37 | 当数据遇上AI，Twitter的数据挖掘实战（一）

2022-01-19 徐文浩

《大数据经典论文解读》 课程介绍 >

讲述：徐文浩

时长 14:29 大小 13.27M

你好，我是徐文浩。

课程进行到这里，我们对于各类大数据系统的论文的解读，就已经结束了。不过，真的要把大数据系统运用到实践当中，我们仍然会遇到很多挫折。在 2010 年，我第一次开始使用 Hadoop。在读完了论文和教程，开始撰写 Java MapReduce 的代码之后，我的第一感觉是“大数据不过如此”。

不过，在逐步深入尝试利用数据做越来越多的事情之后，我们遇到了越来越多意料之外的问题。大部分困难的来源，往往并不是纯粹的技术问题。毕竟，那些问题都可以靠读代码、记日志、找个环境复现问题来解决。更多的挑战，来自于**系统和系统之间的“鸿沟”，不同团队和角色之间的“鸿沟”**。

日志格式的含义、工程师和数据科学家之间的技能树的差异，乃至于不同数据报表之间的依赖关系，都会成为我们快速分析数据、产出结论的鸿沟。难以精确定义的业务目标，与实际需要精确定义才能变成代码的技术开发之间的鸿沟，更是挡在大部分数据科学家面前的一座大山。

那么，对于这样看似软性的问题，我们是不是只能靠“加强沟通”来解决呢？接下来，我们就一起来看看，Twitter 这家公司是怎么做的吧。今天，就请你和我一起来读一读《Scaling big data mining infrastructure: the twitter experience》这篇论文。

在读完这篇论文之后，希望你能够了解到面对这些“沟通问题”，Twitter 是如何通过**技术手段**来填补知识鸿沟、提升团队效率的。我也希望，你在未来遇到更多看似是“沟通问题”的困扰时，能够找到通过技术手段提升团队杠杆的解决方案。

学术界与工业界的鸿沟

Twitter 的这篇论文很有意思，他的两位作者，一位来自学术界，是马里兰大学的教授 Jimmy Lin，当时他正处于学术休假期，所以去了 Twitter 工作。另一位 Dmitriy Ryaboy，则是 Twitter 后来数据分析基础设施团队的工程经理。所以，我们在论文里很容易就能看到真实互联网世界的数据，给学术界带来的“文化冲击”。

没有从事过工业界的“数据科学”工作的人，很容易有一个错误的假设，那就是工作的大部分时间是在“研究算法”。而在实践中，大部分的时间，其实是在“**定义问题**”和“**清理数据**”。

学术研究要解决的问题，往往是定义得非常清晰的（Well-defined）。我们在媒体里听到的，也往往是这样的，比如如今已经到处都在用的人脸识别，或者是你听说过的“Netflix 通过大数据拍出了纸牌屋”。我们总会以为，基于大数据做数据挖掘、机器学习，就是找到一个新想法、新算法来解决这样的问题。

但是事实远不是这样，数据科学家在 Twitter 这样的公司面临的第一个问题，就是**问题本身是模糊的**。比如，Twitter 想要解决的问题，是“加快用户增长”，但是这个问题，并不能立刻转化成一个算法。

所以，实际的数据科学家的**工作的第一步，往往是探索性分析**（Exploration Analysis）。也就是，我们先去看一看，我们有哪些数据，做一些粗浅的统计、相关性分析。期望通过

这个步骤，能够找到解决问题的思路。

而这第一步的探索工作，常常就把很多数据科学家给卡住了。因为，在上了大数据系统之后，我们的日志往往不是太少了，而是太多了。

一般来说，我们的系统会是分布式的，每个服务都会去记录自己的日志。以 Twitter 为例，可能你的时间线（Timeline）是访问一个服务，而搜索功能又是访问另外一个服务的。而这两个服务，可能各自都会调用某一个排序算法服务。每个服务都是由一个独立的小团队开发的，也就有各自的日志。他们日志的格式、字段名称都各不相同。

这样不同格式的日志可能在一个公司里，有个几十种并不稀奇。而不同日志里，可能不同的字段名称会代表相同的意思，比如在首页的时间线里，我们用 uid 这个字段来作为用户标识，到了搜索服务里，可能用了 user_id，而到了用户管理自己 Profile 信息的服务里，它又变成了 userID。

Schema 和 HCatalog，把隐式知识显式化

那对于这样的数据，想要去做数据分析，最耗费时间的往往是搞清楚到底有哪些数据可以用，以及这些数据各个字段到底是什么含义。

你可能会说，那我们就写个文档就好了呗。但是不要忘了，我们的**系统不是静态的**，往往还在不停地迭代之中。在这个过程中，记录的日志的字段会增加，旧的字段会废弃。比起没有文档，更糟糕的是，文档里写了 uid 用来唯一标识用户，但是在数据里早就没有这个字段，只剩下一个 user_id 字段，这种情况往往会为我们带来更大的困扰。

Twitter 在论文里举了一个看似极端，但是相信在很多公司里都发生过的事情。那就是原本某个字段 feature_enabled，只是为了记录当前用户的某个功能是否开启的标识。但是随着时间的推移，系统里的 A/B 测试越来越多，这样的一个字段变成了可以记录多个功能标识，于是就用逗号分割。乃至后面可能又有人用冒号，乃至用 JSON 格式去记录这一个字段。

这样，我们下游的数据科学家，往往会在尝试多次失败之后，才发现数据里有这样三种不同的格式。于是不得不写上一段堆满 if...else 的代码，然后几个月之后，当有新的数据科学家加入的时候，还要再向他解释一遍，我们为什么要干这种傻事。

当然，我们可以靠“努力沟通”“努力写文档”“努力地写兼容性代码”，去尝试解决这样的问题。但无论是 Twitter，还是业界其他的标准解决方案，都是**把这些元数据，也“显式”地变成了系统的一部分**。

在 Twitter 的论文里，Twitter 是这样做的。

首先，自然是让日志从原本的纯文本，或者是 JSON 这样的弱 Schema 的格式，变成 Thrift 这样**强 Schema 的格式**。这样，日志格式会始终和代码保持一致。而在 Thrift 的定义文件里，加上几行注释，说明某个字段是什么含义，就不是一个困难的事情了。

更进一步地，和 Hive 一样，Twitter 还针对这些日志**引入了 HCatalog**，来做数据的元数据管理。通过一个中心化的管理平台，让所有人知道，我们有哪些日志，它们存放在哪里、格式是怎么样的、字段的含义是什么。其实这一点，我们在之前介绍 Hive 的论文的时候就已经强调过了。良好管理的元数据，会大大提升数据科学家们的工作团队效率。

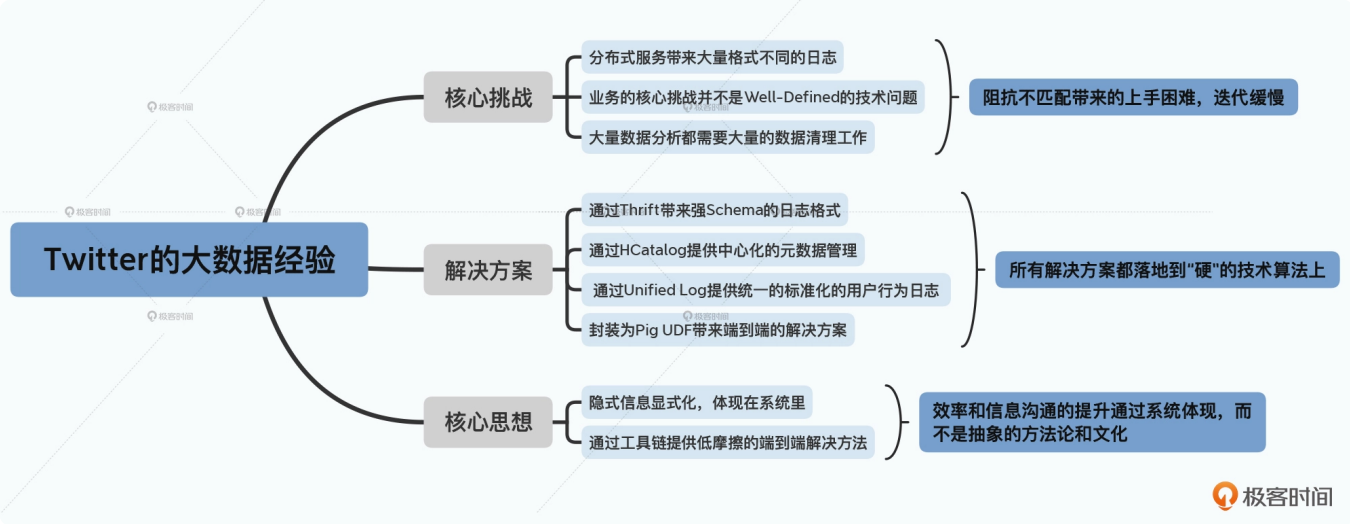
而在元数据里，光光给出了“数据是什么”还不够。Twitter 还进一步地给出了**数据是从哪里来的**，也就是数据是通过什么数据源生成的。这个，也会有助于数据科学家们去理解有哪些数据可以使用，并且避免重复劳动。

不过，即使是这样，虽然理解数据没有问题了，但是数据科学家们往往仍然需要花费大量的时间去清理数据。我们虽然知道了 Timeline 服务的日志里的 uid，与搜索服务的日志里的 user_id 是同一个字段，但是在做任何分析工作之前，往往我们还需要把这些日志 Join 起来，找到我们需要的字段。这也是为什么，很多从学校进入工业界的数据科学家会有很大落差的原因。

所以，进一步地，Twitter 更是单独启动了一个项目，将最常用的用户行为日志，从散落在各个服务里各自记录，**统一放到了在客户端用统一的格式记录**。针对这个系统，Twitter 也专门发表了一篇论文，叫做《The Unified Logging Infrastructure for Data Analytics at Twitter》，你可以去仔细读一下。我们在下节课介绍 Twitter 采用的各类数据挖掘的方法和技巧的时候，也会做更多深入的解读。

而很多其他公司采用的策略，则是有一个专门的数据工程团队，来进行 **ETL 的数据处理**。他们会把所有的用户行为统一清理出来，变成一份干净的日志提供给数据科学家使用。也就是把最常见的日志处理清理工作一次做完，让其他人可以“复用”。

可以看到，在这些优化中，Twitter 始终采用的，是通过技术手段去减少重复劳动。并且让需要跨团队、跨角色进行沟通的工作，变成系统里的表架构、数据这些显性知识，通过将隐性的知识显性化，来提升工作效率。



封装 Pig，用一种语言处理整个流程

在有了干净的日志和清晰的元数据之后，数据科学家就可以很容易对数据进行探索了。很快，我们就可以找到一些有助于“快速增长用户”的数据挖掘方案。

Twitter 在论文里并没有给出他们具体的业务案例，那我们就想想我们身边的例子。比如，我们可以尝试根据用户行为，去预测哪些用户有比较高的流失的可能性。我们可以利用机器学习算法，找到有高概率流失的用户，通过消息推送，或者是优惠券把他们吸引回来。

你可能会想，那这回我们的重点应该回到研究算法了吧？不幸的是，事情并非如此。在大数据的场景下，很多时候多灌一点高质量的数据，然后采用简单的模型，就会有很好的预测效果了。特别是在这篇论文发表的 2013 年，深度学习还没有进入主流。在大部分互联网公司，采取的都是通过海量数据信号，来获得更好的模型预测效果的策略。

而在这个过程中，数据科学家们会遇到**第二个“鸿沟”**，就是需要在多个不同的系统、不同的运行环境里面切换。

数据科学家先需要在 Hadoop 上，撰写一系列的数据处理代码，把原始的日志变成机器学习所需要的训练样本数据。然后，在实际的机器学习的过程里，大家就需要各显神通了。

有时候，他们需要把对应的数据，下载到特定的单机去运行机器学习算法。有时候，他们会直接在 Hadoop 上运行 Mahout 这样的分布式算法。

而在模型训练完成之后，他们又要单独撰写 Hadoop 上的代码，去验证模型训练的结果。一旦一个模型上线，随着数据的不断变化，数据科学家们还要定时让模型根据时间滚动更新，并去监测模型的效果。

在整个过程中，我们会发现，机器学习并不是一个“研发”过程，而是一个**探索性交互的过程**。我们会修改一些提取特征的代码，提交 Hadoop 任务。我们也会换一个模型，重新运行算法。在这个过程中，我们也可能频繁地在服务器和本机之间去搬运数据。而我们也可能常常写出一点小 Bug，让程序跑不出结果。

在这个过程中，真正的挑战并不在于怎么写程序，而是我们要不停地在上下文里面切换。很多时候，我们不得不人工记忆，对应的数据、程序、版本在哪里，好让自己知道目前算法优化的进展。而我们的编程语言，也会在 Pig 脚本、Java 程序，调用 Mahout 或者 Python 的库里不停地切换，而上下文切换本身，就是效率的大敌。

Twitter 的解决方案很简单直接，他们做了两件事情：

> 第一件事情，是**在原始的完整日志之外，提供了多份采样的日志**。这个，使得数据科学家探索、测试的迭代周期大大缩短。我们先可以在这份日志上，完成大部分的研究探索工作，确保在一个小数据集上是有效的，再在完整的数据上运行。通过缩短整个交互的时间，来提升效率。
>
> 第二件事情，就是让数据科学家**通过 Pig 就能运行所有的数据处理逻辑，以及机器学习算法**。机器学习算法，被包装成了 Pig 的 UDF。因为 Pig 脚本写起来，本身就类似于 R 这样的交互式数据分析代码。这样让整个端到端的数据挖掘过程，通过同一种语言管理起来，就减少了上下文切换的成本，从而提升了效率。

复制代码

```
1 training = load ‘training.txt’ using SVMLightStorage() as (target: int, featur
2 store training into ‘model/’ using FeaturesLRClassifierBuilder();
```

> Twitter 通过实现 FeaturesLRClassifierBuilder，直接实现了逻辑回归算法，简单通过两行脚本就能实现机器学习的过程。

```
复制代码
1 define Classify ClassifyWithLR('model/');
2 data = load 'test.txt' using SVMLightStorage() as (target: double, features: m
3 data = foreach data generate target, Classify(features) as prediction;
```

> Twitter 在 Pig 里同样通过 UDF 实现了模型的预测功能，方便数据科学家能够快速使用模型验证和分析算法效果。

虽然 Pig 在今天已经逐步淡出大数据的主流系统了，但是这个决策背后的思路并没有错，这也是为什么 Spark 在数据科学家里快速流行了起来。

因为 Spark 在推出没有多久，就支持了 Python，并且支持了 DataFrame 风格的 API。这个使得我们在 Spark 里，很容易通过 Python 这一种语言，就能完成整个数据处理、数据挖掘的全流程。更好的一点在于，Python 本身还有 Matplotlib 这样很优秀的可视化库，补上了前面这个流程的最后一个短板。

小结

Twitter 的这篇论文其实对于我们大部分人更有借鉴意义。

相比于 Google、Facebook 这样拥有海量的基础设施和人才的公司来说，Twitter 在那个时候其实也还是一个中型公司，他们拥有的 Hadoop 集群服务器，也才是几千台的规模而已。相信大部分工程师会经历的大数据系统的开发、运用，也都是在几十台到几千台之间，所以 Twitter 的经验可以说对我们很有借鉴意义。

通过这节课的学习，我们可以看到，一旦进入“混乱”的日常运营，大数据系统面临的主要问题不再是单纯的技术问题，而是来自于不同团队、不同系统之间的“**鸿沟**”。

散落在各个服务里不同格式的日志、维护基础设施但是对于业务场景不够熟悉的工程团队、能做分析和研究但是并不负责撰写线上服务的数据科学团队，以及始终模糊只有逐步迭代才会慢慢清晰的目标，都是切实在每一个要使用大数据的公司都会面临的实际情况。

而 Twitter 的解决方案也并不是什么“神奇”的新系统新科技，而是**通过各种实战的手段来缝合这些问题之间的鸿沟。**

通过 Thrift 为所有的日志加上 Schema，通过 HCatalog 为各种日志提供一个中心化元数据管理，通过在元数据里提供各个数据生成的依赖关系，这几点使得 Twitter 让大部分的数据分析人员不用去“猜”、去“试”、去“问”，数据到底是什么意思、是从哪里来的，大大降低了团队的上手和沟通成本。

通过将原本散落在多个服务里的日志，统一成层级化的、标准的从客户端发起的用户行为日志，Twitter 简化了日志处理和清理的成本。

通过为日志提供采样的 Staging 和本地数据副本，Twitter 加速了数据科学家快速探索数据的过程。

通过将常见的数据 ETL 的流程和机器学习算法包装成 Pig 脚本，Twitter 减少了各个系统之间的“阻抗不匹配”问题，数据科学家们不用在不同的语言、系统之间切换，来完成整个分析过程。

而贯穿所有这些实践方案的主线，其实仍然是**通过“技术手段”来解决“沟通问题”**。

Twitter 并不是靠嘴上说说“加强沟通”来解决沟通问题的，而是通过把原本需要沟通才能解决的问题，变成系统里的一部分。他们用 Thrift、HCatalog 来管理元数据，提供 Pig 包来进行数据处理和机器学习，这些都是通过技术手段把需要交流的信息，变成系统里固定的一部分。而往往也只有这样的方案，才能做到长治久安。

推荐阅读

Twitter 其实发表了很多论文，分享了自己在大数据处理和数据挖掘层面的工程实践。如果你在搭建大数据体系的过程中，遇到很多难以决策的困难，大可以从里面找找有什么可以借鉴的思路和解决方案。我把我自己读过觉得有价值的论文的链接放在了这里，你可以有空的时候去读一下。

- Scaling big data mining infrastructure: the twitter experience
- Large-Scale Machine Learning at Twitter
- The Unified Logging Infrastructure for Data Analytics at Twitter

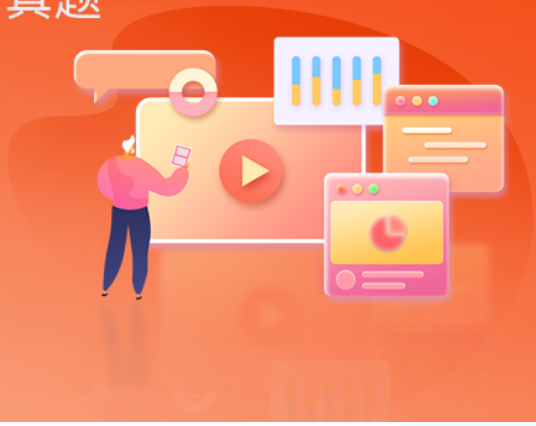
思考题

如果你自己也在开发、使用大数据系统。你所在的团队，有遇到和 Twitter 一样的挑战和困扰么？你们是通过什么样的方式来解决的呢？Twitter 在 2013 年的这些经验中，你看到还有哪些做得不好的，你又是采用了什么样的解决方案呢？

欢迎在留言区分享下你的思考和见解，也欢迎你把今天的内容分享给更多的朋友。

上一篇 36 | 从Omega到Kubernetes：哺育云原生的开源项目

下一篇 用户故事 | 陈煌：唯有自强不息，方能屹立不倒



精选留言 (3)

piboye

2022-01-20

clickhouse 这些sql 系统的 udf 可以用python 来实现机器学习吗？


在路上

2022-01-19

徐老师好，当数据遇上AI，需要从源头来规范日志格式，用一种语言处理整个流程，这一切都是为了提升数据科学家的工作效率。这说明数据的价值来自于分析，以及所得出的结论。在阅读《Streaming System》时我以为流式处理是大数据未来的方向，但是我最近在读《数学之美》和《智能时代》，发现大数据的价值在于服务于AI，而AI的重点不在于实时性，而在于海量、完备性和相关性。实时处理是当今的热点，不过比起追逐潮流，更…

展开


那时刻

2022-01-19

良好管理的元数据，会大大提升数据科学家们的工作团队效率。这个深有体会，我们之前数据字典不规范，导致部门之间的信息不均衡，后来通过统一数据字典以及可视化的方式，来统一信息的一致性