



Q search

📄 首页

📄 资讯

📁 项目

🚩 招募

技术 | 强化学习入门以及代码实现

Xtecher原创 | 行业洞察

👁 10900 👍 1573 2017-08-26 🔥

AI科技大本营 (微信
ID:rgznai100)

Xtecher特稿作者

[关注](#)

专访 | 商汤HPC负责人刘文志 (风辰) : 未来战略的两大方向及招人的4个标准

苹果最新博文剑指汉字手写识别！
专家回应：并没有技术含量

四个月就能学成机器学习？我们认真准备了一下该怎样做到

[More >>](#)

AI科技大本营



介绍

目前，对于全球的科学家而言，“如何去学习一种新技能”已经成为最基本的研究课题之一。解决这个问题的心愿显而易见——如果能够解决这个问题，那么人类就有望做到某些从未想过的事情。换句话说，我们可以训练机器去做更多原本人类要做的工作，迎来真正的人工智能时代。虽然，对于上述问题，目前我们还没有一个完整的回答，但有一些事情是十分明确的。不考虑技能方面的学习，我们首先的是在与环境的交互过程中进行学习。不管是学习开车，还是婴儿学习走路，学习的基础都是与环境的交互过程。在互动中学习是所有学习理论以及智力发展理论的最根本的概念。

强化学习

今天，我们将探讨强化学习。其中与环境的交互是深度学习的基础，通常伴有明确的目的。有些人认为，强化学习是实现强人工智能的真正希望。这么说确实没错，因为强化学习拥有着巨大的潜力。

目前，有关强化学习的研究正在快速增长，人们为不同的应用程序生成了各种各样的学习算法。因此，熟悉强化学习的技术变得尤其重要。如果你还不是很熟悉强化学习，那么我建议你去看我我以前有关强化学习的文章和一些开源的强化学习平台。



Xtecher



Q search



首页



资讯



项目



招募

如果你已经掌握并理解了强化学习的基础知识，那么请继续阅读这篇文章。读完本文之后，你将会对强化学习以及代码实现过程有一个透彻的了解。

注：在代码实现部分，我们假设你已经有了 Python 的基本知识。如果你还不知道 Python，那么你应该先看看这篇教程。

<https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-learn-data-science-python-scratch-2/>

目录

强化学习问题的表现形式

与其他机器学习方法的比较

解决强化问题的框架

强化学习的实现

增加强化学习的复杂性

深入了解强化学习的最新进展

其他资源

1. 强化学习问题的表现形式

强化学习不仅需要学习做什么，也需要学习如何根据与环境的交互采取相应的行动。强化学习的最终结果，就是要实现系统回报信号的最大化。学习者事先并不知道要去执行什么行为，需要自己去发现哪种行动能产生最大的回报。让我们通过一个简单的例子来解释一下。

我们以一个正在学习走路的孩子为例进行讲解。



search



首页

资讯

项目

招募



以下是孩子在学习走路时所采取的步骤：

孩子关注的第一件事，就是观察身边的人是如何走路的。身边的人使用两条腿走路，一次走一步，一步一步按照次序往前走。孩子会抓住这个概念，然后努力去模仿这个过程。

但很快他/她又会明白，在走路之前，必须先站起来！在学习走路的过程中，站立对于孩子来说是一个挑战。因此，孩子试图自己站起来，他/她不断跌倒，但是仍然决定站起来。

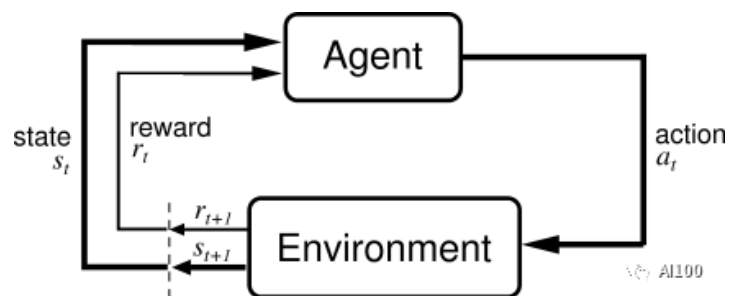
然而接下来，还有另外一个挑战需要应付。站起来是相对容易的，但是要保持站立状态就是另一个挑战了。在狭小的空间中找到支撑，孩子就能成功保持站立状态。

现在，孩子的真正任务就是开始学习走路了。但是学习走路说起来很容易，而实际做起来就不是那么容易了。在孩子的大脑中需要处理很多事情，比如平衡身体、决定接下来放哪个脚、放在哪里。



这听起来像是一个很困难的任务，对吗？事实上，这确实是一个挑战，先要学习站立，然后才能学习行走。但是，现在我们不都学会了走路嘛，再也不会被这个问题所困扰了。现在你应该可以明白，为什么学习走路对于孩子来说非常困难了。

让我们用具体的形式来表述上面的例子。例子所要陈述的问题是“走路问题”，其中孩子是一个试图通过采取行动(走路)来操纵环境(在地上走路)的智能体，他/她试图从一个状态(即，他/她走的每一步)转移到另一个状态。当他/她完成任务的一个子模块(即孩子走了几步)时，孩子会获得奖励(比如，一些巧克力)，但是当他/她不会走路时，他/她不会收到任何巧克力(这是一个负反馈过程)。这就是对强化学习问题的简单描述。



下面的链接，是介绍强化学习的很好的视频。

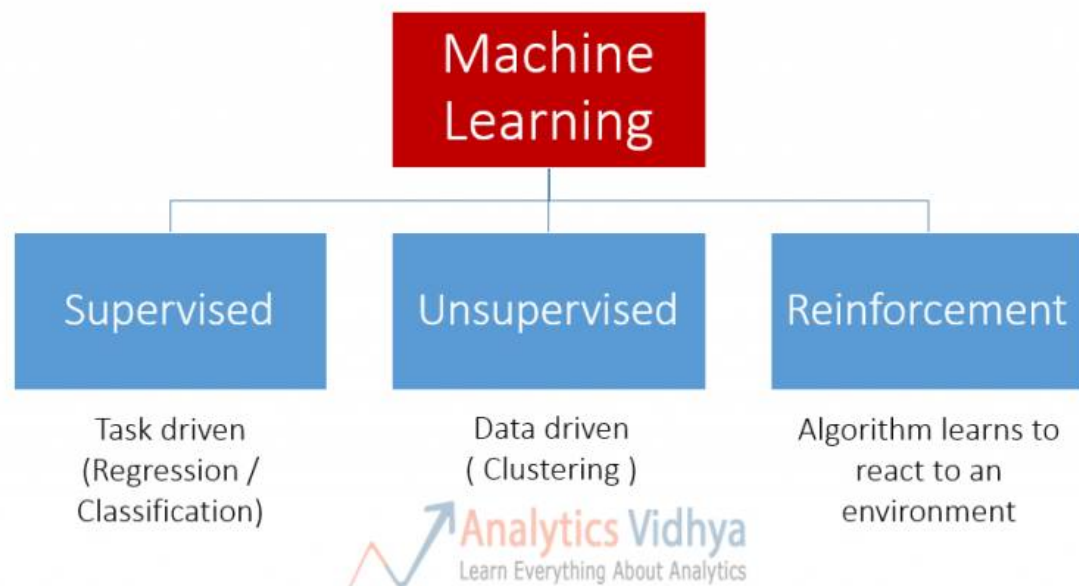
<https://youtu.be/m2weFARiE8>

2. 与其他机器学习方法的比较

强化学习是机器学习算法中的一类。以下是有关机器学习算法类型的描述。



Types of Machine Learning



让我们比较一下强化学习算法和其他类型算法之间的区别：

监督学习与强化学习：在监督学习中，其外部有一个“监督主管”，这个“监督主管”拥有环境方面的知识，并且与智能体一起共享这个知识，从而帮助智能体完成任务。但是这存在一些问题，因为存在如此多的子任务之间的组合，智能体要实现其目标的话，这些组合都是可以利用的。所以，创建一个“监督主管”几乎是不切实际的。例如，在象棋游戏中，存在数万个移动方法。因此，创建玩法知识库的任务将会单调而乏味。有一个更加合理可行的方法，就是设法从自己的经历中学习，并以此获得所需的知识。这就是强化学习和监督学习的主要区别。在监督学习和强化学习中，输入和输出之间都存在映射。但是在强化学习中，存在的是对智能体的奖励反馈函数，而不是像监督学习一样，直接告诉智能体最终的答案。

无监督学习与强化学习：在强化学习中，有一个从输入到输出的映射过程，但是这个过程在无监督学习中是不存在的。在无监督学习中，主要任务是找到一种最基本的模式，而不是映射关



search



首页



资讯



项目



招募

系。例如，如果任务是向用户推荐新闻文章，那么无监督学习算法首先将会查看该人以前读过的类似文章，并把它们推荐给其他人。而强化学习算法则是，通过用户阅读的某文章，不断获得用户的反馈，从而构建一个“知识图谱”，推测用户喜欢的文章。

还有第四种类型的机器学习，称为半监督学习。**半监督学习本质上是监督学习和无监督学习的组合。**它不同于强化学习，而是与监督学习相类似。半监督学习会直接给出参照答案，而强化学习不会。

3. 解决强化学习问题的框架

为了理解解决强化学习问题的过程，让我们通过一个经典的例子来解释一下强化学习问题——多臂赌博机。首先，我们需要了解探索与开发的基本问题，然后去定义解决强化学习问题的框架。

**xtecher**

Q search



首页

资讯

项目

招募







如上图中的老虎机，假设你已经在老虎机上面玩了很多次了。

现在你想做的是从老虎机上面获得最大的回报，并且尽可能快地获得这个回报。你会怎么做呢？





Q search

 首页 资讯 项目 招募

一个比较天真的想法是，只选择一个老虎机，然后一整天都去玩它。这听起来非常无聊，但老虎机确实可能会给你一些“回报”，即让你赢钱。使用这种方法，你可能中奖的概率大约是0.00000.....1。也就是说，大多数时间你可能知识坐在老虎机面前亏钱。正式说明一下，我们可以将其定义为一种纯粹的开发方法。但是这是最佳选择吗？答案当然是否定的。

让我们看看另外一种方法。我们可以拉每个老虎机的拉杆，并且向上帝祈祷，让我们至少打中一个。当然，这是另一种天真的想法，它只会让你一天都在拉动拉杆，但只是给你一点点报酬。正式说明一下，这种方法只是一种纯粹的探索方法。

这两种方法都不是最优的，我们必须在它们之间找到适当的平衡点，以获得最大的回报。这就是强化学习中“探索VS开发”的两难选择。

首先，我们来正式地定义一下解决强化学习问题的框架，然后列出可能的方法来解决这个问题。

马尔科夫决策过程

在强化学习场景中，用于解决问题的数学框架叫做马尔科夫决策过程。这可以被设计为：

状态集合： S

动作集合： A

奖励函数： R

策略： π

价值： V

要想从开始状态转变到结束状态(S)，我们必须采取一定的行动(A)。每当我们采取一个行动之后，我们都会得到一定的回报作为奖励。当然，所获得的奖励的性质(正面奖励还是负面奖励)是由我们的行动决定的。

策略集合(π)取决于我们的动作集合，而我们得到的回报将会决定我们的价值(V)。在这里，我们的任务就是选择正确的策略来最大化我们的价值。所以我们必须最大化下面的方程：



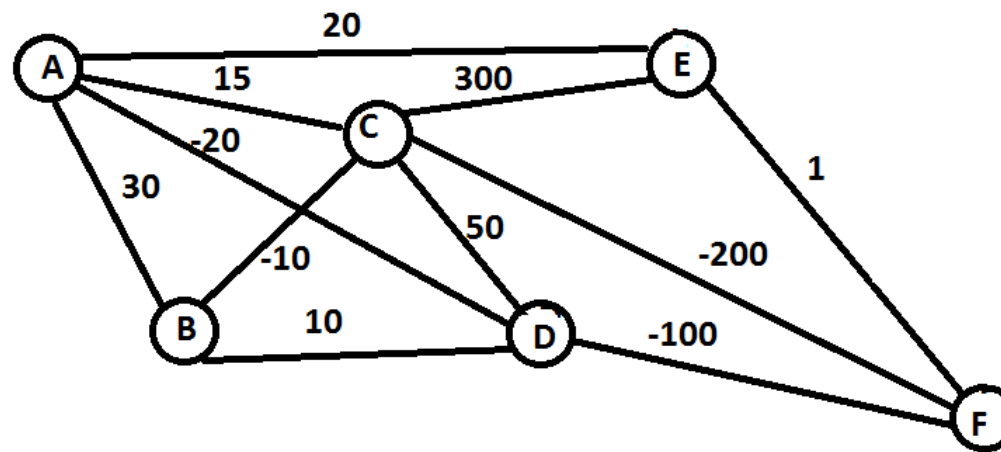
$$E(r_t | \pi, s_t)$$

AI100

在时间点 t ，我们必须最大化 S 中所有可能的值。

旅行推销员问题

让我们通过另外一个例子来说明一下。



AI100

这个问题是一系列旅行商(TSP)问题的代表。任务是以尽可能低的成本从地点A到达地点F。两个字母之间的每条边上的数字表示两地之间的花费。如果这个值是负数，那么表示经过这条路，你会得到一定的报酬。这里我们定义的价值(Value)是运用选择的策略走完整个路程时，所获得的总价值。

这里说明一下符号：



search



首页

资讯

项目

招募

所有状态的集合相当于图中的节点： $\{A, B, C, D, E, F\}$

动作集合相当于是从一个地点到另一个地点的过程： $\{A \rightarrow B, C \rightarrow D, \text{etc}\}$

奖励函数相当于边上的值，例如“成本”

策略函数相当于完整的路径规划，比如： $\{A \rightarrow C \rightarrow F\}$

现在假设你在地点A，唯一你能看见的路就是你的下一个目的地(也就是说，你只能看见B，D，C，E)，而别的地点你是不知道的。

你可以采取贪心算法，选择当前状态下最有利的步骤，也就是说，从 $\{A \rightarrow (B, C, D, E)\}$ 中选择采取 $\{A \rightarrow D\}$ 这种方法。同样地，如果现在你所在的地点是D，想要到达地点F，你就可以从 $\{D \rightarrow (B, C, F)\}$ 中采取 $\{D \rightarrow F\}$ 这个方法，这可以让你得到最大的报酬。因此，我们选取这一条路。

至此，我们的策略就是 $\{A \rightarrow D \rightarrow F\}$ ，这种策略所获得的回报是-120。

恭喜！你刚刚就已经实现了强化学习算法。这种算法被称之为 epsilon 贪婪算法。这是一种通过逐步测试而解决问题的贪婪算法。现在，如果见你(推销员)想再次从地点A到达地点F的话，你大概会一直选择这条路线了。

其他旅游方式？

你能猜出我们的策略是属于哪个类别(纯粹的探索VS纯粹的开发)吗？

请注意，我们采取的策略并不是最佳的策略。要想寻找最佳的策略，我们必须更具“探索精神”。

在这里，我们采取的方法是局域策略学习，我们的任务是在所有可能的策略中找到最佳的策略。有很多的方法都可以解决这个问题，在这里，我简要的列出一些主要的内容：

策略优先：我们的重点是找到最佳的策略

回报优先：我们的重点是找到最佳的回报价值，即累计奖励

行动优先：我们的重点是在每个步骤上采取最佳行动

在以后的文章中，我会深入讨论强化学习算法。到那时，你可以参考这篇关于强化学习算法的调研论文



Q search



首页



资讯



项目



招募

<https://www.jair.org/media/301/live-301-1562-jair.pdf>

4. 强化学习的实现

接下来，我们将使用深度Q学习算法。Q学习是一种基于策略的学习算法，它的函数表示和神经网络近似。Google就是采用了这种算法打败了Atari游戏。

让我们看看Q学习的伪代码：

初始化价值表 ' $Q(s, a)$ '。

观察当前的状态值 ' s '。

基于动作选择一个策略(例如，epsilon贪婪)作为该状态选择的动作。

根据这个动作，观察回报价值 ' r ' 和下一个新的状态 ' s '。

使用观察到的奖励和可能的下一个状态所获得的最大奖励来更新状态的值。根据上述公式和参数进行更新。

将状态设置为新的状态，并且重复上述过程，直到达到最终状态。

Q学习的简单描述可以总结如下：



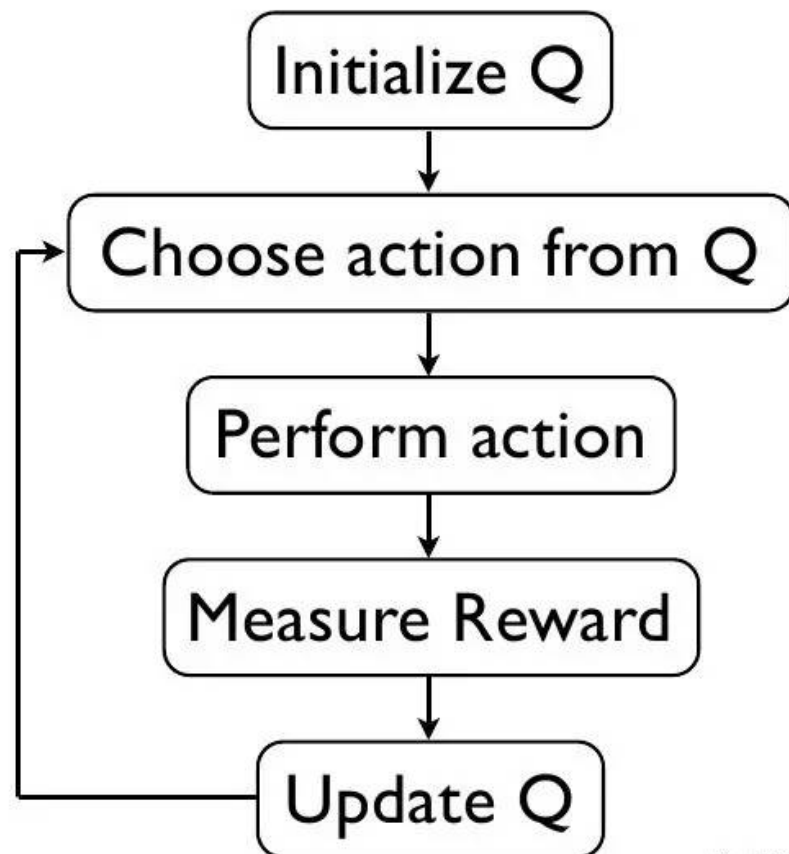
Q search

首页

资讯

项目

招募



AI100

我们首先来了解一下 Cartpole 问题，然后再继续编写我们的解决方案。

当我还是一个孩子的时候，我记得我会挑选一根木棍，并试图用一只手指去使它保持平衡。过去，我和我的朋友有过这样一个比赛：谁能让木棍保持平衡的时间更久，谁就能得到一块巧克力作为奖励。

以下链接是一个简短的视频，描述的是真正的 Cart-Pole 系统。

<https://youtu.be/XiigTGKZfks>



Q search



首页

资讯

项目

招募

让我们开始编写代码吧！

在开始编写之前，我们需要先安装几个软件。

步骤一：安装keras-rl包

在终端中，你可以运行以下命令：

```
git clone https://github.com/matthiasplappert/keras-rl.git
cd keras-rl
python setup.py install
```

步骤二：为CartPole环境安装依赖程序

我们假设你已经安装好了pip，那么你只需要使用以下命令进行安装：

```
pip install h5py
pip install gym
```

步骤三：开始编写代码

首先我们需要导入一些我们需要的模块。

```
import numpy as np
import gym

from keras.models import Sequential
from keras.layers import Dense, Activation, Flatten
from keras.optimizers import Adam
```



search



首页



资讯



项目



招募

```
from rl.agents.dqn import DQNAgent
from rl.policy import EpsGreedyQPolicy
from rl.memory import SequentialMemory
```

然后，设置相关变量。

```
ENV_NAME = 'CartPole-v0'

# Get the environment and extract the number of actions available in the Cartpole
problem
env = gym.make(ENV_NAME)
np.random.seed(123)
env.seed(123)
nb_actions = env.action_space.n
```

之后，我们来构建一个非常简单的单层神经网络模型。

```
model = Sequential()
model.add(Flatten(input_shape=(1,) + env.observation_space.shape))
model.add(Dense(16))
model.add(Activation('relu'))
model.add(Dense(nb_actions))
model.add(Activation('linear'))
print(model.summary())
```

接下来，我们对智能体进行相关的配置并进行编译。我们运用的策略是 Epsilon 贪婪算法。同时，我们还将我们的存储空间设置为序列存储，因为我们需要存储执行操作后的结果以及每一个操作所获得的奖励。



🔍 search



📄 首页

📄 资讯

📁 项目

🚩 招募

```
policy = EpsGreedyQPolicy()
memory = SequentialMemory(limit=50000, window_length=1)
dqn = DQNAgent(model=model, nb_actions=nb_actions, memory=memory,
nb_steps_warmup=10,
target_model_update=1e-2, policy=policy)
dqn.compile(Adam(lr=1e-3), metrics=['mae'])
```

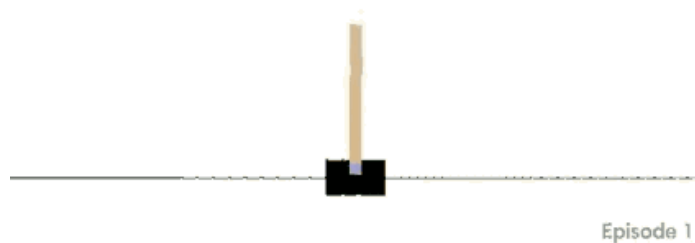
Okay, now it's time to learn something! We visualize the training here for show, but this slows down training quite a lot.

```
dqn.fit(env, nb_steps=5000, visualize=True, verbose=2)
```

现在，让我们来测试一下我们的强化学习模型。

```
dqn.test(env, nb_episodes=5, visualize=True)
```

下图是模型的输出结果：



瞧，你刚刚就建立了一个强化学习机器人！



5. 增加复杂性

现在，你已经看到了一个强化学习的基本的实现过程，让我们开始学习更多的问题吧，每次增加一点点复杂性。

汉诺塔问题



有些人可能还不知道汉诺塔问题：汉诺塔问题是在1883年发明的，由3根木棍和一系列大小不一的圆盘组成(比如上图中的3个)。从最左侧的木棍开始，目的是以最少的移动次数，把最左边的圆盘移动到最右边的圆盘上。

要解决这个问题，我们需要先从其状态开始谈起：

初始状态：三个圆盘都在最左边的木棍上(从上到下，依次编号为1,2,3)

结束状态：三个圆盘都在最右边的木棍上(从上到下，依次编号为1,2,3)

所有可能的状态：

这里，我们会得到27种可能的状态：



xtecher



Q search



首页



资讯



项目



招募

All disks in a rod	One disk in a Rod	(13) disks in a rod	(23) disks in a rod	(12) disks in a rod
(123)**	321	(13)2*	(23)1*	(12)3*
(123)	312	(13)*2	(23)*1	(12)*3
** (123)	231	2(13)*	1(23)*	3(12)*
	132	*(13)2	*(23)1	*(12)3
	213	2*(13)	1*(23)	3*(12)
	123	*2(13)	*1(23)	*3(12)

其中, (12)3* 表示, 圆盘1和圆盘2在最左边的木棍上面(从上往下编号), 圆盘3在中间那个木棍上面, 最右边的木棍没有圆盘。

数值奖励：

由于我们想要以最少的移动步数解决这个问题, 所以, 我们可以将每次移动的奖励设为 -1。

策略：

现在, 如果不考虑任何的技术细节, 那么从前一个状态过渡到下一个状态过程中, 我们可以预测出其可能的状态。比如, 当数值奖励为-1时, 状态 (123)** 会转移到状态 (23)1*, 或者状态 (23)*1。

如果你现在看到了一个同时进行的状况, 那么上面提到的这27个状态的每一个都可以表示成一个类似于旅行商问题的图, 我们可以通过实验各种状态和路径来找到最优的解决方案。

3 x 3 魔方问题



search



首页



资讯



项目



招募

虽然我可以为你解决这个问题，但是我想让你自己去解决这个问题。你可以按照我上述提到的思路来进行，应该就可以解决了。

从定义开始状态和结束状态开始，接下来，定义所有可能的状态、转换过程、奖励和策略。最后，你应该就可以使用相同的方法来构建自己的解决方案了。

6. 深入了解强化学习的最新进展

你可能已经意识到，魔方的复杂性要比汉诺塔问题高出很多倍，可供选择的数量要比汉诺塔问题多得多！现在，让我们来想象一下棋类游戏的状态和选择的策略数量吧，比如围棋。最近，Google DeepMind公司创建了一个深度强化学习算法，并且打败了李世石。

在深度学习不断取得成功的过程中，人们关注的焦点正逐渐发生变化，视图应用深度学习来解决强化学习问题。最近有一则消息，如洪水般向我们涌来：由Google DeepMind创建的深度强化学习算法打败了李世石。在视频游戏中也出现了类似的情况，深度强化学习算法达到了人类水平的准确性，甚至在某些游戏中，超越了人类。研究和实践仍然需要共同发展，工业界和学术界需要共同努力，以建立出更好地、能自我学习的机器人。



Q search



首页

资讯

项目

招募



<http://www.tomshardware.com/news/alphago-defeats-sedol-second-time,31377.html>

以下是几个已经应用强化学习的重要领域：

博弈论和多个智能体交互

机器人

计算机网络

车载导航

医学

工业物流

我们可以看到，仍然有许许多多的领域尚未得以开发，加之人们对深度学习的热情非常高涨，我相信以后肯定会有更多的突破！

下面是一则最近的消息：



search



首页



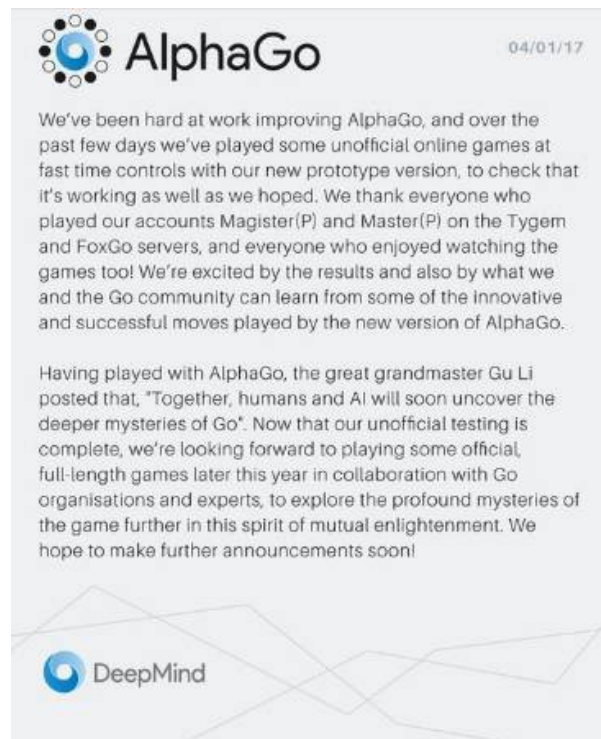
资讯



项目



招募



Demis Hassabis
@demishassabis

Follow

Excited to share an update on #AlphaGo!

11:00 PM - 4 Jan 2017

2,984 2,895

AI100

7. 其他资源

我希望你现在能够深入了解强化学习的工作原理。这里还有一些额外的资源，可以帮助你学习更多有关强化学习的内容。

有关强化学习的视频

<https://www.analyticsvidhya.com/blog/2016/12/21-deep-learning-videos-tutorials-courses-on-youtube-from-2016/>



xtecher



search



首页



资讯



项目



招募

有关强化学习的书籍

<https://webdocs.cs.ualberta.ca/~sutton/book/bookdraft2016sep.pdf>

GitHub上有关强化学习的资料库

<https://github.com/aikorea/awesome-rl>

David Silver的强化学习课程

https://www.youtube.com/playlist?list=PLV_1KI9mrSpGFoaxoL9BCZeen_s987Yxb

本文作者 Faizan Shaikh 是 AnalyticsVidhya 的数据科学实习生，目前致力于研究深度学习，目标是利用自己的技能，推动AI研究的进展。

xtecher

Innovation That Inspires

Xtecher，发现最具潜力的科技项目。寻找对科技、媒体或品牌痴迷的人。

关注未来的人 都关注了Xtecher



联系我们

public@xtecher.com

北京市朝阳区工体北路8号三里屯SOHO A座1201

© 2017 Xtecher 版权所有 | 京ICP备15019674号 - 1

关于我们 | 加入我们 | 联系我们 | 校园大使计划