Social Media Data Science Pipelines Project 1: 4chan and Reddit Data Collection System

Devang Jagdale
djagdale@binghamton.edu
Binghamton University
Binghamton, New York, USA

Tejas Hiremath
thiremath@binghamton.edu
Binghamton University
Binghamton, New York, USA

Chaitanya Jha
cjha@binghamton.edu
Binghamton University
Binghamton, New York, USA

ACM Reference Format:
Devang Jagdale, Tejas Hiremath, and Chaitanya Jha. 2024. Social Media Data Science Pipelines Project 1: 4chan and Reddit Data Collection System. In . ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

This project implements a continuous data collection system that gathers data from two social media platforms: Reddit and 4chan. The collected data is intended to support future data science projects, focusing on sentiment analysis, hate speech detection, and the analysis of online discourse. The system fetches data dynamically from selected subreddits and 4chan boards, processes it, and stores it in a TimescaleDB-powered PostgreSQL database for efficient time-series analysis.

2 DATA SOURCES

2.1 Reddit API

The Reddit API provides structured access to its extensive repository of user-generated content. We use the following endpoint to target relevant subreddits and collect comments from posts:

Comments API: https://oauth.reddit.com/r/{subreddit}/comments retrieves comments from posts within a specific subreddit.

The subreddits list is configurable, allowing the dynamic addition or removal of subreddits based on research needs.

2.2 4chan API

4chan does not have an official API, so data is collected via scraping from specific boards such as `/pol/`. The 4chan API is used for retrieving thread information and metadata.

Catalog API: http://a.4cdn.org/pol/catalog.json retrieves metadata from all active threads on `/pol/`.
Thread API: http://a.4cdn.org/pol/thread/{thread_number}.json retrieves all posts from a specific thread.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3 DATA COLLECTION WORKFLOW

3.1 Crawler Classes

We implement two custom Python clients for Reddit and 4chan, encapsulating their respective API interaction logic:

- `RedditClient`: Handles Reddit API calls to fetch comments from subreddits.
- `ChanClient`: Scrapes 4chan boards and retrieves threads and posts from `/pol/`.

3.2 Process Flow

The data collection workflow follows these steps:

(1) **Fetch Latest Comments/Threads**: The system queries APIs and collects new comments or threads.
(2) **Detect Inactive Threads**: It checks for threads or posts no longer receiving new comments.
(3) **Data Storage**: Collected data is preprocessed, removing irrelevant parts, and stored in a TimescaleDB-powered PostgreSQL database.

4 SYSTEM ARCHITECTURE

The system architecture which is Figure1 of the Reddit crawler consists of several key components that work together to fetch, process, and store subreddit data. The primary components are as follows:

4.1 Scheduler

The scheduler will trigger data collection at regular intervals, ensuring continuous data collection from both Reddit and 4chan.

4.2 Crawler

The crawler will handle API calls to Reddit and 4chan, fetching data as it becomes available. For Reddit, the crawler will collect new posts and comments from specified subreddits, while for 4chan, it will focus on specific boards. The subreddit and board lists will be configurable.

4.3 Database (PostgreSQL)

The preprocessed data will be stored in a PostgreSQL database. This structure ensures that the data is well organized and accessible for future data analysis and modeling projects.

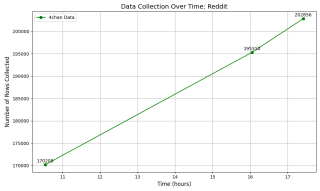

Figure 1: Reddit Data Overtime

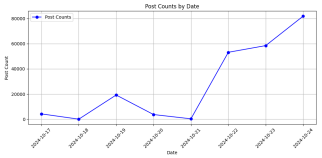

Figure 2: 4chan Data Overtime

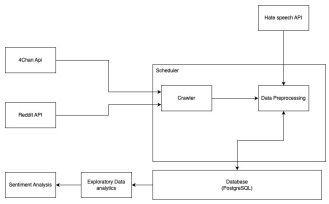

Figure 3: System Architecture

4.4 Description of current implementation

- **Environment Setup:** The application utilizes environment variables to manage configurations such as database connection strings and Reddit API credentials. The `dotenv` library is employed to load these variables from a configuration file.
- **Logging:** A logging mechanism is set up using the `logging` library to track the application's activities and errors. The logger is configured to display messages in a specific format, ensuring visibility of important events.
- **Reddit API Interaction:** The application fetches data from Reddit's API using the `requests` library. An OAuth2 token is obtained for authentication, allowing the crawler to access subreddit comments and posts.
- **Data Storage:** Fetched data is stored in a PostgreSQL database. The `psycopg2` library is used for database interaction, including inserting and updating records with support for JSON data types.
- **Job Scheduling and Management:** The `pyfaktory` library is utilized to manage background jobs. The system can schedule crawls using the Faktory job queue, enabling asynchronous processing of tasks to avoid blocking operations.
- **Continuous Scheduling:** A continuous loop is implemented to ensure that the crawler regularly fetches subreddit data. The `crawl_subreddit` function is called to initiate data fetching and storage, with scheduling of subsequent jobs occurring after each crawl. Used screen command for this purpose.
- **Cold Start Functionality:** The `cold_start_reddit_crawl` function allows the system to initiate the crawling process, pushing the first job to the Faktory queue when the application starts.

This architecture allows for a robust and scalable data collection system that efficiently retrieves and stores data from Reddit for further analysis.

4.5 Python Libraries

The following Python libraries are utilized in the implementation of the Reddit crawler:

- **requests**: For making HTTP requests to the Reddit API.
- **time**: For handling time-related functions, such as introducing delays.
- **requests.auth**: For managing HTTP Basic Authentication.
- **logging**: For tracking events and errors during execution.
- **pyfaktory**: For interacting with the Faktory job queue.
- **psycopg2**: For connecting to and querying a PostgreSQL database.
- **datetime**: For date and time manipulations.
- **sys**: For accessing system-specific parameters.
- **os**: For interacting with environment variables.

5 FILE IMPLEMENTATION DETAILS

The implementation consists of several key components:

5.1 Crawler Scripts

- `reddit.py`: This script is responsible for collecting data from Reddit. It interacts with the Reddit API to retrieve comments and metadata from specific subreddits.
- `chan_crawler.py`: A custom Python script designed to collect data from 4chan boards. It scrapes the catalog and individual threads from 4chan's boards.
- `cold_start_board.py` and `reddit_coldStart.py`: These are additional scripts aimed at performing an initial or cold-start data collection from 4chan boards and Reddit, respectively, to populate the database with baseline data.

5.2 Database Management

The project uses PostgreSQL with TimescaleDB for storing the collected data. TimescaleDB is a good fit for this project, given the time-sensitive nature of the data being collected, making it easier to perform time-series analysis.

The repository includes migration scripts managed using `sqlx`, which allow for easy version control of database schema changes. The database is run inside a Docker container, making it straightforward to set up and manage without complicated local configurations.

5.3 Faktory Job Queue

The system uses Faktory, a job processing system, for handling background tasks such as data fetching and storing. This enables the system to scale by distributing the crawling tasks over multiple workers, ensuring efficient data collection even as the amount of data increases.

6 CHALLENGES FACED

- **API Rate Limiting**: Reddit's API enforces rate limits, requiring us to implement efficient backoff strategies.
- **Storage Constraints**: The system collects a large amount of data, raising concerns about storage capacity.

7 PRELIMINARY DATA COLLECTION RESULTS

Based on initial data collection on 4chan and Reddit, here are the recorded data points:

- At 10:54 AM, Reddit had 25,970 rows and 4chan had 170,209 rows.
- At 4:05 PM, Reddit had 40,756 rows and 4chan had 195,310 rows.

8 PROJECTIONS AND ESTIMATES

Based on these collection rates, the estimated data collection per time unit is as follows:

- **Reddit**: Approximately 1.48 rows/sec, 88.65 rows/min, 5,319 rows/hour.
- **4chan**: Approximately 2.37 rows/sec, 142.16 rows/min, 8,530 rows/hour.

From this data, we project the following data volumes over different time spans:

- **Per Day (24 hours)**:
  - Reddit: 127,656 rows
  - 4chan: 204,720 rows
- **Per Week (7 days)**:
  - Reddit: 893,592 rows
  - 4chan: 1,432,080 rows
- **Per Month (30 days)**:
  - Reddit: 3,829,680 rows
  - 4chan: 6,141,600 rows
- **Per 3 Months (90 days)**:
  - Reddit: 11,489,040 rows
  - 4chan: 18,424,800 rows

8.1 Storage Projections

Assuming the average size of a Reddit row is approximately 500 bytes and a 4chan row is 600 bytes, we can estimate the following storage requirements for the data.

- **Reddit Storage Projections**:
  - **Per Day**: 127,656 rows × 500 bytes/row = 63,828,000 bytes ≈ 60.88 MB
  - **Per Week**: 893,592 rows × 500 bytes/row = 446,796,000 bytes ≈ 426.13 MB
  - **Per Month**: 3,829,680 rows × 500 bytes/row = 1,914,840,000 bytes ≈ 1.78 GB
  - **Per 3 Months**: 11,489,040 rows × 500 bytes/row = 5,744,520,000 bytes ≈ 5.35 GB
- **4chan Storage Projections**:
  - **Per Day**: 204,720 rows × 600 bytes/row = 122,832,000 bytes ≈ 117.76 MB
  - **Per Week**: 1,432,080 rows × 600 bytes/row = 859,248,000 bytes ≈ 819.82 MB
  - **Per Month**: 6,141,600 rows × 600 bytes/row = 3,684,960,000 bytes ≈ 3.43 GB
  - **Per 3 Months**: 18,424,800 rows × 600 bytes/row = 11,054,880,000 bytes ≈ 10.30 GB

9 CONCLUSION

The system has successfully demonstrated the feasibility of continuously collecting and analyzing data from both Reddit and 4chan. The implementation of the scheduler, crawler, and database integration provides a robust foundation for future data science applications.