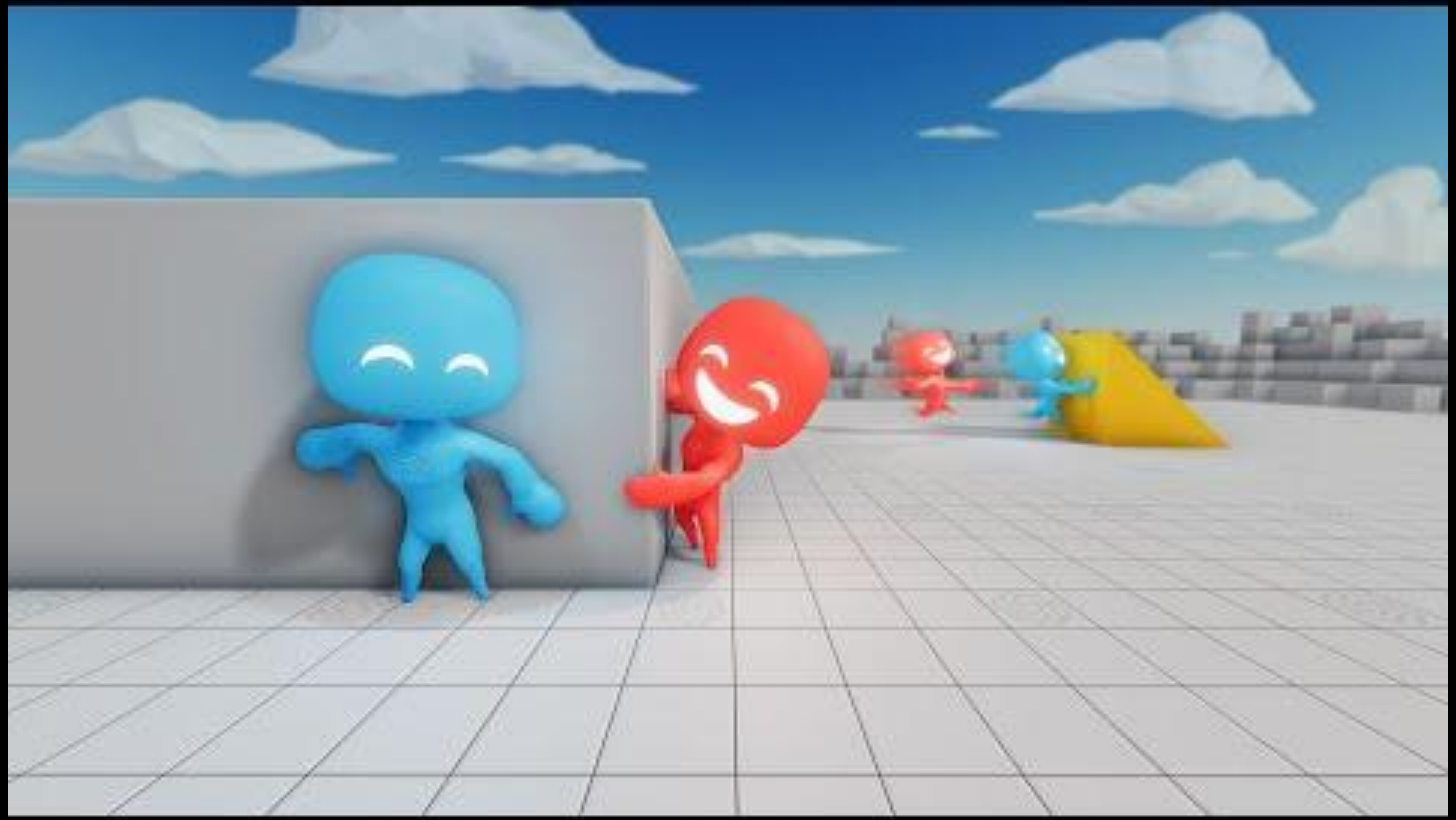
SCORE MANA 0 - 5 ALPHASTAR

BLIZZARD ENTERTAINMENT



SEPTEMBER 17, 2019 • 9 MINUTE READ

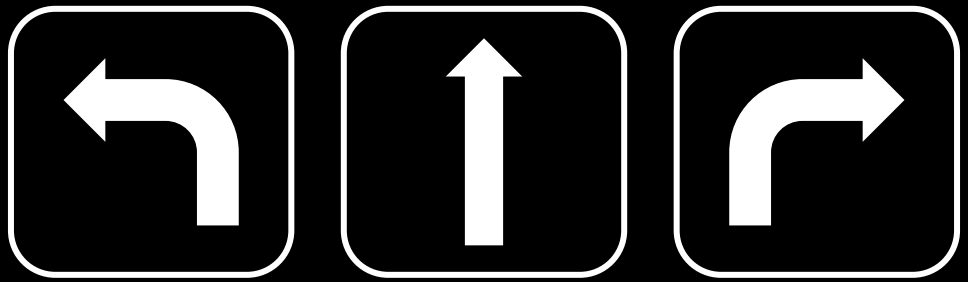
Emergent Tool Use from Multi-Agent Interaction

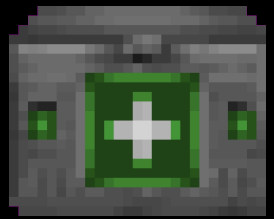


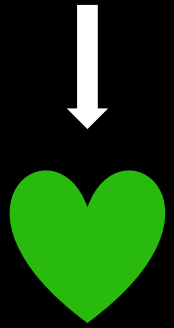
Foraging and Deep Reinforcement Learning


60%
AMMO
HEALTH
2 3 4
5 6 7
ARMS
0%
ARMOR
0 / 200
0 / 50
0 / 50
0 / 300


60%
AMMO
HEALTH
ARMS
0%
ARMOR
0 / 200
0 / 50
0 / 50
0 / 300








60%
AMMO
HEALTH
ARMS
0%
ARMOR
200
50
50
300

On the Development of an Autonomous Agent for a 3D First-Person Shooter Game Using Deep Reinforcement Learning

Paulo B. S. Serafim* Yuri L. B. Nogueira Creto A. Vidal Joaquim B. Cavalcante-Neto

Federal University of Ceará (UFC), Department of Computing, Brazil

Figure 1: The three ViZDoom's scenarios played by the agent. **Left**: Basic. **Center**: Defend the Line. **Right**: Medikits and Poisons.

ABSTRACT

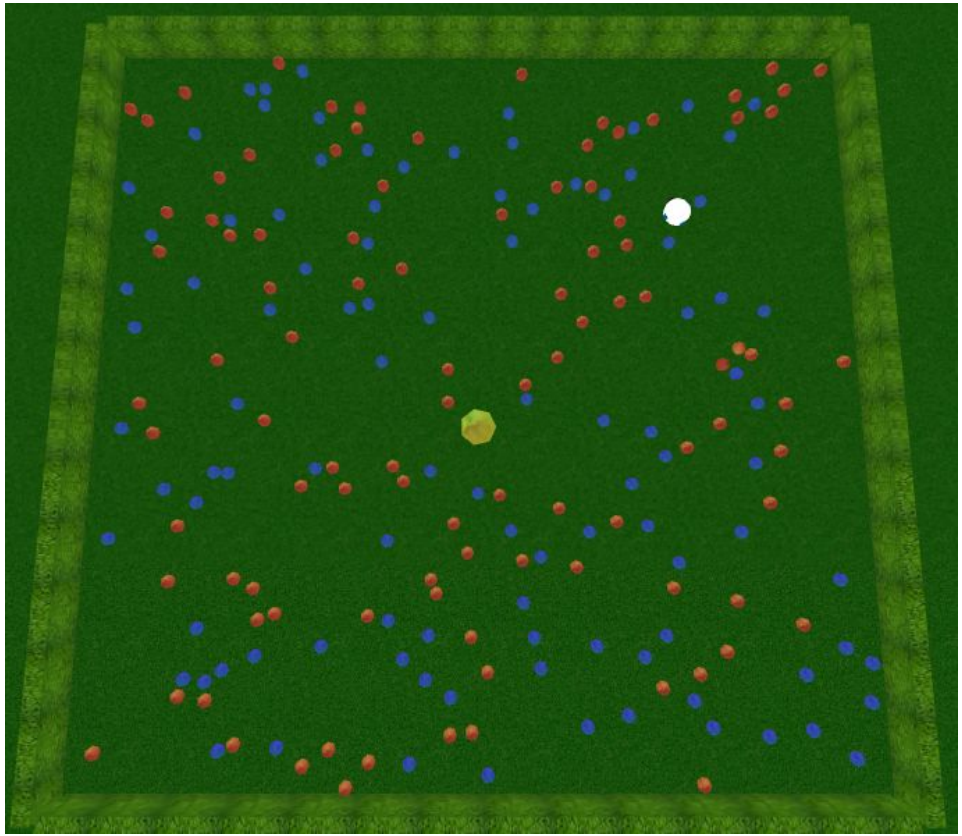
First-Person Shooter games have always been very popular. One of the challenges in the development of First-Person Shooter games is the use of game agents controlled by Artificial Intelligence because they can learn how to handle very distinct situations presented to them. In this work, we construct an autonomous agent to play different scenarios in a 3D First-Person Shooter game using a Deep Neural Network model. The agent receives as input only the pixels of the screen and should learn how to interact with the environments by itself. To achieve this goal, the agent is trained using a Deep Reinforcement Learning model through an adaptation of the Q-Learning technique for Deep Networks. We evaluate our agent in three distinct scenarios: a basic environment against one static enemy, a more complex environment against multiple different enemies and a custom medikit gathering scenario. We show that the agent achieves good results and learns complex behaviors in all tested environments. The results show that the presented model is suitable for creating 3D First-Person Shooter autonomous agents capable of playing different scenarios.

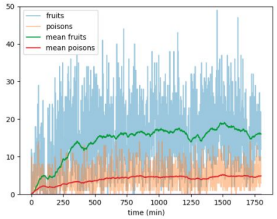
Keywords: 3D first-person shooter, autonomous agent, reinforcement learning, deep neural networks.

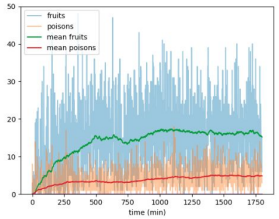
are becoming more easy to make, they still need to have all behaviors explicitly created by the programmers.

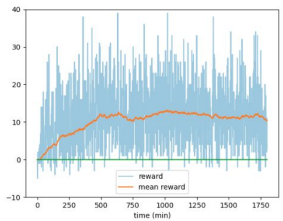
To solve this problem, the creation of an autonomous agent that learns how to behave in different environments is a desired objective. However, creating an autonomous agent to play a complex game like First-Person Shooters is very hard. Such agent must be capable of taking a wide range of actions, like exploring your surroundings, aiming and shooting enemies, picking up items and surviving as long as possible. Ideally, the agent should be capable of learning complex behaviors with the minimum number of information given to him directly, i.e., scripted.

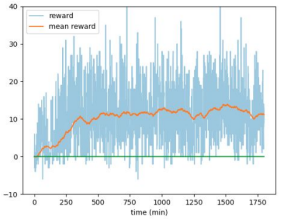
One desired characteristic is that the input received by the agent is the current screen view. Using only the pixels of the screen guarantees input fairness, which means that the input is the same for every player, even if he is a human player. However, the construction of an autonomous virtual player that receives as input only raw pixels is a very difficult task. Deep Learning [11] is a technique which can handle this type of problem. Since its development, [7] it was extended to solve different tasks. Recently, it is also being used to model autonomous game agents that receive as input only the pixels of the screen [15]. Thus, the creation of an autonomous game character that learns complex behaviors by himself begins to

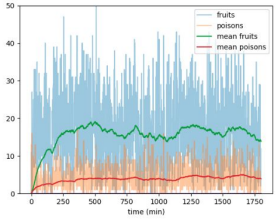


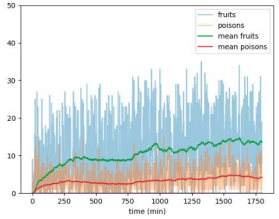

fruits
poisons
mean fruits
mean poisons
time (min)

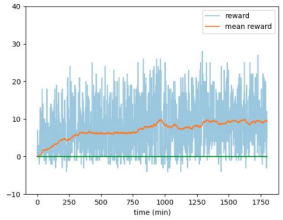

fruits
poisons
mean fruits
mean poisons
time (min)

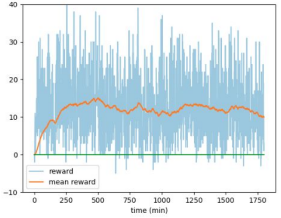

reward
mean reward
time (min)

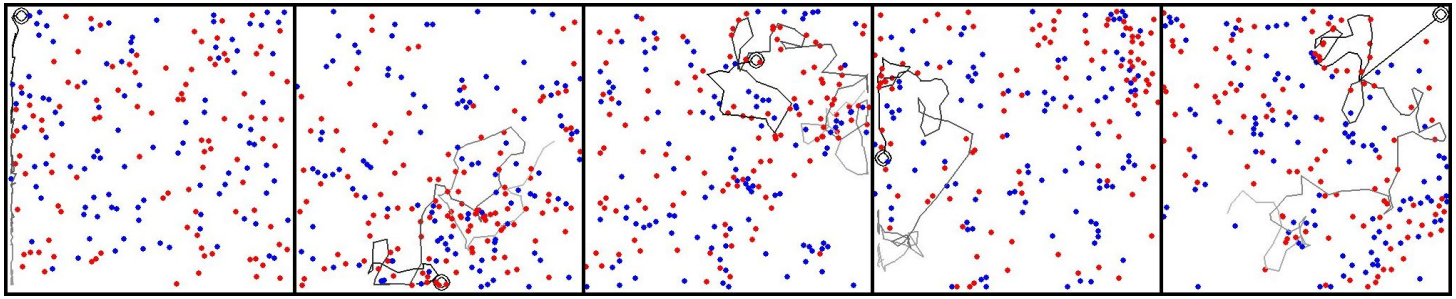

reward
mean reward
time (min)

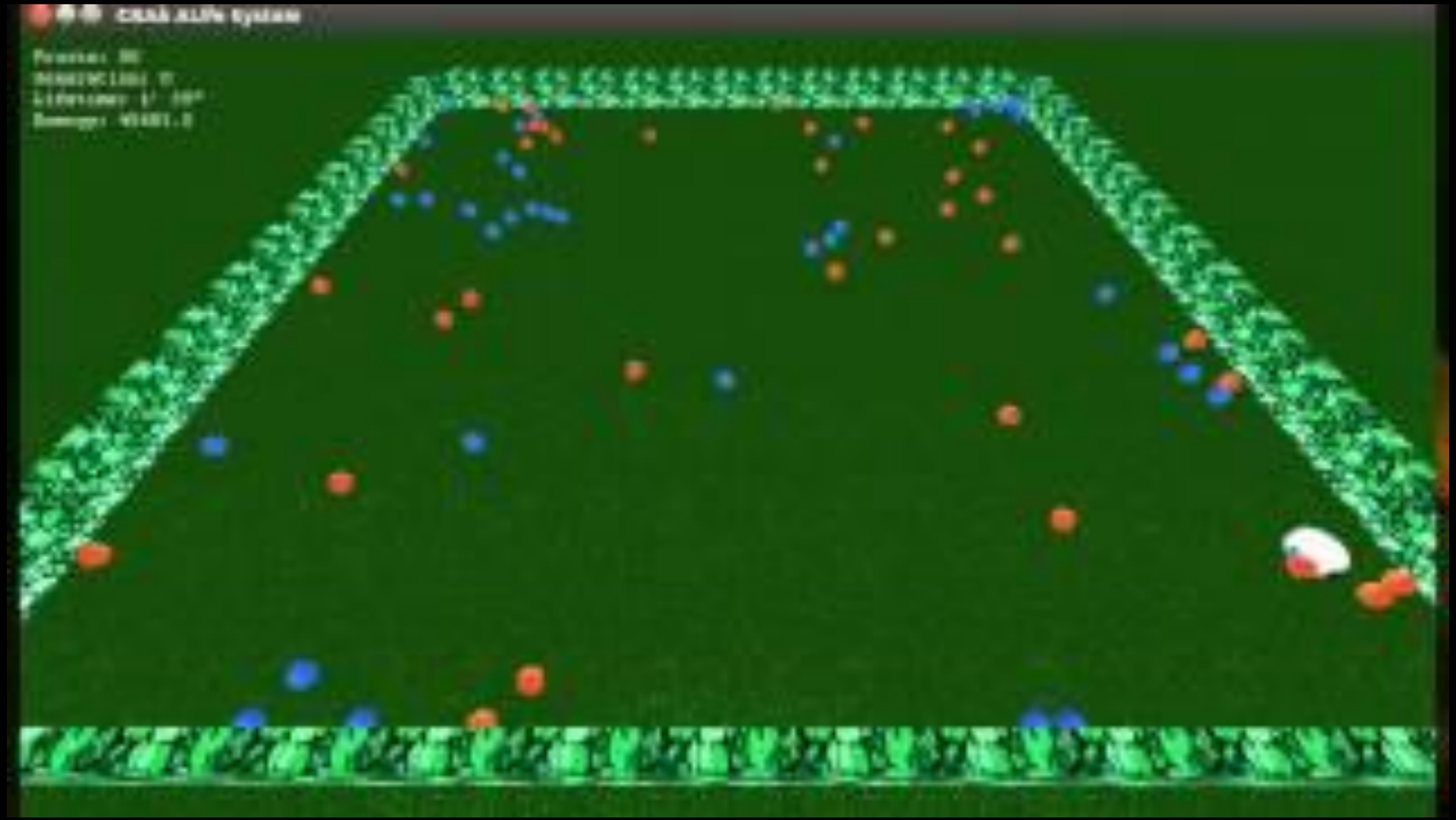

fruits
poisons
mean fruits
mean poisons
time (min)


fruits
poisons
mean fruits
mean poisons
time (min)


reward
mean reward
time (min)


reward
mean reward
time (min)





Autonomous Foraging with SARSA-based Deep Reinforcement Learning

Anderson Mesquita, Yuri Nogueira, Creto Vidal, Joaquim Cavalcante-Neto
Department of Computing (DC)
Federal University of Ceará (UFC)
Fortaleza, Brazil
andersonmesquita@alu.ufc.br; {yuri, cvidal, joaquimb}@dc.ufc.br

Paulo Serafim
Instituto Atlântico
Fortaleza, Brazil
paulo_serafim@atlantico.com.br

***Abstract*—This work proposes a system capable of autonomous behavior using Computer Vision and Deep Reinforcement Learning. These learned behaviors are those related to foraging in an environment that has food and poisons distributed throughout a scenario. We apply the Deep Learning framework for color image processing. These images simulate the agent's vision. The foraging task is modeled as a reinforcement learning problem, in which an input constituted by raw pixels is processed by a convolutional neural network resulting in a set of actions. A Deep Learning algorithm based on SARSA was used. During training, the agent selects the actions based on a probability distribution called Softmax. The objective of this work is to present an agent capable of searching for food and distinguishing it from poisons through continuous learning and without the help or external intervention from humans. The experiments show that the agent is able to distinguish food from poisons without hints or markings in its vision input. This highlights the advantages of combining Deep Learning with reinforcement learning for the foraging problem. The results of this work form an initial basis for understanding the relationship among autonomy, cognition, and perception in artificial agents.**

***Index Terms*—autonomy, deep learning, reinforcement learning, computer vision**

to the concept of surviving in the environment and avoiding precariousness conditions. Thus, the autonomous agent must make decisions in order to maintain its own organization and constitution. In engineering, an autonomous agent is associated with controlling systems capable of carrying out tasks without the intervention of a human being [4]. When an autonomous agent interacts with the environment, it learns a set of behaviors to solve a given problem, which is predefined externally or generated as an internal objective [5] . Finding this set of behaviors and select those most useful is a challenging task, especially when the environment is not stationary and the agent's sensors have high dimensionality, such as a vision sensor.

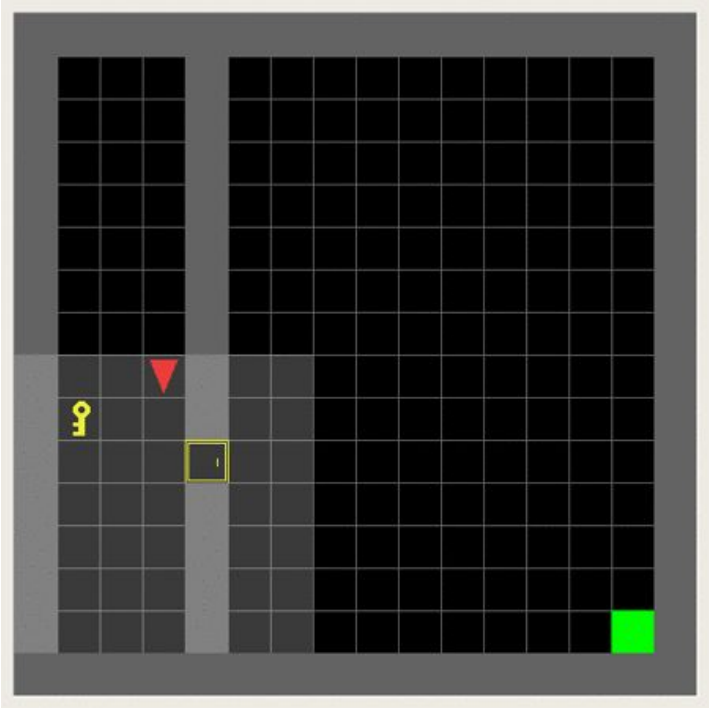
However, works that used the Deep Learning framework [6] contributed to the creation of autonomous agents to solve diverse tasks [7] and presented positive results in respect to agents being able to abstract elements of the environment with high information without any help or external intervention. Nonetheless, many pioneering Deep Learning applications have been directed to tasks of classification and identification

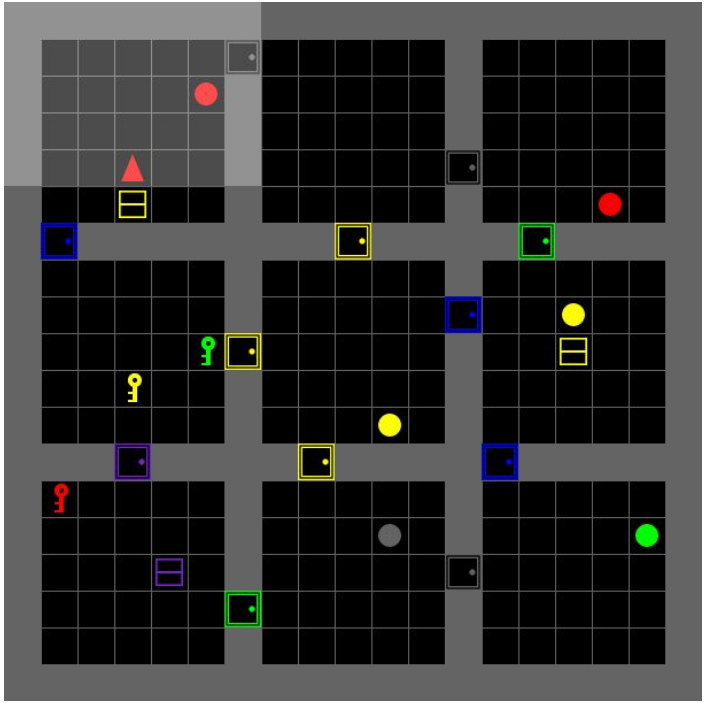
Multi-Sensory Agents

Work In Progress!
(sorry for the quality 😅)

Reinforcement Learning Environments

Minigrid

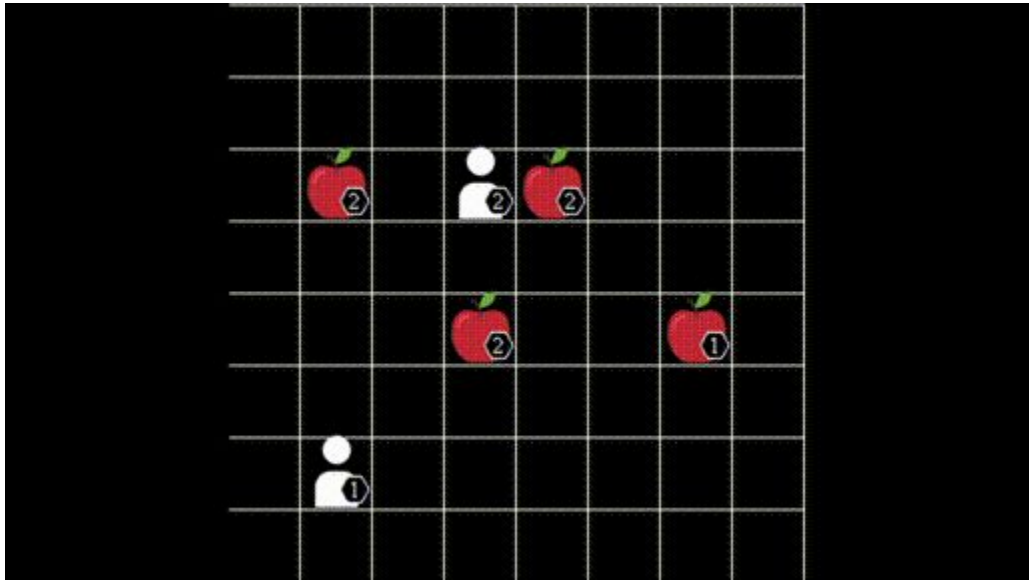


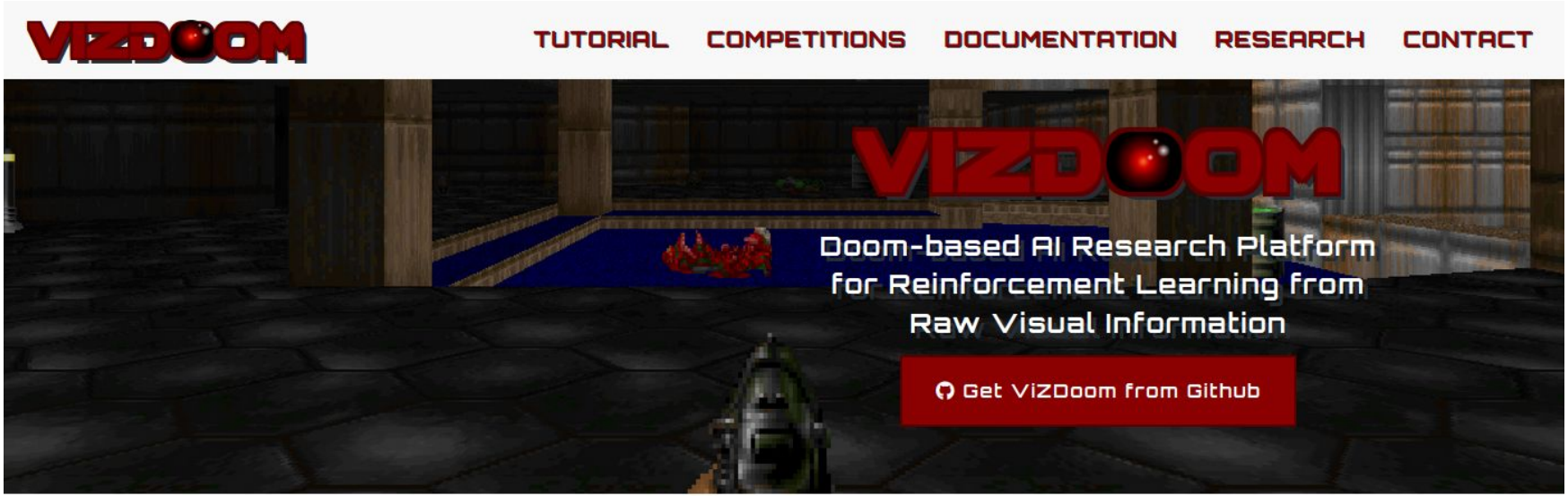


LBF

Level-based Foraging

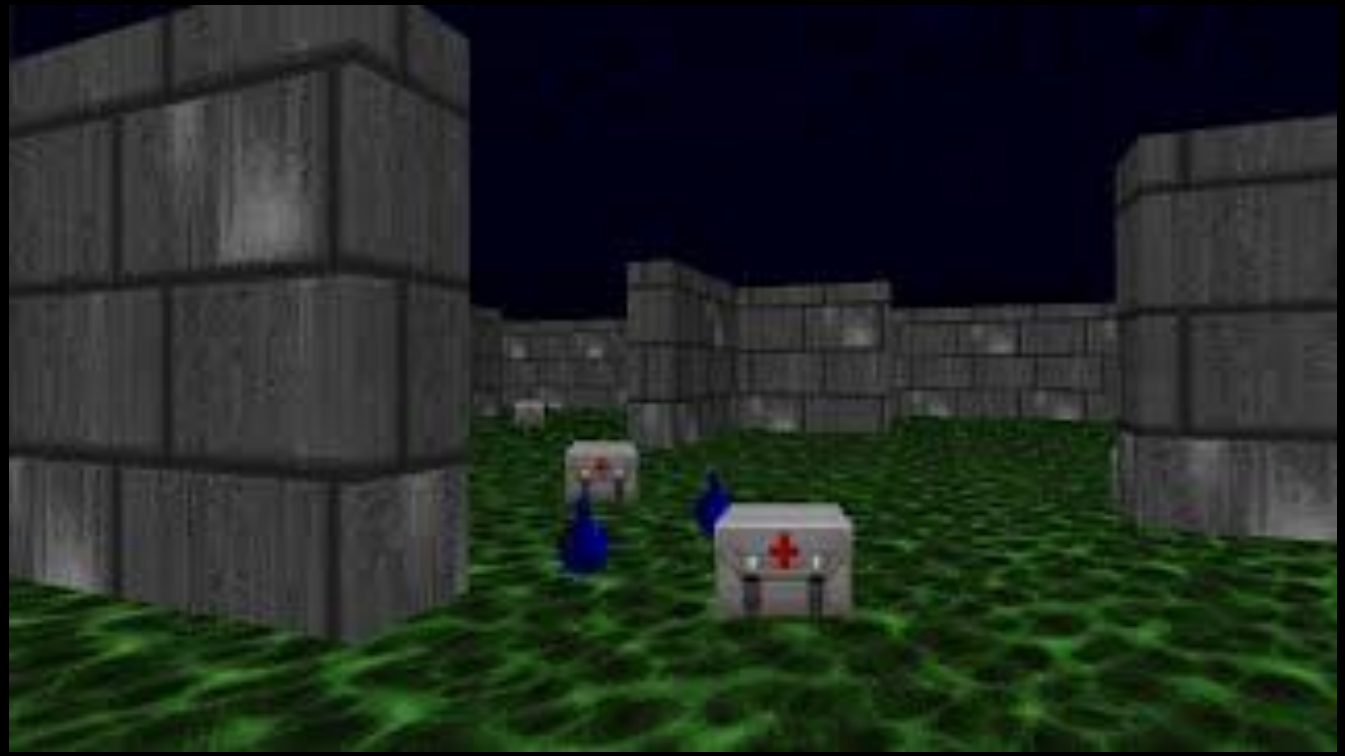
A multi-agent reinforcement learning environment





VIZDOOM allows developing AI bots that play **DOOM** using visual information (the screen buffer). It is primarily intended for research in machine visual learning, and deep reinforcement learning, in particular.

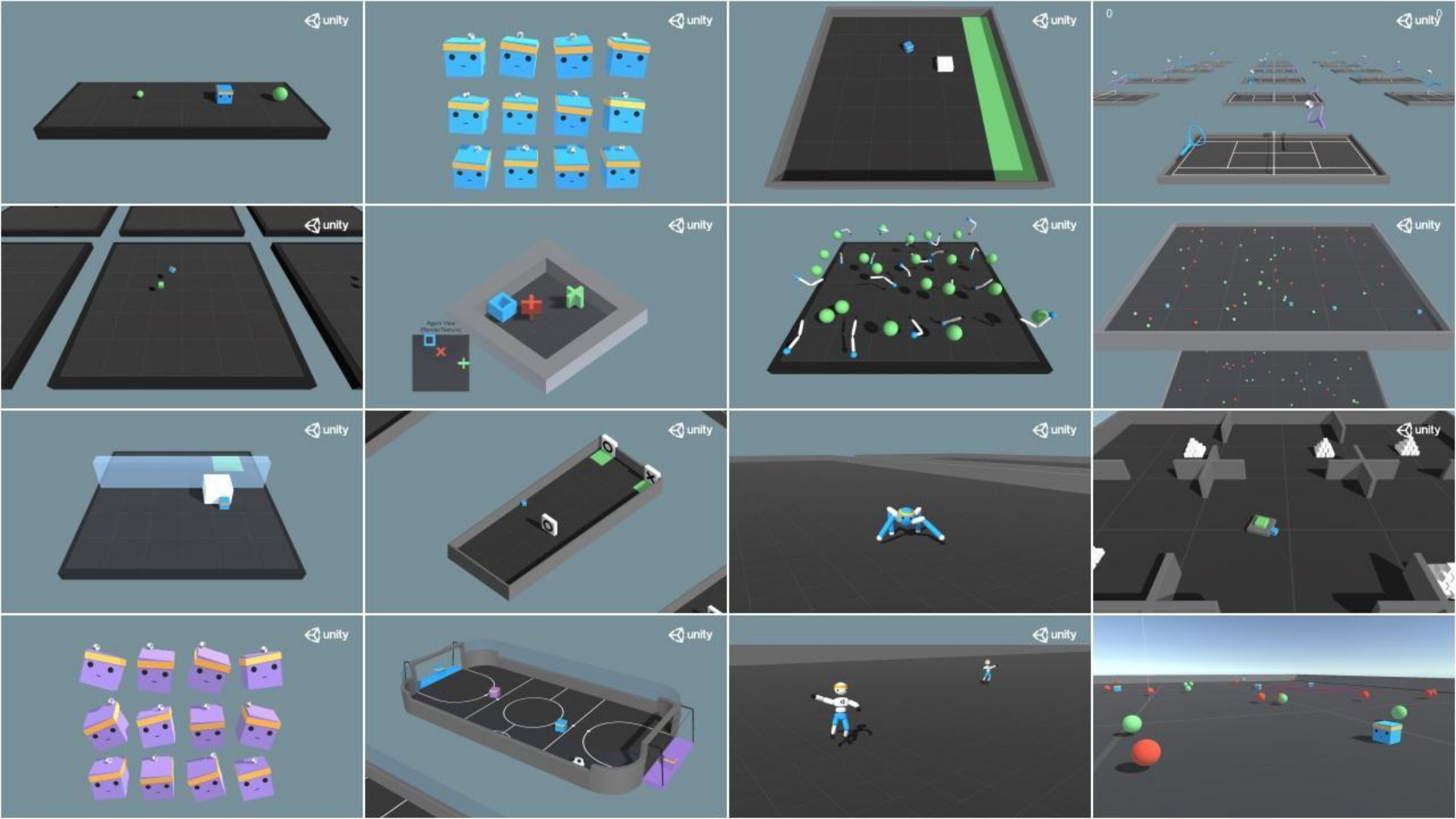
https://youtu.be/re6hkcTWVUY

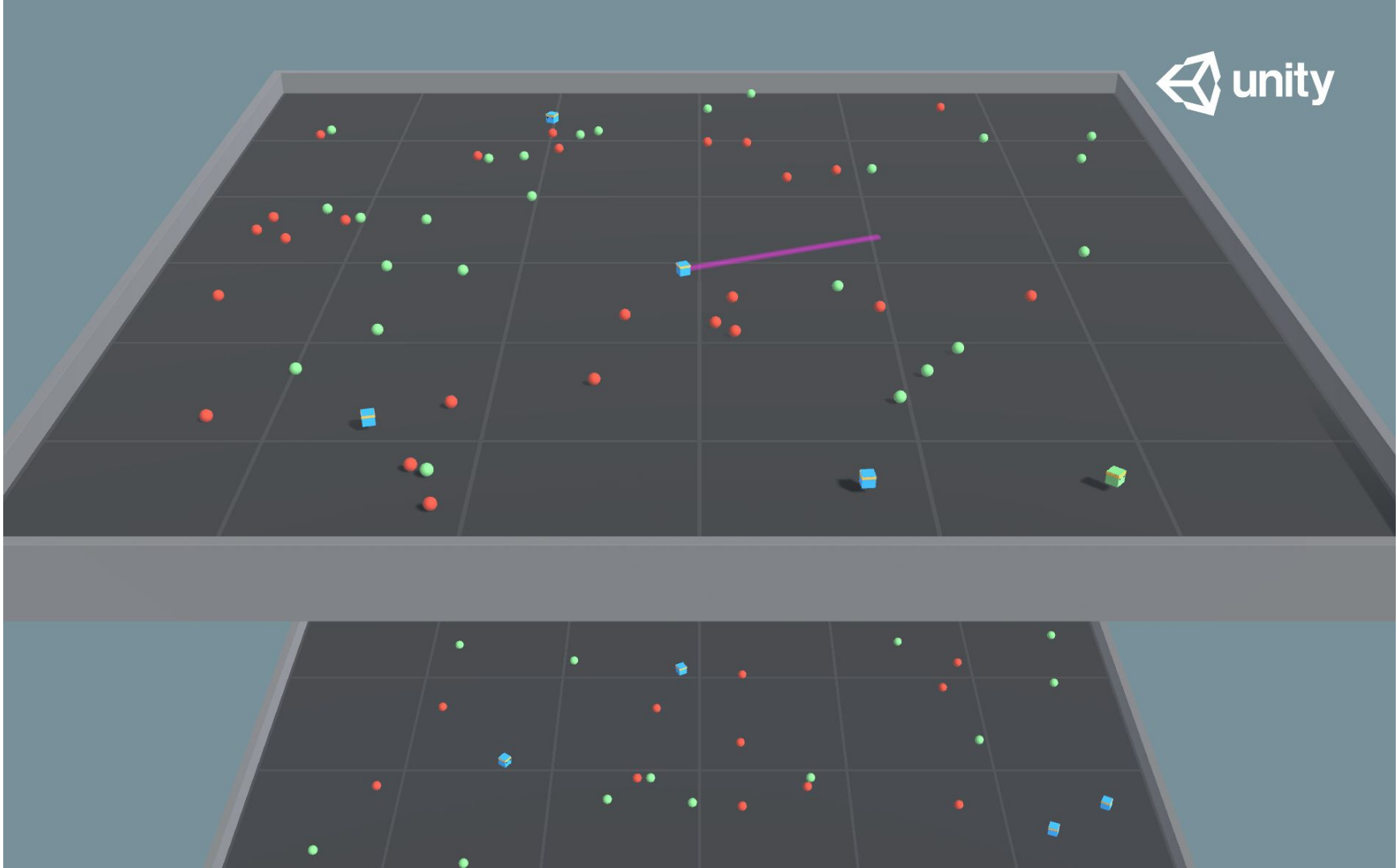


UNITY MACHINE LEARNING AGENTS

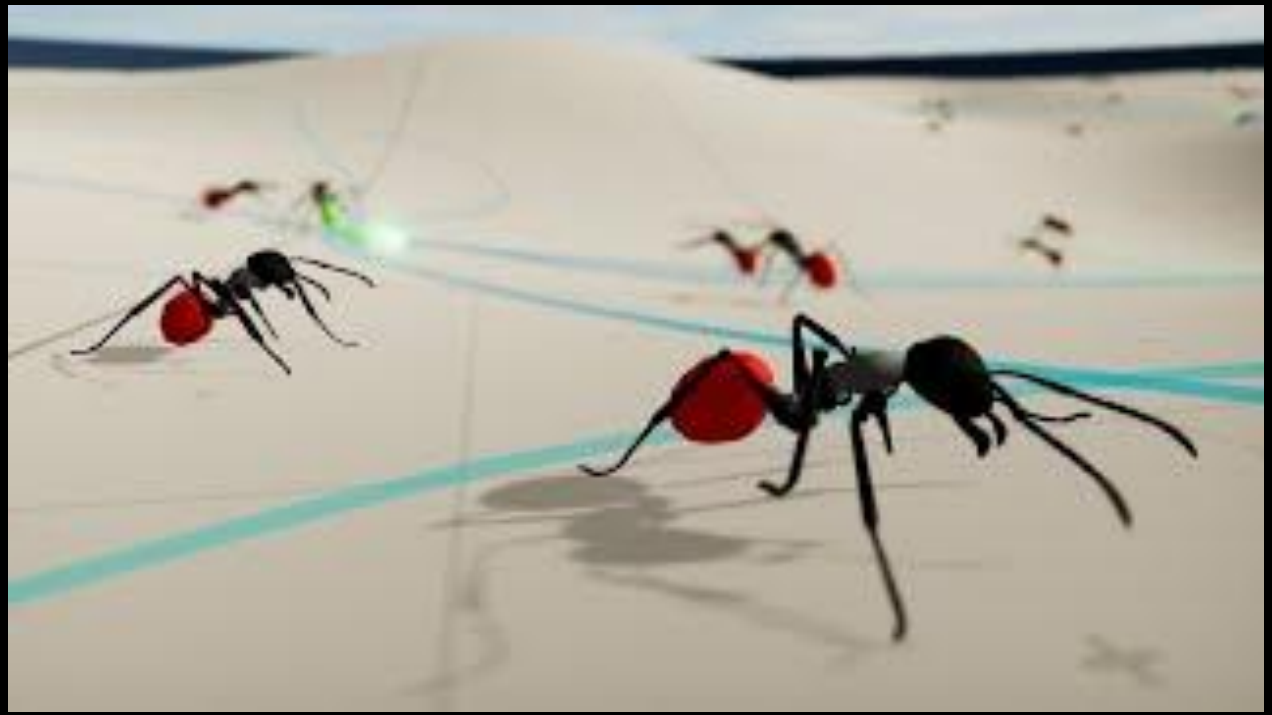
Train and embed intelligent agents by leveraging state-of-the-art deep learning technology.

Download from Github

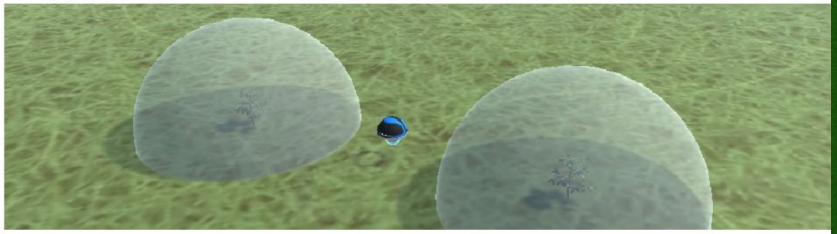


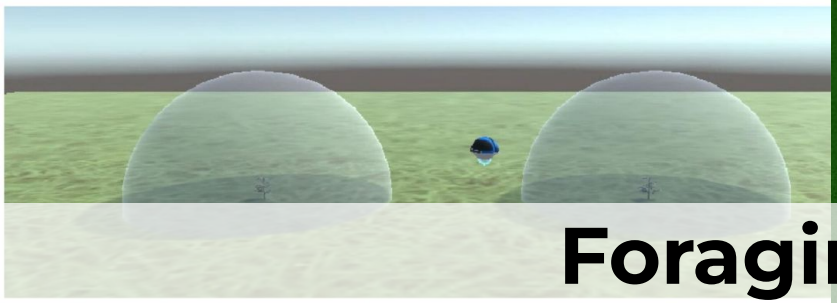

unity

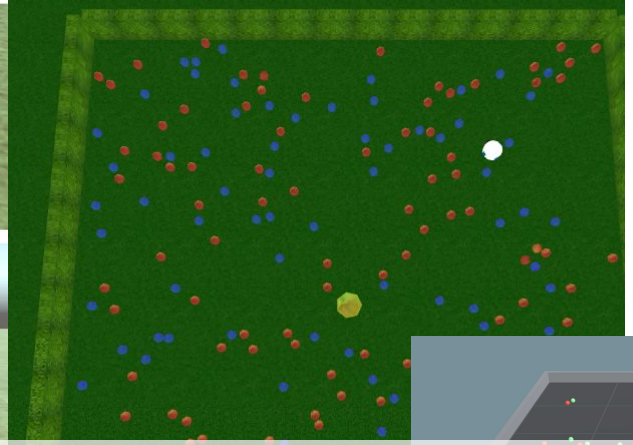
https://youtu.be/EwB8XXCYOsc

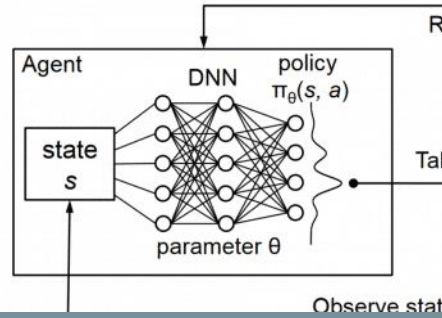


Foraging and Deep Reinforcement Learning

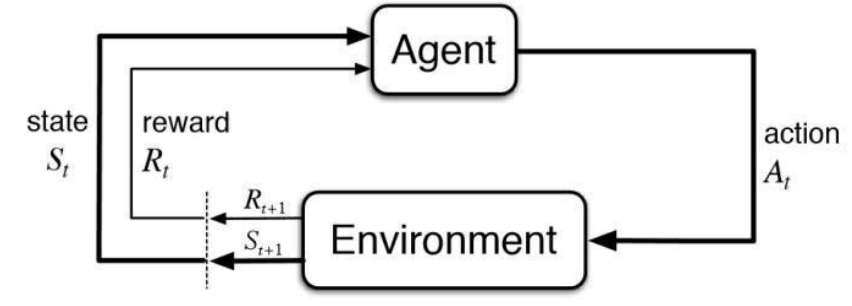


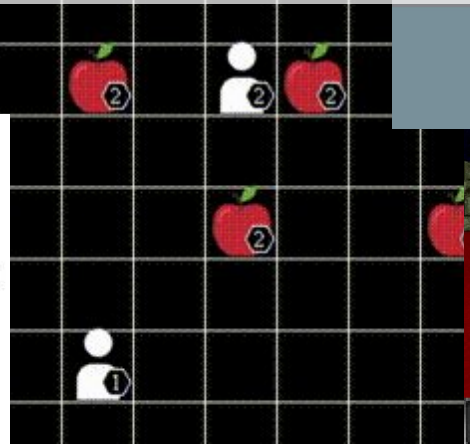






Foraging and Deep Reinforcement Learning





THE FUTURE OF FORAGING

Thank You!

GRAN SASSO SCIENCE INSTITUTE
SCHOOL OF ADVANCED STUDIES
Scuola Universitaria Superiore

paulo.desousa@gssi.it

paulobruno.github.io

MDCC UFC

References

https://en.wikipedia.org/wiki/Reinforcement_learning
https://en.wikipedia.org/wiki/Markov_decision_process
https://en.wikipedia.org/wiki/Q-learning

Sutton and Barto. Reinforcement Learning. 2018.

Mnih et al. Playing Atari with Deep Reinforcement Learning. 2013. https://arxiv.org/abs/1312.5602

Mnih et al. Human-level control through deep reinforcement learning. 2015.
https://www.nature.com/articles/nature14236

Baker et al. Emergent tool use from multi-agent autocurricula. 2019. https://arxiv.org/abs/1909.07528

https://vizdoom.cs.put.edu.pl/
https://unity.com/products/machine-learning-agents

Content

Sponge Bob
https://www.pngaaa.com/detail/698133

mouse, shock
freepik @ flaticon

cheese
Bartama Graphic @ flaticon

Mario diagram
https://paulovasconcellos.com.br/explicando-deep-reinforcement-learning-com-super-mario-ao-inv%C3%A9s-de-matem%C3%A1tica-4c77392cc733

ANN Timeline
http://beamlab.org/deeplearning/2017/02/23/deep_learning_101_part1.html

Neurons
https://cs231n.github.io/neural-networks-1/

Convnets
http://colah.github.io/posts/2014-07-Conv-Nets-Modular/

Deep RL Diagram
http://people.csail.mit.edu/hongzi/content/publications/DeepRM-HotNets16.pdf

Content

Hinton's Speech
https://youtu.be/vShMxxqtDDs?t=419

Turing Award
https://www.nytimes.com/2019/03/27/technology/turing-award-ai.html

Breakout
https://youtu.be/TmPfTpjtdgg

AlphaGo Movie
https://www.youtube.com/watch?v=WXuK6gekU1Y

OpenAI Five
https://openai.com/blog/openai-five-defeats-dota-2-world-champions/

AlphaStar
https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii

MaNa's Speech
https://youtu.be/UuhECwm31dM?t=216

ViZDoom Health Gathering Supreme
https://huggingface.co/learn/deep-rl-course/unit8/hands-on-sf?fw=pt

Unity Food Collector
https://unity-technologies.github.io/ml-agents/Learning-Environment-Examples/#food-collector