OSSFinder

Project Summary

Yuxing Ma, Alex Klibisz, Muhammed Benkhayal

Contents

Abstract

- Problems
  - Millions of open-source software projects.
  - Discover projects that match requirements.
  - Discover projects that offer minimal learning curve.
- Our Solution
  - ***Build an open source software recommendation engine based on user-generated data.***

Abstract

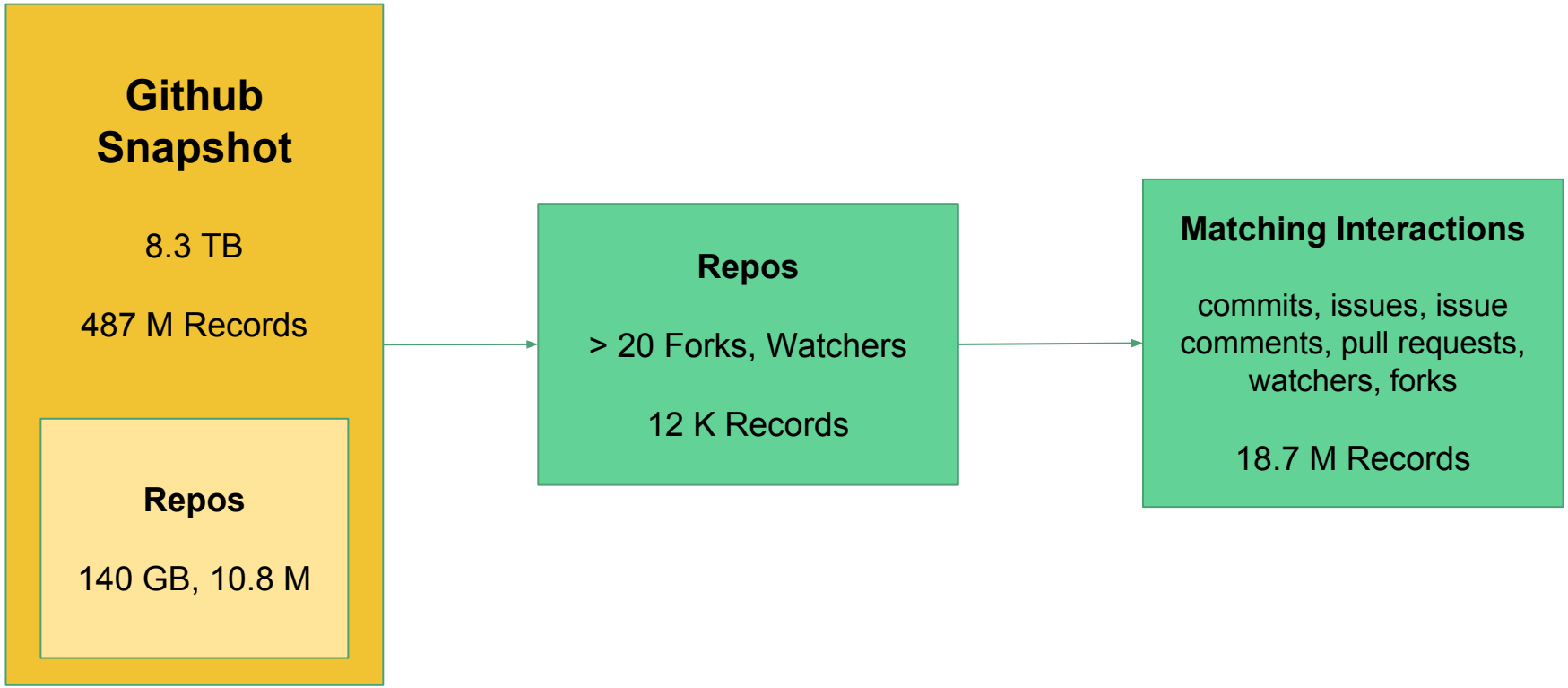
1. Feature Search
    a. Repo READMEs searched for features
2. Repo Relationships
    a. Users interact with repos in many ways
    b. Interactions generate relationships
3. Recommendations
    a. Combining search and relationships creates recommendations

Search + Relationships = Recommendations

Implementation: Tooling

- Python
- Pymongo
- MongoDB
- Javascript
- Html

Implementation: Data Collection

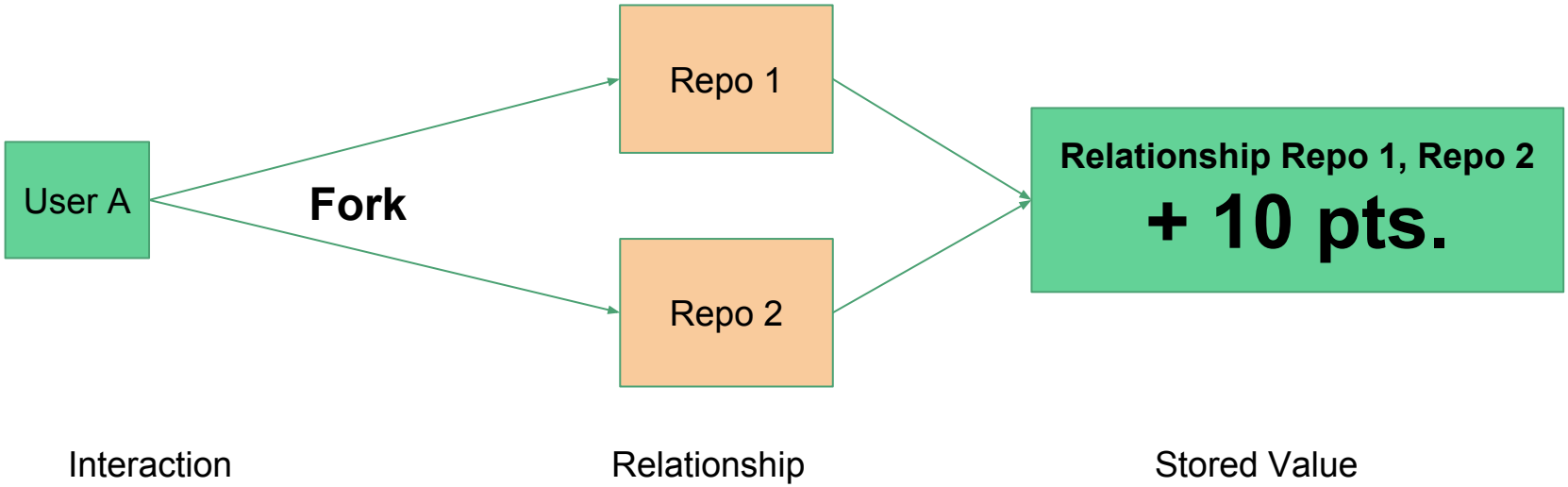


Implementation: Feature Search

- Using READMEs, build a feature search engine
- Simserver
  - converts READMEs to semantic representation
  - use TF-IDF + LSI to train and create index
  - given query, returns ids of the most similar documents
- In other words
  - *feature query => [repo_0, repo_1, ... repo_n]*

Implementation: Relationship Calculations

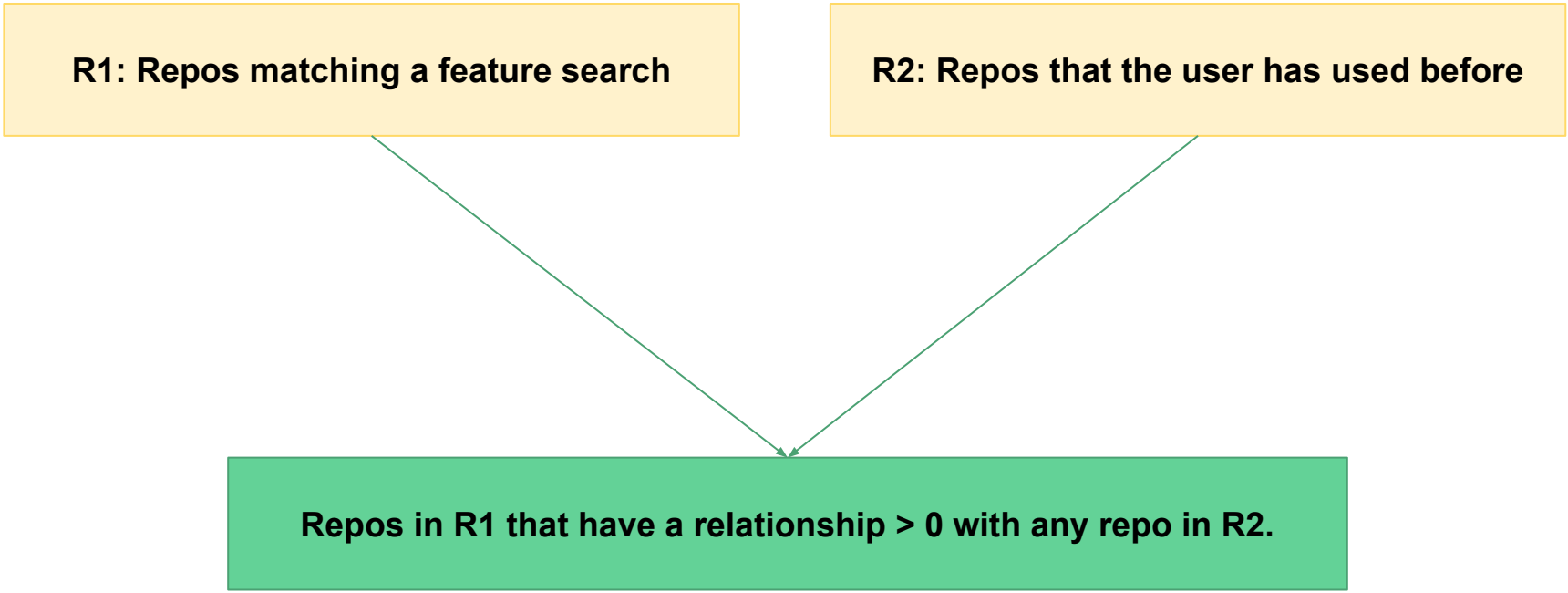
- 1 user, 2 repos = interaction
- Stored separately, then aggregated
- Multipliers weight relationships



Implementation: Relationship Calculations

Relation	Approx. Amount	Point Value
Issue Comments	6.5 M	1
Watchers	6 M	1
Issues	3 M	2
Forks	2.5 M	2
Pulls	2 M	3
Commits	500 K	13

Implementation: Recommendations



Challenges

- Search Implementation
  - readme retrieve
  - repo filter
- Quantity of Data
  - Serial vs. Threaded Filtering, Transferring
  - Ended up with 30GB data
- New material
  - Pymongo
  - SimServer
  - Flask

Demo!

The feature I need:

Mapping

I’m familiar with:

- easyXDM: cross-site JS messaging
- etsy/statsd - node.js stats aggregation
- punjab - real-time browser connections
- morris.js - time series line graphs
- angular-google-maps