Reinforcement Learning:
From Playing Games to Trading Stocks

Dr. Yves J. Hilpisch

Virtual Meetup, 23. April 2020

Introduction

THE PYTHON QUANTS

SERVICES
for financial institutions globally

EVENTS
for Python quants &
algorithmic traders

CERTIFICATION
in cooperation with university

TRAINING
about Python for finance
& algorithmic trading

BOOKS
about Python and
finance

PLATFORM
for browser-based
data analytics

OPEN SOURCE
Python library
for financial analytics

http://tpq.io

The AI Machine

http://aimachine.io

http://certificate.tpq.io/tpq_top_algo_2019.pdf

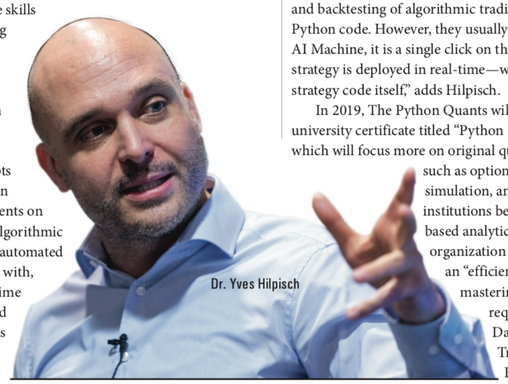
The Python Quants

First University Certificate in Python for Algorithmic Trading

Python programming has become a key skill in the financial industry. In areas such as financial data science, computational finance or algorithmic trading, Python has established itself as the primary technological platform. At the same time, the level of Python sophistication the industry is expecting from its employees and applicants is increasing steadily. The Python Quants Group is one of the leading providers of Python for Finance training programs.

Among others, The Python Quants have tailored a comprehensive online training program leading to the first University Certificate in Python for Algorithmic Trading. Be it an ambitious student with intrigue for algorithmic trading, or a major financial institution, The Python Quants, through this systematic training program, is equipping delegates with requisite skills and tools to formulate, backtest and deploy algorithmic trading strategies based on Python.

The topics covered in the training programs offered by The Python Quants are generally not found in the typical curriculum of financial engineering or quantitative finance Master programs. Dr. Yves Hilpisch, the firm's founder and managing partner, explains, "There are courses out there that show students how to apply machine learning for the formulation and backtesting of algorithmic trading strategies. However, none of them explains the difficulties or the skills required in deploying such algorithmic trading strategies in the real world. Besides providing an introductory course that teaches Python and financial concepts from scratch, we train our delegates and clients on how best to deploy algorithmic trading strategies in automated fashion in the cloud, with, among others, real-time risk management and monitoring," explains Hilpisch, an author of three books on the topic, with "Python for Finance" (2nd ed., O'Reilly) being the standard reference in the field.


Dr. Yves Hilpisch

The organization's "Python for Algorithmic Trading University Certificate" consists of 200 hours of instruction, 1,200 pages of documentation and 1,000s of lines of Python code. In addition to offering both online and offline Python training, Hilpisch and his team also organize bespoke training events for financial institutions, hedge funds, banks, and asset management companies. "Most of the training is online since we have students and delegates from about 65 different countries in general. Most recently, we noticed that it's not just financial firms and students who want to deepen their algorithmic trading knowledge, but even professors of finance who want to get more involved in this popular topic," says Hilpisch.

While the Quant Platform is the most popular choice, especially for users in the financial sector who don't have access to a full-fledged, interactive, financial analytics environment, the team at The Python Quants is currently developing The AI Machine—a new platform which leverages artificial intelligence to formulate and deploy algorithmic trading strategies in a standardized manner. Hilpisch explains that it's relatively easy to write Python code for an algorithmic trading strategy, but the same can't be said about the deployment of such a strategy. "There are a few platforms out there that allow the formulation and backtesting of algorithmic trading strategies by the use of Python code. However, they usually stop exactly there. With The AI Machine, it is a single click on the 'GO LIVE' button and the strategy is deployed in real-time—without any changes to the strategy code itself," adds Hilpisch.

In 2019, The Python Quants will be introducing a new university certificate titled "Python for Computational Finance," which will focus more on original quantitative finance topics, such as option pricing, Monte Carlo simulation, and hedging. As financial institutions begin to perceive Python-based analytics as a prerequisite skill, the organization will continue to provide an "efficient and structured way of mastering all the tools and skills required in Python for Financial Data Science, Algorithmic Trading, and Computational Finance."CM

Dr. Yves J. Hilpisch is founder and CEO of **The Python Quants** (http://tpq.io), a group focusing on the use of open source technologies for financial data science, artificial intelligence, algorithmic trading, and computational finance. He is also the founder and CEO of **The AI Machine** (http://aimachine.io), a company focused on AI-powered algorithmic trading based on a proprietary strategy execution platform.

Yves has a Diploma in Business Administration, a Ph.D. in Mathematical Finance and is Adjunct Professor for Computational Finance.

Yves is the author of five books (https://home.tpq.io/books):

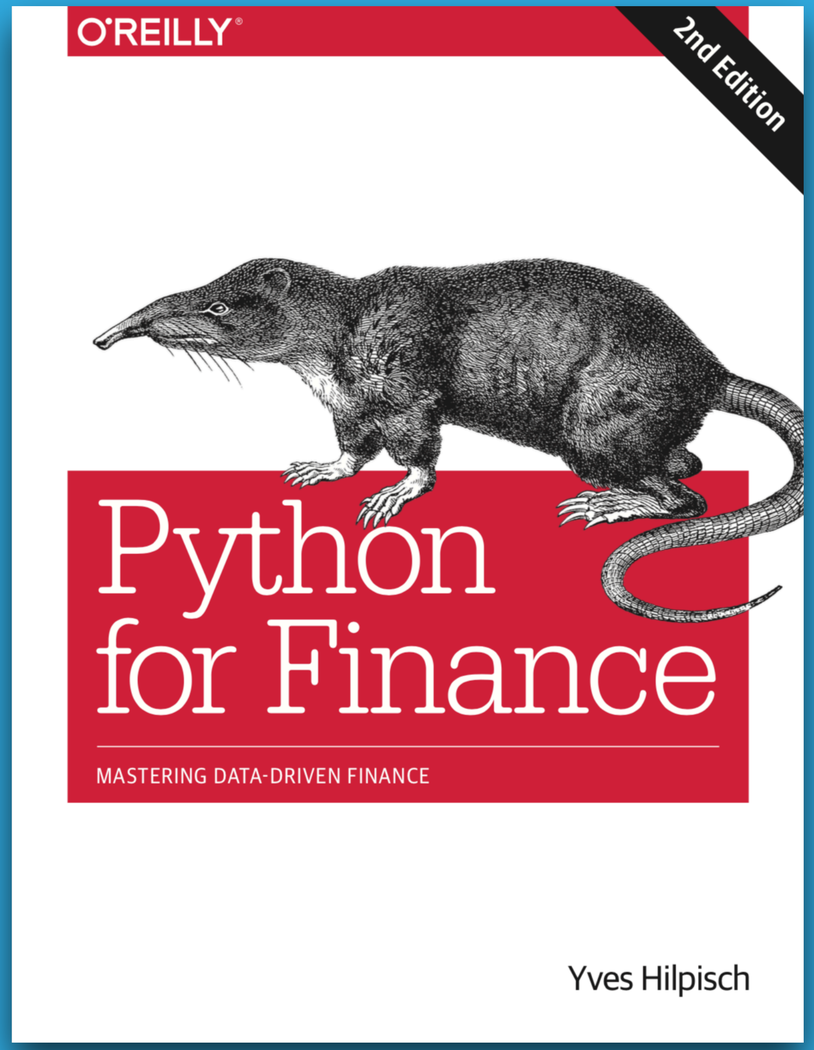
* Artificial Intelligence in Finance (O'Reilly, forthcoming)
* Python for Algorithmic Trading (O'Reilly, forthcoming)
* Python for Finance (2018, 2nd ed., O'Reilly)
* Listed Volatility and Variance Derivatives (2017, Wiley Finance)
* Derivatives Analytics with Python (2015, Wiley Finance)

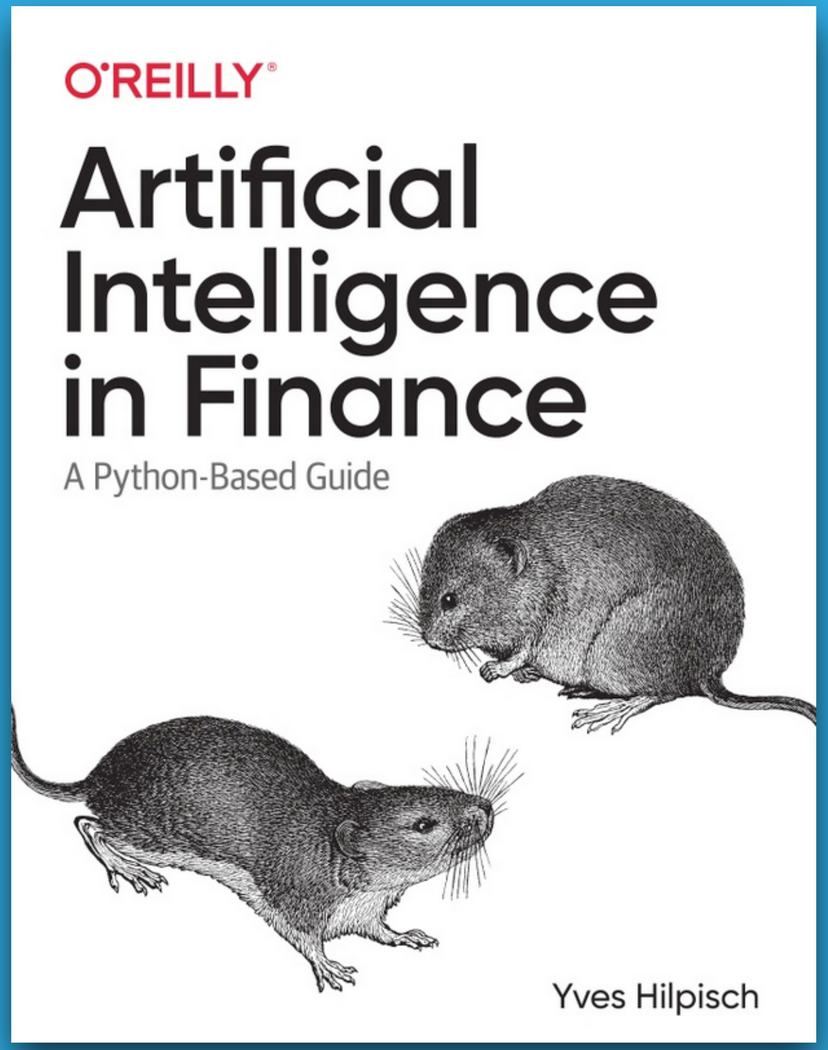
Yves is the director of the first online training program leading to **University Certificates in Python for Algorithmic Trading** (https://home.tpq.io/certificates/pyalgo) and **Computational Finance** (https://home.tpq.io/certificates/compfin). He also lectures on computational finance, machine learning, and algorithmic trading at the **CQF Program** (http://cqf.com).

Yves is the originator of the financial analytics library **DX Analytics** (http://dx-analytics.com) and organizes Meetup group **events**, **conferences**, **and bootcamps** about Python, artificial intelligence and algorithmic trading in London (http://pqf.tpq.io), New York (http://aifat.tpq.io), Frankfurt, Berlin, and Paris. He has given **keynote speeches** at technology conferences in the United States, Europe, and Asia.

http://hilpisch.com

Python & AI for Finance



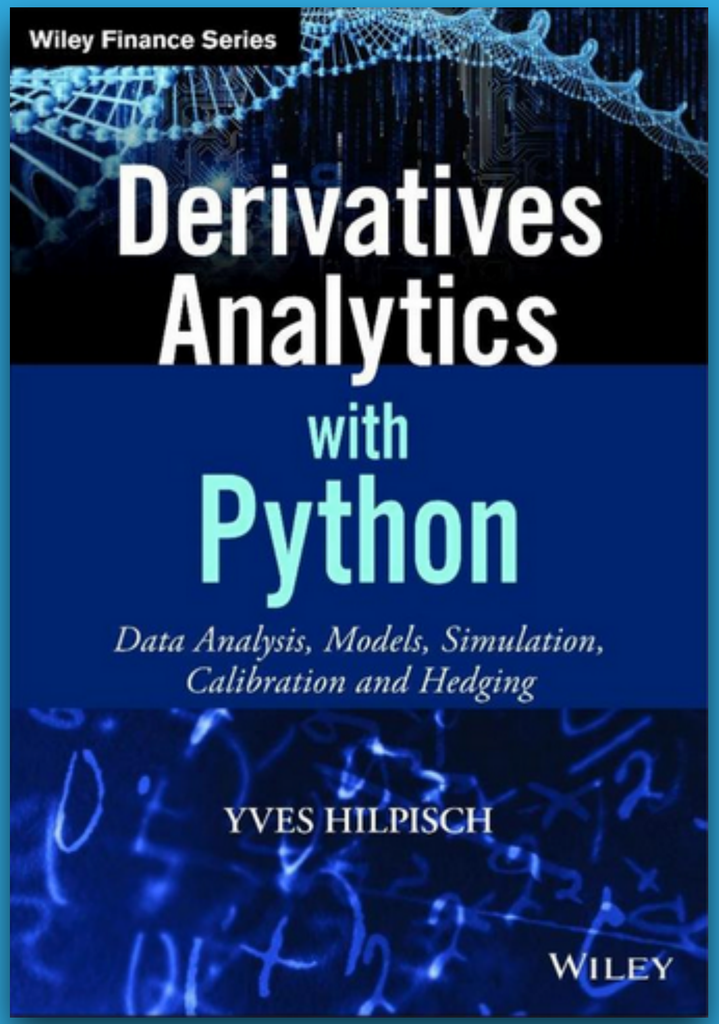


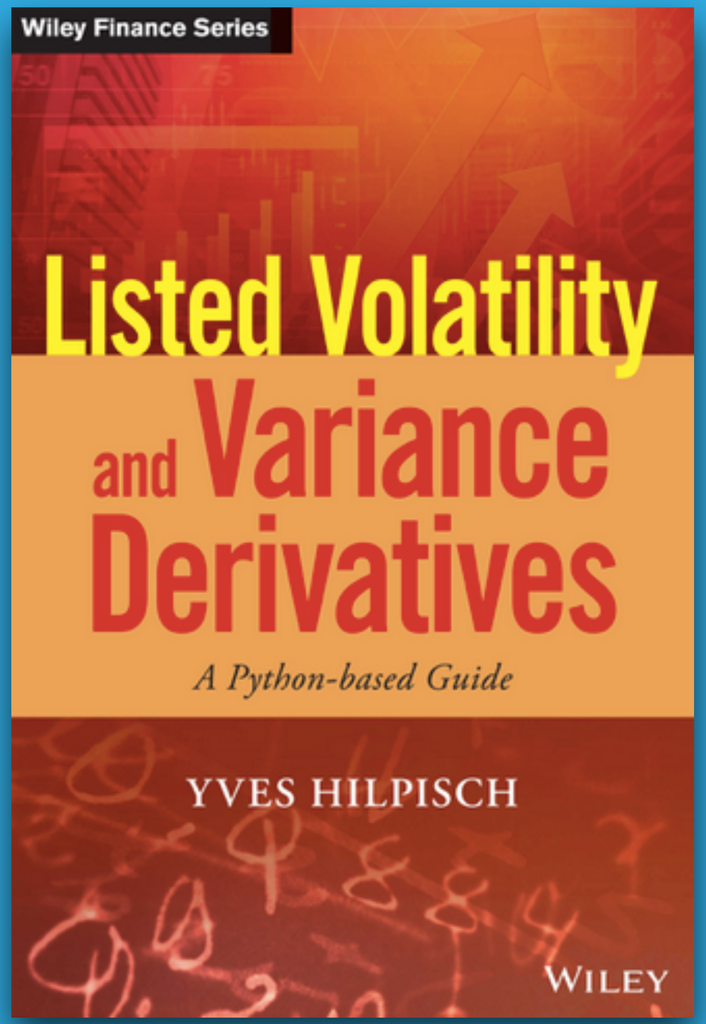
Next book project:

Python for Algorithmic Trading

http://books.tpq.io

Quant Finance with Python





http://books.tpq.io

Our Certificate Program

16 week program

150+ hours of instruction

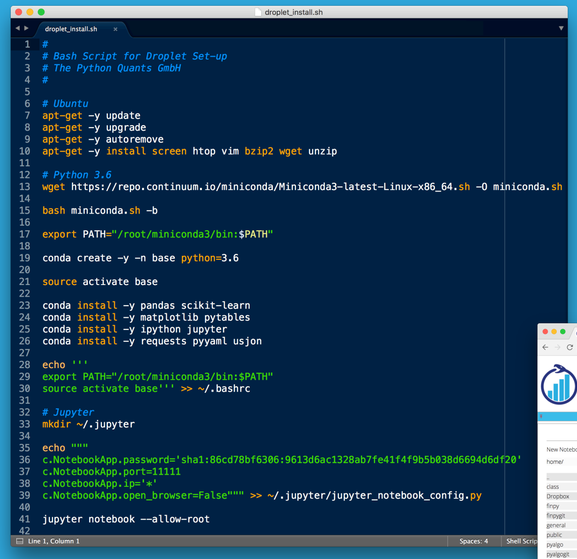
5,000+ lines of code

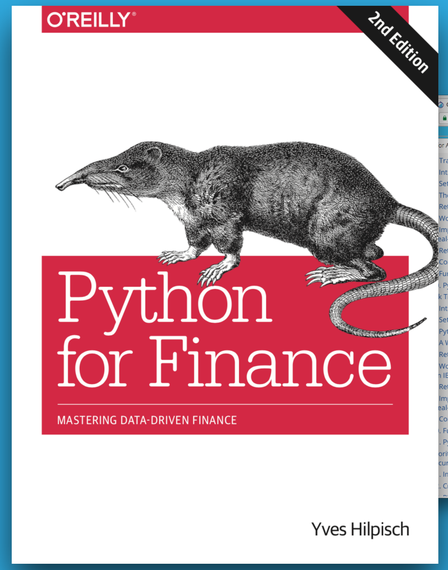
1,200 pages PDF

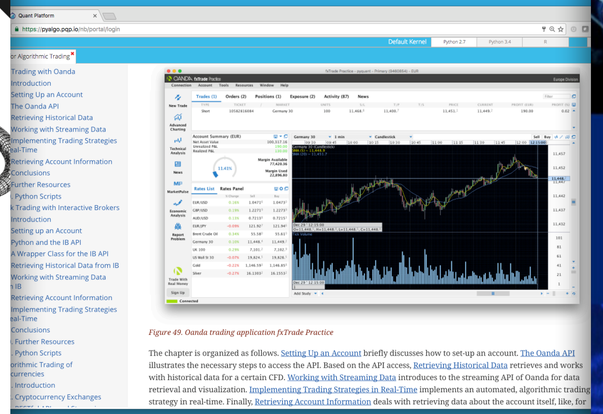
http://certificate.tpq.io

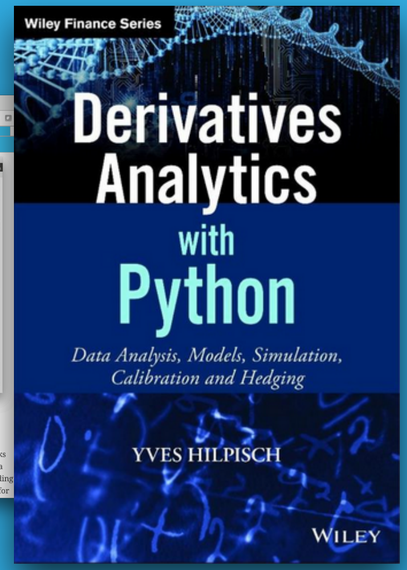
1,000+ pages of
Finance with Python,
Python for Finance,
Algorithmic Training,
Derivatives Analytics

10,000+ lines of code



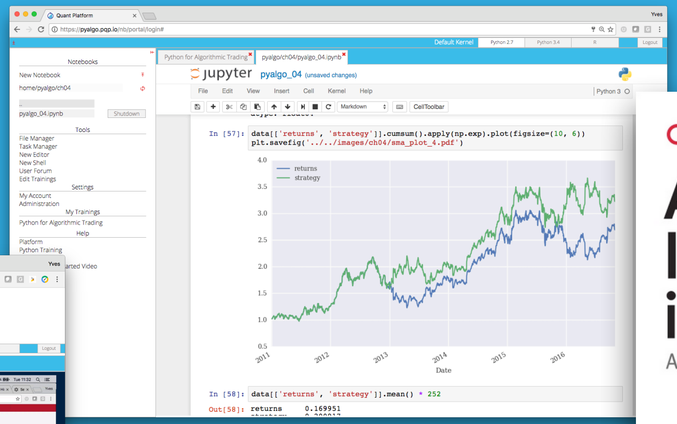


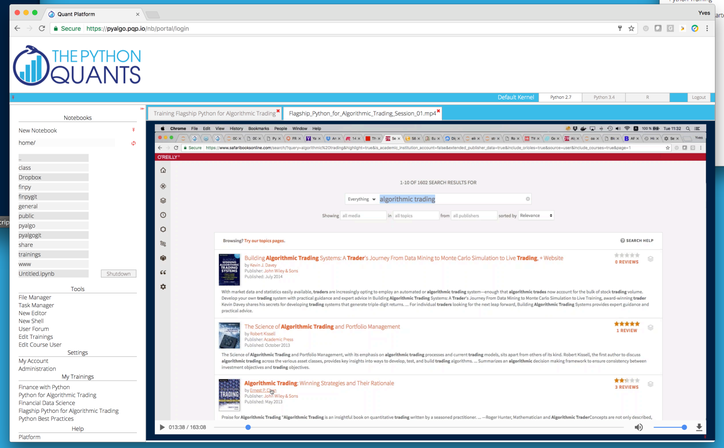




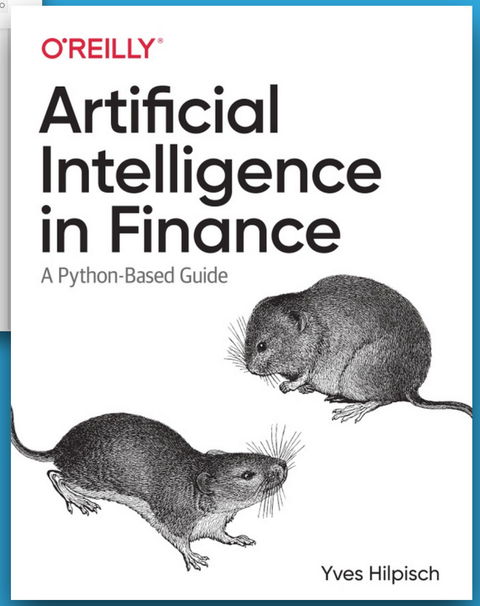
150+ Jupyter Notebooks

200+ hours
of pre-recorded
video instruction



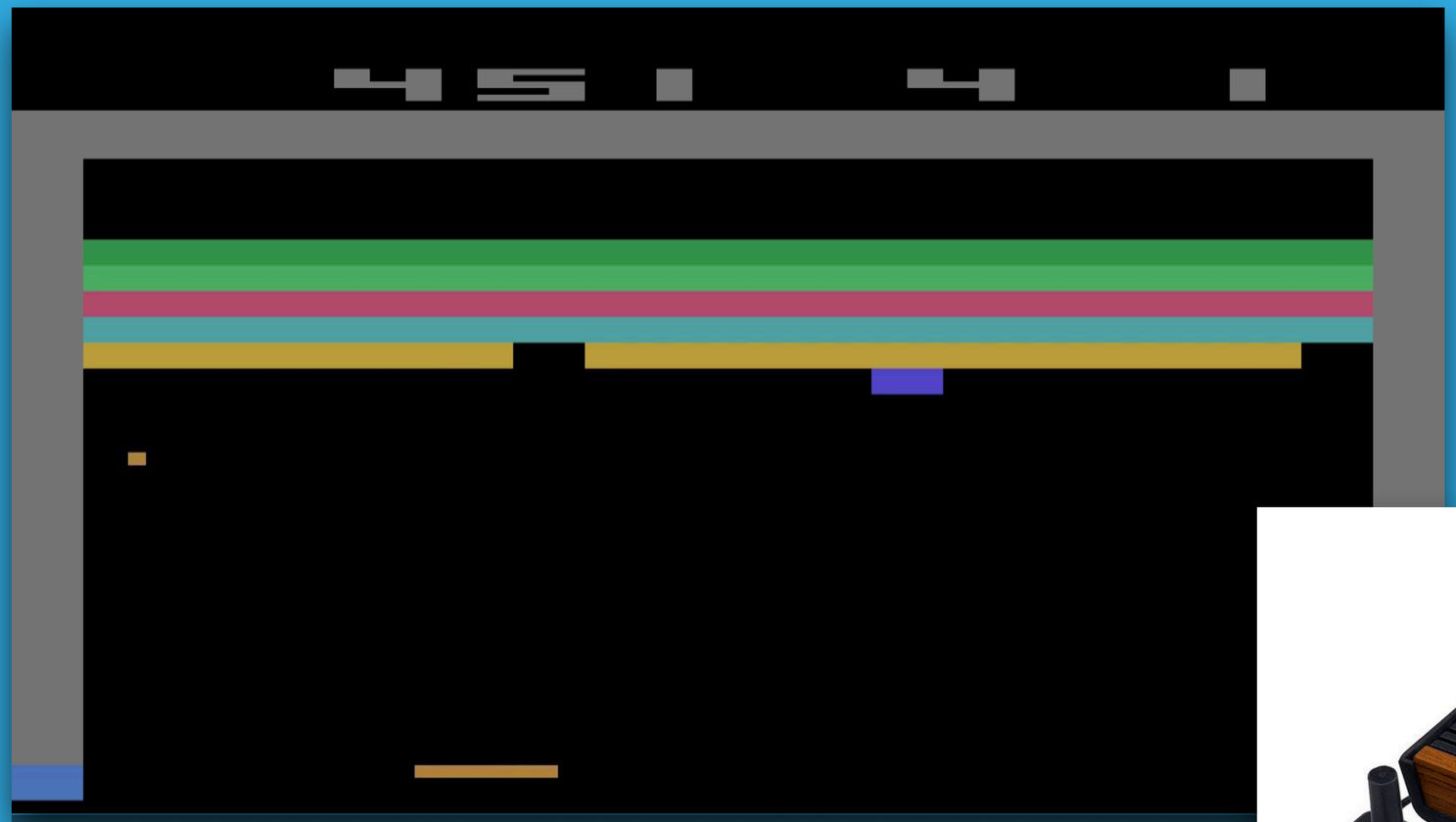


many hours of
additional
live sessions



RL Success Stories

—Atari Games and
Reinforcement Learning


45 4


POWER
ON
OFF
COLOR
B-W
TV TYPE
VIDEO COMPUTER SYSTEM™
GAME SELECT
GAME RESET
ATARI®

“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.”

Mnih, V. (2013): “Playing Atari with Deep Reinforcement Learning”. https://arxiv.org/pdf/1312.5602v1.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.

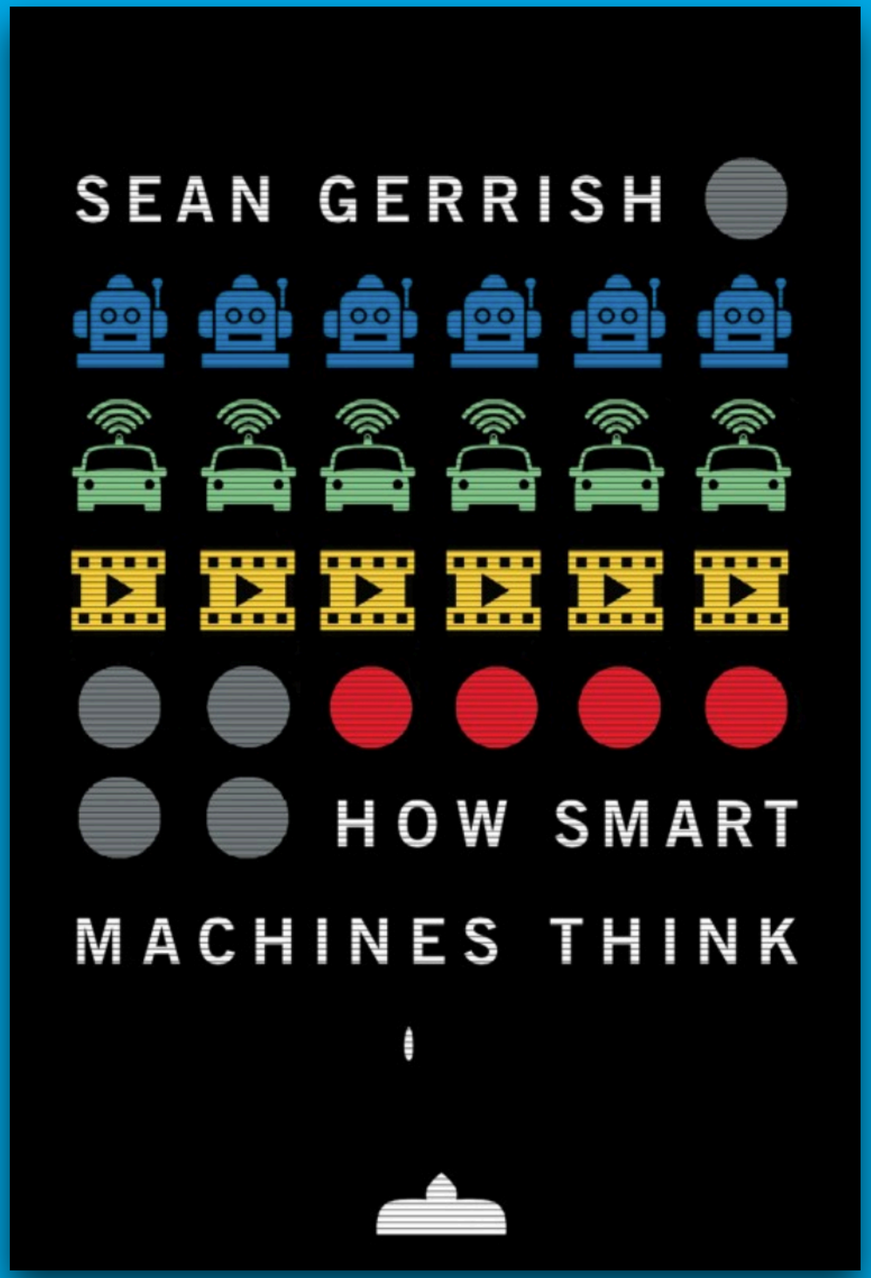
1 Introduction

Learning to control agents directly from high-dimensional sensory inputs like vision and speech is one of the long-standing challenges of reinforcement learning (RL). Most successful RL applications that operate on these domains have relied on hand-crafted features combined with linear value functions or policy representations. Clearly, the performance of such systems heavily relies on the quality of the feature representation.

Recent advances in deep learning have made it possible to extract high-level features from raw sensory data, leading to breakthroughs in computer vision [11, 22, 16] and speech recognition [6, 7]. These methods utilise a range of neural network architectures, including convolutional networks, multilayer perceptrons, restricted Boltzmann machines and recurrent neural networks, and have exploited both supervised and unsupervised learning. It seems natural to ask whether similar techniques could also be beneficial for RL with sensory data.

However reinforcement learning presents several challenges from a deep learning perspective. Firstly, most successful deep learning applications to date have required large amounts of hand-labelled training data. RL algorithms, on the other hand, must be able to learn from a scalar reward signal that is frequently sparse, noisy and delayed. The delay between actions and resulting rewards, which can be thousands of timesteps long, seems particularly daunting when compared to the direct association between inputs and targets found in supervised learning. Another issue is that most deep learning algorithms assume the data samples to be independent, while in reinforcement learning one typically encounters sequences of highly correlated states. Furthermore, in RL the data distribution changes as the algorithm learns new behaviours, which can be problematic for deep learning methods that assume a fixed underlying distribution.

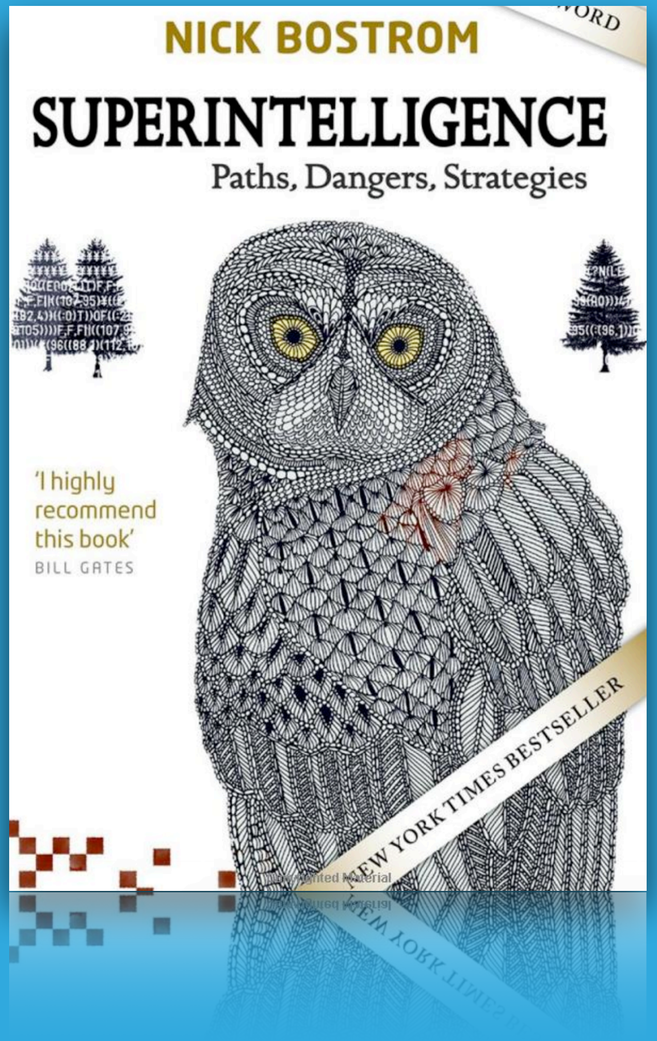
This paper demonstrates that a convolutional neural network can overcome these challenges to learn successful control policies from raw video data in complex RL environments. The network is trained with a variant of the Q-learning [26] algorithm, with stochastic gradient descent to update the weights. To alleviate the problems of correlated data and non-stationary distributions, we use



Success Stories about Deep Learning and Deep Reinforcement Learning:

- Self-Driving Cars
- Recommendation Engines
- Playing Atari Games
- Image Recognition & Classification
- Speech Recognition
- Playing the Game of Go

—Go and AlphaGo



"Go-playing programs have been improving at a rate of about 1 dan/year in recent years. If this rate of improvement continues, they might beat the human world champion in about a decade."

Nick Bostrom (2014): Superintelligence.

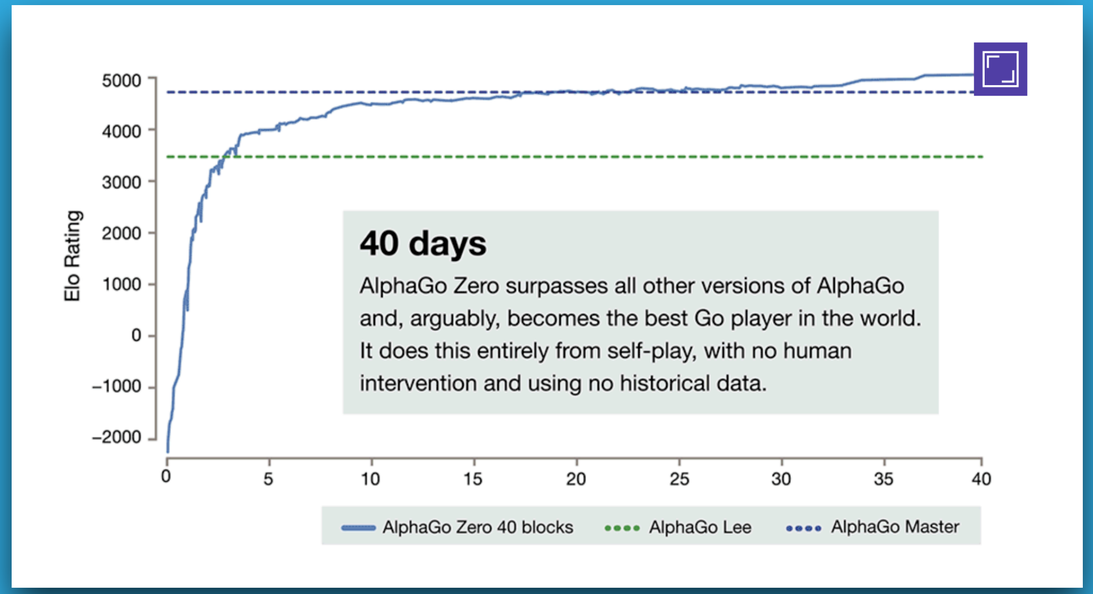
The story of AlphaGo so far

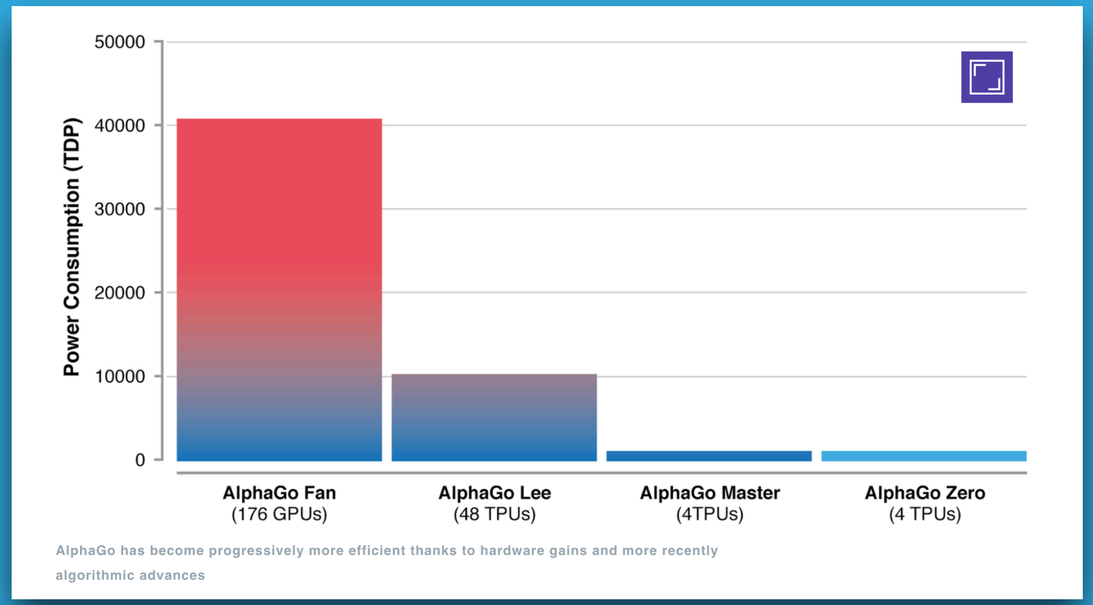
AlphaGo is the first computer program to defeat a professional human Go player, the first program to defeat a Go world champion, and arguably the strongest Go player in history.

AlphaGo's first formal match was against the reigning 3-times European Champion, Mr Fan Hui, in October 2015. Its 5-0 win was the first ever against a Go professional, and the results were published in full technical detail in the international journal, **Nature**. AlphaGo then went on to compete against legendary player Mr Lee Sedol, winner of 18 world titles and widely considered to be the greatest player of the past decade.

AlphaGo's 4-1 victory in Seoul, South Korea, in March 2016 was watched by over 200 million people worldwide. It was a landmark achievement that experts agreed was a decade ahead of its time, and earned AlphaGo a 9 dan professional ranking (the highest certification) - the first time a computer Go player had ever received the accolade.

During the games, AlphaGo played a handful of **highly inventive winning moves,** several of which - including move 37 in game two - were so surprising they overturned hundreds of years of received wisdom, and have since been examined extensively by players of all levels. In the course of winning, AlphaGo somehow taught the world completely new knowledge about perhaps the most studied and contemplated game in history.



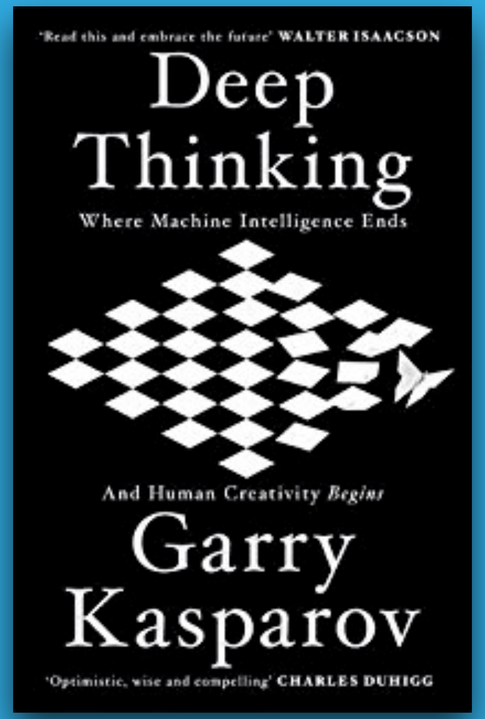


AlphaGo has become progressively more efficient thanks to hardware gains and more recently algorithmic advances

Podcast with David Silver

Netflix Documentation

—Chess, Deep Blue & AlphaZero



"It was a pleasant day in Hamburg in June 6, 1985, … Each of my opponents, all thirty-two of them, was a computer. … it didn't come as much of a surprise, …, when I achieved a perfect 32—0 score."

"Twelve years later I was in New York City fighting for my chess life. Against just one machine, a $10 million IBM supercomputer nicknamed 'Deep Blue'."

"Jump forward another 20 years to today, to 2017, and you can download any number of free chess apps for your phone that rival any human Grandmaster."

AlphaZero: Shedding new light on the grand games of chess, shogi and Go

"Traditional chess engines — including the world computer chess champion Stockfish and IBM's ground-breaking Deep Blue — rely on **thousands of rules and heuristics handcrafted by strong human players** that try to account for every eventuality in a game. …

AlphaZero takes a totally different approach, replacing these hand-crafted rules with a **deep neural network** and **general purpose algorithms** that know nothing about the game beyond the basic rules."

"The amount of **training** the network needs depends on the style and complexity of the game, taking **approximately 9 hours for chess**, 12 hours for shogi, and 13 days for Go."

"In Chess, for example, it searches **only 60 thousand positions** per second in chess, compared to roughly 60 million for Stockfish."

Source: http://deepmind.com

Fundamental Notions

Reinforcement Learning

"Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment in order to maximize some notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning."

https://en.wikipedia.org/wiki/Reinforcement_learning

Environment
The environment defines the problem at hand. This can be a computer game to be played or a financial market to be traded in.

State
A state can be thought of as a vector containing all relevant parameters describing the environment at a certain point (in time). In a computer game this might be the whole screen with all its pixels. In a financial market, this might include current and historical price levels, financial indicators such as moving averages, macroeconomic variables, and so on.

Agent

The term agent subsumes all elements of the RL algorithm that interacts with the environment and that learns from these interactions. In a gaming context, the agent might represent a player playing the game. In a financial context, the agent could represent a trader (trading bot) placing bets on rising or falling markets.

Action

An agent can choose one action from a (limited) set of allowed actions. In a computer game, movements to the left or right might be allowed actions, while in a financial market going long or short could be admissible.

Reward
Depending on the action an agent chooses, a reward (or penalty) is awarded. For a computer game, points are a typical reward. In a financial context, profit (or loss) is a standard reward.

Target
The target specifies what the agent tries to maximize. In a computer game this in general is the score reached by the agent. For a financial trading bot, this might be the trading profit.

OpenAI Gym

About OpenAI

OpenAI is a research laboratory based in San Francisco, California. Our mission is to ensure that artificial general intelligence benefits all of humanity. The OpenAI Charter describes the principles that guide us as we execute on our mission.

Environments Documentation

Gym

Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball.

View documentation ›

View on GitHub ›

RandomAgent on Pendulum-v0

RandomAgent on SpaceInvaders-v0

Q-Learning

Reward Function

The reward function *R* assigns to each state-action (*S*, *A*) pair a numerical reward.

$$R : S \times A \to \mathbb{R}$$

Action Policy

An action policy *Q* assigns to each state *S* and allowed action *A* a numerical value. The numerical value is composed of the **immediate reward** of taking action *A* and the **discounted delayed reward** — given an optimal action taken in the subsequent state.

$$Q : S \times A \to \mathbb{R},$$

$$Q\left(S_t, A_t\right) = R\left(S_t, A_t\right) + \gamma \cdot \max_a Q\left(S_{t+1}, a\right)$$

Representation

In general, the optimal action policy Q can not be specified in closed form (e.g. in the form of a table). Therefore, Q-learning relies in general on approximate representations for the optimal policy Q.

Neural Network

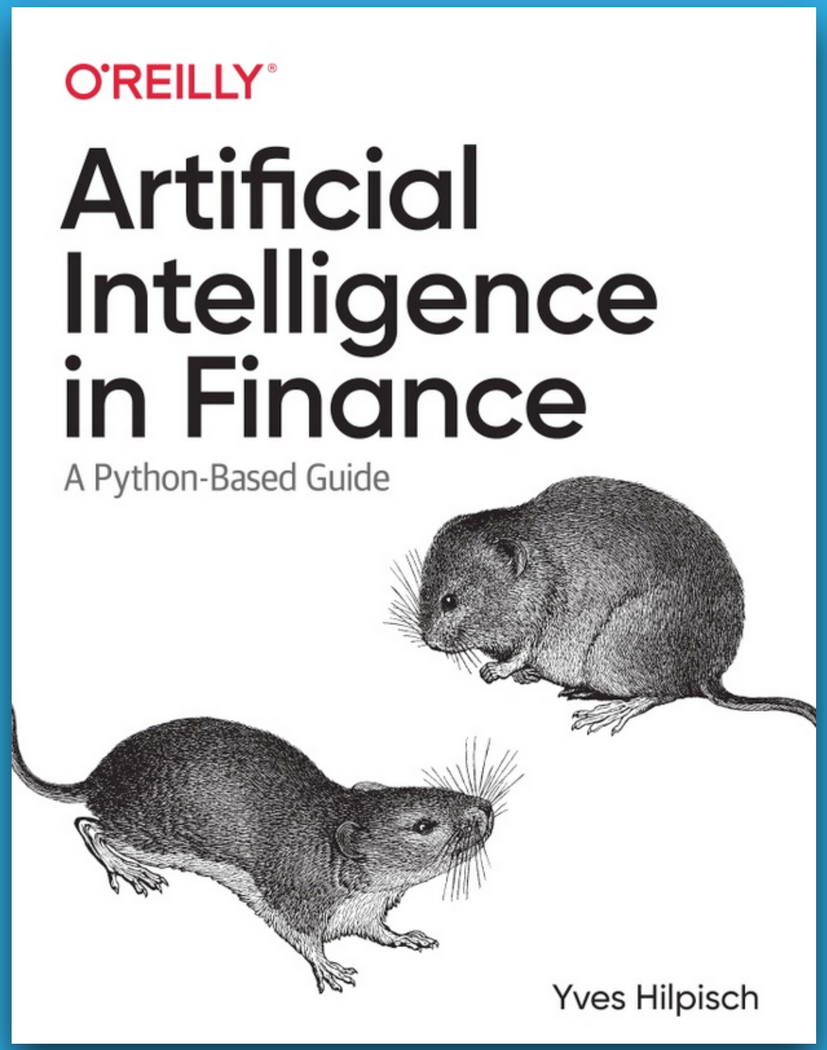
Due to the approximation capabilities of neural networks (“Universal Approximation Theorems”), neural networks are typically used to represent optimal action policies Q. Features are the parameters that describe the state of the environment. Labels are values attached to each allowed action.

Exploration
This refers to actions taken by an agent that are random in nature. The purpose is to explore random actions and their associated values beyond what the current optimal policy would dictate.

Exploitation
This refers to actions taken in accordance with the current optimal policy.

Replay
This refers to the (regular) updating of the optimal action policy given past and memorized experiences (by re-training the neural network).

Demo



Chapter 9
— Reinforcement Learning

Introductory Examples

- **CartPole Game**
- **Algorithmic Trading**

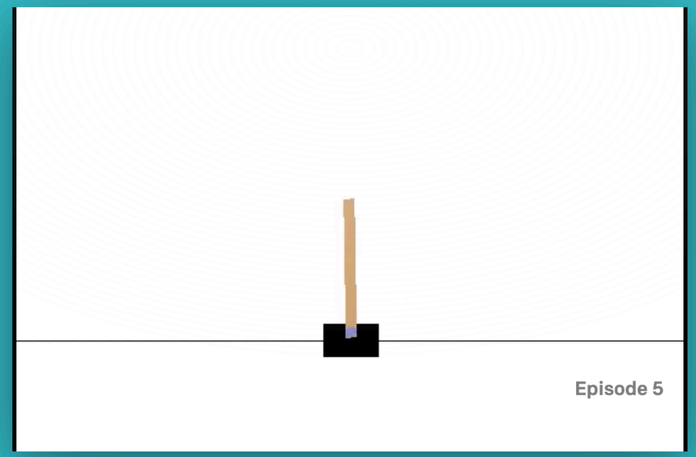
Gym

gym.openai.com/envs/CartPole-v1/

Environments Documentation

CartPole-v1

A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of +1 or -1 to the cart. The pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for every timestep that the pole remains upright. The episode ends when the pole is more than 15 degrees from vertical, or the cart moves more than 2.4 units from the center.



RandomAgent on CartPole-v1

This environment corresponds to the version of the cart-pole problem described by Barto, Sutton, and Anderson [Barto83].

[Barto83] *AG Barto, RS Sutton and CW Anderson, "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problem", IEEE Transactions on Systems, Man, and Cybernetics, 1983.*

VIEW SOURCE ON GITHUB

Environments Documentation

OpenAI

The Python Quants GmbH

Dr. Yves J. Hilpisch
+49 3212 112 9194
http://tpq.io | team@tpq.io
@dyjh