Paper presentation: An in-depth study of the promises and perils of mining GitHub

Tyler Brown

What are the promises and perils of mining GitHub for software engineering research?

What can this paper do for you?

- Identifies potential threats to validity which can undermine a statistical analysis using GitHub data
- Highlights general observations about project participation and user behavior which may help you formulate a research question for your project

Summary of the paper

- Provides 13 perils to avoid when analyzing GitHub repositories
- Reviews Mining Software Repositories (MSR) 2014 Papers for susceptibility to these perils
- Peril discovery process:
  1. *Quantitative analysis of project metadata.* Used GHTorrent dataset.
  2. *Manual analysis of a 434-project sample.* Random manual sample of 434 projects from the 3 million projects in the GHTorrent dataset.

Repositories are Part of Projects

- **Peril I:** A repository is not necessarily a project
  - *Avoid:* To analyze a project hosted on GitHub, consider the activity in both the base repository and all associated forked repositories
- **Peril II:** Most projects have low activity
  - *Avoid:* Consider the number of recent commits on a project to select projects with an appropriate activity level
- **Peril III:** Most projects are inactive
  - *Avoid:* Consider the number of recent commits and pull requests

On the Contents of Projects

- **Peril IV:** Many projects are not software development
  - *Avoid:* Do not rely just on the types of files within the repositories, also review descriptions and README files to ensure the projects fit the research needs

On the Users Involved with Projects

- **Peril V:** Most projects are personal
  - *Avoid:* Consider the number of committers

On the use of Non-Github Infrastructure

- **Peril VI:** Many active projects do not use GitHub exclusively
  - *Avoid:* Stay away from projects that have a high number of committers who are not registered GitHub users and projects with descriptions that explicitly state they are mirrors.

On Pull Requests

- **Peril VII:** Few projects use pull requests
  - *Avoid:* When researching the code review process on GitHub, consider the number of pull requests before selecting a project.
- **Peril VIII:** Merges only track successful code
  - *Avoid:* To analyze the full set of commits involved in a code review, do not rely on the commits reported by GitHub.
- **Peril IX:** Many merged pull requests appear as non-merged
  - *Avoid:* Do not rely on GitHub's merge status, but consider using heuristics to improve merge detection when analyzing merged pull requests

On Users

- **Peril X:** Not all activity is due to registered users
  - *Avoid:* For empirical studies that need to map activity to specific users, use heuristics for email unification to improve the validity of the results
- **Peril XI:** Only the user's public activity is visibile
  - *Avoid:* This peril is unavoidable when using data from public websites–acknowledging this partial view in the discussion of results and replicating a study in other contexts can help reduce its impact
  - *Not entirely true:* User's "Contributions calendar" will show the number of public and private contributions by default (github-help/viewing-contributions-on-your-profile/)[https://help.github.com/articles/viewing-contributions-on-your-profile/]

GitHub is an Evolving Entity

- **Peril XII:** GitHub's API does not expose all data
  - *Avoid:* Obtaining data from one of the services which archives the data from the GitHub API (like `GHTORRENT`) can help avoid this peril but comes with assumptions about how the data is collected.
- **Peril VIII:** GitHub is continuously evolving
  - *Avoid:* Understand how both the Github `API` and the website have evolved over time. Changes to the website are often posted to the GitHub blog, but this is not guaranteed

Relationship Between Perils

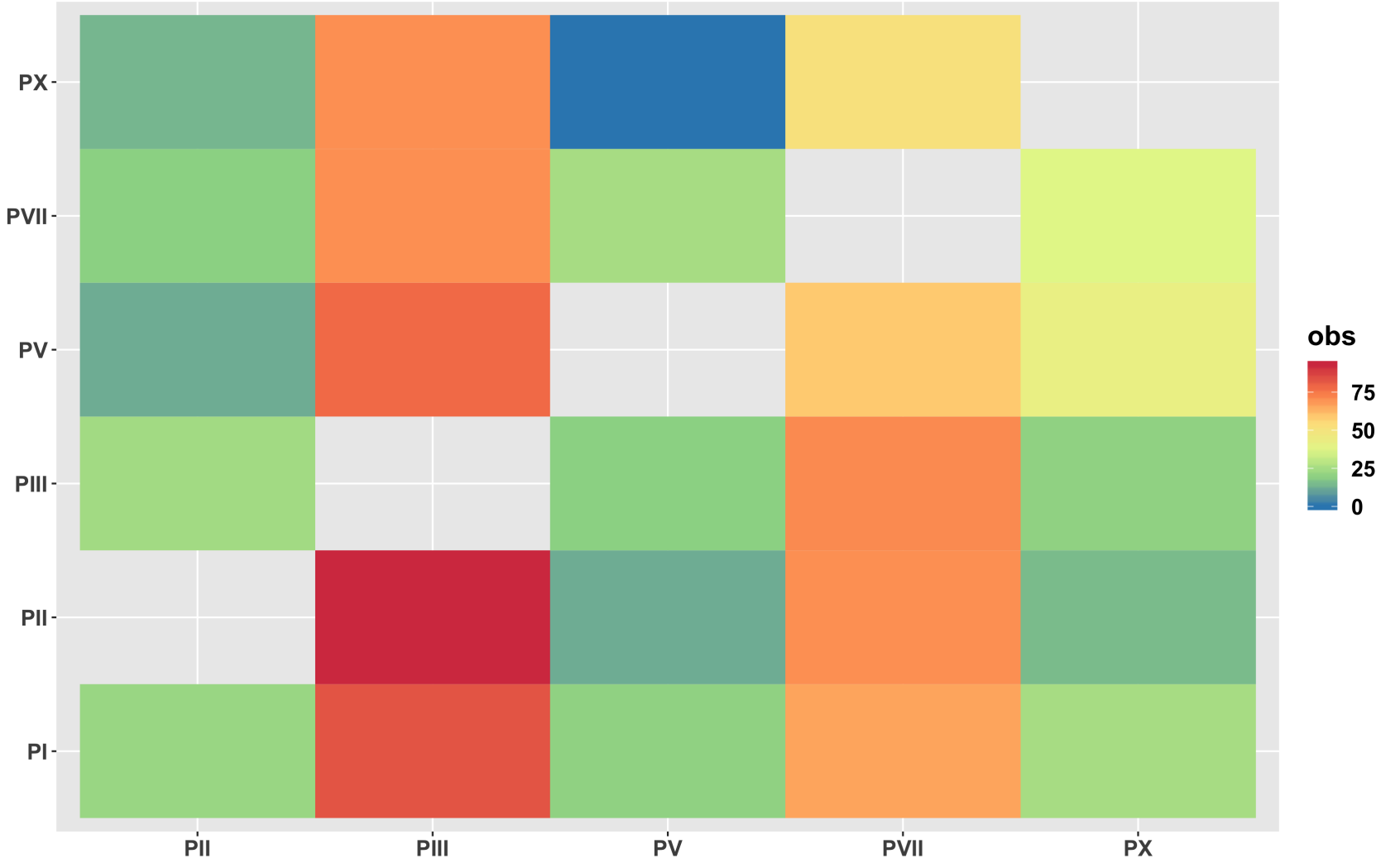

Figure 1: (Table 4) GitHub Project Peril Matrix by Percentage

An Analysis of the MSR 2014 Mining Challenge

- Researchers were given a subset of the GHTorrent dataset to analyze and derive new insights
- Competition resulted in 9 accepted papers; these papers are their dataset

MSR 2014 Papers

- **Peril XII:** GitHub's API does not expose all data
  - Do watchers become contributors? (Sheoran et. al. 2014)
    - GHTorrent may not recognize the dates that Watchers become Watchers, just the date GHTorrent noticed.
- **Peril IX:** Many merged pull requests appear as non-merged
  - What makes a pull request successful? (Rahman and Roy 2014)
    - The analysis considered merged pull requests as successful, while marked-as-non-merged as unsuccessful
  - How often are commits merged into the master branch? (Padhye et al. 2014)
    - The number of recognized committers can be effected by whether a pull request was marked as merged

Discussion

- What project ideas were you considering and how might lessons from this paper impact those ideas?

Questions/Comments?

- Thank you for your time!