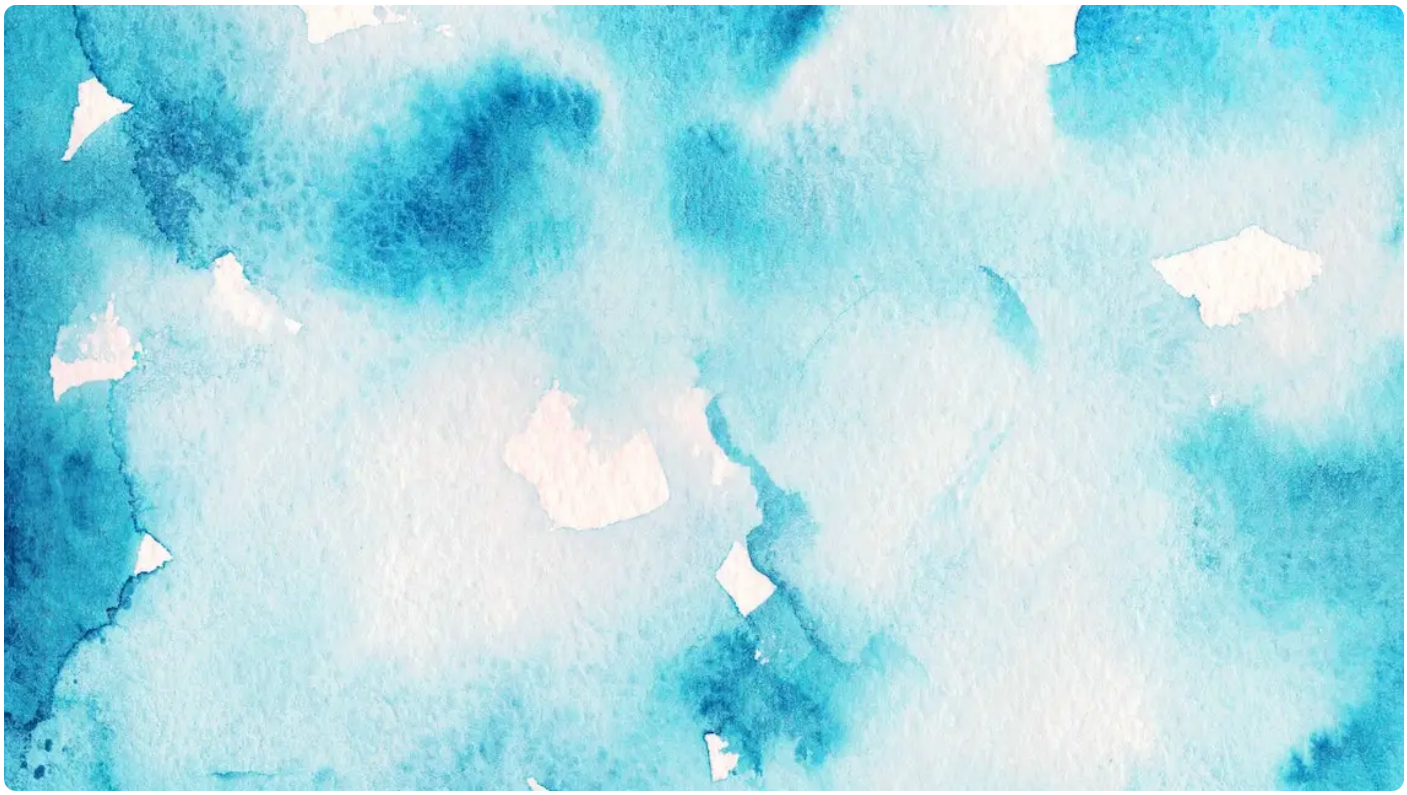
06 | 免费的宝库: 什么是网络爬虫？

2022-10-22 郑建勋 来自北京

《Go进阶 · 分布式爬虫实战》

课程介绍 >



讲述：郑建勋

时长 21:14　大小 19.39M

你好，我是郑建勋。

网络爬虫（Web Crawler）又称为网络蜘蛛（Web Spider），是一种自动获取互联网信息的网络机器人（Web Robot）。想想还真是非常形象，蜘蛛在相互连接的网站中，辛苦地从一个网站爬到另一个网站获取信息，又像一个不知疲倦的打工人。

互联网是一个充满了庞大免费数据的地方，但是数据本身并不产生价值，有价值的是从数据中提炼出来的知识与智慧。就像金块一样，这些零散的数据可以被收集、过滤、组合和提炼，生产出极具价值的产品。

凭借正确的知识、技能和一点点创造力，你可以构建一个以爬虫引擎为核心的价值数百亿的商业公司（想想今日头条是如何起家的），这是多么让人兴奋的领域呀。但是网络爬虫合法吗？这一领域需要掌握哪一些知识？基于爬虫可以构建哪些有用的产品？这节课，我们就来深入讨论一下网络爬虫这个领域。

网络爬虫合法吗？

近年来，不断出现爬虫相关的犯罪案件，所以很多人对爬虫敬而远之，甚至将它戏称为“面向监狱编程”。

> 全国首例短视频平台领域网络“爬虫”非法获取用户数据案宣判
> 爬虫失控导致政府网站奔溃，CTO 和程序员双双被判刑
> 新三板挂牌公司涉窃取 30 亿条个人信息，非法牟利超千万元

从各种爬虫犯罪的案例中，我们可以分析出触犯法律的主要原因：

- 用爬虫程序抓取未公开、未授权的个人敏感信息，甚至违规留存、使用、买卖这些隐私数据，严重扰乱市场经济秩序；
- 破解用户密码或利用系统安全漏洞，访问了非公开的系统；
- 恶意对网站进行 DDoS 攻击，超出了服务能够承载的能力导致服务崩溃。

根据《刑法》第二百八十五条“非法获取计算机信息系统数据罪”，以及《中华人民共和国网络安全法》，上面这些行为确实应当受到惩罚。

但是，正如刀本身并不犯法，使用刀去伤人才犯法一样。爬虫这门技术本身是不违法的，只有用爬虫非法窃取用户数据，攻击网站，恶意与竞争对手商业竞争，这些行为才会有法律风险。

要降低使用爬虫的法律风险，我们需要提前确认好爬取的数据是我们有权访问的数据。它主要包括下面几类：

- 数据发布者已决定将数据公开，例如暴露了 API；
- 用户无需创建帐户或登录即可访问的数据；
- 该网站的 robots.txt 文件允许访问的数据。

其实，在合法限度内，爬虫是一门非常必要的技术。例如，搜索引擎本身就是网络爬虫，通过爬取并分类整理网络中海量的信息，用户能够轻松地基于关键词和时间等因素搜索出需要的信息和网站。搜索引擎蕴含着非凡的商业价值，也支撑起了像百度、谷歌、雅虎这样的互联网巨头。

再举个例子，字节跳动的产品今日头条，在初期也是通过爬取各种信息，聚合资讯，智能内容推荐的方式快速占领了市场。这些例子都让人惊叹于网络爬虫的商业潜力。

同时，如果我们去互联网上搜索，也能够看到爬虫相关的书籍汗牛充栋，可以看出这是一个非常热门的技术领域。

简而言之，网络抓取行为并不违法。但是，我们需要遵守一些规则。如果想要提取非公开数据，那网络爬虫就成了违法行为。

下面就让我们来看一看基于爬虫可以做出哪些有商业价值的产品。

网络爬虫的商业价值

信息聚合

刚才也提到了，网络爬虫可以将某一个领域中有价值的信息整合起来：

- 如果你是房地产经纪商，可以通过爬虫来补充自己待售或出租信息的资源；
- 如果你是新闻聚合商或者证券经纪商，可以通过爬虫快速获取各大新闻网站的热点事件，**再通过个性化推荐系统将它们分发给感兴趣的用户；**
- 如果你是一家提供出行服务的聚合商，可以爬取各大酒店、打车服务、机票的定价，并为用户提供某一条件下最低的价格；
- 如果你在做**政策风向研究，可以利用爬虫第一时间收集各地区各部门的政务公告。**

即便是上述最基础的信息整合，也可以预见到巨大的商业潜力。我举一个例子，一条爆炸性的消息会在资本市场中掀起巨大的波澜，这时候，谁能够找到准确的渠道，更快地获取准确有用的信息，谁就能够快人一步，得到丰厚的回报。

行业见解

如果我们将整合的信息稍微分析一下，提炼出有价值的观点，就能进一步增加爬虫的商业价值。

许多公司使用网络爬虫将特定行业的海量信息存储到数据库中，并通过 Excel、Tableau 这样的数据分析软件分析判断，从中获得特定行业的见解。例如，一家公司可能会抓取和分析大量

有关石油价格、出口和进口的数据，经过分析后将他们的见解出售给世界各地的石油公司。一些公司通过网络爬虫获取数据，分析企业的实际经营情况，来判断是否要进行投资或者做空。

预测

爬虫技术本质上获取的是信息，但是要放大信息的价值，更多时候需要对信息进行预测。

例如，知道俄乌开战的信息，能够预测出未来石油、黄金价格的暴涨。又如政府进行舆情监测，需要将特定事件相关的全部新闻资讯采集下来，以监控并预测事件发展态势、及时进行疏导与评估疏导效果。

未来可能是无法预测的，但现实的种种迹象却表明了未来大概率的模样，这就和天气预报一个道理，看起来不可思议，但是蕴含了科学。和你分享一段我的体会，2022 年 1 月 22 日封城前夜，我看到一篇医学文章，文中通过流行病学研究，结合国际旅行概率、潜伏期、发现病例的平均时间，就用概率预测了背后实际可能的发病人数。

Methods

Using internationally reported cases, it is possible to infer the magnitude of comparable cases within Wuhan City that may have occurred thus far.

The total number of cases requiring healthcare is given by:

$$Total\ number\ of\ cases = \frac{number\ of\ cases\ detected\ overseas}{probability\ any\ one\ case\ will\ be\ detected\ overseas}$$

where the probability any one case will be detected overseas (p) is given by:

$$p = daily\ probability\ of\ international\ travel \times mean\ time\ to\ detection\ of\ a\ case$$

The daily probability of travel is calculated by:

$$daily\ probability\ of\ international\ travel = \frac{daily\ outbound\ international\ travellers\ from\ Wuhan}{catchement\ population\ of\ Wuhan\ airport}$$

Finally, the mean time to detection can be approximated by:

$$mean\ time\ to\ detection = incubation\ period + mean\ time\ from\ onset\ of\ symptoms\ to\ detection$$

论文的公式推导

我当时恍然大悟，也意识到了信息和预测的重要性。从这个案例也可以看出，一个事物的爆发需要经历一段时间的发酵，而在这中间其实就有种种迹象预示着未来的走向，如果能够提前预知到，也许就能避免国家和社会的巨量损失。

我再分享另一个我看到的商业机会。之前我提到过，谁能够更快、更准确地获取准确爆炸信息，谁就能够快人一步。但其实有一些信息在成为爆炸信息之前，在成为热搜之前，是经历了信息的传播和发酵的。就好像我知道了一个信息，我需要传播给身边的 10 个人，而这 10 个人又需要传播给他们身边的其他人。如果我们可以在其他人知道之前，追踪到这些可能成为热门的信息，我们就可以先人一步，成为那只春江鸭。

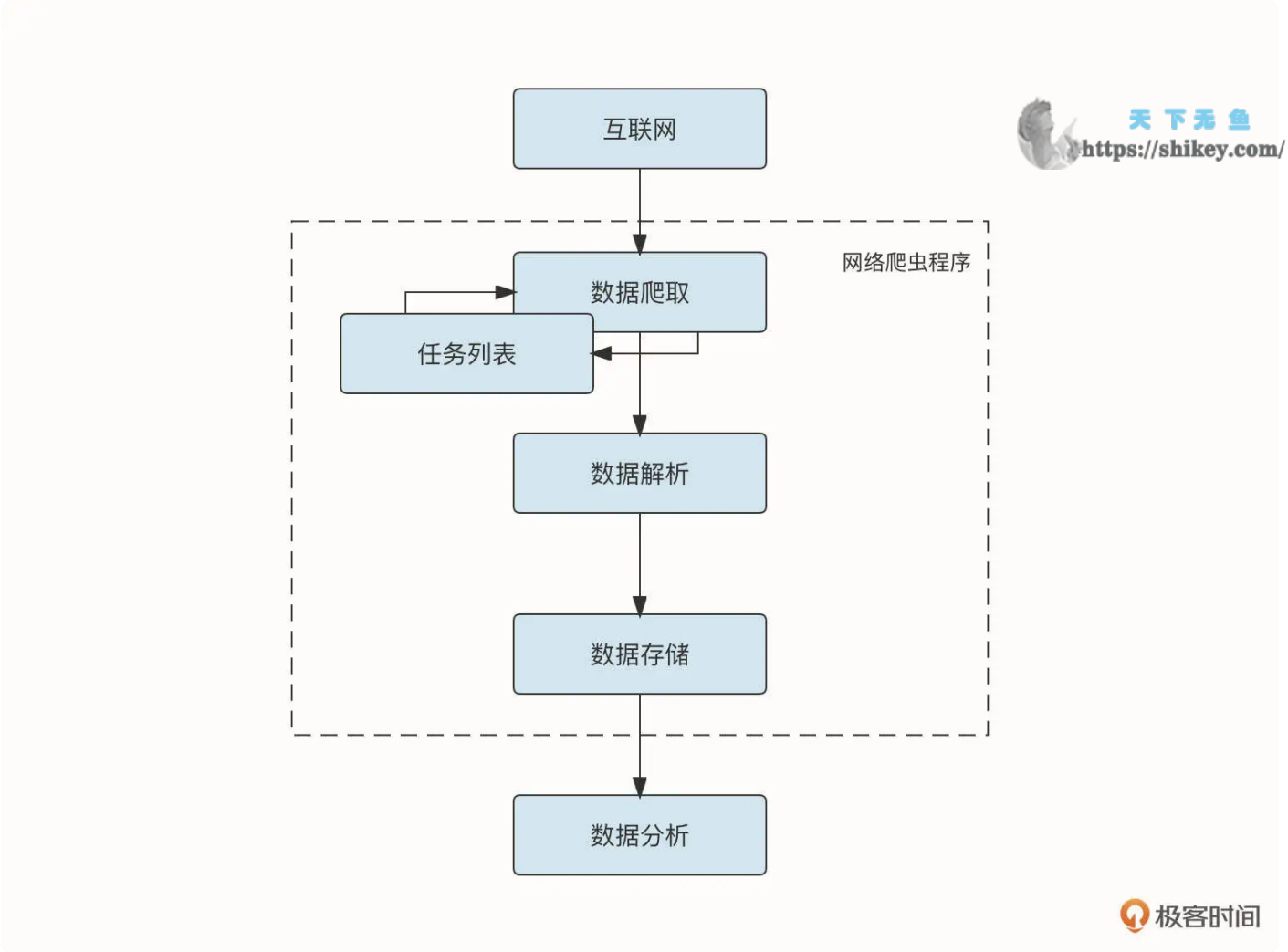
机器学习

在人工智能时代，大公司需要海量的数据来完成机器学习，但是公司自身的数据常常是不够的。举一个例子，要识别一个视频中是否有人摔倒，这需要输入大量真实的视频。这时候大公司通常会选择在互联网中爬取对应的数据并存储下来，完成后续模型的训练。

总之，你可以用网络爬虫做各种各样的事情，这完全取决于你想用收集到的数据做什么，取决于你能够创造多大的价值。

网络爬虫的流程与技术栈

好了，畅想了这么久网络爬虫的价值，我们也来看看实现爬虫需要用到什么技术。一个典型爬虫程序的流程如下图所示，我们结合这个流程来看一下爬虫程序涉及到的技术栈。



数据爬取协议

网络爬虫的第一步是爬取数据。爬虫程序根据一个初始化的任务队列，收集对应网站上的数据。过去的客户端一般通过 HTTP 协议访问网站，但是现在的网站越来越多地使用 HTTPS 协议对数据进行加密和鉴权，这也是未来的趋势。

HTTP 和 HTTPS 协议本身是基于 TCP 协议实现的，另外在网站请求的过程中还涉及到网站域名的 DNS 解析。如果继续深入挖掘，你会发现网络爬虫涵盖了主流网络协议的各个方面。由于 HTTP、TCP 具有的缺陷，你甚至可以进一步探索 HTTP/2、HTTP/3 协议。

除此之外，网络爬虫有时候也是一门斗智斗勇的学科，对于初学者来说，可能很难理解其中的一些现象。例如为什么用浏览器能够访问网站，但是写的脚本和程序就无法获取数据？又或者为什么获取到的数据和真实浏览器中看到的数据不一致，是获取的时间太短了吗？这其中就涉及到一些服务器端的反爬机制以及浏览器的工作机制。

不仅如此，随着移动互联网的兴起，有些程序只能在手机上访问，在浏览器中我们能够方便看到访问的地址，那我们又怎么确定手机程序中访问的是哪一个网站呢？这中间就涉及到网络的

抓包。而这些问题都是进一步理解计算机科学、浏览器工作原理、网络协议处理流程的良好契机。

数据爬取策略

通过初始网站列表爬取到的网页中可能包含了可以进一步爬取的网站列表，这样我们爬取的网站列表就可以像一棵树展开。

以什么样的策略爬取网站呢？假设 A 网站包含 B、C、D 三个链接，而 B 链接中又包含了 E、F 两个链接。我们在抓取 A 网站后，是用深度优先搜索的方式先搜索 B，再搜索 B 包含的 E、F。还是用广度优先搜索的方式先将 B、C、D 网站都爬取完毕，之后再搜索 E、F 呢? 这中间涉及到我们设计合适的算法与数据结构来满足特定爬取需求。

另外在爬取过程中也不是一帆风顺的，这中间我们需要使用合适的超时控制、限流与重试机制保证服务的健壮性，还要使用代理等机制突破服务端的封锁。

最后在爬取过程中，如何设计高并发的模型来保证海量任务的并发执行，这涉及到对服务进行合适的架构设计甚至是分布式的架构设计。这可能就牵涉到如何解决分布式系统的一致性与故障容错问题。除此之外，我们还要考虑对任务进行合理的分配，采用合理的负载均衡策略。

数据解析

一般我们从网页上收集的数据是 HMTL 格式的（当然，有时候我们也希望搜集 CSS 文件、js 文件，以及图片、音频、视频等各种形式的文件），所以，我们必须要了解前端的知识，例如 HTML 的组成、HTML 常见的标签及其作用、文档对象模型、JavaScript 语法以及 CSS 的渲染规则等。

浏览器会根据 CSS 文件中的规则对 HTML 元素进行渲染。有时服务器会借助 CSS 的这一特性将一些数据伪装成另一种形式，阻止你直接获取到像飞机票价格这类敏感数据。这时候就需要了解 CSS 是如何修饰数据的，然后反推出真实的价格。

在对获取的本文数据进行解析时，可能涉及到多种文本处理技术，这里包括：

- 语言标准库中对基本字符串的处理，例如 Go 语言中的 strings 包，strconv 包；
- 正则表达式对文本进行复杂规则的匹配；

- XPath 遍历 **XML 文档节点；**
- CSS 选择器获取指定 HTML 标签中的数据；
- 自然语言处理（Natural Language Processing，NLP），例如在文本中提取特定单词或短语（人名、公司名、地理位置等）。

最终，我们需要定义一个结构，将解析到的数据整合为结构化的数据。例如，我们希望在豆瓣网站中获取图书的信息，但是图书的信息散落在各处，我们需要将这些信息收集起来，并存储到对应的结构体中：


```
type BookDetail struct {
  BookName  string // 书名
  Author    string // 作者
  Publicer  string // 出版社
  Bookpages int    // 页数
  Price     string // 价格
  Score     string // 评分
  Into      string // 简介
}
```

数据存储

下一步，我们还要将爬取到的数据存储到文件或者数据库中。根据存储数据的规模、性质和后续处理方式的不同，我们要选择不同类型的存储。

- 如果我们要存储的数据总量比较小，可以考虑将其存储到 CSV 文件或者 Excel 文件中。
- 如果我们要存储的数据结构比较确定，关系比较简单，可以使用传统的 MySQL、PostgreSQL 等**关系型数据库。**
- 如果我们要存储的数据结构需要有比较强的扩展性，需要以类似 JSON 对象的方式进行存储和查询，可以考虑使用 MongoDB 这类**面向文档的数据库。**
- 如果存储的某一部分都只包含键和值这样的 key-value 存储方式，可以考虑使用 DynanoDB 这样的**键值数据库。**
- 如果你存储的数据关系复杂，例如社交网络这样的场景使用 Neo4j 和 JanusGraph 这样的**图形数据库**是比较好的选择。

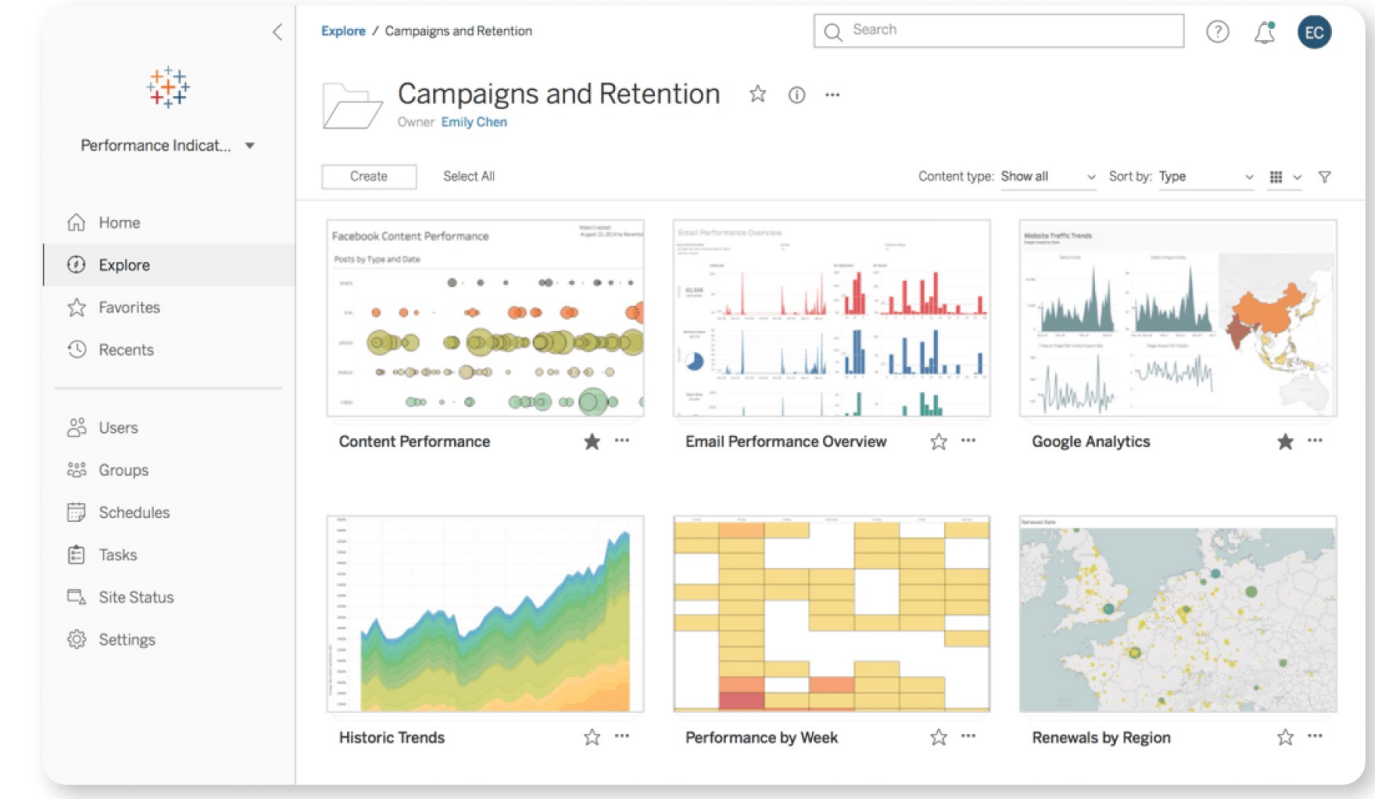
- 如果你存储的数据主要用于决策，不需要太强的实时性，数据会涉及大批量的读取与写入，可以考虑使用像 ClickHouse 这样的适合OLAP 场景的数据库。

总之，数据存储也是计算机科学的基石之一，借助爬虫项目可以深入挖掘不同类型数据库适用的场景，探索数据库内部的存储结构（B-Trees、LSM-Trees），了解分布式数据库的一致性保证与实现方案。

数据分析与可视化

当然，爬取数据最终目的是分析数据中蕴含的价值。常见的数据分析工具包括下面几种。

- Excel：Excel 是微软提供的办公软件，我相信大多数人对它都不陌生。对于少量的数据（一般不超过 100 万行），使用 Excel 中简单的工具（筛选、排序、函数）就可以对数据进行多维度的计算和统计。除此之外，Excel 还提供了数据透视表，方便我们可视化和启发式地发现数据中蕴含的规律。对于更加复杂的逻辑，还可以使用专门为 Excel 设计的 VBA 语言。
- Tableau、**Microsoft Power BI** 等商业软件，这些软件能够处理更大规模的数据，具有更加强悍的可视化能力。



- R 语言：R 语言内置了丰富的函数，可以对海量数据进行专业的分析，主要用于统计分析、绘图以及数据挖掘。
- Python 语言: Python 中拥有众多应用广泛的库，例如 spaCy、TensorFlow、Matplotlib 都可以满足自然语言处理、机器学习、专业可视化等需求。

从前端到后端，从网络到存储，从数据结构算法再到可视化数据分析，可以看到爬虫涉及到了丰富的技术栈，通过爬虫项目将众多的技术串联在一起是一种极佳的选择。我们在后面的课程中实战的爬虫项目，也会涉及到包括分布式系统的设计、高并发模式的选择、文本的解析与存储、HTTP 网络协议、代理在内的核心技术栈。

常见的反爬虫措施

与此同时，我们也必须知道，服务器为了保证服务的质量，保护数据不被恶意获取常常会采用一系列方式阻止爬虫的执行。常见的反爬虫措施包括 IP 校验、HTTP Header 校验、验证码、登陆限制、CSS 数据伪装、sign 参数签名等。

IP 校验

对于不需要登录就能够访问的网站来说，信息具有公开性，服务器无法识别到访问者的具体身份。但是这并不是说来访者就没有办法被追踪到了，服务器可以用间接的方式识别用户，例如识别并监控客户端的访问 IP 等。当特定 IP 在一段时间内访问的频率、次数达到一定限定阈值后，服务器可以采取返回错误码或者拒绝服务的方式起到反爬虫的目的。

在当下，由于 IPV4 地址不足，出现了 NAT 等技术，局域网内的用户进行外部访问时会共享同一个公网 IP 地址，因此，如果服务器对这种 IP 地址进行阻断，会导致大量正常的用户被拦截在网站之外。客户端解决 IP 校验比较有效的方式是，使用大量网络代理隐藏源 IP 地址，让服务器以为是不同的 IP 在访问一样。

HTTP Header 校验

还有一些服务器会校验客户端传递的 HTTP Header，例如，User-Agent 字段用于表明当前正在使用的应用程序、设备类型、操作系统及版本、CPU 类型、浏览器及版本、浏览器渲染引擎等。浏览器会在该字段自动填充数据，例如，当前我的谷歌浏览器的 User-Agent 字段为：

复制代码

```
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like
```

如果服务器识别 User-Agent 字段发现不是用户通过浏览器发出的，服务器可能会拒绝服务。解决这类 HTTP Header 校验的方式是在请求头中添加浏览器的标识，让你的请求看起来就像是通过浏览器发出的。

验证码

验证码又被称为全自动区分计算机和人类的公开图灵测试（**CAPTCHA**）。

顾名思义，验证码是一种区分用户是机器还是人类的自动程序。验证码包括简单的数字验证码、字母数字验证码、字符图形验证码、极验验证码等，能够输入正确验证码的访问者被服务器认定是人类，否则被认为是爬虫。

一些简单的验证码测试可以借由打码平台辅助完成，这些平台通过脚本上传验证的图片，再由打码公司雇用人工进行识别。对于一些更加复杂的验证码，破解的难度和成本还会更高。考虑到验证码一般是在 IP 地址访问过于频繁之后才会出现，一个解决思路就是当页面弹出验证码时，通过切换 IP 的方式避开验证码的输入。

登录限制

此外，登录限制也是一种有效保护数据的方式。当用户需要访问重要的数据或者更多的数据时，需要登录才能继续查看。例如，知乎用户如果想查看更多数据就需要先登录网站。这种策略也是一把双刃剑，因为需要登录的页面是不能被搜索引擎检索的，这就降低了网站的曝光度。

解决登录限制的方式是提前登录，然后借助 cookie 在已经登录的情况下访问数据。如果单个用户的访问频率受到了限制，还可以准备大量的账号来操作，但这样做的成本太高了。

CSS 数据伪装

一些网站借助了 CSS 对 HTML 元素的渲染功能来实现反爬虫机制。也就是说，不在 HTML 元素中放入真实的值。例如，一个产品的实际价格为 888 元，服务器会将特定 HTML 标签的数字修改为 999 元，并利用 CSS 的规则巧妙地将 999 渲染为 888。但是如果我们单纯地获取 HTML 文本的数据，就可能出错。

要解决这一问题，我们需要先手动识别出这种数据伪装的规则。由于网站每次更新后这种数据伪装规则都可能发生变化，所以还需要识别当前网站的版本。更复杂的解决方案则是使用 OCR，对区域内的图像进行文字识别。

sign 参数签名

一些 API 会对参数进行签名（sign），以此拒绝非法请求。这种机制常见于手机 App 中。签名通常包含了时间戳、请求的参数体等信息。这样即便请求被非法抓包或捕获，也无法修改请求的内容或者重新访问，因为服务器会对时间戳和参数进行验证，并且只有在一定时间范围内这个请求才是有效的。

在下面这个例子的 HTTP GET 方法中，在 url 中加入的 time 参数为当前的时间戳，sign 为生成的参数签名（如果是 POST 方法，这些参数会放入 content 中）。

```
<http://cosapi.myqcloud.com/api/cos_create_bucket?accessId=9999&bucketId=abc&ac
```

要破解 sign 参数签名的规则一般比较困难，除了试错法，一种可能的机制是使用反编译技术获得加密算法。此外我们还可以模拟用户操作应用，并通过抓包的方式截获流量中的信息，但这种方式效率较低。

总结

互联网就像一个免费的宝库，你只需要有一些创造力，就可以应用爬虫技术实现信息的聚合，获取有洞察力的行业见解，预测未来的走势，从而创造非凡的商业价值。

就像刀可以伤人但刀本身并不违法一样，爬虫技术本身并不违法，但我们需要遵守一些正确的规则，提前确认好爬取的数据有权访问，不侵犯个人和企业的权益。

爬虫技术非常有趣而且有料。典型的爬虫流程涉及到数据的爬取、数据的解析、数据的存储以及数据的分析。在这中间爬虫涉及到众多的技术栈，涵盖了前端、网络、存储、算法与数据结构、代理、分布式系统设计，自然语言处理等，这也使得爬虫成为我们进一步学习这些技术领域的契机。

课后题

最后，我也给你留一道思考题。

我们在爬取网页的时候，常常会出现通过浏览器能够正常获取网页信息，但是通过程序访问的方式就无法获取网页信息，你知道可能是什么原因吗？

欢迎你在留言区与我交流讨论，我们下节课再见！

上一篇　05 | 全局视野：洞悉项目开发流程与规范

精选留言

写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。