

GitHub 2Q20 数字洞察报告



X-lab

开放实验室



开源社会工程研究院



开源社
kaiyuanshe



DaSE
Data Science
& Engineering

SHOPEN

上海开源信息技术协会

目 录

摘要	02	说明 1.1 GH Archive	06
引言	03	说明 1.2 GitHub 事件日志	06
一、 总体情况	06		
二、 开发者分析	07	说明 1.3 活跃仓库与开发者统计方法	06
1、 全域开发者活跃情况			
2、 GitHub Apps 使用情况		说明 2.1 开发者活跃度	07
3、 全域日志时间分布			
4、 典型开源开发者工作时间画像		说明 2.2 开发者活跃仓库数量	08
5、 全球开发者时区分布			
6、 开发者使用语言分布		说明 2.3 GitHub Apps	08
三、 项目分析	13		
1、 全域项目总体数据		说明 2.4 UTC 标准时间	10
2、 项目活跃度 Top 20			
3、 中国项目活跃度排名		说明 2.5 工作时间分布统计方法	10
4、 项目语言分布情况			
5、 OpenGalaxy		说明 2.6 工作时间分布图理解方法	10
四、 案例分析	22		
1、 案例分析方法说明		说明 2.7 开发者时区估计方法	11
2、 CNCF 基金会项目分析			
3、 Linux Foundation AI & Data 基金会项目分析		说明 2.8 开发者使用语言统计方法	12
4、 Apache 软件基金会大数据领域项目分析			
5、 Apache 软件基金会中国项目		说明 3.1 项目活跃度	13
6、 VSCode 案例分析			
五、 每月之星	32	说明 3.2 中国项目及企业项目归属说明	15
1、 每月之星评判方法说明			
2、 2020 年度每月开源之星		说明 3.3 OpenGalaxy 的构建方法说明	18
六、 总结与展望	34		
七、 致谢	34	说明 4.1 开发者协作网络构建方法	31

摘要

开源软件已经成为人类数字社会的基石，是全人类共同努力的结晶，开源协作对人类数字文明的发展起到了巨大的推动作用。GitHub 作为全球范围内最主要的开源协作平台，无数个开源社区在其上孕育而生，其背后海量的开发者行为数据蕴含了大量的个体贡献规律、群体协作模式、社区健康状况、生态发展趋势、以及商业战略价值。

《GitHub 2020 数字洞察报告》是由 X-lab 开放实验室发起，联合多家科研机构与开源社区所共同完成的一个反映全球开源现状与趋势的开源项目。报告涵盖了当今全球开源的总体情况分析、开发者分析、项目分析、领域案例、每月之星等众多内容，希望以此绘制人类的“开源数字生态地图”，推动开源社会创新，繁荣开源数字文明。

引言

2020 注定是个不平凡的数字，开源的 2020 也是如此。

1月，由广大开发者自愿发起的 Wuhan2020 开源公益项目就像一座丰碑，激励着我们通过开源协作的方式让这个世界变得更加美好。开放式自组织协作取得的一小点成绩，却让我们看到了人类的未来。

5月，首届“开源软件供应链点亮计划”正式拉开帷幕。由中科院软件所与 openEuler 社区共同发起的一项面向高校学生的暑期活动，旨在鼓励高校在校学生积极参与开源软件的开发维护，促进国内优秀开源软件社区的蓬勃发展。

7月，GitHub 完成北极代码保险库项目，共保存了 21TB 的代码数据。所有这些数据通过 186 卷胶卷被送上飞机，从挪威首都奥斯陆飞往斯瓦尔巴群岛，这些开源代码将被保存至少长达 1000 年。

9月，中国首个开源软件的基金会“开放原子开源基金会”正式官宣。开放原子开源基金会是一个致力于开源产业的全球性非营利公益机构，为各类开源项目提供中立的知识产权托管服务，以及战略咨询、法务咨询、项目运营和品牌营销服务。

12月，Linux 基金会旗下的 CHAOSS 开源社区正式登录中国上海，开始传播开源项目与开源社区健康度的度量体系、工具与方法论。开源社区生态与治理不再像以往空泛的进行着讨论，而是可以通过大数据与数字化工具进行着更加有效的落地。

开源就是这样，即便是在新冠疫情肆虐的今天，依旧大发展，甚至发展的更加迅速。从各项数据指标来看，我们都能发现，开源呈现出一个全球大繁荣的趋势：GitHub 的日志数 2020 年达到了 **8.6** 亿条，相较 2019 年增长了 **43%**；活跃代码仓库达到了 **5,421** 万个，相较 2019 年增长了 **36%**；活跃开发者数达到了 **1,454** 万人，相较 2019 年增长了 **22%**。

作为数字经济创新创业公共基础设施，开源以**开放**（Openness）、**对等**（Peering）、**分享**（Sharing）、**透明**（Transparency）以及**全球运作**（Acting Globally）——正在吞噬着传统的商业教条。

开源经过了 20 多年的发展，已经演化出诸如 Linux、MySQL、Hadoop、Kubernetes、TensorFlow、React、VS Code 这些数字化的基础设施，而且整个技术栈还在不断向上生长，吞噬着整个软件世界，并形成了一个开源生态，一个开源文明。所有代码、数据、开发者行为、社区规范等等所组成的**开源数字生态系统**，就像一个新物种一样，不断演化与成长，支持

着整个数字世界的基础设施，也在逐渐改变人类的协作与行为方式。

开源是一场社会创新，可以克服传统市场的种种缺陷，例如信息的充分获取与高效传播。而诸如开源协作平台也已经迭代成为一个**海量数据市场**，在这个市场里，我们将拥有大量透明化的信息，同时开始拥有能够做出决策和进行协作的**自动化数字工具**。这一切将产生巨大的影响，不只是对组织和管理者，而且对所有生态中的各类参与者，包括开发者、社区经理、工程师、还有消费者。

自由的开源软件产业是否能够战胜封闭的传统软件产业，毋庸置疑，就是因为其拥有透明高效的海量数据市场以及自动化协作数字工具。全球化的分布式协作，全球 24 小时不间断的代码接力，使得软件产业以前所未有的速度向前推进。

管理学大师彼得·德鲁克曾经说过：“**你如果无法度量它，就无法管理它**” (If you can not measure it, you can not manage it)，**进而也无法提高它**，而软件行业至今也还没有找到一个可以有效度量软件开发生产效率 (Productivity) 的方法。整个开源生态系统更是如此，个体如何度量、社区如何度量、管理者如何利用这些数字做更好的决策，这些都是问题。但在我看来，这些既是挑战，也是机遇。要想有效的开展开源治理的工作，就难以绕开度量的问题，GitHub 全域数据给予了我们这样一个极佳的机会。

度量也是一把双刃剑。度量具有极强的引导性，它会激励你重视并改善能够度量的元素，但也可能使你忽视无法度量的元素并使之恶化。在全球大规模的开源社区与生态构建的过程当中，如何找到合理的度量，并合理的利用这些度量呢？希望本报告能够给大家带来一些启示，这也是本报告的重要目的之一。

在去年《GitHub 2019 数字年报》的基础上，今年的《GitHub 2020 数字洞察报告》主要的变化包括：

- 整个报告的迭代以开源项目的形式协作完成，涉及数据、代码以及文字内容；
- 提出更加全面的度量指标，以及更加科学的计算方法；
- 用更加专业化、丰富的手段进行数据可视化与洞察；
- 在活跃度的基础上，更加关注时间维度、多样性维度、协作网络维度上的信息；
- 首次提出开源星系 (OpenGalaxy) 与开源象限 (OpenQuadrant) 的概念，并进行了落地实现；
- 增加了单个项目开发者协作网络的深度分析案例；
- 增加了短期内受到大量关注的开源每月之星内容。

今年，我们将“GitHub 数字年报”更名为“GitHub 数字洞察报告”，一个重要的原因就是对“洞察 (insight) ”的理解，不再是对一些统计数字进行简单罗列，而是会挖掘开源世界背后

具有数据支撑的规律，并结合自身的专业知识进行科学解释。本次报告的主要开源洞察如下：

- 全球开源事业大发展，社区活跃行为、开发者数量、开源仓库数量均大幅提升；
- 开源软件生产流水线自动化程度大幅提升，多样化的数字协作机器人开始成为主流；
- 基于海量数据的活跃度模型能够有效地持续反映开发者与社区的整体状况；
- 主流开发者的工作时间具有较强规律，并和工作时间开始重合，公司化开源成为绝对主流，996 开源项目开始出现；
- 美洲开发者分布最多，欧洲拥有最高的单时区开发者比例，亚洲开发者数量依然较少，中国相较与其他亚洲国家还是具有较高的开源活跃度；
- JavaScript 和 Python 依旧是语言排行榜上的冠亚军，HTML 和 CSS 在全域开发者语境下更受欢迎，而 TypeScript 和 Rust 语言则上升明显；
- 谷歌、微软等老牌企业依旧为活跃的开源贡献大户，国内的阿里活跃度排名第一，PingCAP 的表现则非常亮眼；
- 第一次通过开源星系认识到 GitHub 开源项目的全貌，主流技术领域的开源生态已经形成，新的开源社群则不断涌现，极少量项目还是协作孤岛；
- CNCF、LF、Apache 等基金会在技术领域上各有所长，通过开源象限能够很好的区分同类开源项目的发展阶段与成熟水平；
- 开发者时区分布图和开发者协作网络成为开源社区多样性与健壮性的有效描述手段，能够更好地指引社区经理的开源治理工作。

以下是《GitHub 2020 数字洞察报告》正文。

一、总体情况

从总体数据来看，2020 年全年，GitHub 全域事件日志数量总计约 8.6 亿条，较 2019 年 6.1 亿条增长约 42.6%，是近五年来增长最快的一年。本次报告通过项目与开发者行为数据，统计得到 2020 年 GitHub 全域活跃项目数量约 5,421 万个，活跃开发者账号约 1,454 万，分别较 2019 年增长 36.4% 与 21.8%。

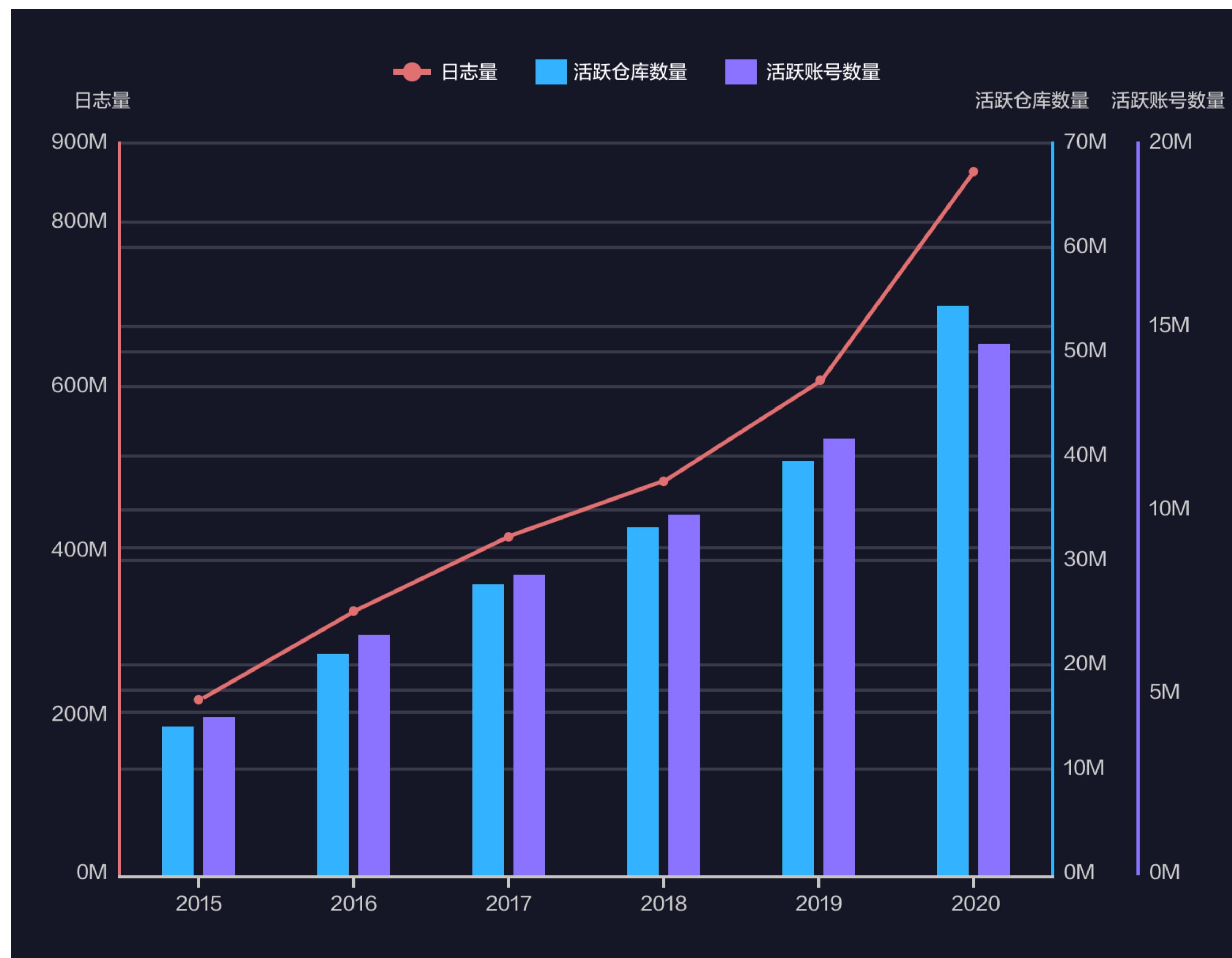


图 1.1 GitHub 2015 - 2020 年事件日志量、活跃仓库数量、活跃账号数量总体情况

说明 1.1: GH Archive

全世界的开源开发人员同时在贡献数百万个项目，为项目编写代码和文档、修复和提交 Bug 等等。[GH Archive](#) 是一个项目，用于记录公共 GitHub 时间线（可以理解为一个记录 GitHub 上所有行为活动的日志）、存档并使其易于访问以进行进一步分析。

说明 1.2: GitHub 事件日志

GitHub 提供了 [20 多种事件类型](#)，范围从新的 Commit 和 Fork 事件，到提交新的 Pull Request, Comment 以及向项目添加成员。这些事件被聚合到每小时的存档中，我们可以使用 HTTP 客户端访问这些存档。

说明 1.3: 活跃仓库与开发者统计方法

若被统计的代码仓库包含事件日志，仓库即被定义为活跃仓库；若开发者有任何代码仓库包含事件日志，开发者即被定义为活跃开发者。

二、开发者分析

开源世界的核心是贡献开源的开发者们，如同 Apache Way 所推崇的 Community Over Code，由开发者组成的社区才是开源生命力的源泉。本报告将从全域开发者活跃情况、GitHub Apps 使用情况、开源开发者典型工作时间画像、全球开发者时区分布、开发者使用语言分布等多个角度对 GitHub 2020 年全域开发者进行全面分析。

1、全域开发者活跃情况

通过对全域开发者进行活跃度与活跃仓库数量的统计，我们得到 GitHub 全域开发者的活跃度分布情况和单个开发者活跃仓库数量分布情况如下：

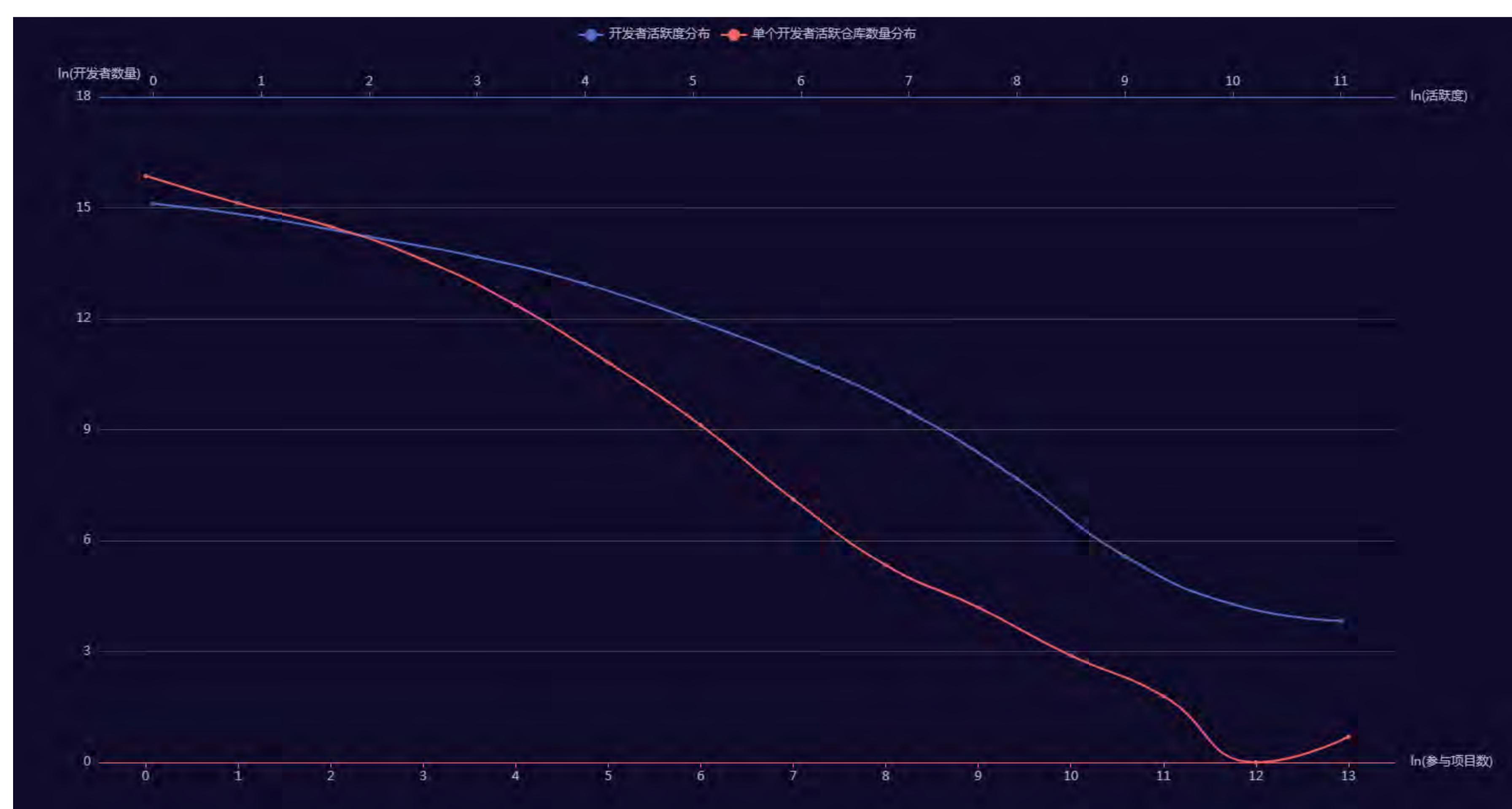


图 2.1 开发者活跃度与活跃仓库数量分布图

上图使用双对数坐标系绘制，图上方横坐标表示开发者活跃度，图下方横坐标表示开发者参与项目数量，纵坐标表示开发者数量，所谓双对数坐标，就是将原来的线性坐标轴都取自然对数，可以看到开发者活跃度与活跃仓库数量的分布符合幂律分布。经统计，活跃度超过 2,000 的开发者数量为 5,445 个，占全域开发者数量不足万分之六。而大部分开发者活跃度都在 [0, 500] 区间内，占全域开发者数量的 99.45%，说明大多数开发者还是处于低活跃度的一个状态。观察曲线尾部，我们发现开发者活跃仓库数量在最后有一个回升，其实是由于部分未被过滤掉的自动化协作类账号的活跃仓库数量巨大，远超正常人类开发者，因此尾部形成 V 形曲线。

另外，我们通过活跃度统计了全域活跃排名最高的 10 个开发者，其中 8 个账号为 GitHub Apps，另外两个账号为自动化协作的开发者账号。

说明 2.1：开发者活跃度

开发者活跃度，其定义为某特定 GitHub 账号在一段时间内在某特定 GitHub 项目中的活跃评价指标。其活跃度由该账号在该项目中的行为数据决定，本报告中所关心的行为包含如下几种：

- Issue comment：在 issue 中参与讨论是最基本的行为，每个评论计入 1 次。
- Open issue：在项目中发起一个 issue，无论是讨论、bug 报告或提问，对项目都是带来活跃的，每个发起的 issue 计入 1 次。
- Open pull request：为项目提交一个 PR，表示已对该项目进行源码贡献，则每次发起一个 PR 计入 1 次。
- Pull request review comment：对项目中的 PR 进行 review 和讨论，需要对项目有相当的了解，并且对项目源码的质量有极大帮助，每个评论计入 1 次。

注：仅通过代码 review 对特定代码行的讨论记为 review，直接对 PR 的评论回复记为 issue comment 事件。

- Pull request merged：若有 PR 被项目合入，即便是很小的改动，也需要对项目有较为深入的理解，是帮助项目进步的真切贡献，则每一个 PR 被合入计入 1 次，同时 PR 合入事件根据该 PR 的代码增加行数。
- Watch：用户 star 项目，被计入 1 次。注：Star 操作在 GitHub 日志事件中记为 Watch 事件。
- Fork：开发者 fork 该项目，被计入 1 次

表 2.1 GitHub 2020 年全域开发者账号活跃度统计 Top 10

#	开发者账号	活跃度	issue_comment	open issue	open pull	pull review comment	merge pull	star	fork
1	dependabot[bot]	36,082,423.2	3,062,227	0	17,914,723	0	1,311,348.0	0	0
2	dependabot-preview[bot]	10,169,281.1	2,097,036	38,354	3,547,903	0	1,214,518.7	0	0
3	pull[bot]	9,591,558.7	0	0	3,204,996	0	2,585,151.0	0	0
4	renovate[bot]	2,668,048.6	57,090	2,450	949,411	0	620,122.0	0	0
5	github-learning-lab[bot]	1,988,666.3	1,578,947	727,017	93,468	53,231	66,450.2	0	0
6	github-actions[bot]	984,674.2	632,314	60,492	128,594	39,412	72,411.8	0	0
7	direwolf-github	894,975.0	0	0	516,714	0	0	0	0
8	codecov[bot]	826,249.8	838,764	0	0	0	0	0	0
9	snyk-bot	654,743.4	0	0	346,908	0	34,277.0	0	23
10	sonarcloud[bot]	527,040.0	755,729	0	0	0	0	0	0

如表 2.1 所示，自动化协作机器人由于运行在服务端，可以同时服务于众多项目，从而具有极高的活跃度和协作仓库数量，在上述统计开发者活跃度和活跃仓库数量时，已经过滤了 GitHub Apps 相关账号的协作行为。

2、GitHub Apps 使用情况

如表 2.1 所示，在全球最活跃开发者账号中，大部分为 GitHub Apps，故本报告对 GitHub Apps 数据做出相关统计，GitHub Apps 年活跃账号数量（活跃数量）与所产生日志总量占全年日志占比（日志占比）的变化如图 2.2 所示。

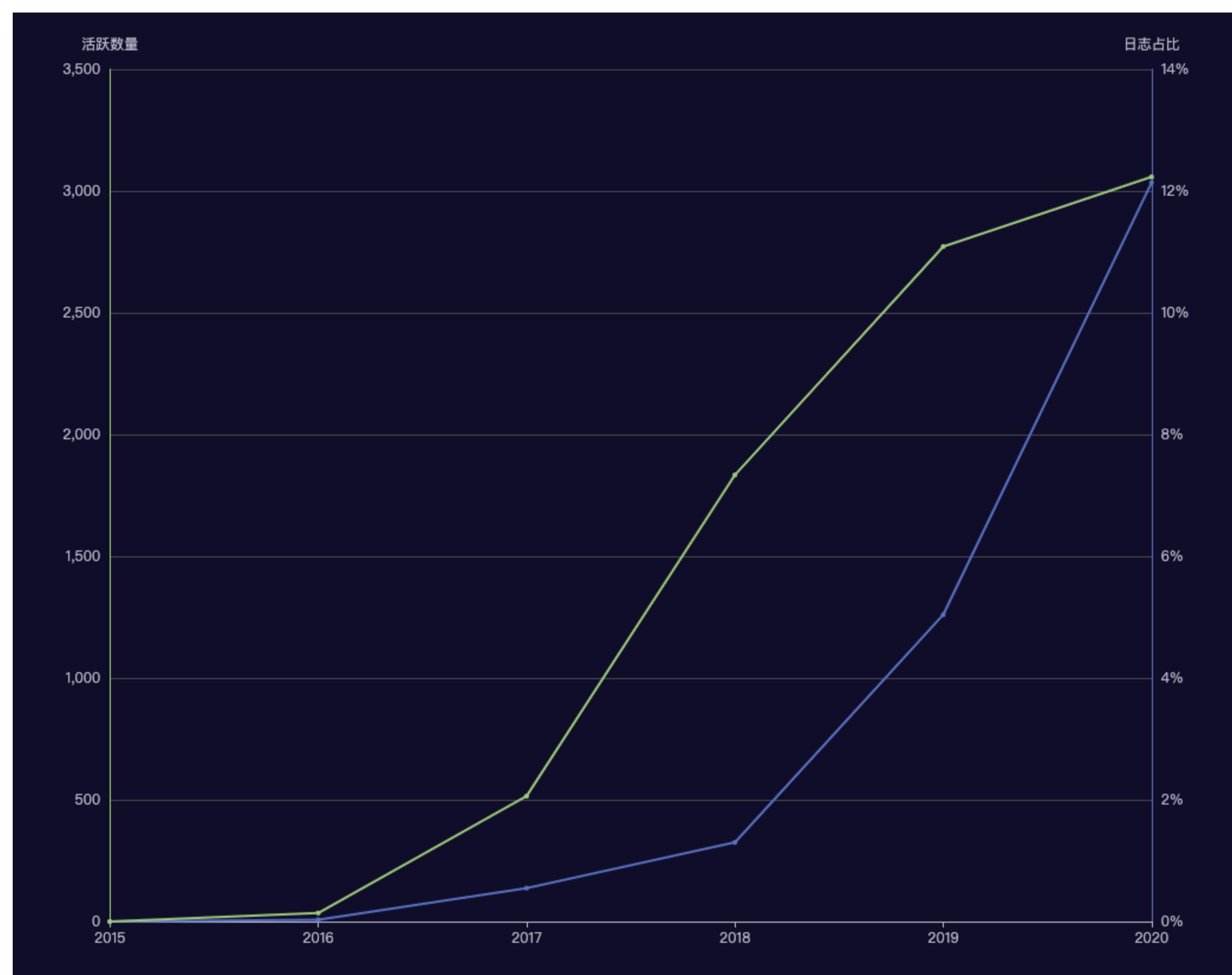


图 2.2 GitHub Apps 活跃账号数量与日志占比

通过图 2.2 可以看到，GitHub Apps 自 2016 年推出以来，在近年得到了迅猛发展。从日志量占比来看，2019 年相较于 2018 年提升了 288%，2020 年相较 2019 年增长 141%，达到了 12% 以上。

然而直到 2020 年，活跃的 GitHub Apps 账号数仅有 3,058 个，说明目前 GitHub Apps 的开发者还并不多。而其中服务仓库数量大于 100 个

以上 7 个种行为在该报告模型中各自独立计数，具有不一样的权重，根据专家经验，加权值分别为 1、2、3、4、2, 1, 2, 即：

$$A_{u-d} = C_{issue_comment} + 2C_{open_issue} \\ + 3C_{open_pr} + 4C_{review_comment} \\ + 2C_{pr_merged} + C_{watch} \\ + 2C_{fork}$$

其中，Pull request merged 的计数由分段函数决定：

$$C_{pr_merged} = \begin{cases} 0.8 + 0.002 \times loc & loc < 100 \\ 1 & 100 \leq loc < 300 \\ 2.5 - 0.005 \times loc & 300 \leq loc < 400 \\ 0.5 & loc \geq 400 \end{cases}$$

其中，loc 表示新增的代码行数。根据 [软件计量学经典数据统计](#)，单次代码变更最佳在 200 行以内，超过 400 行的代码变更会导致审阅困难，故通过分段函数关联了代码新增行数与 PR 的权重指标。

说明 2.2：开发者活跃仓库数量

开发者活跃仓库数量，定义为在上述开发者活跃度定义下，由每个开发者所产生的活跃度大于 0 的仓库数量。

说明 2.3：GitHub Apps

在此，简单介绍一下 GitHub Apps 是什么以及 GitHub Apps 的使用。

首先，GitHub Apps 的官方定义如下

GitHub Apps are first-class actors within GitHub. A GitHub App acts on its own behalf, taking actions via the API directly using its own identity, which means you don't need to maintain a bot or service account as a separate user.

的 GitHub Apps 仅 165 个，大于 1,000 个仓库的仅 59 个，说明大部分 GitHub Apps 还是个人开发小规模使用，而头部的几个 GitHub Apps 则占领了大量的市场（如使用最广泛的 `dependabot[bot]` 与 `dependabot-preview[bot]` 服务仓库数量超过 450 万个，超过了其他所有 GitHub Apps 服务仓库数量总和），成为 GitHub 上最为活跃的一批账号。

同时，对于 GitHub 平台上全域最活跃的一些 GitHub Apps，我们也对其进行了简单的分类与说明，如下表所示：

表 2.2 GitHub 2020 最活跃 GitHub Apps 账号说明

actor_login	actor_activity	involved areas	functions
<code>dependabot[bot]</code>	内置于 GitHub 的自动依赖项更新	升级依赖	拉取依赖项文件，并查找任何过时或不安全的需求。
<code>dependabot-preview[bot]</code>		升级依赖；安全监控	保持最新状态的PR;每次更新的兼容性打分;自动化安全建议;简单的上手流程。
<code>pull[bot]</code>	通过自动拉取请求让分支与上游保持最新	上下游更新	确保分支更新;自动集成来自上游的新更改;自动合并或硬重置拉取请求以匹配上游;分配reviewer以拉取请求
<code>renovate[bot]</code>		更新依赖项;自定义分组和计划	自动更新依赖项;支持多种语言;广泛的可配置性;支持共享预设作为代码。
<code>github-learning-lab[bot]</code>		使用引导	帮助用户学习如何使用 GitHub;通过机器人获取交互式说明和活动。
<code>github-actions[bot]</code>	自动化从创意到生产的工 作流程	自动化工作流	使用世界一流的 CI/CD 实现所有软件工作流的自动化。从 GitHub 构建、测试和部署代码。
<code>codecov[bot]</code>	为开发人员提供工具，以提高代码质量 和测试。	提高代码质量和测试	分组、合并、存档和比较覆盖范围报告。
<code>sonarcloud[bot]</code>	在线连续代码质量和代码安全性的领 先产品，完全免费用于开源项目。	提高代码质量和安全性	支持所有主要编程语言;检测错误、漏洞和代 码风格。

值得一提的是，用于本报告对应开源项目的协作机器人账号 `analysis-report-bot[bot]` 虽然仅在 2020 年 8 月后上线且仅服务于 GitHub 数字分析项目，但在全域 GitHub Apps 活跃度排行中位列全球第 289 位。相信在未来，基于 GitHub Apps 的自动化协作机器人会被更加广泛的用于项目的自动化协作，更好的帮助开源项目进行大规模协作的管理。

3、全域日志时间分布

由于 GitHub 事件日志具有详细的时间戳信息，故可以通过对时间维度的统计分析进行洞察，例如在 UTC 标准时间下，全球的工作时间分布如图 2.3 所示：

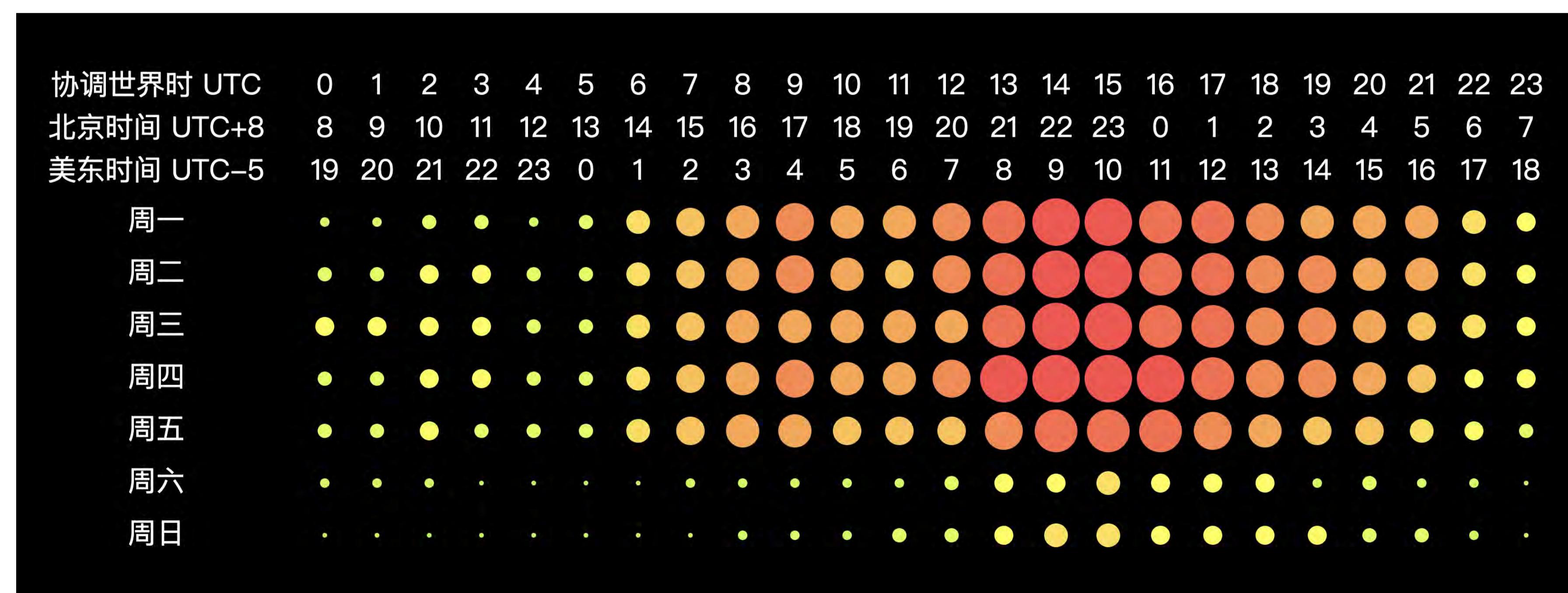


图 2.3 GitHub 2020 年全球日志时间分布情况

GitHub Apps 是 GitHub 官方提供的一类产品，它本身是一种应用程序，用户可以借由它使用 Github 提供的认证信息去调用 Github API 来完成一系列的操作。GitHub Apps 在运行时，以自己的身份来运行，它不代表任何其他的用户，它本身就是 GitHub 用户，但是，它有区别于一般用户的明显特征，即用户名以 “[bot]” 结尾，这很好的帮助我们将其从大量的日志中筛选出来。

GitHub Apps 的优点有很多，一个比较突出的优势是，GitHub Apps 可以做到对权限的精细管理，比如负责持续集成的 GitHub Apps 可以请求对仓库内容的读取权限和对状态的写入权限，又比如一个 GitHub Apps 可能没有对代码的读取或写入权限，但仍能管理 issues、labels 和 milestones。

图 2.3 为打孔图展示形式，图中横轴为一天 24 个小时，纵轴为一周 7 天，圆点大小表示该时段产生的日志量的相对大小，该图 0 时为 UTC 标准时间。

如 2.3 图所示，若我们认为主流开发者正常的工作时间为每日 9 时至 21 时，则在全球视角下，通过日志量来看，可以看到 GitHub 平台上的开发者应由欧美主导。而且周末的活跃明显低于工作日，也与 [GitHub Octoverse 2020](#) 报告中更多开发者使用 GitHub 工作而不仅仅是基于兴趣开发相吻合。

事实上，该打孔图用于特定社区或项目分析时更为有效，可以有效反应项目的工作时间分布情况。如图 2.4 所示，某项目在 2020 年全年的工作时间分布中，我们可以明显看到该项目周六的活跃与工作日相当，是一个较典型的 996 项目。

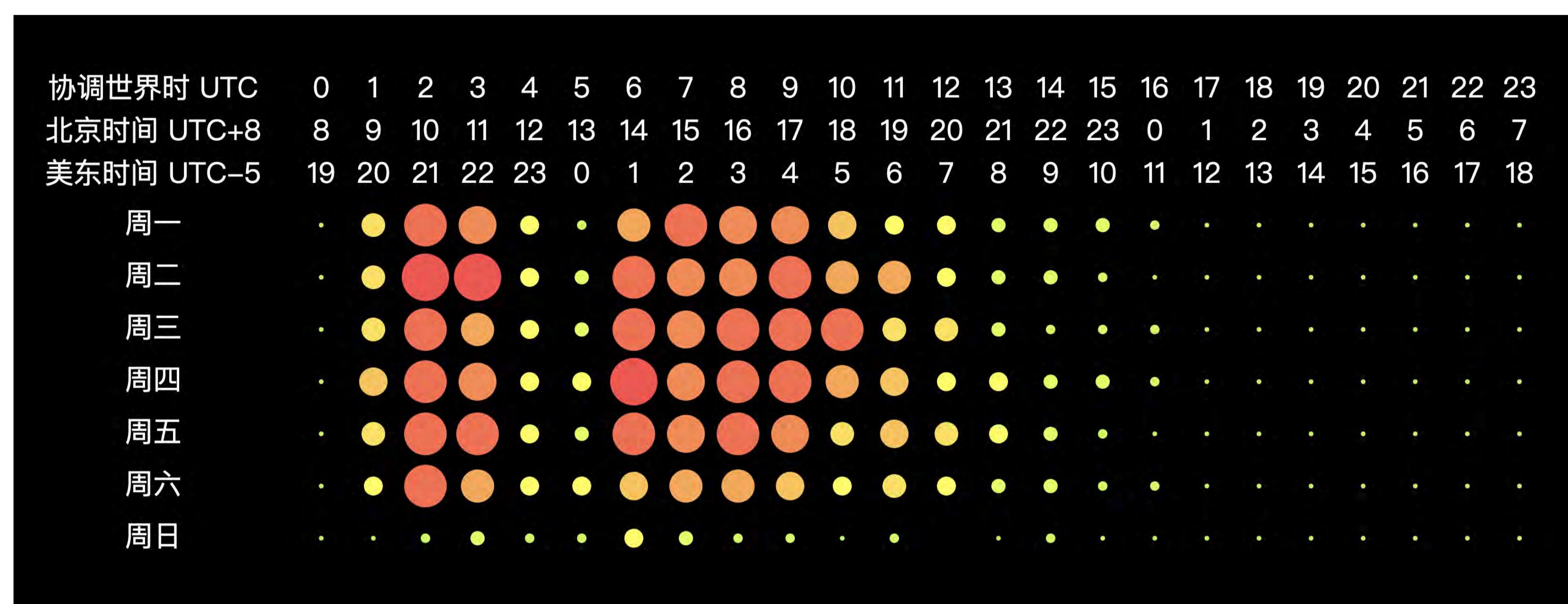


图 2.4 某项目在 GitHub 2020 年全年日志时间分布情况

4、典型开源开发者工作时间画像

我们还可以从日志中抽取每个开发者独立的日志在每日不同时间段的分布情况，并通过对开发者进行时区估计后将其移动合并到同一时区，从而得到开源开发者的典型工作时间分布情况。由于该估计需要开发者有足够的行为日志用于分析，故在剔除 GitHub Apps 的情况下，我们仅保留了全域活跃度排名前 5 万名的开发者账号用于分析，得到的工作时间画像如图 2.5 所示。

说明 2.4：UTC 标准时间

UTC 即协调世界时间，是国际通用的时间标准，将全球各地的时间进行同步协调。UTC 时间是经过平均太阳时（以格林威治时间 GMT 为准）、地轴运动修正后的新时标以及以秒为单位的国际原子时进行综合精算而得到的。

世界时区使用 UTC 的正或负偏移量表示。最西端的时区使用 UTC-12，比 UTC 落后十二小时；最东部的时区使用 UTC+14，比 UTC 早 14 小时。出现 UTC+13、UTC+14 的原因如下：

- 1、因为国际日期变更线，虽然 1884 年划定时避免在一个国家中同时存在着两种日期，但是 1979 年成立的基里巴斯领土却跨越了国际日期变更线。基里巴斯的 UTC 有：莱恩群岛(UTC+14)、菲尼克斯群岛(UTC+13)、吉尔伯特群岛(UTC+12)，这样保证一个国家内的日期为同一天。

- 2、因为夏令时，即天亮早的夏季人为将时间调快一小时。比如新西兰是 UTC+12，在夏时制改用 UTC+13。

说明 2.5：工作时间分布统计方法

工作时间分布统计通过开发者在每周内每日每小时的日志量进行统计，并线性最大最小归一化到 1 - 10 区间范围内，不包含活跃度信息。

说明 2.6：工作时间分布图理解方法

该图中横轴的 24 个刻度即对应一天内的 24 个小时，纵轴的 7 个刻度即对应一周下的所有工作日和周末，其中横纵坐标交叉处的圆点大小表示该时段产生的 GitHub 日志数量的相对大小，横轴的 0 刻度即 0 时，是 UTC 标准时间。圆点由小变大，颜色也渐变由黄转红。圆点越大，说明该时段产生的日志数量的越多，反映出该时间段下的开发者很活跃；通过横向比较圆点大小，可以看出一天内全球开发者活跃的时间段；通过纵向比较圆点大小，可以对比一周内每天的活跃情况。

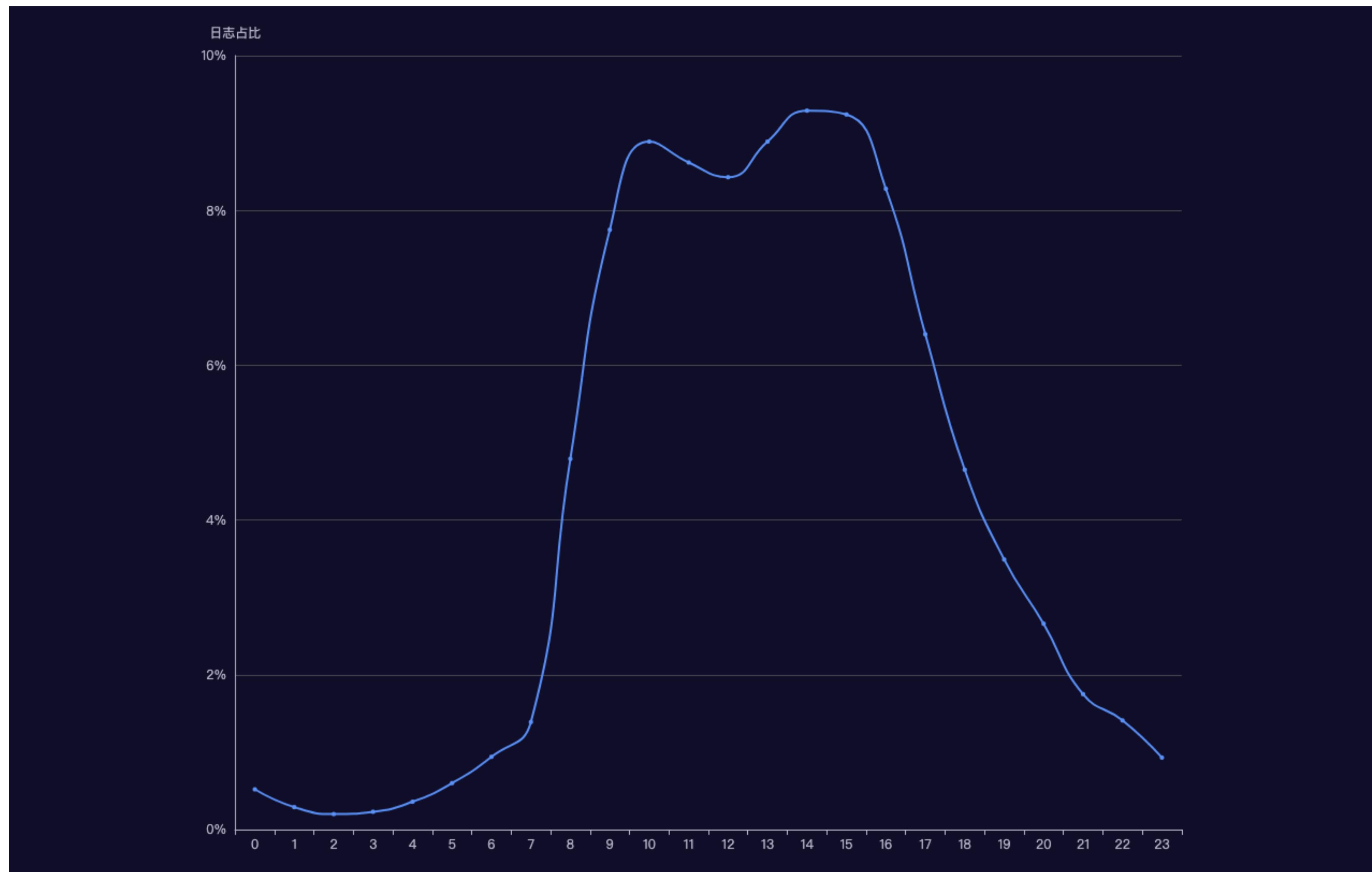


图 2.5 典型开源开发者工作时间画像

从上图不难看到，一般开发者会选择从当地时间早上 8 时开始工作，中午 11 - 13 时有短暂的午休时间，随后在下午 15 - 16 时达到一个生产力高峰并持续产出到晚间，较符合一般的工作时间情况，但可以看到开发者在晚间的工作产出比例还是较高，甚至可以工作到凌晨 1 点，活跃程度应该明显高于一般职业。

5、全球开发者时区分布

开发者的地理分布情况一直是开源项目全球化指标的一个重要方面，然而一直以来没有较好的方式来估计开发者所在时区，但在全域日志数据下，开发者的个人行为为我们提供了有效的估计手段。可以在全域范围或特定开发者群体内估计开发者在不同时区的占比情况，从而有效判断 GitHub 全域或特定项目内的全球化覆盖情况。通过对 GitHub 全域开发者活跃度前 5 万名开发者的统计，我们得到全球开发者在各时区分布估计如图 2.6 所示。

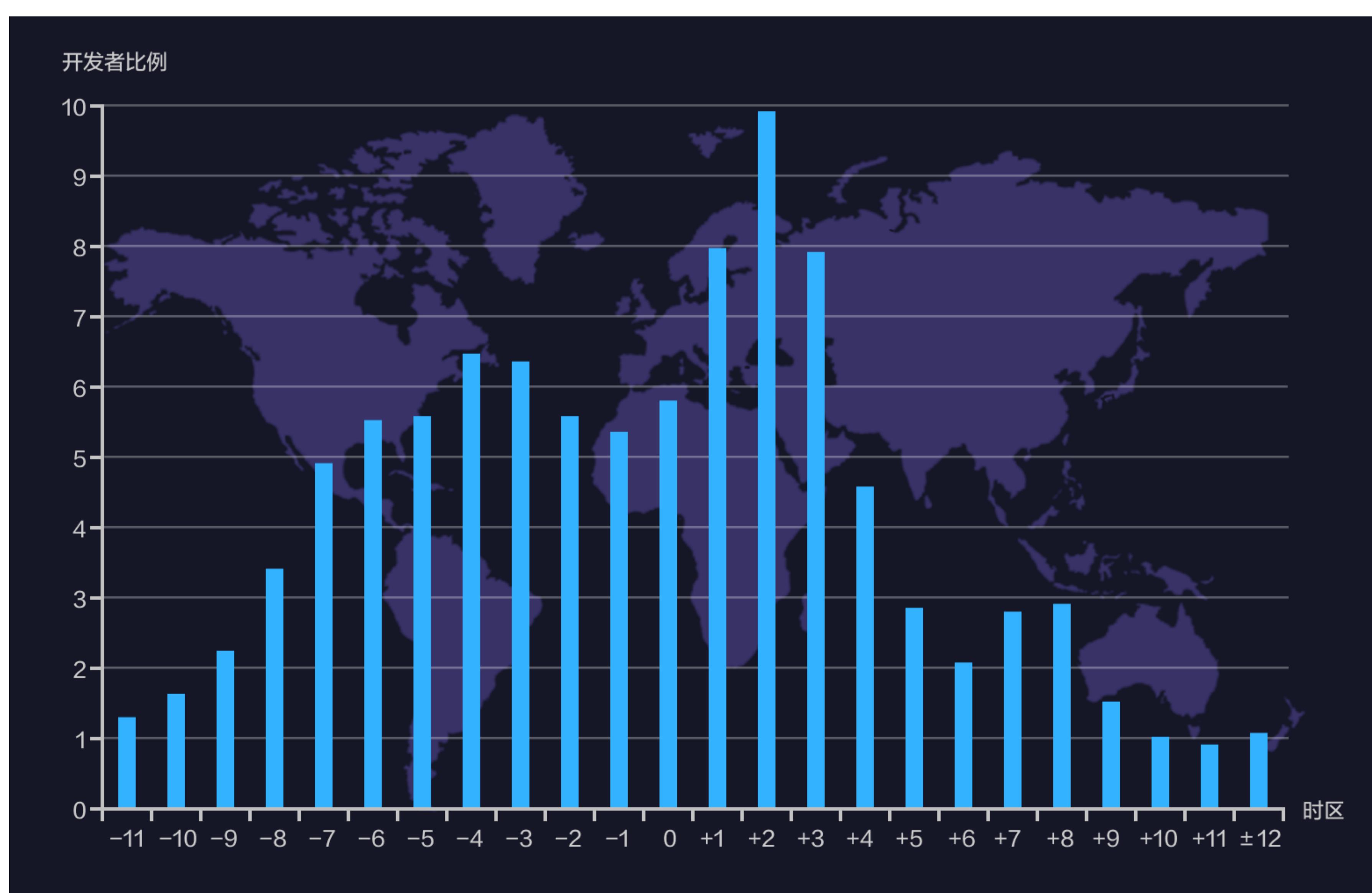


图 2.6 GitHub 2020 年全球开发者时区人数分布图

说明 2.7：开发者时区估计方法

根据开发者全年每小时产生日志的数量，计算获取其事件最多的连续 12 个小时，令其时间为该开发者本地时间的 9 时至 21 时，则可以大致确定该开发者的所属时区。

因为日志所记录的时间为 UTC 标准时间，我们考虑开发者一天连续 12 个小时产生日志数最多的最后一个小时，其最后一小时（即本地时间 21 时）对应的 UTC 时间记为 k ，则时区计算公式如下：

$$\text{zone} = \begin{cases} 20 - k^*, & k^* > 8 \text{ 对应本地为东时区} \\ -4 - k^*, & k^* \leq 8 \text{ 对应本地为西时区} \end{cases}$$

其中 $k^* = \text{argmax}\{\sum_{i=k-11}^k c_i\}$, $k = 0, 1, \dots, 23$

c_i 表示当前小时产生的日志数量，当 i 为负值时，意味着为前一天的时间，需进行转换，即使用 $i = i + 24$ 进行转换。

注：本方法很难准确估计开发者所属时区，因单个开发者的行为主具有偶发性与特殊性，故不可用于单开发者时区的精确估计，但在统计维度上有意义；另一方面，活跃度较低的开发者，其行为主也具有偶发特性。因此，本次报告在对开发者时区判断时，要求开发者活跃度排名要在前 5 万名。

通过图 2.6 我们可以看到，在高活跃开发者中，UTC-8 至 UTC-3，即美洲（美国、加拿大、南美）开发者分布最多，虽然单时区的开发者比例不是最高，但总体开发者占比高达 33% 左右。而 UTC+1 至 UTC+3，即欧洲拥有最高的单时区开发者比例，UTC+1 时区高达近 10%，三个时区的总占比约为 26% 左右。总体而言，亚洲的开发者数量依然较少，但在 UTC+7 至 UTC+8 区域有一个小的波峰，说明中国、俄罗斯开发者相较于其他国家还是有较高的开源活跃。而太平洋地区（UTC+9 至 UTC-9）则由于人口分布原因，开发者比例最低。

说明 2.8：开发者使用语言统计方法

开发者使用语言在该报告中定义为一个开发者账号在 2020 年使用频率最高的语言，具体实现为一个开发者账号在 2020 年提交并合入 PR 最多的项目所使用的编程语言。

6. 开发者使用语言分布

表 2.3 为 2020 年全域活跃的开发者使用的语言分布和活跃度 Top 10 万的开发者使用的语言分布情况，对比项目语言分布，可以看到排名有所波动。

表 2.3 GitHub 2020 年全域活跃开发者和 Top10 万活跃开发者语言分布对比

#	全域开发者语言榜	开发者账号数	#	Top 10W 万开发者语言榜	开发者账号数
1	JavaScript	305,814	1	JavaScript	15,858
2	Python	175,610	2	Python	10,866
3	HTML	159,303	3	TypeScript	7,419
4	Java	139,673	4	Java	6,665
5	Ruby	87,780	5	Go	5,094
6	TypeScript	85,116	6	C++	4,204
7	C#	54,343	7	Ruby	3,802
8	PHP	52,915	8	HTML	3,490
9	C++	47,799	9	PHP	3,000
10	CSS	46,528	10	C#	2,892

JavaScript 和 Python 是排行榜上岿然不动的冠亚军，而 HTML 和 CSS 两种语言放在全域开发者语境下显然更受欢迎，这和 GitHub 上大量的博客网站等类型的仓库有关，它们一般为小型独立项目，由个体开发者维护，但是体量巨大。

三、项目分析

1、全域旅游项目总体数据

基于开发者活跃度定义，我们也给出开源项目活跃度的计算方法。在给定的活跃度计算方法下，过滤了 GitHub Apps 相关账号的协作行为，共统计得到 2020 年有效的总活跃项目数量约 **1,167** 万个。这些项目的活跃度分布情况与项目中的参与开发者数量分布情况如下图所示。

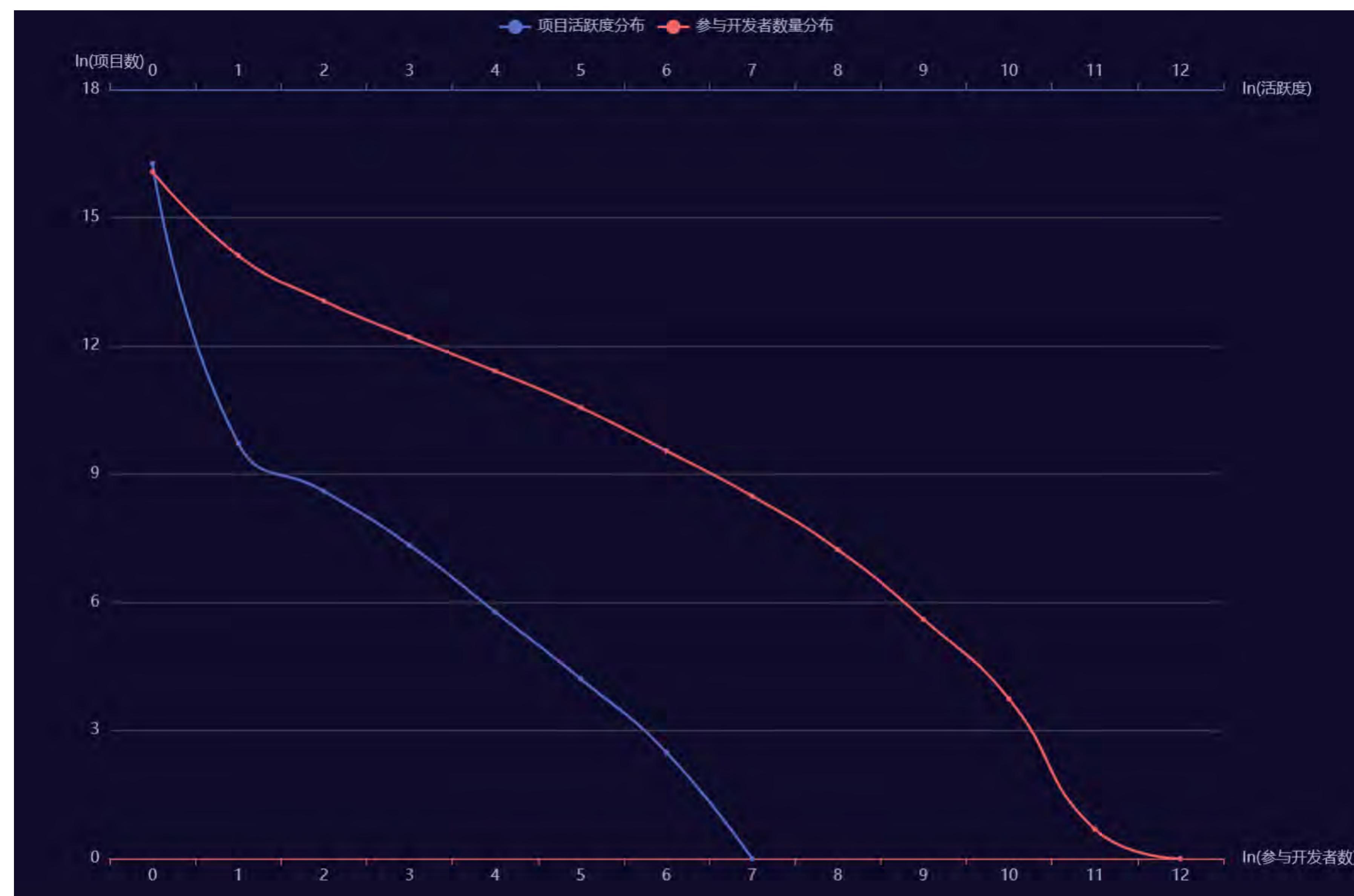


图 3.1 GitHub 2020 活跃项目的活跃度及参与开发者数量分布情况

上图使用双对数坐标系绘制，图上方横坐标表示项目的活跃度，图下方横坐标表示项目的参与开发者数量，纵坐标表示相应的项目数量。据统计，2020 年全年项目的活跃度值最高为 971.2，但高活跃度的项目数量占比极低，超过 99.95% 的项目活跃度值都低于 10，即绝大部分项目处于低活跃状态。另外，单个项目中的参与开发者数量最高达 85,532，即在 2020 年内最多有 85,532 个开发者参与了同一个项目。但有 71.21% 的项目参与开发者数量都低于 10，即 2020 年全年 GitHub 上大部分项目的参与人数都在 10 人以内。

说明 3.1：项目活跃度

项目活跃度，其定义为某特定项目在一段时间内的活跃评价指标。其活跃度由单开发者在一天内的活跃度加总计算得到（时间点与时间段计算方式均基于开发者个人活跃度方式计算）：

$$A_r = \sum (A_{u_d} / day_count)$$

2. 项目活跃度 Top 20

根据项目活跃度定义，我们对 2020 年全年活跃项目进行了活跃度统计与排名，这里给出世界活跃度 Top 20 项目的列表。由于该报告为 2020 数据报告，故该表中所有数据均为 2020 年新增数量，并未陈列历史数据。

表 3.1 GitHub 2020 年全域项目活跃度 Top 20

#	项目名	活跃度	参与开发者数量	issue comment	open issue	open pull	pull review comment	merge pull	star	fork
1	pddemo/demo	971.2	2	0	251,354	0	0	0.0	1	0
2	google-test/signcla-probe-repo	871.5	6	138,621	0	104,135	0	0.0	1	0
3	test-organization-kkjeer/app-test	703.1	6	109,172	40,226	47,731	7,529	18,169.3	1	0
4	elastic/kibana	698.0	5326	145,309	9,838	22,891	45,224	17,094.3	2,202	1,293
5	test-organization-kkjeer/bot-validation	691.3	8	104,083	40,242	47,729	7,425	17,724.2	2	0
6	ouyanxia2/hgmgmg	627.3	52	4	162,355	0	0	0.0	0	0
7	imamandrews/imamandrews.github.io	621.2	13	79,182	104,764	13	0	0.0	3	2
8	kubernetes/kubernetes	601.6	23279	23,5006	3,897	6,868	30,721	3,984.3	13,327	6,181
9	flutter/flutter	583.8	52414	126,337	17,538	7,222	18,438	4,124.3	33,130	6,714
10	NixOS/nixpkgs	538.1	5467	84,113	4,489	26,407	30,328	18,371.8	1,798	1,956
11	microsoft/vscode	432.2	46612	99,180	23,349	1,977	1,780	1,220.8	23,347	6,470
12	rust-lang/rust	422.8	14532	103,937	4,799	7,809	25,194	5,039.9	10,912	1,820
13	Ramos-dev/jniwebshell	417.7	3	0	108,092	0	0	0.0	2	0
14	dotnet/runtime	415.2	8914	81,740	7,886	7,942	40,000	5,662.5	4,007	1,616
15	pytorch/pytorch	413.4	19643	67,923	5,805	12,165	38,319	283.2	11,781	4,313
16	home-assistant/core	388.6	21407	76,135	6,101	7,987	30,681	5,621.0	9,832	5,526
17	MicrosoftDocs/azure-docs	339.0	18766	85,504	17,367	5,165	1,015	2,791.0	1,709	3,828
18	apache/spark	324.7	7465	138,874	0	3,877	32,252	1.8	4,581	3,263
19	elastic/elasticsearch	319.5	12865	40,330	3,610	12,515	22,599	9,791.7	7,848	4,202
20	odoo/odoo	312.8	8304	54,666	1,706	19,392	19,509	26.8	4,521	2,879

可以发现，前 5 名项目中，除了 elastic/kibana 外，其他项目均是有人机参与协作的项目。位于表格榜首的项目是 pddemo/demo，其 open issue 的数量特别得高，该项目的简介是 “A new issue is created in this repo every minute”，与表现特征相吻合。

google-test/signcla-probe-repo 位于第 2 名，该项目用于验证 SignCLA 是否正确运行，位于第 3 名和第 4 名的 test-organization-kkjeer/app-test 和 test-organization-kkjeer/bot-validation 均用于 bot 的验证。

由于自动化行为的影响较大，在各指标均大于 0 的前提下，我们对 2020 年全年活跃项目进行了活跃度统计和排名，表 3.2 是当各指标均大于 0 时，世界活跃度 Top 20 项目的列表。

表 3.2 GitHub 2020 年全域项目活跃度 Top 20（项目的各项指标均大于 0）

#	项目名	活跃度	参与开发者数量	issue comment	open issue	open pull	pull review comment	merge pull	star	fork
1	elastic/kibana	698.0	5,326	145,309	9,838	22,891	45,224	17,094.3	2,202	1,293
2	kubernetes/kubernetes	601.6	23,279	23,5006	3,897	6,868	30,721	3,984.3	13,327	6,181
3	flutter/flutter	583.8	52,414	126,337	17,538	7,222	18,438	4,124.3	33,130	6,714
4	NixOS/nixpkgs	538.1	5,467	84,113	4,489	26,407	30,328	18,371.8	1,798	1,956
5	microsoft/vscode	432.2	46,612	99,180	23,349	1,977	1,780	1,220.8	23,347	6,470
6	rust-lang/rust	422.8	14,532	103,937	4,799	7,809	25,194	5,039.9	10,912	1,820
7	dotnet/runtime	415.2	8,914	81,740	7,886	7,942	40,000	5,662.5	4,007	1,616
8	pytorch/pytorch	413.4	19,643	67,923	5,805	12,165	38,319	283.2	11,781	4,313
9	home-assistant/core	388.6	21,407	76,135	6,101	7,987	30,681	5,621.0	9,832	5,526
10	MicrosoftDocs/azure-docs	339.0	18,766	85,504	17,367	5,165	1,015	2,791.0	1,709	3,828
11	elastic/elasticsearch	319.5	12,865	40,330	3,610	12,515	22,599	9,791.7	7,848	4,202
12	odoo/odoo	312.8	8,304	54,666	1,706	19,392	19,509	26.8	4,521	2,879
13	tensorflow/tensorflow	308.8	33,089	62,952	7,277	3,126	7,960	1,909.5	17,321	8,392
14	labuladong/fucking-algorithm	285.7	85,532	679	261	360	37	123.1	81,970	15,273
15	istio/istio	257.2	8,512	82,976	4,338	5,519	14,572	3,826.9	5,361	1,465
16	SmartThingsCommunity/SmartThingsPublic	251.7	16,391	878	14	37,512	1,790	484.7	463	16,247
17	kubernetes/websit	242.6	4,911	54,180	2,222	5,139	21,281	3,345.1	706	3,304
18	cockroachdb/cockroach	236.0	3,218	55,887	7,682	6,819	4,096	4,957.7	2,424	564
19	pandas-dev/pandas	235.3	11,008	32,697	2,906	5,282	21,512	3,870.2	5,835	3,563
20	jwasham/coding-interview-university	234.2	70,867	246	92	138	22	37.8	63,162	16,011

从表 3.2 中可以看到，活跃度最高的项目是来自 elastic 社区的开源数据可视化仪表板 Kibana。Kibana 是一种开源分析和可视化工具，可通过基于浏览器的界面轻松搜索，可视化和探索大量数据。

谷歌发起的前端跨平台开发框架 Flutter 和容器编排系统 Kubernetes 分别位于第 2 名和第 5 名，这说明了谷歌在开源上的努力和影响力获得了业内的认可。

微软发起的跨平台代码编辑器 microsoft/vscode 和微软使用开源的方式来建设其 Azure 云平台的项目 MicrosoftDocs/azure-docs 分别位于第 7 名和第 10 名，这说明了微软在开源上的努力获得了程序员的认可。

值得注意的是，rust-lang/rust 的排名上升得很快，位于第 4 名。在 2020 年，Rust 的 GitHub Star 数达到了 51k，Reddit Fans 则涨至 125k，全年合并 PR 达 8,114 个，这些数据都在说明，Rust 正成为越来越受欢迎的编程语言。就像 Tiobe software 的首席执行官 Paul Jansen 说得：“Rust 解决了所有其他冗长的编程和其他语言的尖锐问题，同时进行了静态强类型化。其类型系统可防止运行时空指针异常，并且可在编译时计算内存管理。”这可能是 Rust 越来越受欢迎的原因之一。

3、中国项目活跃度排名

同时，我们也通过各种渠道采集了中国的开源项目列表，并同样给出了中国项目的活跃度排名情况，如表 3.3 所示。

表 3.3 GitHub 2020 中国项目活跃度 Top 20

#	项目名	活跃度	参与开发者数量	issue comment	open issue	open pull	pull review comment	merge pull	star	fork
1	pingcap/tidb	210.1	5,831	53,022	2,801	4,969	10,928	3,459.2	4,862	1,052
2	ant-design/ant-design	193.3	23,620	32,026	4,836	3,131	3,320	2,130.7	12,709	8,052
3	PaddlePaddle/Paddle	127.4	4,842	15,329	2,256	5,656	9,625	3,478.2	3,574	786
4	tikv/tikv	81.7	2,593	17,817	997	2,019	5,547	1,279.9	2,129	434
5	apache/shardingsphere	75.3	5,267	9,055	1,713	3,235	1,858	2,539.5	3,834	1,443
6	apache/incubator-tvm	70.4	2,148	7,961	437	2,112	8,506	1,540.1	1,454	662
7	pingcap/docs-cn	65.1	532	8,202	96	2,965	6,959	2,315.9	140	320
8	apache/incubator-echarts	64.2	11,638	7,650	1,620	324	346	194.5	6,664	4,463
9	pingcap/pd	60.9	437	13,325	667	1,667	4,972	1,297.7	214	224
10	alibaba/nacos	59.9	9,956	7,042	1,640	706	827	410.0	6,347	3,450
11	NervJS/taro	54.7	7,469	9,339	2,231	917	135	551.5	5,250	1,012
12	youzan/vant	54.2	9,806	4,897	1,661	715	201	554.4	4,672	4,502
13	pingcap/docs	53.9	314	7,014	64	2,736	5,226	2,257.8	90	164
14	ElemeFE/element	52.7	11,749	4,993	1,762	297	10	33.3	6,853	3,411
15	apache/skywalking	51.9	5,556	6,783	1,084	860	3,455	583.4	4,201	1,471
16	PaddlePaddle/PaddleOCR	47.9	9,394	4,039	1,033	573	622	420.0	8,430	1,664
17	apache/incubator-dolphinscheduler	47.1	2,588	9,364	1,269	1,407	730	902.7	1,835	909
18	apache/apisix	45.4	2,923	5,855	1,109	1,029	3,383	715.0	2,496	579
19	seata/seata	45.1	7,339	3,754	785	517	1,805	313.5	5,261	2,296
20	pingcap/tidb-operator	45.1	425	8,627	703	1,498	3,683	1,172.1	240	140

从这个列表中，我们发现，PingCAP 在开源领域的表现非常地亮眼。Top 20 项目中上榜的项目有 6 个，包括位于榜首的由其自主设计、研发的开源分布式关系型数据库 pingcap/tidb，分布式事务型的键值数据库 tikv/tikv，文档型项目 pingcap/docs-cn、pingcap/docs 等等，表明 PingCAP 很重视项目文档的建设。

说明 3.2：中国项目及企业项目归属说明

我们通过多方数据源收集由中国本土开发者和本土科技公司开源的项目，具体见[开源项目配置列表](#)，如发现疏漏或错误，欢迎提交 Issue 或 PR 进行补充和修正。

阿里在开源领域中的成绩也是非常不错。Top 10 项目中上榜的项目有 2 个，分别是蚂蚁金服采用 React 封装的一套组件库 ant-design/ant-design（位于第 2 名），以及致力于配置和管理微服务的特性集 alibaba/nacos。

百度在人工智能领域的表现非常不错，其深度学习平台 PaddlePaddle 占据了 2 个项目，分别是核心框架 Paddle 以及相关工具库。

中国的 Top 20 项目列表中，包括阿里的 Ant-Design 组件库，京东基于 React 前端框架的开发框架 taro，由饿了么（已被阿里收购）前端团队开源的 Vue UI 组件库 Element 等等，这说明了在国内，前端群体在社区更为活跃；另外前端代码一般也不太涉密，因此公司在心态上更开放一些。不过这其中也有一点需要引起注意，上榜的前端项目组件库居多，但是缺少核心项目。

值得关注得是，与 GitHub 2019 报告中的中国 Top 20 项目相比，GitHub 2020 的中国 Top 20 项目列表中，由 Apache 孵化得项目占据的席位由 2 席增加到了 6 席，从侧面表现出大家对 Apache 项目的关注度以及参与度越来越高。

而在各大开源项目的背后，基本都有科技公司的支持，我们计算出了科技公司所维护的开源项目在 2020 年的活跃情况，结果如表 3.4 所示：

表 3.4 GitHub 2020 中国企业开源项目活跃度一览

#	公司	活跃度	项目数量	issue comment	open issue	open pull	pull review comment	merge pull	star	fork
1	Alibaba	1,571.1	1,496	130,558	33,947	29,097	22,615	17,471.6	216,980	68,864
2	PingCAP	778.4	151	139,255	8,138	25,401	61,538	18,880.4	18,008	5,058
3	Baidu	671.2	540	55,265	12,592	20,720	23,380	13,475.9	70,960	22,148
4	Tencent	432.3	388	21,446	8,599	10,264	2,870	7,088.8	69,198	19,348
5	JD	153.0	74	20,126	4,504	4,483	2,043	3,214.2	13,119	3,316
6	Huawei	101.8	200	10,322	1,709	2,930	3,867	2,005.2	8,758	3,168
7	DiDi	89.4	63	3,114	1,290	827	207	508.7	20,489	3,907
8	Youzan	88.6	58	7,259	2,760	1,409	634	1,068.1	9,509	5,820
9	Bytedance	59.2	85	1,973	645	785	659	514.9	14,034	1,671
10	WeBank	57.9	59	2,197	718	3,501	596	2,411.9	5,225	1,902
11	Xiaomi	50.4	98	1,767	1,604	1,007	3,001	691.6	5,823	1,760
12	Meituan	46.9	68	1,356	564	305	17	147.0	10,879	2,573
13	Bilibili	42.7	51	1,306	446	132	52	66.4	10,278	2,295
14	360	39.8	147	1,769	810	441	40	231.7	8,105	1,914
15	Juejin	39.5	26	3,866	578	661	3,624	546.9	4,208	810
16	CTrip	36.9	25	2,346	537	216	276	130.5	6,196	2,562
17	Linux China	34.1	16	226	10	3,862	11	3,123.3	482	302
18	Netease	25.0	119	1,603	777	313	32	149.6	3,880	1,445
19	Deepin	18.6	267	2,555	931	326	21	132.7	1,339	821
20	Qunar	7.1	43	113	54	56	10	9.9	1,653	478
21	Vipshop	7.1	14	112	127	66	0	14.0	1,604	421
22	Douban	3.7	41	98	43	158	58	128.6	508	99

在国内企业的开源数据中，我们可以看到阿里巴巴的数据在很多指标上的表现里都非常不错，有的指标值甚至是其它公司之和，并且在社区化/开放等方面也做得不错。

可以看出，各大企业近年来均在不断加大开源社区生态建设。PingCAP 在 2020 年下半年宣布完成 2.7 亿美元的 D 轮融资，创造了全球数据库历史新的里程碑，同样，PingCAP 今天在开源方面的表现也是非常亮

眼，已经超越百度跃居排行榜第二位；其中，pull review comment 的数量更是超过了阿里，可见 PingCAP 的开源社区非常的活跃。AI 是百度开源最鲜明的竞争力，比如深度学习平台 PaddlePaddle 和自动驾驶平台 Apollo。有赞的排名上升得非常快，这可能得益于其开源项目 youzan/vant 的优秀表现，该项目是轻量级的移动UI组件。

4、项目语言分布情况

基于项目活跃度的定义，我们针对全域项目计算了不同开发语言的项目的活跃度总和，如图 3.2 所示。可以发现使用 JavaScript 的项目总活跃度最高，紧随其后的是 Python 和 TypeScript。

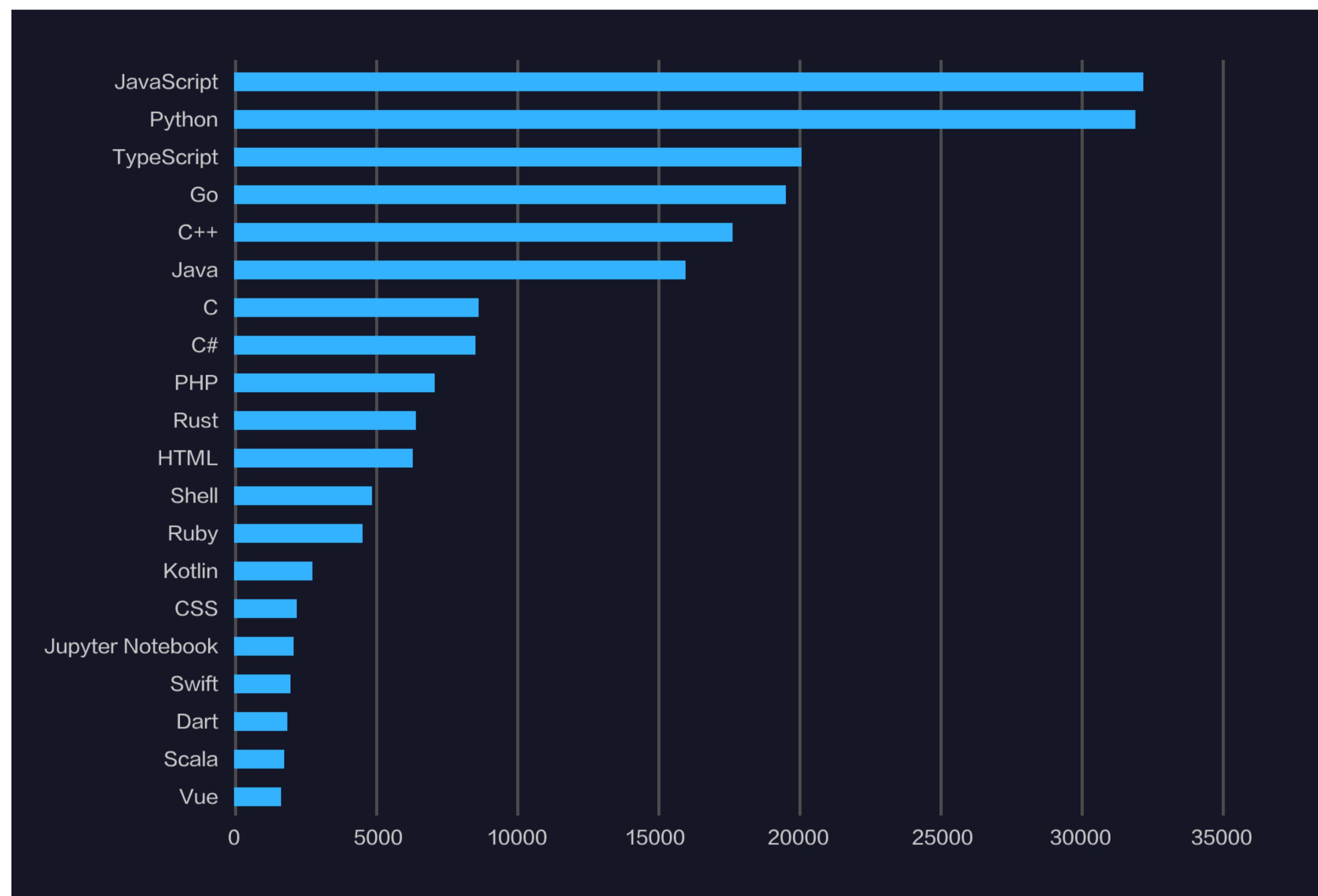


图 3.2 GitHub 2020 项目活跃度语言分布情况

JavaScript 排名居高不下的原因分析如下：

- JavaScript 是一门直接被嵌入到 HTML 中的脚本语言，是一门可以被 Web 浏览器理解的语言。它无需编译、在浏览器环境可以直接运行的特性，让 JavaScript 项目在活跃度的提升上占据了优势。
- 谷歌开发的 Angular 和 Facebook 开发的 React，另外还有 Vue，这些主流前端框架都是 JavaScript 生态圈中的一部分。
- 这几年随着行业的发展，JavaScript 变得几乎“无所不能”，现在基本所有的主流互联网应用，其前端都有大量的 JavaScript 代码。比如我们日常使用的邮件、社交工具等等。

Python 的热度也是未曾减少，排名居于第二位，分析可能的原因为：

- Python 相较于其他主流编程语言具有更好的可读性，简单易学、易于维护。
- Python 应用范围广，它自带的各种模块加上丰富的第三方模块，免去了很多“重复造轮子”的工作，可以更快地实现多种功能。

- 人工智能的浪潮进一步推动了 Python 的发展，很多人工智能任务以及大数据分析都会优先使用 Python 实现。

值得关注的是，TypeScript 的受欢迎程度急剧攀升，排名出现了较大的提升，可能的原因如下：

- TypeScript 是一门比较年轻的语言，发行到现在只有八年时间，但它继承了 JavaScript 的部分人气。
- TypeScript 只需编译一次，就可以在服务器、浏览器或你喜欢的任何地方来运行它。
- 在某些领域，TypeScript 是不可代替的，且代码的可读性和可维护性较强。

5. OpenGalaxy

如 2.2 小节所示，通过活跃度的数据统计得到的结果会受到自动化协作行为的影响，并且不同生命周期阶段的项目的活跃度可能不具备可比性，故在本次报告中我们引入了全域项目协作关系网络，开源星系——OpenGalaxy。

OpenGalaxy 通过开发者在 GitHub 平台上的所有协作行为构建起项目间的协作关系，并通过图算法提供了一种影响力和聚类计算方法。在该计算方法下，影响力较大的项目由开发者的行而关联在一起，项目的影响力将不再是简单的数据统计，优秀的开发者和优秀的项目互相吸引从而可提升项目的影响力，即便在不过滤自动化行为的情况下也可以给出较好的结果。

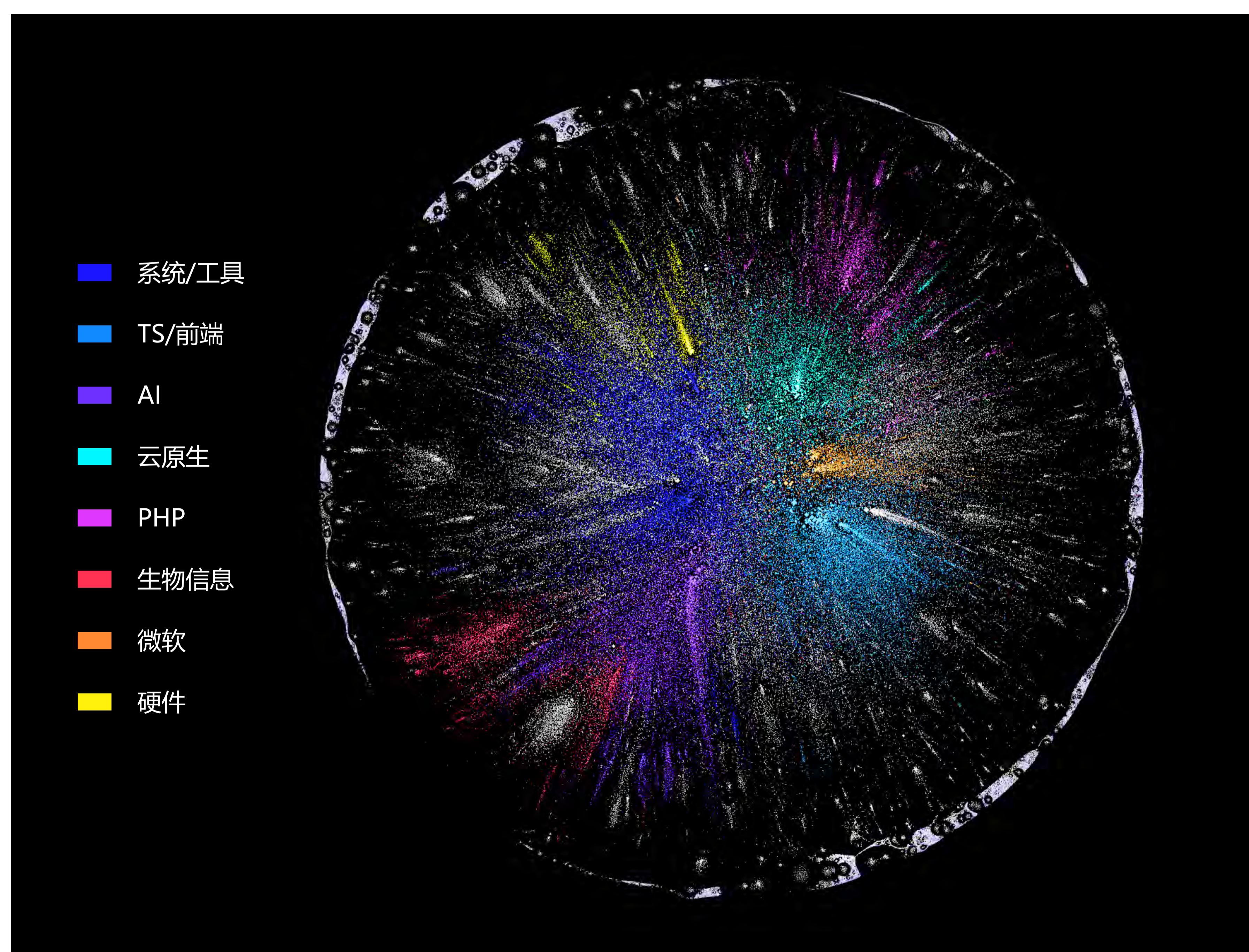


图 3.3 GitHub 2020 全域项目协作关系网络——OpenGalaxy 2020

说明 3.3: OpenGalaxy 的构建方法说明

OpenGalaxy 即全域项目协作网络是基于说明 2.1 中开发者活跃度的定义，通过开发者在多个项目中的协作关系进行构建的。具体计算方式为：

$$R_{ab} = \sum_i \frac{A_{ia} A_{ib}}{A_{ia} + A_{ib}}$$

其中 A_{ia} A_{ib} 分别为开发者 i 在项目 a 和项目 b 上的活跃度，遵循说明 2.1 中的活跃度计算方法（由于 star 行为具有盲从性，对聚类结果有影响，故 star 不计入）。 R_{ab} 为项目 a 和项目 b 在的协作关联度。即两个项目的协作关联度为其所有共同活跃开发者的活跃度的调和平均值之和。

基于上述方法构建得到全域活跃项目数 **105.7 万**，协作关系共计 **2,607 万**，即得到的全域项目协作网络为 **105.7 万个节点、2,607 万条边的无向图**。在该图上应用加权 PageRank 算法计算至收敛（共计迭代 128 次），得到每个节点的 PageRank 值为该节点影响力指标。在该图上应用 Louvain 社区发现算法，得到的一级社区划分记为项目聚类结果，即项目在协作网络下的所属领域。

各项分析均基于该图进行，考虑到可视化效果，OpenGalaxy 渲染图使用了 PageRank 大于 1 的所有项目共计约 **22.1 万个**，协作关系使用协作关联度大于 2 的边共计约 **292.2 万条**。

5.1 全域项目下的影响力

图 3.3 为 GitHub 2020 最活跃的 22.1 万个开源项目组成的协作网络图。该图中节点的大小表示项目的影响力大小，节点的着色表示节点所属的协作聚类结果。此处影响力区别于活跃度，具有更好的算法稳定性，更能体现开发者的整体活跃情况对项目的影响，影响力的计算方式请参考说明 3.3。在协作网络的影响力评估下，得到 GitHub 2020 全域影响力最高的项目 Top 20 如下表所示。

表 3.5 OpenGalaxy 2020 全域项目影响力 Top 20

#	项目名	影响力
1	microsoft/vscode	977.3
2	flutter/flutter	593.3
3	tensorflow/tensorflow	484.8
4	microsoft/TypeScript	448.5
5	DefinitelyTyped/DefinitelyTyped	446.7
6	microsoft/WSL	431.7
7	golang/go	404.6
8	gatsbyjs/gatsby	381.3
9	kubernetes/kubernetes	370.1
10	vercel/next.js	354.5
11	pytorch/pytorch	340.2
12	NixOS/nixpkgs	335.4
13	rust-lang/rust	328.9
14	home-assistant/core	317.9
15	Homebrew/homebrew-core	311.5
16	microsoft/terminal	301.1
17	nodejs/node	292.9
18	MicrosoftDocs/azure-docs	285.5
19	electron/electron	276.5
20	facebook/create-react-app	276.3

从表 3.5 可以看到，VSCode 的影响力从活跃度的第 5 跃升为第 1，且高于排名第 2 的 flutter 约 64.7%，以巨大的优势成为全球最具影响力项目。

事实上这是由于 VSCode 在成为全球最流行的 IDE 的同时，也与其他各领域的顶级项目产生了大量的协作关联。如表 3.6 所示，除 VSCode 自己的编写语言和技术相关的 TypeScript、electron 等之外，VSCode 还与其相关的语言支持插件有紧密的关系。例如 Go 语言支持项目 microsoft/vscode-go，该项目协作关联度最高的项目即为排在影响力第 7 的 golang/go；Python 语言支持项目 microsoft/vscode-python，该项目协作关联度前 10 位中就包含影响力排名第 3 的 tensorflow 和著名的 AI 类库 pandas（影响力 201）；Dart 语言支持项目 Dart-Code/Dart-Code，由于 Dart 主要用于 flutter 的开发，故 Dart-Code 的作者 DanTup 也建立起了 VSCode 与 flutter 的桥梁，在与 VSCode 协作关联度的排名中，flutter 也高居第 9 位，而前八位则几乎全部来自微软或 VSCode 生态项目。

表 3.6 VSCode 2020 年全年全域协作关联项目 Top 10

#	项目名	关联度	影响力
1	microsoft/TypeScript	803.3	448.5
2	microsoft/vscode-remote-release	576.7	214.2
3	microsoft/vscode-python	500.2	205.3
4	DefinitelyTyped/DefinitelyTyped	392.1	446.7
5	microsoft/terminal	374.6	301.1
6	microsoft/vscode-cpptools	316.8	131.9
7	microsoft/WSL	289.6	431.7
8	electron/electron	261.4	276.5
9	flutter/flutter	246.3	593.3
10	microsoft/vscode-docs	236.3	38.6

综上所述，OpenGalaxy 具有优秀项目会因优秀开发者而关联在一起的特性，从而不会因自动化行为导致影响力指标虚高，具有较好的算法稳定性，影响力指标的变化背后一定意味着开发者群体的活跃行为迁移，从而可以较好的反映出 GitHub 全域的项目影响力状况。

5.2 全域项目下的协作网络聚类

由于使用协作网络来描述 GitHub 全域项目的协作关系，故可以通过项目协作关联度和社区发现算法对项目进行聚类，从而得到基于协作行为的聚类效果，可以用于项目的大致分类。我们通过 Louvain 社区发现算法对上述网络进行聚类，并给出如下的分类结果。

表 3.7 OpenGalaxy 2020 全域项目聚类结果 Top 8

#	所属领域	顶级项目	项目数量
1	系统/工具	rust-lang/rust,Homebrew/homebrew-core,ytdl-org/youtube-dl	26,601
2	TS/前端	microsoft/vscode,microsoft/TypeScript,vercel/next.js	25,693
3	AI	tensorflow/tensorflow,pytorch/pytorch,conda-forge/staged-recipes	21,024
4	云原生	golang/go,kubernetes/kubernetes,moby/moby	12,637
5	PHP	laravel/framework,symfony/symfony,composer/composer	8,365
6	生物信息	rstudio/rstudio,bioconda/bioconda-recipes,apache/arrow	7,258
7	微软	microsoft/WSL,microsoft/terminal,dotnet/runtime	6,626
8	硬件	home-assistant/core,espressif/arduino-esp32,arduino/Arduino,	6,352

图聚类算法可以根据图中边的权重信息，将在协作行为中关系较紧密的项目聚成一类，从而帮助我们在全域数据中发掘项目的类型信息，对于开源项目技术战略方向的观察提供了新颖的视角。由于低活跃项目的聚类结果并不准确，表 3.7 中的项目数量以 OpenGalaxy 中影响力最高的 22 万个项目为基准。如果以全部项目为基准则前端项目群数量第 1，而生物信息项目群为第 8。说明更多的开源低活跃开发者更倾向于前端项目，而生物信息项目的数量不多，但整体质量较高。

事实上上述通过协作网络得到的聚类结果并不完全准确，尤其如系统/工具类项目，其实并没有很好的协作粘性，不具有技术栈上的一致性或关联性。而其他领域项目则技术领域性质较为明显。其中有意思的是微软的开源项目自成一个协作类别，也说明了微软对开源战略的贯彻非常彻底，直接使用 GitHub 作为项目研发的平台，导致其项目群有很高的关联性，也使得微软的项目在影响力的表现上也更加亮眼。

5.3 全域项目下的协作连通性与孤岛

由于使用图网络形式构建协作关系，网络连通性是基本的分析手段之一，而对于 GitHub 项目协作网络的分析为我们带来了诸多新的视角与洞见。

在全域 105 万个开源项目中，根据连通性分析，共分成了 44,990 个连通子图，其中 935,231 个项目在协作关系下构成一个巨大连通子图，其他 44,989 个连通子图中最大的子图包含 200 个项目，最小的包含 1 个项目。包含 100 个项目以上的连通子图仅 9 个。

即全域活跃项目中 89% 的项目构成了一个巨大的协作网络，也就是

GitHub 开源世界的核心。而另外还有将近 4.5 万个小的协作孤岛，这些小的协作网络与开源世界不连通，意味着这些项目上的所有开发者都仅在自己的项目群中协作，从未在其他项目上有过活跃行为，同时在开源核心中的任意开发者也都没有在这个项目群中产生过协作行为。这是一个非常强的条件，因此对于这些数量众多的小型封闭协作世界的探索，可以为我们带来诸多有趣的信息。其中 9 个包含 100 个项目以上的项目群落的数据如表 3.8 所示。

表 3.8 GitHub 2020 全域协作孤岛

#	项目	项目数量	影响力
1	devfactory-dev/*	200	自动化测试
2	labagithub2020-andrew3/*	187	自动化测试
3	ICS3UI-2020-Q1/*	169	教学
4	natewachter/astr-119-hw-5, ThuraDwe/astr-119	167	教学
5	MIEE-ACS/*	158	教学(俄语)
6	bertikq/pibd21-NemovAL, SpokyOky/TimashevISebd21	147	俄语
7	PowerShellForGitHubTeam/*	147	自动化测试
8	team-grilled-cheese/back-end, Kayla-SA-W/NannyHelper-Client	132	学习小组
9	mednajjar/DevSpace, terbouchi/repo	120	法语

可以看到在 2020 年的协作孤岛上，有三个项目群是用于自动化测试的，这些组织或账号下的活跃项目数量众多，大部分是自动创建、测试和删除，所以在其主页上并看不到这么多项目。

还有一些用于教学目的的协作小组，例如 ICS3UI-2020-Q1 和 MIEE-ACS 两个组织下的项目均是通过 GitHub Classroom 创建的，其中 MIEE-ACS 是来自俄罗斯的项目群。而排在第四的项目群，看似来自不同的个人开发者，事实上通过调研可以发现这是来 UC Santa Cruz (加利福尼亚大学圣克鲁兹分校) 的学生们的学习群落，老师/助教 Adriantsh 通过 GitHub 布置作业，同学们则通过 fork 老师/助教的仓库并二次开发来进行学习。

其他的三个项目群则是自发的协作网络群体，也以学习为主要目的，其中有两个项目群的开发者分别来自俄罗斯和法国，有可能是语言问题导致他们没有与开源世界核心的项目产生任何关联。事实上在一些更小的项目群中，我们可以发现来自日本、白俄罗斯、乌克兰、越南等国的协作孤岛。

另外，大量的自动化测试为我们带来了另一个重要的洞见，即在 GitHub 的开放性越来越好，被集成度越来越高的今天，GitHub 作为开源服务平台本身并不开源带来了诸多的问题。其中最重要的一点是与 GitHub 集成的相关应用的自动化测试如果希望进行集成测试，则必须直接使用 GitHub 的线上环境，而无法像 GitLab 一样在沙箱环境中启动一个实例进行完整的集成测试。这不仅导致与 GitHub 集成的工具开发变得困难，而且使 GitHub 线上环境中出现了大量测试仓库，同时这些测试仓库的行为日志数据也被记录下来，成为数据分析中的噪音，使得对 GitHub 数据的分析也变得愈加困难。虽然实验室的同学早在 2019 年已经向 GitHub 社区提出了该问题，但到目前还没有很好的解决。

四、案例分析

1、案例分析方法说明

1.1 领域（案例）分析开发者时区分布计算方法

同 GitHub 全域分析中的开发者时区分布计算方法类似，在筛选出 GitHub 全域开发者活跃度前 5 万名开发者的基础上，进一步筛选出案例（领域）项目所参与的开发者，并判断开发者所属的时区，最后，估计出案例（领域）项目在各时区的开发者人数占比情况。

注：开发者时区判断方法参见上文 3.4 说明1

1.2 开源象限分析方法

本报告提出一种开源象限（OpenQuadrant）的方法来刻画一个开源项目在影响力、全球化、社区规模三个核心特性方面的表现。基于该开源象限分析，使用散点图来表示，横纵两个维度为项目影响力指标和项目全球化指标，为了方便可视化，我们采用取对数的形式呈现上述两个指标，而使用散点图上的点的大小来刻画项目参与的活跃人数，用来反映一个项目的社区规模。

基于以上，开源象限将整个平面分成了四块区域，分别是：

- **前瞻** (foresighted)：项目影响力强、项目全球化程度高；
- **引领** (leading)：项目影响力强、项目全球化程度较低；
- **行动** (acting)：项目影响力较弱、项目全球化程度高；
- **进入** (incubating)：项目影响力较弱、项目全球化程度较低。

需要说明的是，这种分类方法并不唯一，也还处于一个初步的阶段，主要还是希望通过一个可视化的方式，让大家能够从一个更加细分与丰富的角度去观察与认知开源技术的发展。

影响力、全球化、参与人数的具体计算方式如下所示。

1.2.1 领域项目影响力指标

领域项目的影响力计算结果是由全域项目的影响力计算结果给出，计算方法参见上文 3.6 说明 1。

1.2.2 领域项目全球化指标

全球化的影响因素众多，目前我们考虑了地域、开发者人数这两个因素用于项目全球化指标的计算。针对地域因素，我们考虑计算参与该项目的开发者的时区分布情况，开发者时区分布趋于均匀分布时，即标准差越小，认为该项目的全球化程度越高。因此，我们首先通过判断项目上所有开发者的时区，之后计算了项目在 24 个时区上对应的开发者人数，在此基础上，计算项目关于时区人数的标准差。针对开发者人数因素，即该项目上参与的开发者总数，开发者人数越多，该项目全球化程度越高。考虑到开发者人数不多，但开发者的

行为具有偶发特性的情况，该情况导致地域因素计算不准确，从而导致全球化指标的结果不准确。因此，我们考虑如下全球化指标计算公式：

$$\sqrt{\frac{24}{\sum_{i=12}^{11}(x_i - \bar{x})^2}} \bar{x}^2, \quad x_i \text{ 表示该项目在 } i \text{ 时区的开发人数。}$$

上述计算公式表明，全球化指标与项目时区开发者人数均值的平方成正比，与项目时区开发者人数的标准差成反比。该计算方式能够有效评估开发者人数较少的项目的全球化程度，另一方面，该计算方式对开发者人数较多的项目，全球化指标计算有正向作用。

注：全球化的影响因素有许多，上述计算方式考虑的因素有限，之后需要考虑更多的因素来评价项目全球化指标。

1.2.3 领域项目参与人数

项目参与人数，即在项目上发生日志行为的参与者总数。在可视化散点图点的处理上，我们对领域项目的参与人数使用线性最大最小归一化，使得数值映射到了1-10区间范围内。

2、CNCF 基金会项目分析

2.1 CNCF 基金会简介

CNCF 的英文全称是 Cloud Native Computing Foundation，即“云原生计算基金会”。其拥有全球基础架构的关键组件，并且基金会内部汇集了全球顶级的开发者、终端用户和供应商，除此之外，CNCF 基金会运营着全球规模最大的开源开发者会议。“云原生基金会”是 Linux 基金会旗下的基金会，它是一个非盈利组织。“云原生基金会”的口号是“坚持和整合开源技术来编排容器作为微服务架构的一部分”，其致力于云原生应用的普及和推广，也专注于 GitHub 上快速成长的开源技术的推广，对开源生态系统的健康发展作出了突出的贡献。

目前，已经从 CNCF 即“云原生计算基金会”毕业的项目有 containerd、CoreDNS、etcd、Fluentd、Harbor、Helm、Kubernetes、Prometheus、Vitess、Envoy、Jaeger、Open Policy Agent、Rook、TiKV、TUF 等。

目前在 CNCF 基金会中正在进行孵化的项目有 Argo、CloudEvents、CNI、KubeEdge、Operator Framework、Thanos、SPIRE、SPIFFE、OpenTracing、Notary、NATS、Linkerd、gRPC、Falco、Dragonly、CRI-O、Cortex、Contour、Buildpacks 等。

2.2 开发者时区分布分析

图 4.1 为 CNCF 下云原生领域开发者时区分布图。从图中可以看到该领域的开发者时区分布与全域项目的开发者时区分布较为接近，说明 CNCF 云原生领域的项目全球化程度较高。

另一方面，UTC+7 至 UTC+8 时区的开发者占比 11% 左右，相对全域项目的占比而言有所提升，说明 CNCF 领域项目与 GitHub 全域项目相比，中国开发者参与度更高。

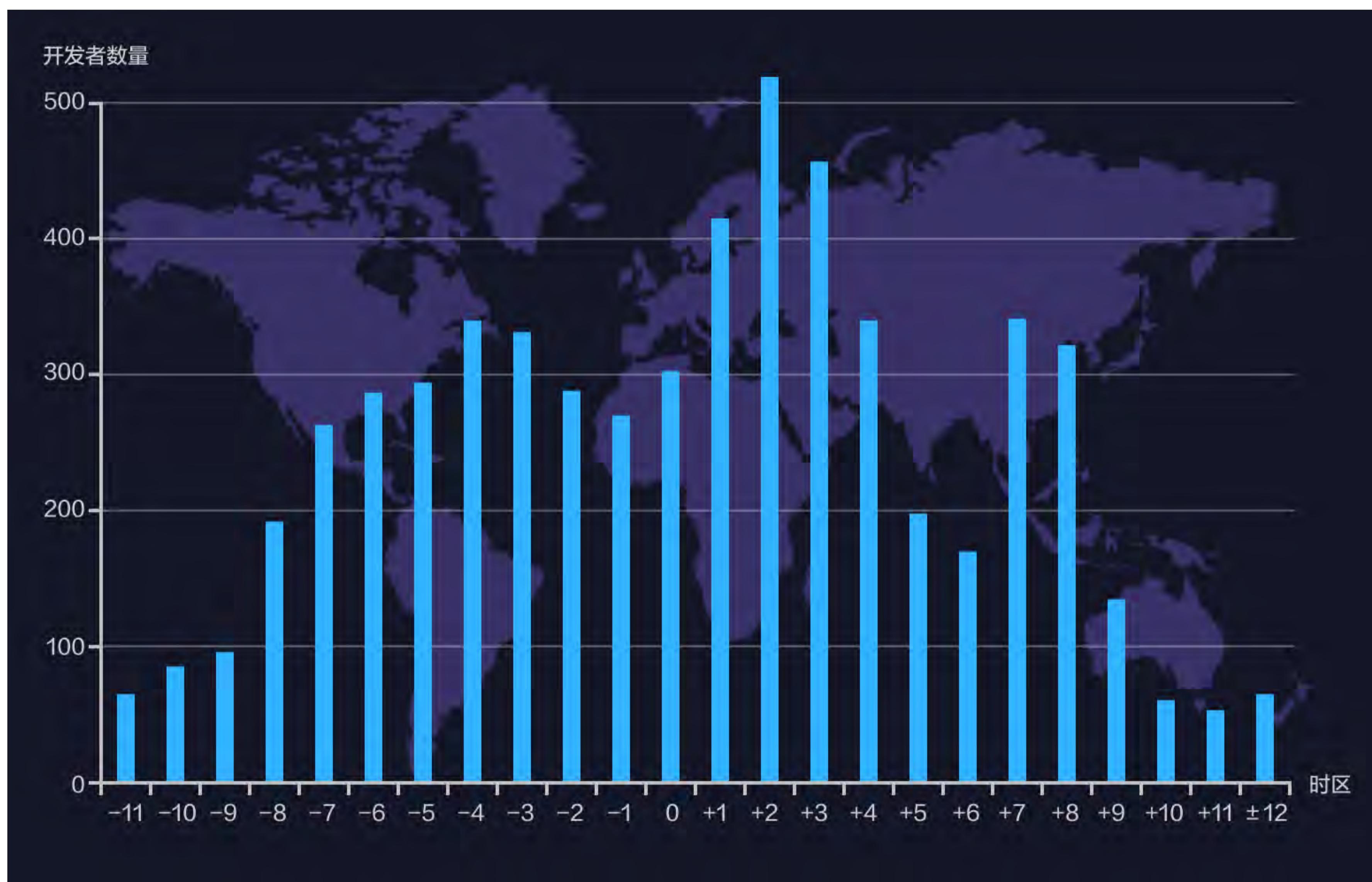


图 4.1 CNCF 下云原生领域开发者时区分布

2.3 开源象限分析

图 4.2 为 CNCF 下云原生领域的开源象限分析可视化结果。Kubernetes 项目无论是在影响力、全球化、还是社区开发者体量上面，当之无愧的处于第一的位置。



图 4.2 CNCF 下云原生领域的开源象限

3、Linux Foundation AI & Data 基金会项目分析

3.1 LF AI & Data 基金会简介

LF AI & Data 是 Linux Foundation 下的一个综合型基金会，支持人工智能、机器学习、深度学习和数据方面的开源创新。创建 LF AI & Data 的目的是支持开源人工智能、机器学习、深度学习和数据，并创建一个可持续的开源人工智能生态系统，使得能够使用开源技术轻松地创建人工智能和数据产品与服务。除了一些扶持性服务，包括成员资格和资金管理、生态系统发展、法律支持、公关/营销/沟通、活动支持和合规扫描，它还为多样化和蓬勃发展的社区中的开放发展项目提供支持。

目前，已经从 LF AI & Data 基金会毕业的项目有 [Acumos](#)、[Angel-ML](#)、[Egeria](#)、[Horovod](#)、[ONNX](#)。

正在孵化的项目有 [Adlik](#)、[Adversarial Robustness Toolkit](#)、[AI Explainability 360 Toolkit](#)、[AI Fairness 360 Toolkit](#)、[Amundsen](#)、[DataPractices](#)、[DELTA](#)、[Elastic Deep Learning \(EDL\)](#)、[Feast](#)、[ForestFlow](#)、[JanusGraph](#)、[Ludwig](#)、[Marquez](#)、[Milvus](#)、[NNStreamer](#)、[OpenDS4All](#)、[Pyro](#)、[SOAJS](#)、[sparklyr](#)。

3.2 开发者时区分布分析

图 4.3 为 LF AI & Data 下数据与人工智能领域开发者时区分布图。从图中可以看到该领域的项目的开发者时区分布相比全域项目有了较大的变化，UTC+7 至 UTC+8 的开发者占比 16% 左右，该领域下中国开发者明显参与更多，从一个侧面也反映了中国有较多的数据与人工智能从业者。

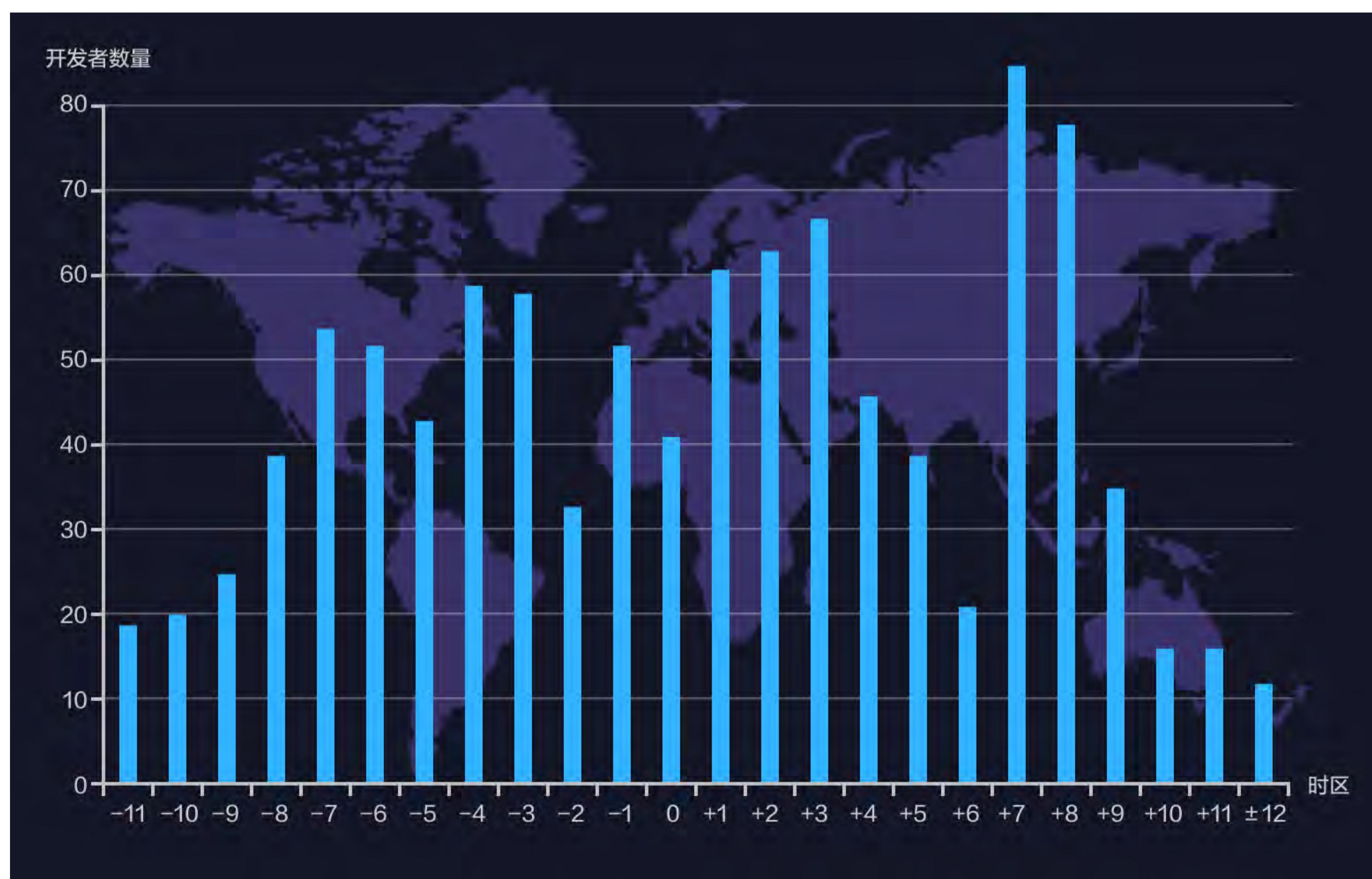


图 4.3 LF AI & Data 下数据与人工智能领域开发者时区分布

3.3 开源象限分析

图 4.4 为 LF AI & Data 下数据与人工智能领域的开源象限分析可视化结果。可以看到该领域有不少全球化做得比较好的项目，非常符合人工智能全球走热的趋势。



图 4.4 LF AI & Data 下数据与人工智能领域的开源象限

4、Apache 软件基金会大数据领域项目分析

4.1 Apache 软件基金会简介

Apache 软件基金会 (ASF) 成立于 1999 年，是一个依据 501(c) 在美国成立的非营利性公共慈善组织。基金会的使命是为公共利益而提供软件。基金会帮助独立个体和组织去理解开源是如何在一个激烈竞争的市场中发挥优势的。其重点不是生产软件，而是指导生产软件的社区。通过被称为“Apache 之道”的精英管理流程，超过 800 个个人会员和 7,000 个提交者成功合作开发了免费的企业级软件，使全球数百万用户受益。

目前 Apache 下的类别标签为“big-data”的开源项目有 [Accumulo](#)、[Lens \(in the Attic\)](#)、[Airavata](#)、[Ambari](#)、[Oozie](#)、[Bigtop](#)、[Zeppelin](#)、[Flink](#)、[Flume](#)、[Spark](#)、[Sqoop](#)、[Storm](#) 等多个大数据相关的项目。

4.2 开发者时区分布分析

Apache 下大数据领域的项目开发者时区分布如图 4.7 所示。从图中可以看到，Apache 下大数据领域项目的开发者时区分布出现了两个明显的人数占比高峰，UTC+2 至 UTC+3 时区占比 16% 左右，以及 UTC+7 至 UTC+8 时区占比 15% 左右。而在 UTC-8 至 UTC-3 时区，开发者占比 26% 左右。虽然 Apache 大数据领域中国开发者参与较多，但是该领域项目仍是以欧美开发者为主导。

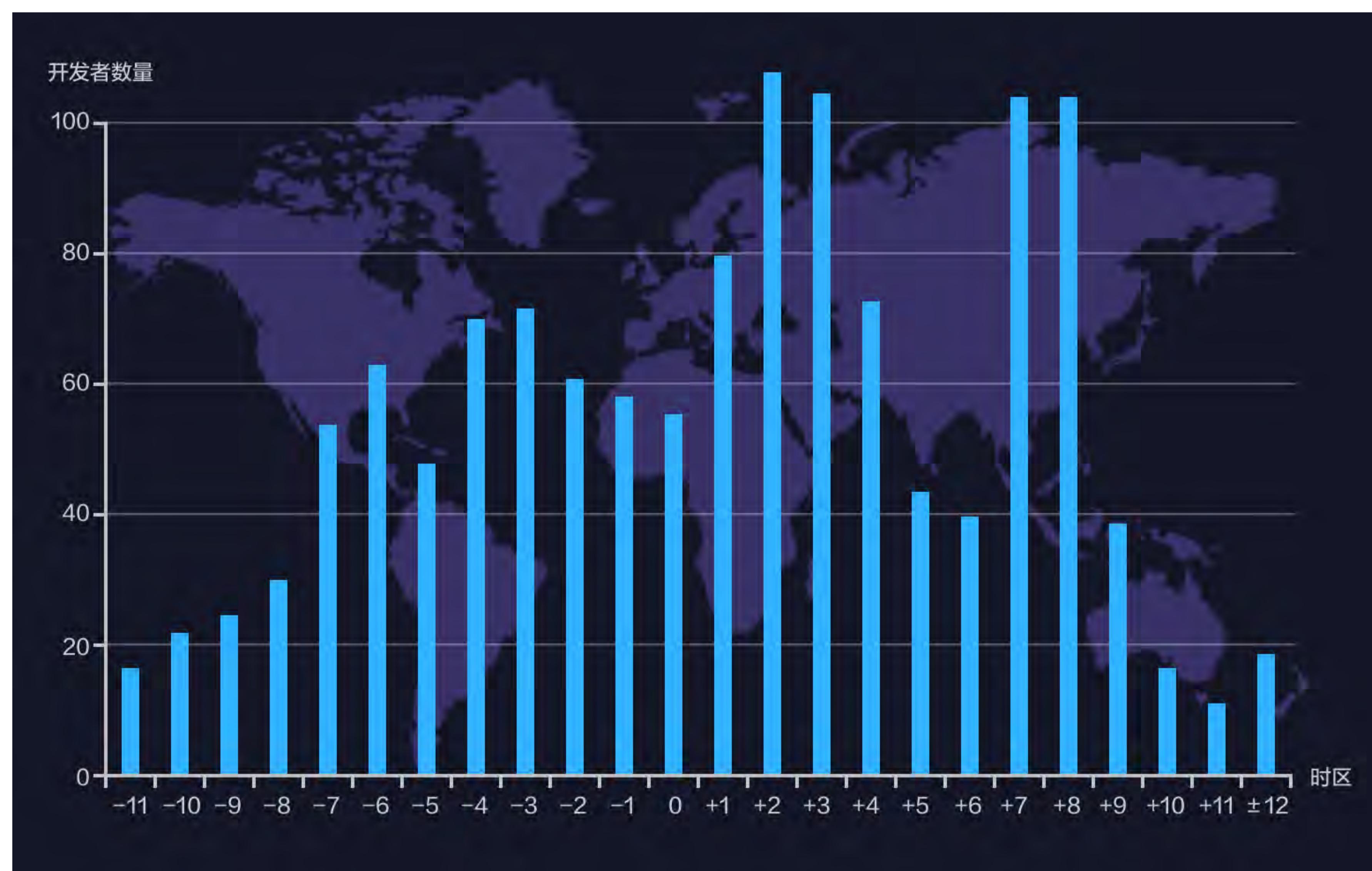


图 4.5 Apache 下大数据领域开发者时区分布

4.3 开源象限分析

图 4.6 为 Apache 下大数据领域的的开源象限分析可视化结果。可以看到该领域的项目整体分布具有较好的分散性，Spark、Flink、Hadoop 等明星项目占据着头牌位置。



图 4.6 Apache 下大数据领域的开源象限

5、Apache 软件基金会中国项目

5.1 简介

Apache 软件基金会作为世界上最活跃的基金会之一，其先进的社区治理方式和和谐的社区氛围吸引着越来越多的中国开源项目加入其中。目前，Apache 软件基金会源自中国地区的开源项目共有 21 个，包括 14 个顶级项目和 7 个正在孵化中的项目，这些项目涵盖了大数据、物联网及云管理等技术领域。

14 个源自中国地区的 Apache 软件基金会顶级项目包括：[Kylin](#)、[HAWQ](#)、[Eagle](#)、[CarbonData](#)、[RocketMQ](#)、[Griffin](#)、[ServiceComb](#)、[Skywalking](#)、[Dubbo](#)、[ShardingSphere](#)、[IoTDB](#)、[APISIX](#)、[Ozone](#)、[ECharts](#)。

7 个正在孵化中的项目包括：[Weex](#)、[Doris](#)、[brpc](#)、[Teclave](#)、[DolphinScheduler](#)、[TubeMQ](#)、[Pegasus](#)。

5.2 开发者时区分布

Apache 下中国项目开发者时区分布如图 4.5 所示。从图中可以看到，Apache 下中国项目明显由中国开发者所主导，UTC+7 至 UTC+8 的时区开发者人数占比 32% 左右，而欧美开发者的占比相对较小。

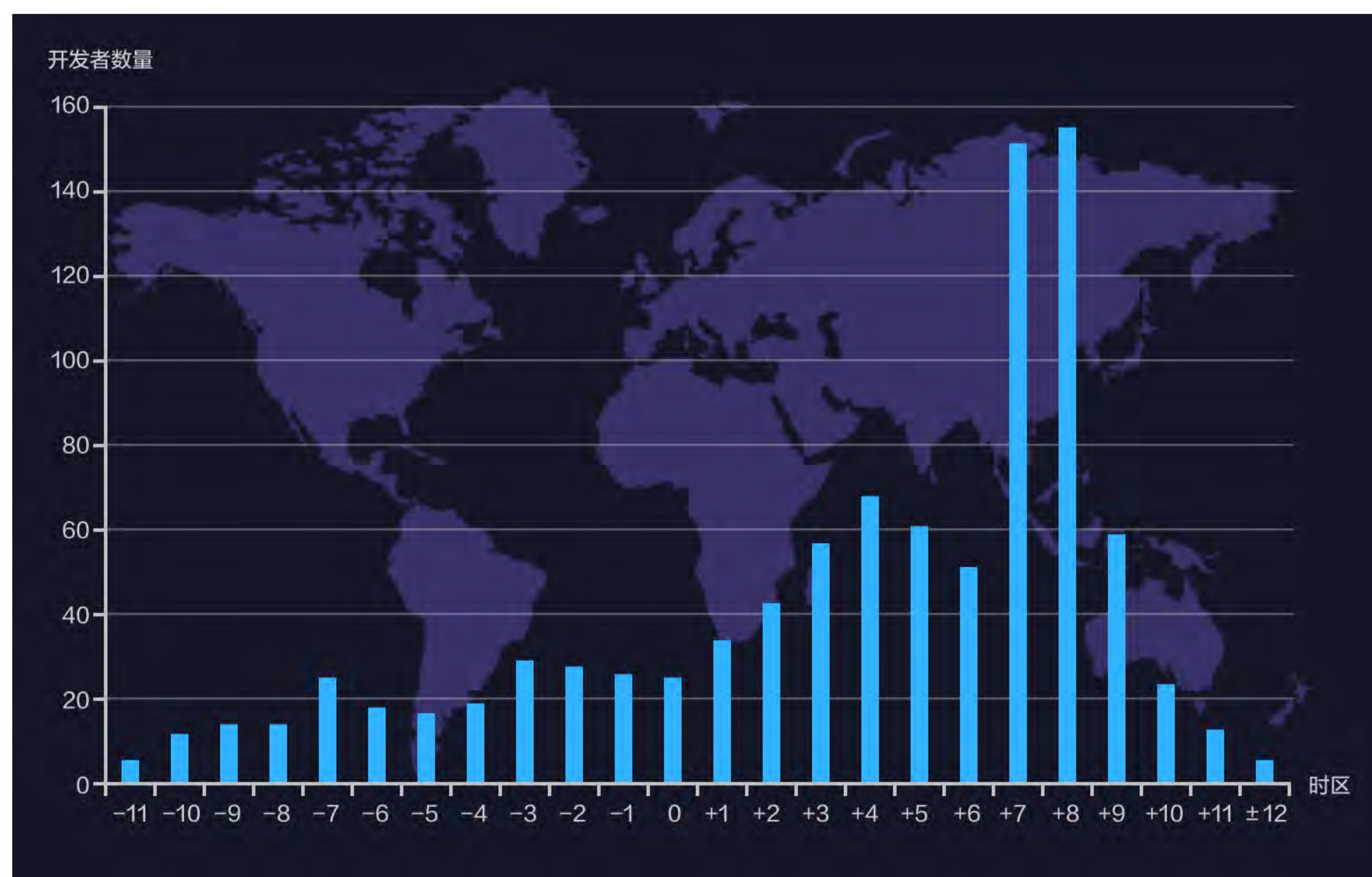


图 4.7 Apache 软件基金会中国项目开发者时区分布

5.3 开源象限分析

图 4.8 为 Apache 软件基金会下中国项目的开源象限分析可视化结果。可以看到该领域的 Echarts、Skywalking、Dubbo 和 Shardingsphere 四个项目处于第一方阵，体现了较强的实力。



图 4.8 Apache 软件基金会下中国项目的开源象限

6、VSCode 案例分析

6.1 总体情况

作为一款现代化轻量级代码编辑器，VSCode 是微软的一个开源杀手锏。它支持几乎所有主流的开发语言的语法高亮、智能代码补全、自定义热键、括号匹配、代码片段、代码对比 Diff、Git 等特性，而插件机制更加保证了它的兼容性与扩展性，在开源生态链中格外耀眼。

VSCode 项目作为开源生态中的一颗北极星，在 2020 年依旧保持着旺盛的生命力，这一年，共有 **206,645** 条记录由 VSCode 产生，相比 **2019** 年的 **121,490** 条，增长了接近一倍；这一年，VSCode 的项目年平均活跃度分值为 **385**，在全域项目中排名第 **7** 位；这一年，有 **46,639** 位开发者（包含协作机器人账号）在项目中活跃过，在这艘开源航母上发出有力的协作信号。

6.2 开发者时区分布分析

VSCode 项目开发者时区分布如图 4.9 所示。从图中可以看出，该项目的开发者时区分布较为均匀，是一个典型的全球化协作项目。活跃开发者在 UTC+1 至 UTC+3 达到一个高峰，说明该项目仍以欧洲为主导。其次是北美地区，占比约 **22%**，并且以美东地区更为活跃。亚洲时区的活跃度相对较弱一些，但是仍旧以约 **15%** 的占比占据着不容忽视的地位。

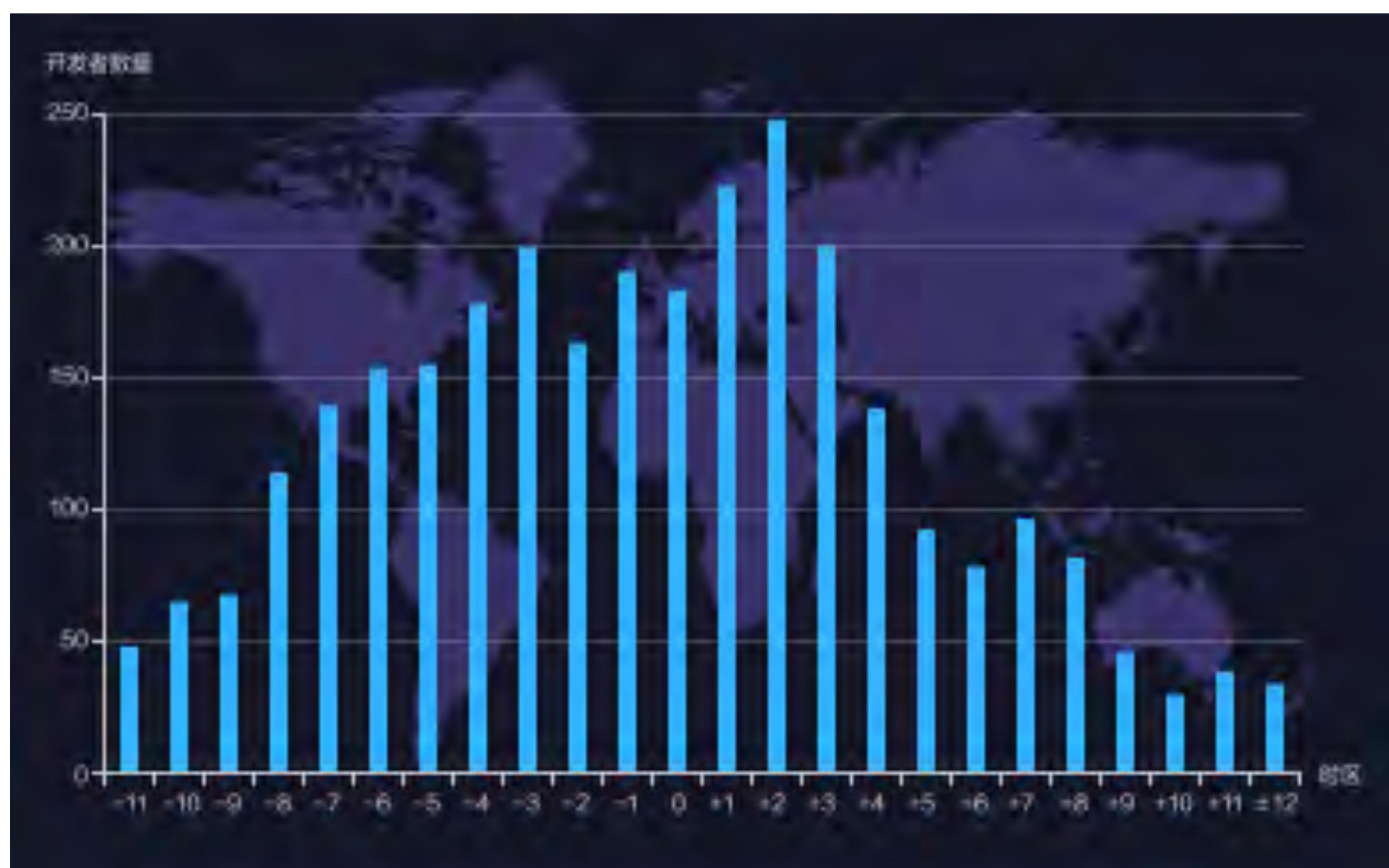


图 4.9 VSCode 项目开发者时区分布

6.3 开发者协作网络分析

同时我们通过开发者在项目中的协作关系构建了 VSCode 在 2020 全年的开发者协作网络，如图 4.10 所示。

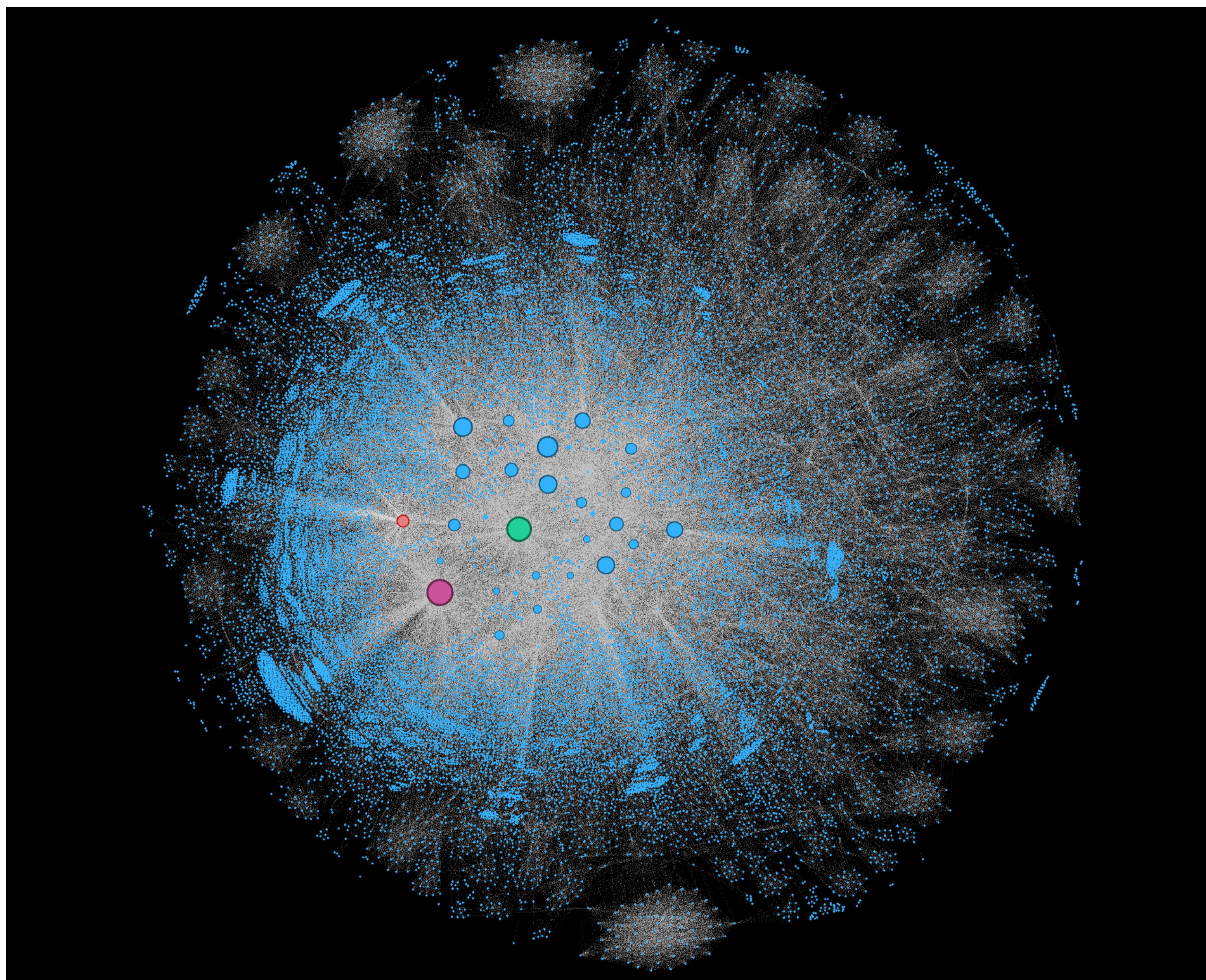


图 4.10 VSCode 项目开发者协作网络

如上图所示，在这个由 2 万多个开发者组成的协作网络中，节点为开发者账号，边为协作关系，节点的大小为对应开发者账号的活跃度。在这个协作网络中，处在网络核心位置的较大的节点是 VSCode 的核心团队成员，他们不仅有很高的活跃度，而且与其他开发者均具有较高的协作关系，这个群体的数量在百人左右。紧接着外侧是 VSCode 的重度使用者或贡献者，他们可能随时提交 Issue 或 PR 进行讨论或贡献，这个群体的数量在千人级别。最外侧，也是最大量的开发者是 VSCode 的一般用户和偶发贡献者，大部分仅在自己关心的问题上提问或讨论。

在这个协作网络的核心位置，我们使用不同的颜色标注出了三个特殊的开发者账号，他们拥有极高的活跃度，他们分别是 vscode-triage-bot, vscodebot[bot] 与 github-actions[bot]。其中 vscode-triage-bot 是专门用于 issue 的生命周期管理，用于提示 feature request 的投票情况、开发进度等，并及时关闭过期的 issue。而 vscodebot[bot] 是 VSCode 团队自主开发的 GitHub Apps，有大量的实验性功能，例如对于新开 issue，自动与历史 Issue 对比内容，推荐相似的 issue；对 issue 进行自动 label 等。github-actions[bot] 是 GitHub 的官方 GitHub Apps 账号，专门用于各种自动化协作任务，从日志中可以看出，VSCode 团队是在 2020 年 3 月底才引入该机器人。目前三个机器人的功能区分并不清晰，但可以确定的是 VSCode 一直在努力使用各种自动化手段进行项目管理，从而处理更大规模的协作。

而在最外侧聚成大簇的开发者则形成了一些协作群落，这些群落是 VSCode 项目中最被关注的一些问题。例如在图正下方的一个巨大群落，这些开发者均在 [issue #52855](#) 上进行过讨论，该问题的内容是 Windows 系统下 VSCode 在更新中关闭操作系统会导致程序彻底丢失，该问题自 2018 年 6 月提出，直到目前依然尚未解决，最新的评论出现在 2021.2.12。而正上方的群落则指向的是 [issue #108447](#)，这个问题是 2020.10.10 提出的，最新的 VSCode 更新导致代码格式化出现问题，短时间内就导致了大量的开发者涌入

讨论，但最终 `github-actions[bot]` 机器人在 2020.12.25 在该 issue 关闭 45 天后触发社区规范锁定该 issue，而在关闭到锁定的这段时间内仍有大量开发者在该 issue 上进行讨论。

由上可见，利用协作关系网络，我们可以很方便的观察到社区中的核心开发者，并通过可视化手段或算法发现社区中的热点，即便对于项目不熟悉的人，也可以通过该工具快速挖掘项目中的深层信息。

说明 4.1：开发者协作网络构建方法

开发者协作网络是基于说明 2.1 中开发者活跃度的定义，通过多个开发者在项目中的协作关系进行构建的。具体构建过程为将每个开发者的全年活跃度计算精细化到具体的 issue/PR 之上，则同时在同一个 issue/PR 上有过活跃的开发者认为其具有协作关系，而协作关系与两人在该 issue/PR 上的活跃度相关，具体计算方式为：

$$R_{ab} = \sum_i \frac{A_{ia} A_{ib}}{A_{ia} + A_{ib}}$$

其中 $A_{ia} A_{ib}$ 分别为开发者 a 和 b 在 issue/PR i 上的活跃度，计算方法遵循说明 2.1 中的活跃度计算方法， R_{ab} 为开发者 a 和 b 在该项目上的协作关联度。即两个开发者在项目的协作关联度为其在所有共同活跃的 issue/PR 上的活跃度的调和平均值之和。该构建方式与 OpenGalaxy 的项目协作网络构建方式类似。

五、每月之星

1、每月之星评判方法说明

除了顶级项目之外，GitHub 上还有一些短期内受到广大开发者大量关注的项目。这些项目可能是现象级项目，也有可能在未来成为顶级项目。这些项目可能与社会热点有关，例如，与新冠疫情相关的项目，与学生毕业求职相关的项目等等。发现这些项目以及解释这些项目在短期内受到大量关注的原因是很有意义的。因此，本部分“每月之星”列举了在 2020 年每个月里受到开发者大量关注的项目。我们根据日志数据，先筛选了每月 star 数排名靠前的项目，之后在每个月里，人工挑选了 1 个比较有特点的项目（当然 GitHub 上每月有特点的项目还有很多），以下是我们挑选出来的项目，以及关于这些项目的一些简要说明。

2、2020 年度每月开源之星

一月：microsoft/playwright

Playwright 是由微软公司 2020 年初发布的新一代自动化测试工具，相较于目前最常用的 Selenium，它仅用一个 API 即可自动执行 Chromium、Firefox、WebKit 等主流浏览器自动化操作。

二月：wuhan2020/wuhan2020

wuhan2020 在 1 - 3 月均在榜单上排名靠前，在新冠疫情的严峻形势下，由各个领域的志愿者们自发组织的开源项目 wuhan2020 应运而生，1 - 3 月由于抗疫队伍、抗疫物资等规模的迅速扩大，急需一个医院、工厂、采购等信息实时同步的数据服务平台，因此该项目以极快的速度成长起来，活跃度非常高。之后由于中国抗疫措施基本完善，现实中疫情也得到了有效控制，虽然现实中的援助还在继续，但需求不再有明显的增加，项目的活跃度开始逐渐消退。

三月：CSSEGISandData/COVID-19

COVID-19 是在 2019 年年底新冠疫情背景下由约翰·霍普金斯大学系统科学与工程中心 (JHU CSSE) 于 2020 年 2 月创建的 Dashboard 项目，在 3 - 4 月时 star 数达到顶峰，与 covid-19 被重视的程度和时间一致，star 数达到顶峰时间（2020 年 3 月）距离项目建立时间接近 2 个月，推测与全球疫情持续关注度、项目所涉及的人力社会资源规模大等因素有关。这是一个非代码类项目，主要记录各国家和组织公布的每日数据档案、并给出数据来源的描述和链接。

四月：labuladong/fucking-algorithm

这是一个基于 leetcode 的思维培养项目，在 3 月份建立，star 数在 4 月、8 月、12 月达到了 3 次顶峰，与开发活跃度的 3 次顶峰时间相比，均落后了 1 个月。在除外开发活跃度达到 3 次顶峰的其他月份中，开发活跃度显著降低。Star 数达到顶峰在 4 月、8 月及 12 月的主要原因是学生升学考试及工作面试。

五月：bradtraversy/design-resources-for-developers

design-resources-for-developers 是一个设计资源索引，整个项目 readme 文件的资源（设计网站地

址) 索引链接占了 90%以上的工作, 24k star, 6k fork, 是由设计开发者自发维护的典型的非代码类项目。一经发起迅速达到 star 顶峰。

六月: [electronicarts/CnC_Remastered_Collection](#)

红色警戒源码! 2020 年 6 月知名游戏公司 EA 开源了《命令与征服》系列中的 2 个游戏, 其中一个便是红色警戒, 吸引了众多开发者关注。

七月: [JaidedAI/EasyOCR](#)

EasyOCR 是一个用 python 编写的 OCR 三方库。支持 80 多种语言的 OCR 识别, 2020 年 6 月底发布 1.1 版本, 在 7 月份, 吸引了广大开发者们的关注。

八月: [geekxh/hello-algorithm](#)

此项目是一套针对小白的完整的算法训练流程, 类似网盘资源分享包和网址导航目录的形式。由于其全免费、求收藏支持原创的主动号召、面向用户广泛 (java 语言学习用户)、覆盖面较广 (算法基础、大厂面试、题典、算法专题应用) 的特性, 有较多的 star 数: 24.1k。在 1-8 月份、12 月份较活跃, 推测与就业有关。

九月: [cli/cli](#)

GitHub 官方命令行工具。2020 年年初 GitHub 开源了自家的命令行工具, 在 9 月正式发布 1.0 版本。该项目目前 star 数有 21.6k, fork 数有 2k, 吸引了众多开发者使用与贡献。

十月: [kamranahmedse/developer-roadmap](#)

2019 年成为 Web 开发人员的路线图。这些路线图可以给出 Web 开发技术的轮廓, 并指导学习的方向, 并可以用来分析为什么某些工具比其他工具更适合用在一些特定情况。readme 排版优秀, 针对人群庞大。目前 star 数 149k, fork 数 21.9k。

十一月: [ytdl-org/youtube-dl](#)

youtube-dl 是一个从 youtube 或其他视频平台上 (比如优酷、爱奇艺、bilibili 等) 下载视频资源的项目, 从 2008 年 7 月至今已经有 12 年多的发展历史了, 随着一些视频网站会修改页面的布局或者代码, 该工程也在持续更新中。

十二月: [beurtschipper/Depix](#)

这是一款 20 年 12 月的一款用匹配方法消除马赛克的 GitHub 开源项目, 上线三天收获近 7,000 star, 有意思的是它只有 3 名贡献者, 3 个分支, 最长分支为 main 分支, 也只是提交了 15 次, 但 fork 数达到了 1.8k, star 数达到了 15k, 有图文并茂简单易懂的使用说明及效果展示。大多数用户再用微信、美图秀秀等工具做马赛克测试, post 文出去表示没有达到展示的效果或是解析他的方法如此简单却能意外的恢复一些信息, 给出不能简单打马赛克的建议。推测此工程因涉及到大多数用户关心的隐私安全的保障问题, 故而被广泛关注。一经发起迅速达到顶峰。

六、总结与展望

《GitHub 2020 数字洞察报告》作为一个数据驱动的可视化工具，主要为大家提供一个新的视角来观察今天的开源世界，进而结合各自的行业经验获得洞见。从本次年报开始，我们将这个作品也作为一个开源项目进行运营，逐渐缩短发布的周期，甚至按需提供个性化的按需服务。

如若发现数据错误或遗漏，欢迎提交 Issue 或 PR 到 GitHub。本报告文本部分采用 CC-BY-4.0 许可协议，项目地址为：<https://github.com/X-lab2017/github-analysis-report>。

七、致谢

《GitHub 2020 数字洞察报告》是由 X-lab 开放实验室发起，由“源光闪烁”开源科技媒体策划，联合了华东师范大学数据科学与工程学院、开源社、上海开源信息技术协会、开源社会工程研究院（筹）等多家科研机构与开源社区所共同完成。



本次数字洞察报告的主要贡献者包括：赵生宇、王伟、周添一、翁振杰、王皓月、夏小雅、朱香宁、杨鸣、宁泽欣、林海铭、王福政、史经犇、娄泽华、顾业鸣、李思颖等，特别感谢吴雪（雪哥）和 Kate（杨佳）作为项目顾问，为数字洞察报告提供指导与建议。我们欢迎更多的开源爱好者加入我们，共同推动开源在世界的发展。



GitHub 2020 数字洞察报告

在线报告

源光闪烁

