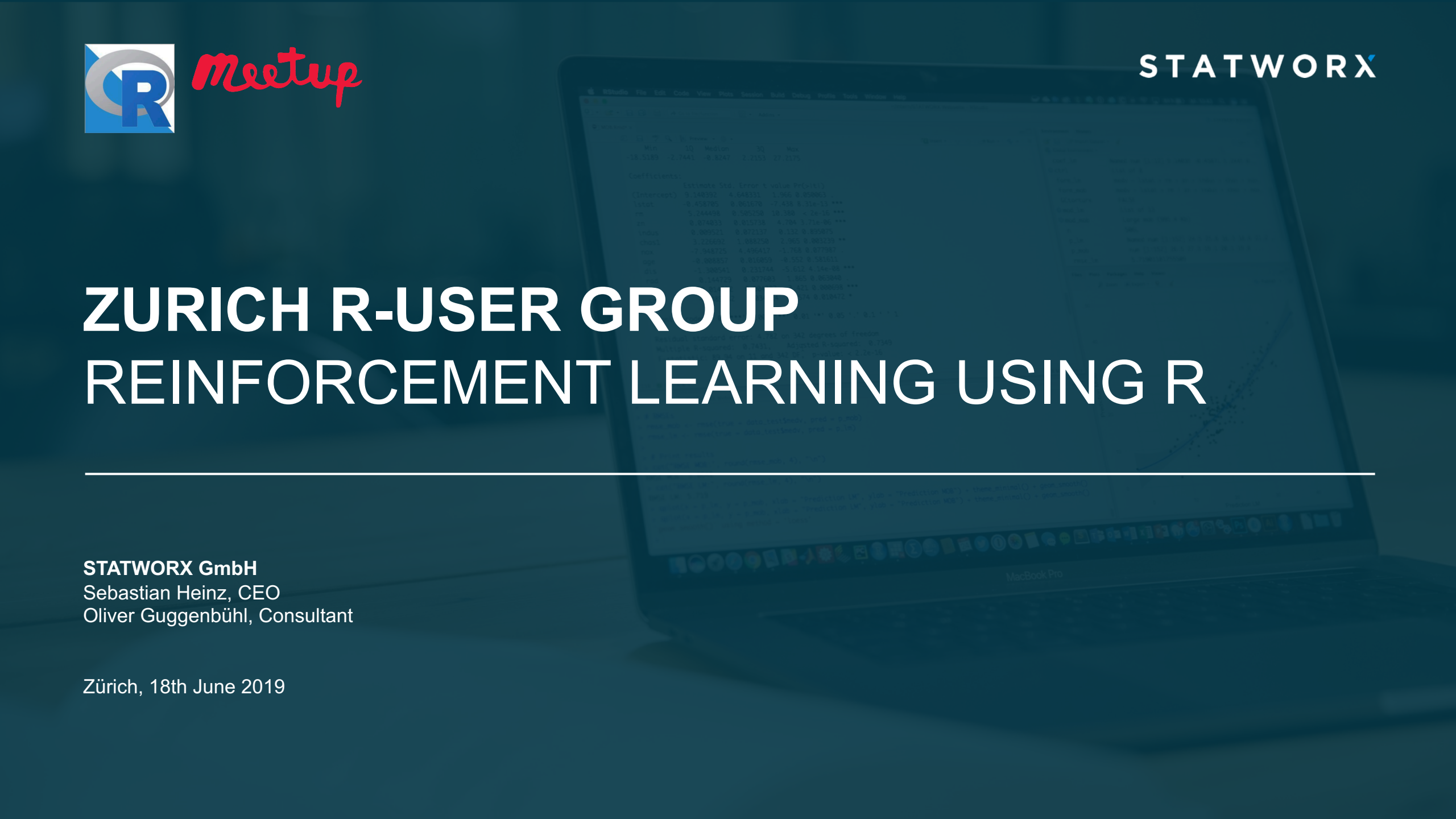


ZURICH R-USER GROUP
REINFORCEMENT LEARNING USING R

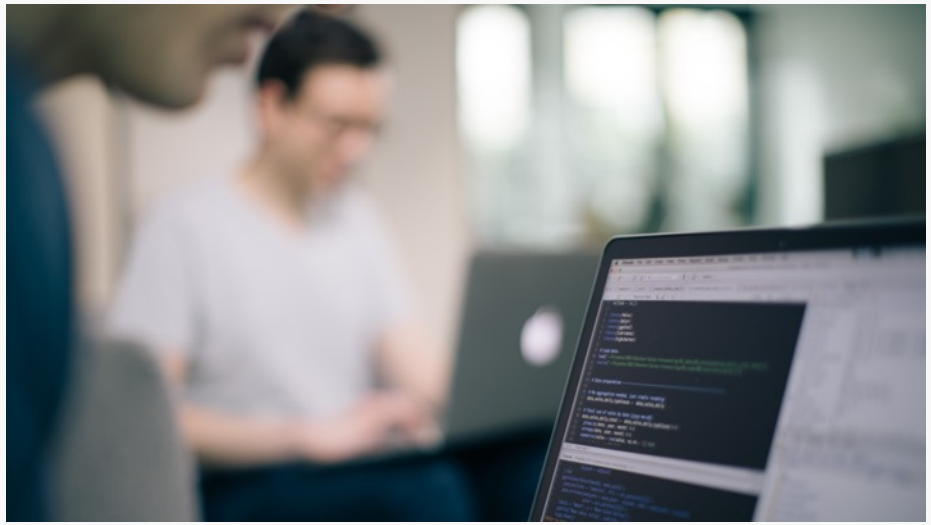
STATWORX GmbH
Sebastian Heinz, CEO
Oliver Guggenbühl, Consultant

Zürich, 18th June 2019

AGENDA

R-Users Zurich meetup

STATWORX



1 COMPANY PROFILE

2 INTRODUCTION TO REINFORCEMENT LEARNING

3 THEORETICAL OVERVIEW

4 IMPLEMENTATION IN R

5 SUPER MARIO AI USE CASE

6 QUESTIONS

STATWORX

STATWORX
COMPANY PROFILE

Information
Services
Project approach

18.06.19 R-Users Zurich meetup

STATWORX

Facts and figures

STATWORX

CLIENT EXCERPT

COMPANY PROFILE

STATWORX is a consulting company for data science, machine learning, and AI located in Frankfurt, Vienna and Zurich. We support our customers in the development and implementation of data science and machine learning projects as well as data driven products.

2011	3	40
FOUNDED	OFFICES	EMPLOYEES

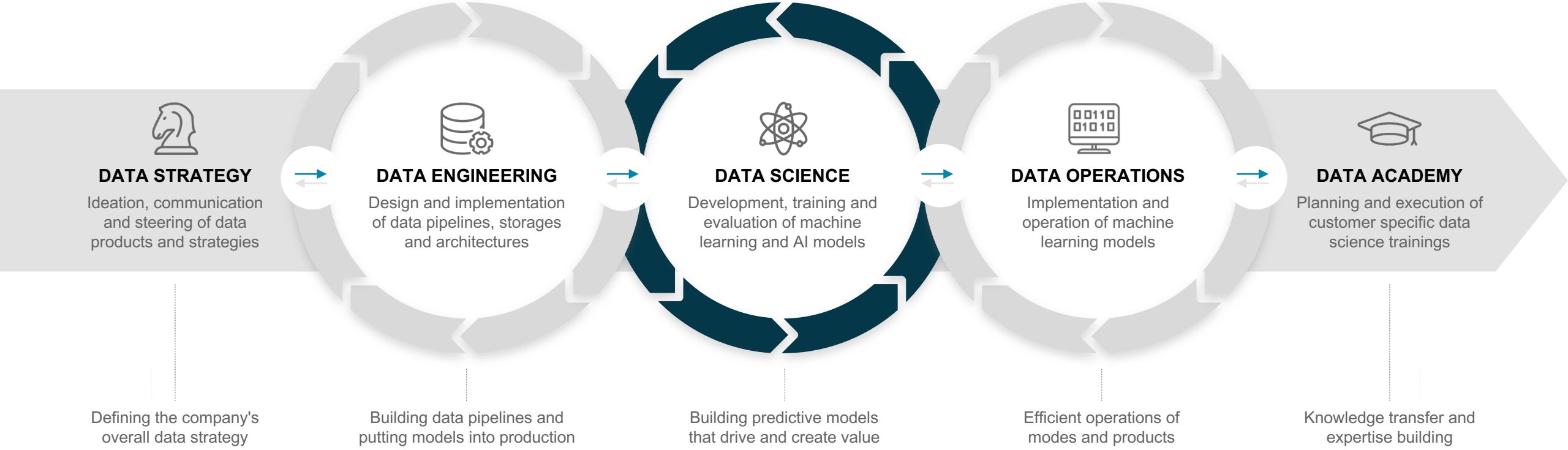
200+	50+	1000+
DATA SCIENCE PROJECTS	INDUSTRY CUSTOMERS	DATA ACADEMY PARTICIPANTS

TOOL STACK & PARTNERS

END-2-END DATA CONSULTING

STATWORX

We support our customers along the whole process of data driven decision making



STATWORX

INTRODUCTION
REINFORCEMENT LEARNING

Where is Reinforcement Learning being used?
A brief history of Data Science
What distinguishes Reinforcement Learning from Supervised & Unsupervised Learning?

INTRODUCTION

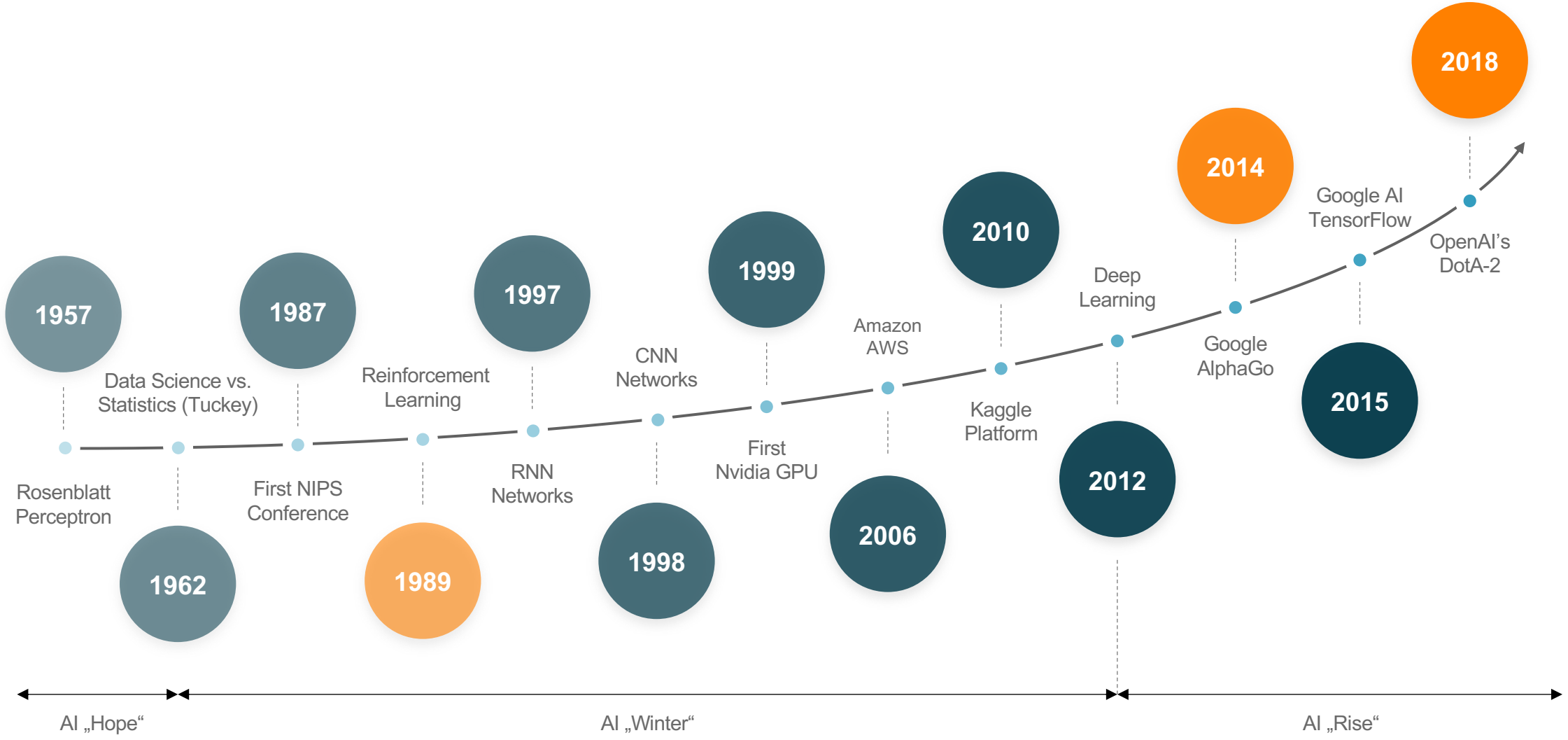
STATWORX

Reinforcement Learning is currently one of the hottest ML topics

A BRIEF HISTORY OF DATA SCIENCE

STATWORX

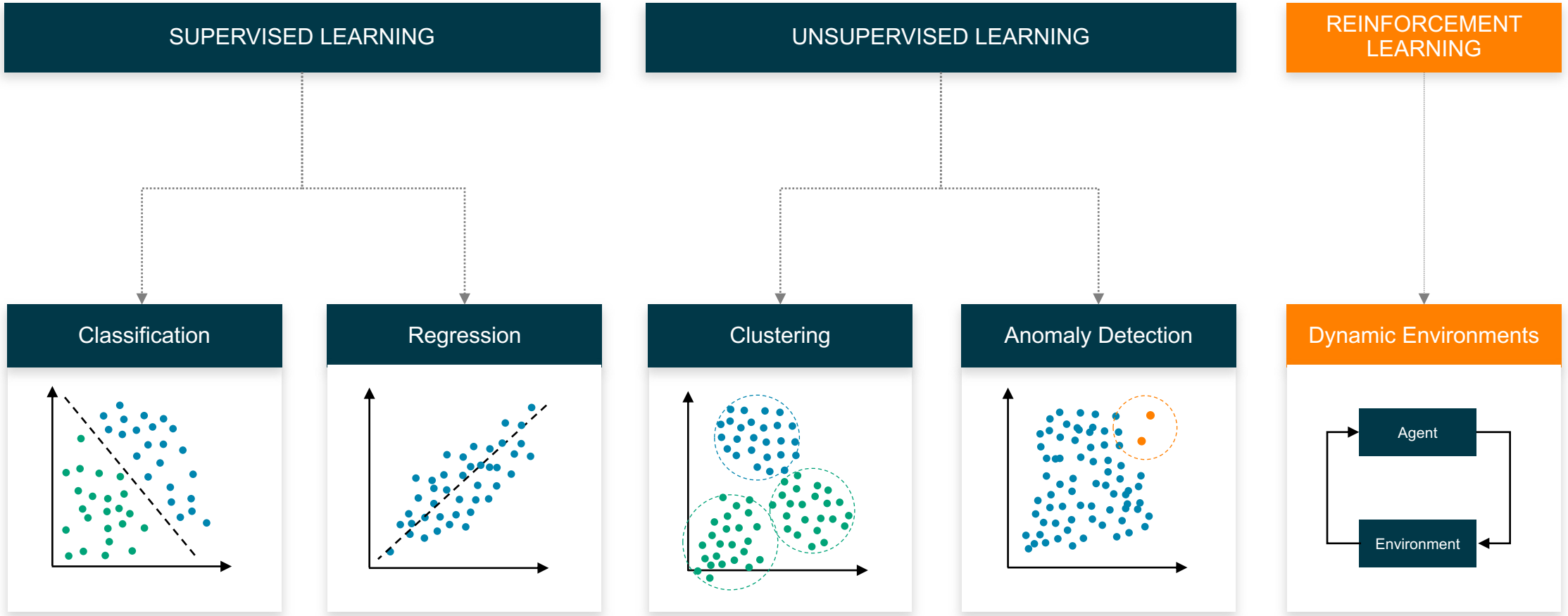
The history of Data Science and AI



MACHINE LEARNING OVERVIEW

STATWORX

Machine Learning Applications



INTRODUCTION

What is Reinforcement Learning?

STATWORX

„Instead of relying on a set of (labelled or unlabelled) training data, Reinforcement Learning relies on being able to monitor the response of the actions taken by the agent.“

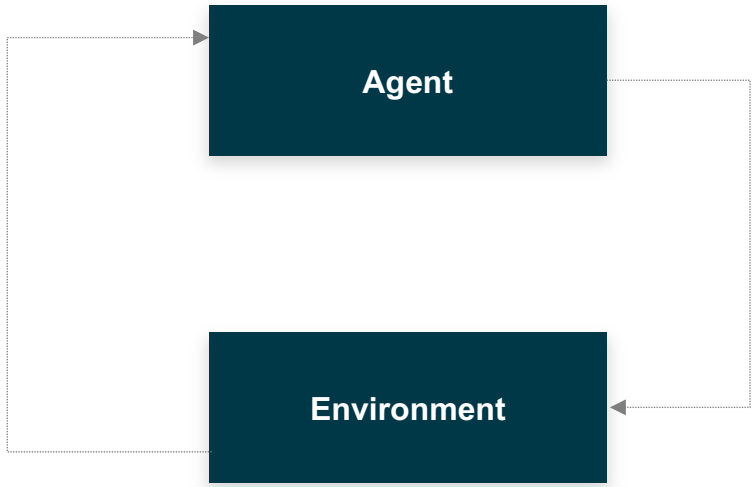
STATWORX

THEORY
REINFORCEMENT LEARNING

How does Reinforcement Learning work?
The Gridworld problem as an example

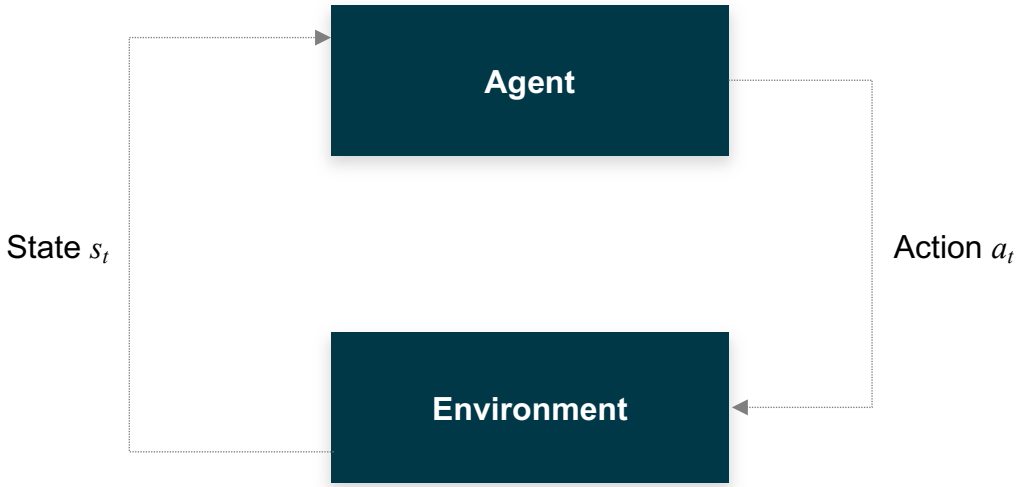
THEORY

How does Reinforcement Learning work?



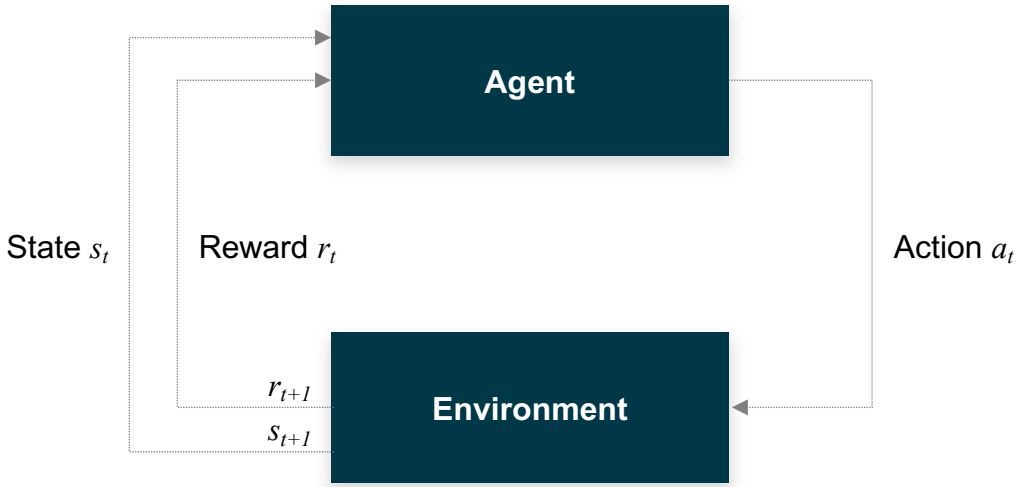
THEORY

How does Reinforcement Learning work?



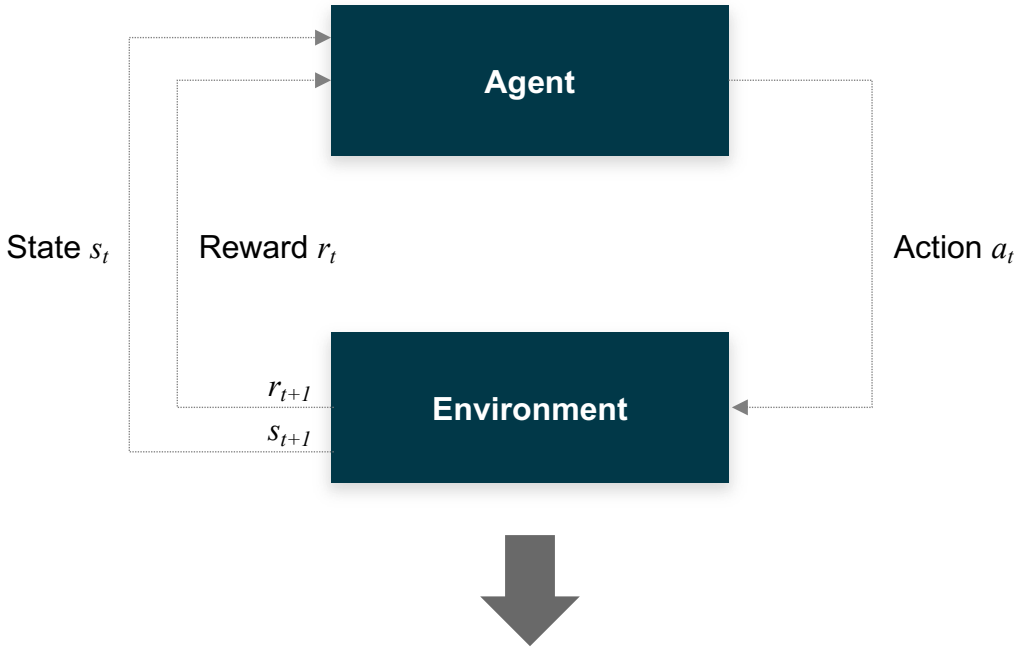
THEORY

How does Reinforcement Learning work?



THEORY

How does Reinforcement Learning work?

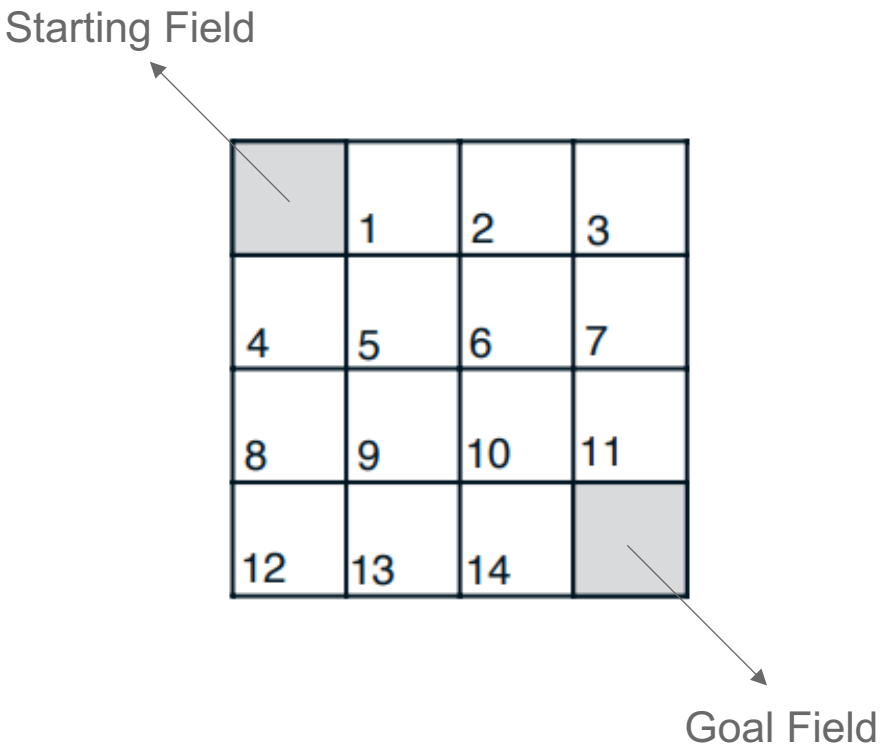


Agent tries to maximize his reward by choosing appropriate actions at a given state of the environment.

EXAMPLE: GRIDWORLD

Reinforcement Learning Use Case

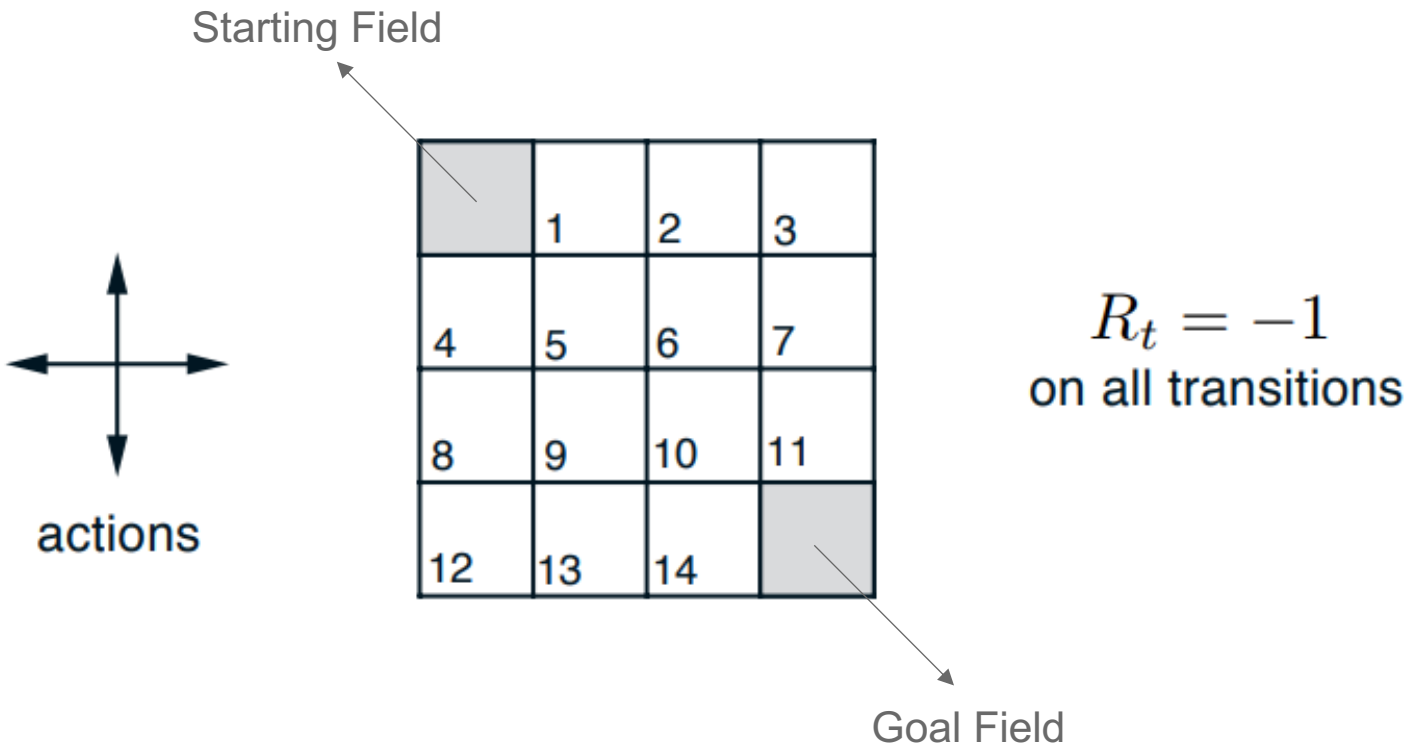
STATWORX



EXAMPLE: GRIDWORLD

Reinforcement Learning Use Case

STATWORX

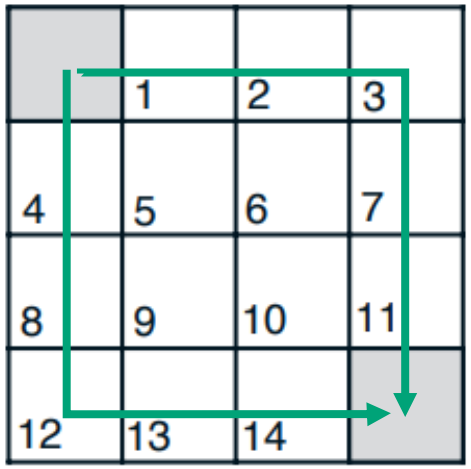


$R_t = -1$
on all transitions

EXAMPLE: GRIDWORLD

Reinforcement Learning Use Case

STATWORX



Ideal return: -6
(6 steps to complete the episode)

Q-LEARNING

Determining the optimal policy for an environment

STATWORX

$$Q(s,a) = r + \gamma max_{a'} Q(s', a')$$

Q-Value — *immediate reward* — *discount factor* — *future reward*

Q-Value

- represents the maximum possible reward at the end of the game for action a in state s
- *does so for each possible action a for the current state s*

Immediate reward

- the immediate reward of a possible action taken, as defined by the environment
- there might be no immediate rewards, but only delayed rewards

Discount factor

- Steers the rate of considering future rewards
- Small values promote decisions that generate immediate reward
- Larger values favor decisions that generate future reward

Future reward

- the maximum possible future reward after transitioning to the next state s' by choosing action a

Q-LEARNING

Determining the optimal policy for an environment

STATWORX

$Q = .59$	$Q = .656$	$Q = .73$	$Q = .81$
$Q = .656$	$Q = .73$	$Q = .81$	$Q = .9$
$Q = .73$	$Q = .81$	$Q = .9$	$Q = 1$
$Q = .81$	$Q = .9$	$Q = 1$	$r = 1$

Assuming that the discount factor $\gamma = 0.9$ and the final reward $r = 1$: $Q(14, right) = 1 + 0.9 \cdot 0$

STATWORX

IMPLEMENTATION IN R
THE `reinforcelearn` PACKAGE

The `reinforcelearn` Package
Live Demo
Advanced Functionalities

THE `reinforcelearn` PACKAGE

Reinforcement Learning implementation in R

STATWORX

`reinforcelearn` package:

The `reinforcelearn` package offers easy tools to create environments, agents and let them interact.

```
library(reinforcelearn)

# Create an environment
env <- makeEnvironment()


# Create an agent
agent <- makeAgent()


# Let the agent interact with the environment
interact(env, agent)
```

THE `reinforcelearn` PACKAGE

Reinforcement Learning implementation in R

STATWORX

Gridworld in `reinforcelearn`:

The Gridworld environment can be easily created with only a few lines of code:

```
library(reinforcelearn)

# Create an environment
env <- makeEnvironment("gridworld",
                       shape = c(4, 4),
                       goal.states = 15,
                       initial.state = 0)
```

THE `reinforcelearn` PACKAGE

Reinforcement Learning implementation in R

STATWORX

Gridworld in `reinforcelearn`:

The agent consists of several parts:

- The policy defines the type of decision rules.
- The value function determines how the current state of the agent is to be evaluated.
- The algorithm determines how the optimal policy is to be found and learnt.

```
library(reinforcelearn)

# Set up the agent
policy <- makePolicy("epsilon.greedy",
                     epsilon = 0.1)
val.fun <- makeValueFunction("table")
algorithm <- makeAlgorithm("qlearning")


# Create the agent
agent <- makeAgent(policy = policy,
                   val.fun = val.fun,
                   algorithm = algorithm)
```

THE `reinforcelearn` PACKAGE

Reinforcement Learning implementation in R

STATWORX

Interaction:

Once the environment and agent have been created we can let them interact with the `reinforcelearn::interact()` function.

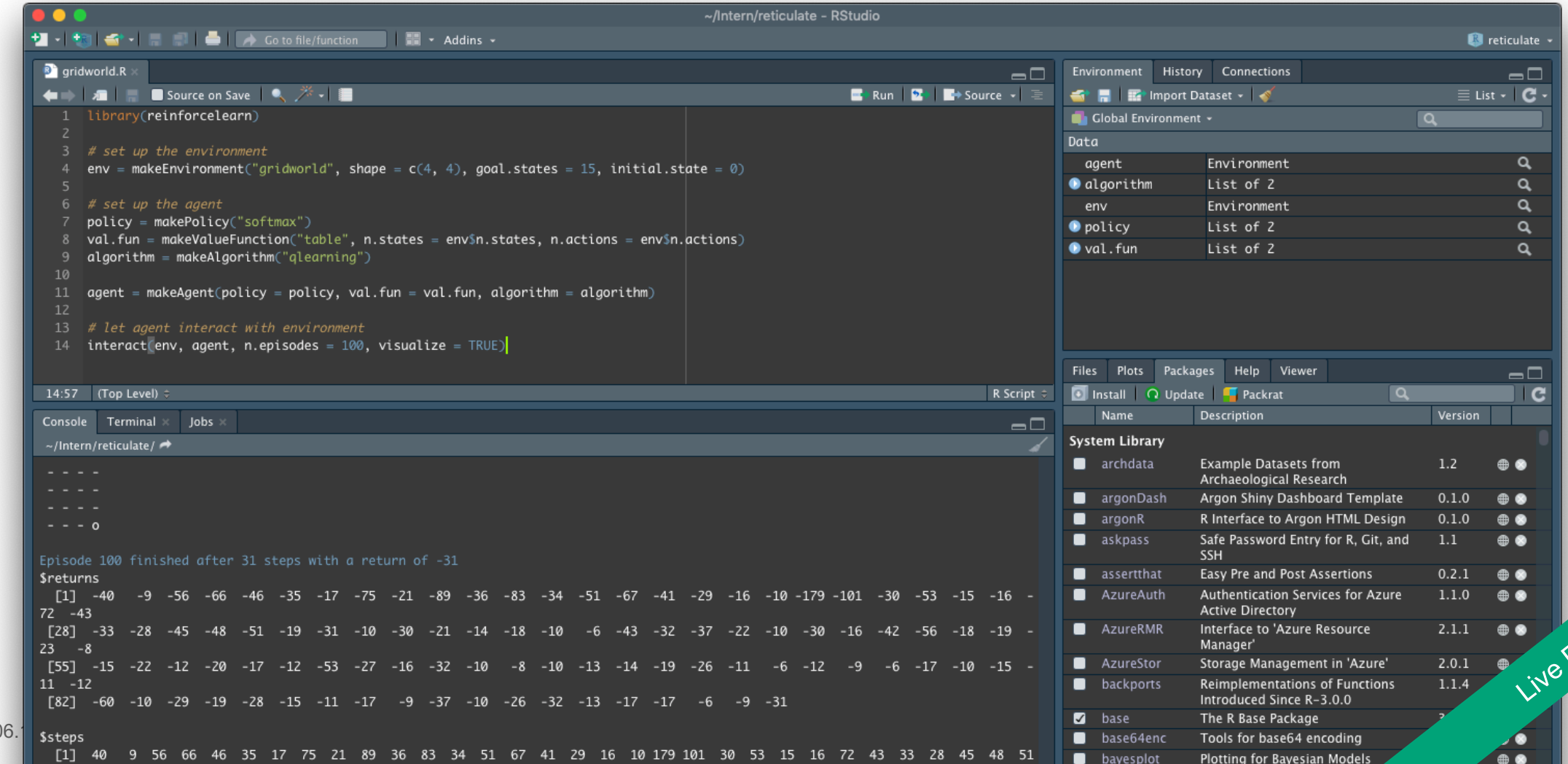
```
library(reinforcelearn)

# Let the agent interact with the environment
interact(env, agent,
         n.episodes = 500,
         visualize = TRUE,
         learn = TRUE)
```

LIVE DEMO

Reinforcement Learning in action

STATWORX



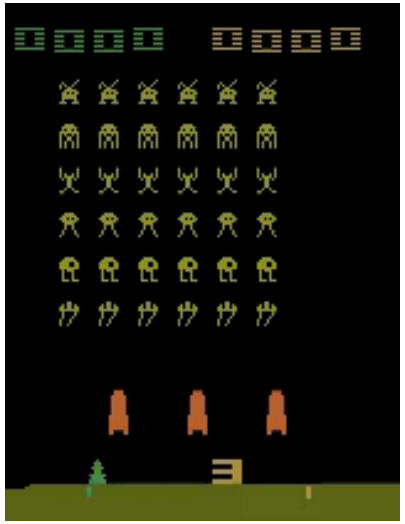
OPENAI GYMS

Advanced functionalities with OpenAI gyms

STATWORX

Using OpenAI gyms in `reinforcelearn`:

`reinforcelearn` allows for easy access to gym environments created by OpenAI.



```r
library(reinforcelearn)
library(reticulate)

# Create an environment
env <- makeEnvironment("gym",
                       gym.name = "SpaceInvaders-v0")
```

NEURAL NETWORKS

Advanced functionalities with Keras

STATWORX

Using neural networks in `reinforcelearn`:

`reinforcelearn` allows for easy integration of neural networks made in `keras` into your value function.

```
library(reinforcelearn)
library(keras)

env <- makeEnvironment("gridworld",
                       shape =c(4, 4),
                       goal.states = 15)

model <- keras_model_sequential() %>%
         layer_dense(units = 4, input_shape = 1,
         activation = "linear") %>%
         compile(optimizer = optimizer_sgd(lr = 0.1),
         loss = "mae")

policy <- makePolicy("epsilon.greedy", epsilon = 0.2)
algorithm <- makeAlgorithm("qlearning")
val.fun <- makeValueFunction("neural.network",
                             model = model)
agent <- makeAgent(policy, val.fun, algorithm)

interact(env, agent, n.episodes = 100)
```

STATWORX

USE CASE DEMONSTRATION
SUPER MARIO BROS. AI

Overview
States, Actions & Rewards
Training and Results

GYM-SUPER-MARIO-BROS

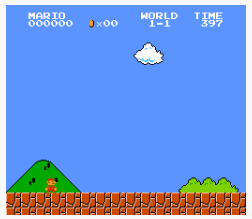
There is a great gym for Super Mario Bros (NES)

STATWORX

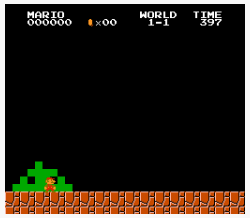
GYM-SUPER-MARIO-BROS

An OpenAI Gym environment for Super Mario Bros. & Super Mario Bros. 2 (Lost Levels) on The Nintendo Entertainment System (NES) using the nes-py emulator.

GAME MODES

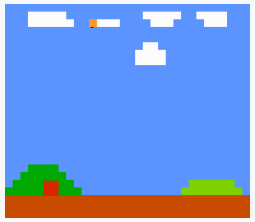


Standard

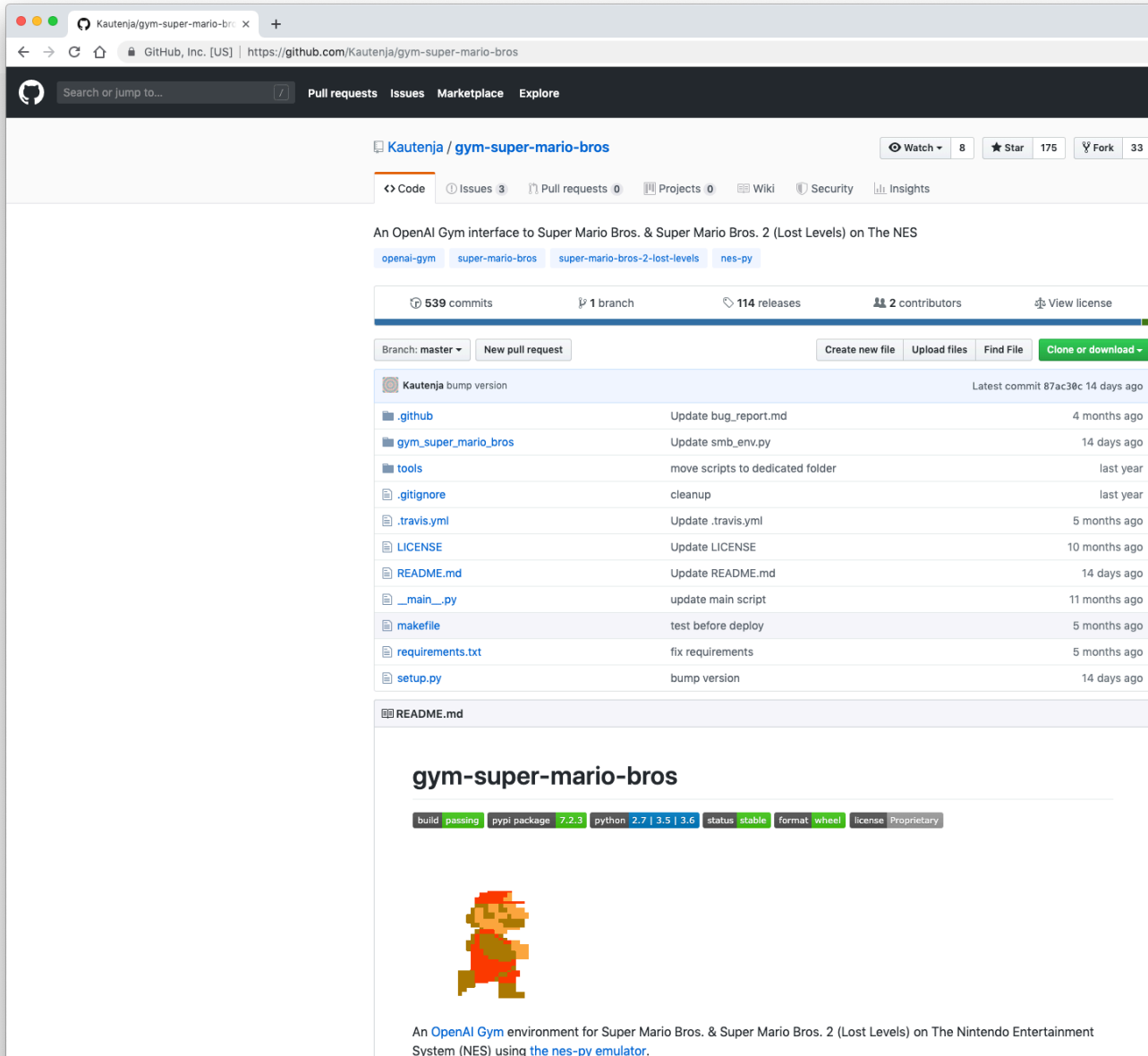


Downsample

Pixel



Rectangle

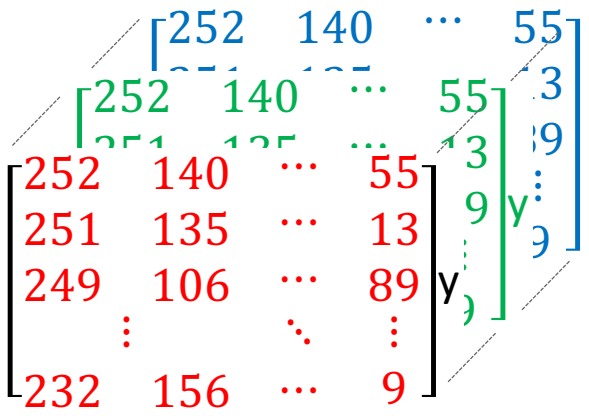


GAME STATES

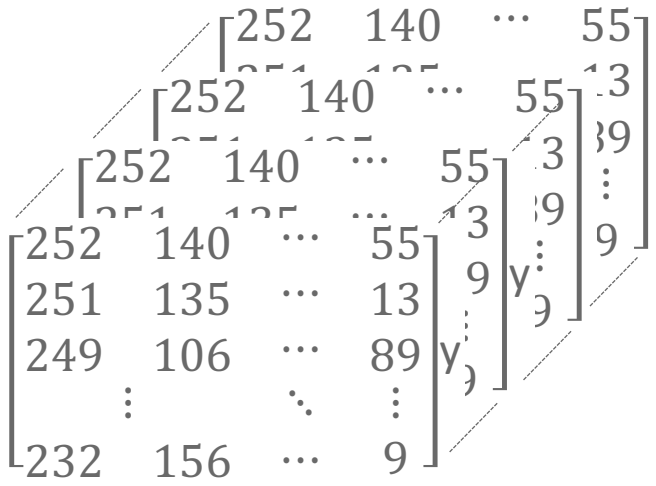
STATWORX

Game states are defined as 4-D tensors containing the last four game states as grayscale matrices

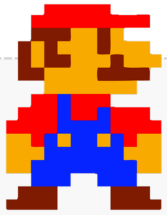
Single Game Screen



$\{n \; x \; p \; x \; c\}$ RGB
Pixel Tensor



$\{n \; x \; p \; x \; 4\}$ Grayscale
Pixel Tensor



"For video game environments, it is important to provide stacked subsequent game frames to the agent. Otherwise, it would not be possible to detect any "movement" on the screen. Furthermore, we use only every n^{th} frame."

Mario Pro-Tip

ACTIONS

STATWORX

Mario's action space is mapped to integers

CONTROL CROSS

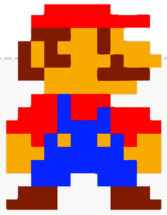
Used to control Mario into 8 different directions: right, right-down, down, down-left, left, left-top, top, top-right.

START / SELECT

Usually used to pause / start / exit the game.

BUTTONS

Used for jumping (A) and running (B hold).



"During training, controller actions (and their combinations) are mapped to integers, usually limited to game-relevant actions. This is done by using warpper functions around the environment."

Mario Pro-Tip

REWARDS

STATWORX

The game's reward is composed of three different components

$$r = v + c + d$$

reward *velocity* *clock* *death*

VELOCITY

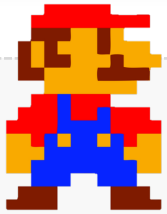
- v: the difference in agent x values between states
- v = x1 - x0
  - x0 is the x position before the step
  - x1 is the x position after the step
- moving right ⇔ v > 0
- moving left ⇔ v < 0
- not moving ⇔ v = 0

CLOCK

- c: the difference in the game clock between frames
- The clock encourages to be fast
- c = c0 - c1
  - c0 is the clock before the step
  - c1 is the clock after the step
- no clock tick ⇔ c = 0
- clock tick ⇔ c < 0

DEATH

- d: a death penalty that penalizes the agent for dying in a state
- this penalty encourages the agent to avoid death
- alive ⇔ d = 0
- dead ⇔ d = -15



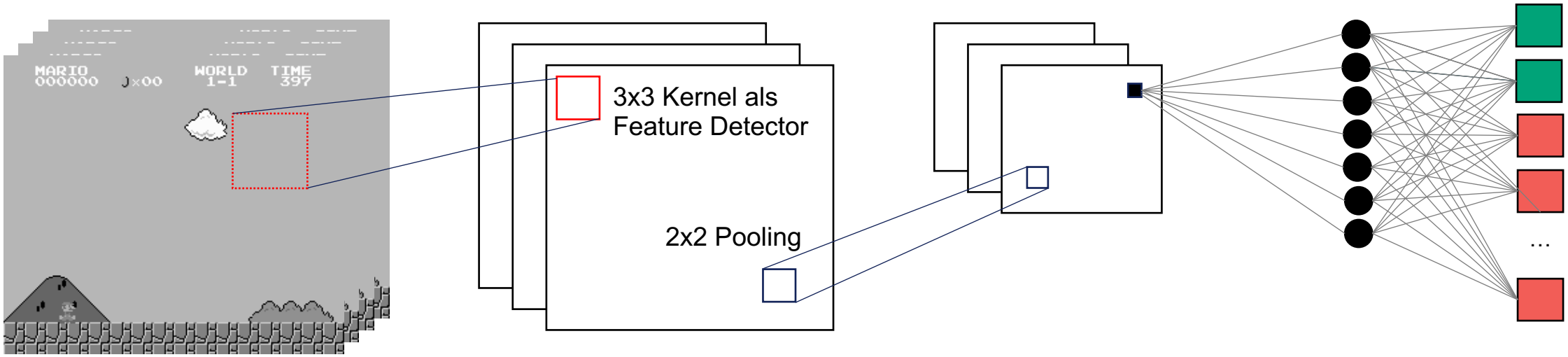
"The reward function assumes the objective of the game is to move as far right as possible (increase the agent's x value), as fast as possible (decrease time), without dying."

Mario Pro-Tip

MODEL

STATWORX

We are using a deep CNN to approximate the Q-value function of our agent



84 x 84 x 4 Tensor

INPUT

We are using the last 4 game frames as input tensor to the neural network.

2D CONVOLUTIONS

Convolutional layers extract relevant information from the game screens.

POOLED FEATURE MAPS

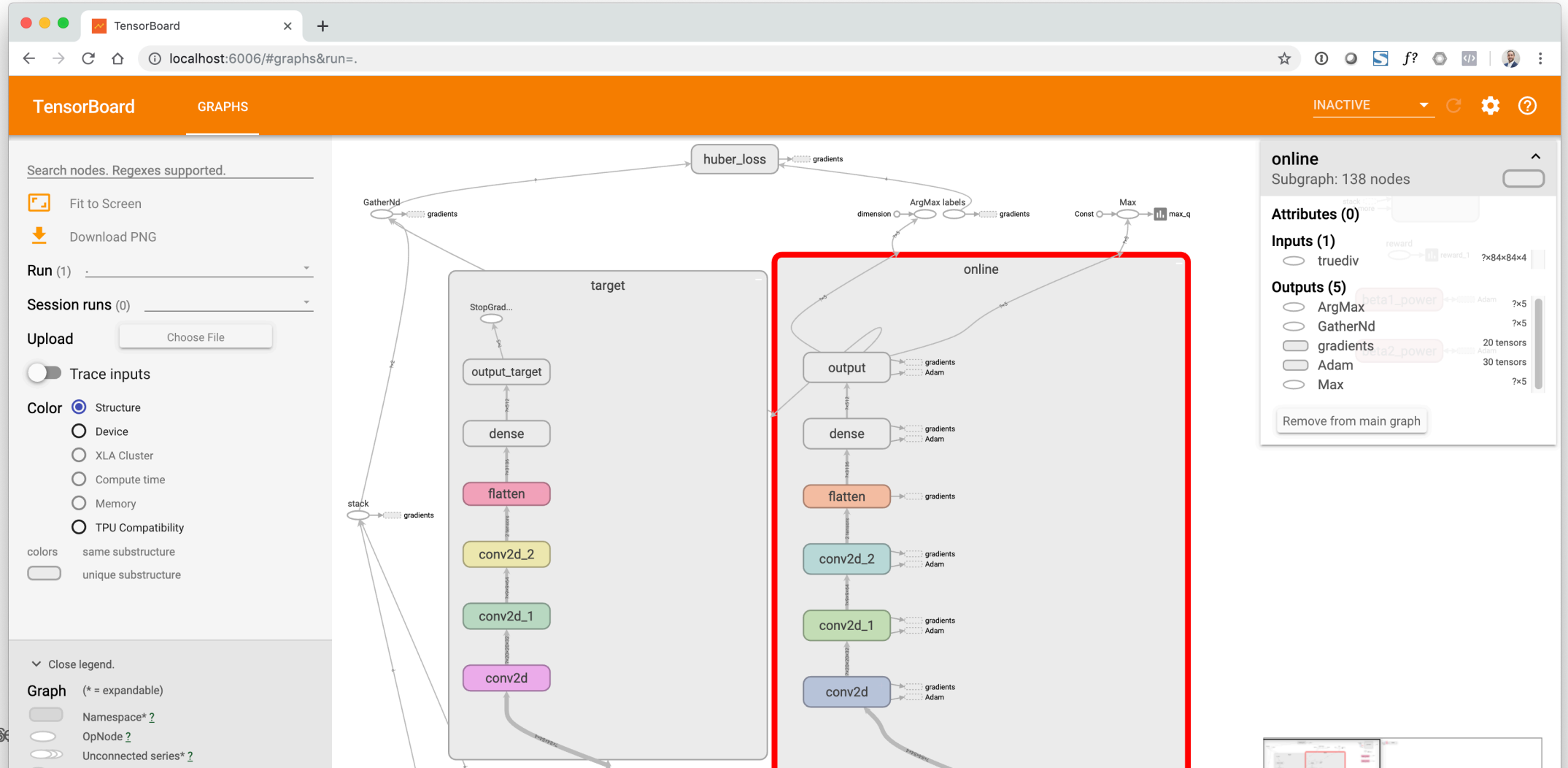
Pooling compresses information and shrinks the dimensionality of the problem.

DENSE & OUTPUT

The network approximates the Q-values by comparing the

MODEL

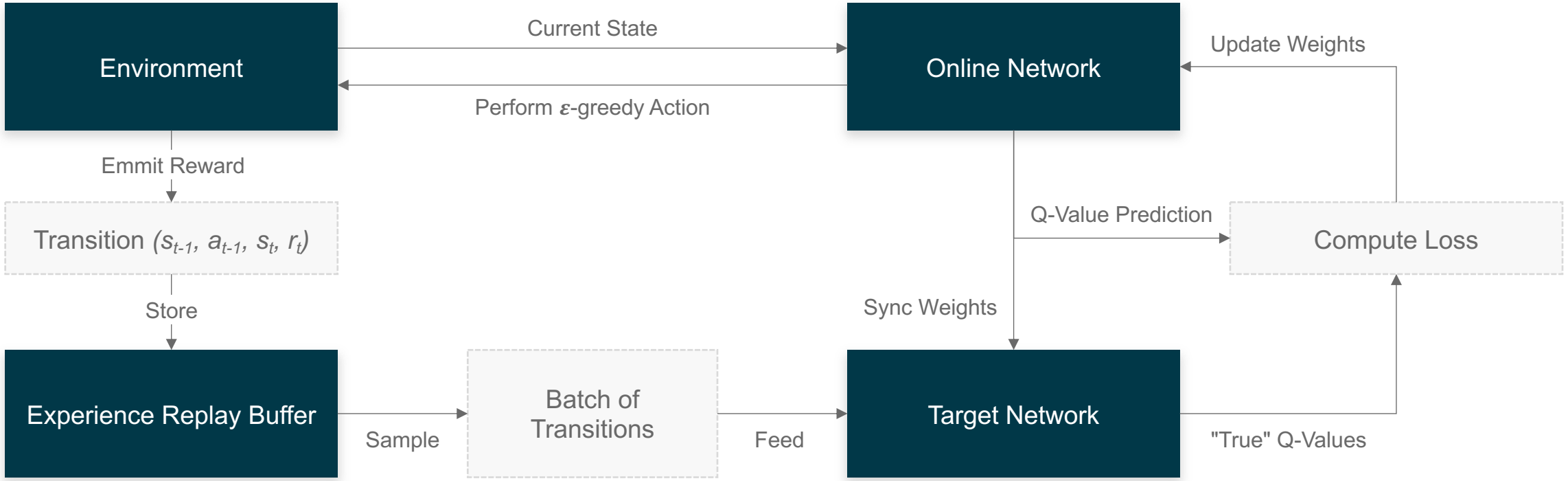
A cool screenshot of our model architecture in TensorBoard ☺

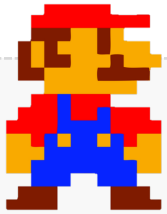


TRAINING PIPELINE

STATWORX

Overview of the actual training process of the agent





"Experience replay is needed because subsequent game states are highly correlated which violates the i.i.d. assumption of stochastic gradient descent, which is used for updating the online network's weights."

Mario Pro-Tip

RESULTS

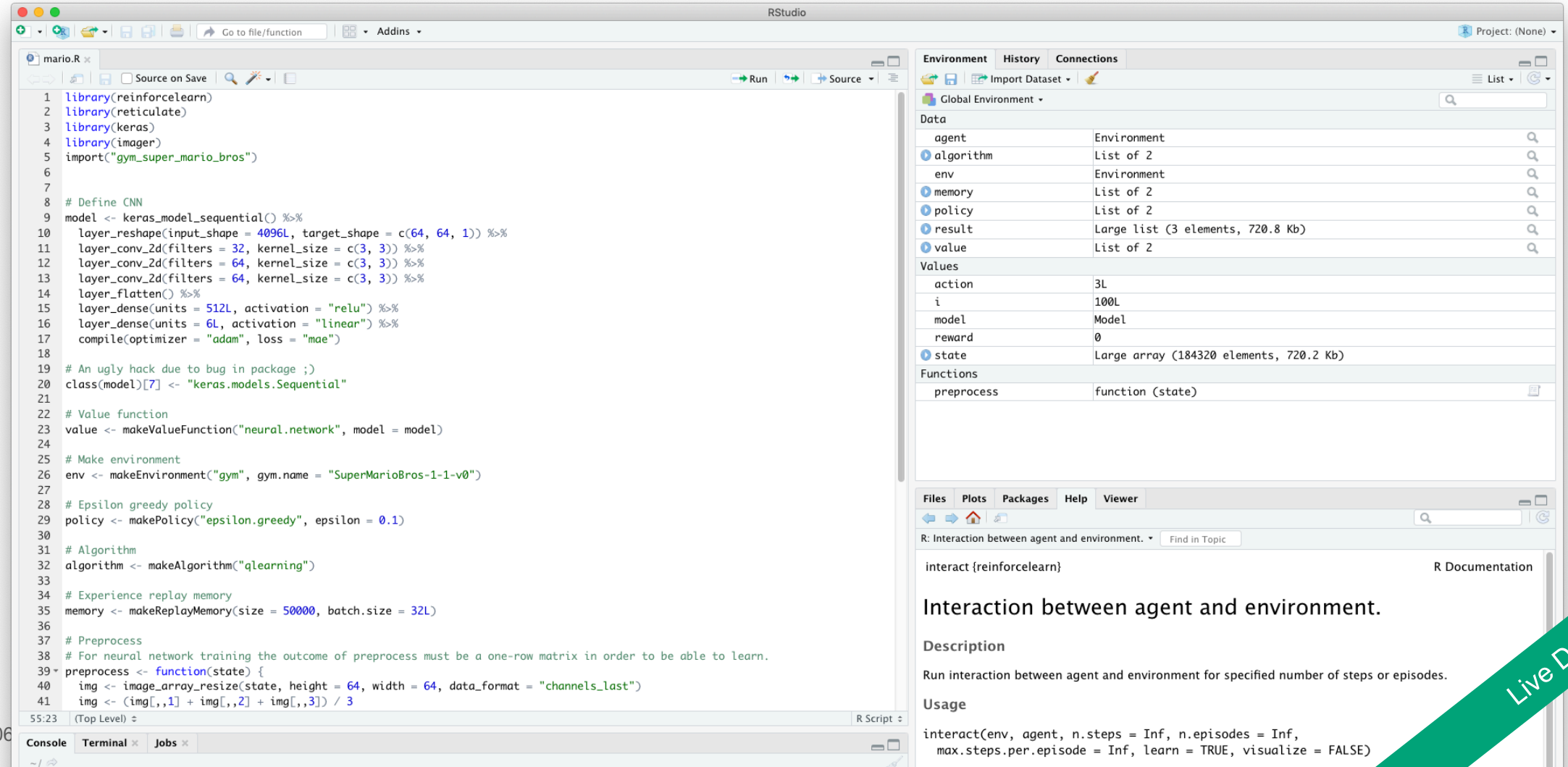
STATWORX

Watch the Super Mario Agent in action as it masters the first level of the game

LIVE DEMO

Learning to play Super Mario using R and reinforcelearn (well, kind of) ☺

STATWORX



KEY FINDINGS

What we have learned

STATWORX

1. Reinforcement learning is a very promising area of ML / AI research
2. R is still quite limited in the area of modern reinforcement learning methods
3. The `reinforcelearn` package is not up-2-date when it comes to network policies
4. reticulate is a way to play around with RL for the moment but not the final solution
5. If you are considering diving deeper into RL, take a look at Python ;)

WANT TO LEARN MORE?

Join one of our next training events!

STATWORX

STATWORX

Data Science Bootcamp

A 5-day introduction to the field of data science and machine learning.

July 1st – July 5th
STATWORX Office
Zurich, Switzerland
German Language

www.statworx.com/ch/academy

JOIN TRAINING

STATWORX

Data University

A 2-day immersive learning experience with 24 sessions from can be selected.

October 9th – October 10th
Goethe University
Frankfurt, Germany
German Language

www.data-university.de

JOIN TRAINING

STATWORX

Deep Learning Bootcamp

A 5-day introduction to the field of deep learning with Keras and TensorFlow.

November 4th – November 8th
STATWORX Office
Frankfurt, Germany
German Language

www.statworx.com/ch/academy

JOIN TRAINING

STATWORX

Q&A and Discussion

STATWORX

THANK YOU
FOR THE ATTENTION.

CONTACT
Sebastian Heinz, CEO
sebastian.heinz@statworx.com
www.statworx.com

STATWORX
Wöhlerstr. 8-10
60323 Frankfurt am Main