Software Analitics, Bug Triaging

Joy Albertini, Jacob Salvi

Data collection

Issues collections

Initially we tried to collect the issues using the 'pygithub' library.
We set up the 'github' object, `Github(auth=auth, per_page=100)`, to fetch 100 issues per page and then tried to get the issues by iterating over the 'PaginatedList' returned by the `repo.get_issues(state="closed")`.
This method revealed to be quite slow since a much larger number of requests than the expected one were performed. With 220000 issue and 100 issues per page a mere 2200 requests should have sufficed, avoiding the 5000 request per hour limit github has in place.
Instead a much larger number of requests was performed and we became subject to the timeout put in place by github.
To solve this problems we have completely ditched the 'pygithub' library, opting to make the request manually.

```
base_url = "https://api.github.com/repos/microsoft/vscode"
url = f"{base_url}/issues?state=closed&page={current_page}&per_page=100&direction=asc"
response = requests.request("GET", url, headers=headers, data=payload)
```

By fetching the issues in ascending order we should get issues with issues number increasing from number 1 onward.
That said, it seems that the github api itself has a small bug. Page 1921 had a out of place issues which caused has to trigger the termination of our fetching loop prematurely. This page contained issues numbered [205001, 205002, 205003, 230000, ...], the obvious outlier was causing issues.
We saved all of the issues data as received in a json file to be used for the preprocessing. Having all the data allowed us to study which part of the data was useful without having to fetch it again.

Pre processing

The data had to be pre processed before being given to the model for training. Of all the information we retrieved in the previous step we decided to keep only the title, body, id, number, url, assignee and the labels.

The title and body required the most preprocessing given that they contain the bulk of the text. Given that the bodies of the issues are written in markdown we have used a markdown library to parse them.
Some elements, such as html tags, links and images have been removed during the parsing. Code blocks have been kept exactly as is.
The main parts that have been modified then are, headings, paragraphs and lists.
The text has been lower-cased, stemmed and it had stop-words removed.
Other elements, such as emojis, html tags, url tags and email tags have also been removed to decrease the noise in the data.

Predictor

Results

All issues

Recent issues