From Threads to Insights: A 4chan-Reddit Dynamic Data Pipeline

Karthik Maganahalli Prakash
Binghamton University
New York, USA
kmaganahalli@binghamton.edu

Sanjana Shivanand
Binghamton University
New York, USA
sshivanand@binghamton.edu

Abstract

The foundation of any data science pipeline is reliable and continuous data collection. In this project, we propose building a continuously running system to collect the data from two sources: 4chan and Reddit. The data will be retrieved using only raw HTTP clients, with OAuth for Reddit and JSON endpoints for 4chan, explicitly avoiding crawler frameworks or Reddit libraries. The system design includes a scheduler, fetcher, parser/normalizer, and checkpointer, with PostgreSQL as the storage backend and logging for monitoring. The planned measurements include volume over time, per-community activity, engagement patterns, and cross-platform comparisons. Napkin math estimates suggest approximately 180,000 items per week (215 MB), or about 2.1 GB over time, which is well within the expected storage limits. This pipeline establishes the foundation for future projects by ensuring continuous, scalable, and structured data collection.

ACM Reference Format:
Karthik Maganahalli Prakash and Sanjana Shivanand. 2018. From Threads to Insights: A 4chan-Reddit Dynamic Data Pipeline. In . ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Data collection is the first step in any data science pipeline, and without continuous collection, data quickly becomes outdated. This project aims to develop a scalable and reliable system that collects social media content in near real time. The system will run continuously without manual intervention, ensuring new posts and comments are captured as they appear and providing the foundation for later analysis of trends, engagement, and community behavior.

We will collect from two sources: 4chan (mandatory) and Reddit. Data will be retrieved using raw HTTP requests with OAuth for Reddit and JSON endpoints for 4chan. Boards and subreddits will be specified through configuration files rather than hard-coded, ensuring flexibility and compliance. By following these principles, the system will be production-ready and capable of continuous automated operation.

Conference'17, Washington, DC, USA

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

2 Data Source

Our project requires collection from two distinct social media sources. The first mandatory source is 4chan, where we will leverage the official JSON API to dynamically discover boards and collect threads and posts. The second source is Reddit, which we selected over Bluesky due to its structured community design (subreddits), higher data volume, and direct alignment with the grading requirements. Both data sources will be configured dynamically from external files, not hard-coded, ensuring flexibility, compliance, and the use of only standard HTTP clients for API calls.

2.1 4chan

We will use 4chan's official JSON API [1]. This API is read-only and provides structured access to boards, threads, and posts. The following endpoints will be used in our collection pipeline:

- `https://a.4cdn.org/boards.json` – dynamically discover all boards.
- `https://a.4cdn.org/{board}/catalog.json` – detect new and active threads on a board.
- `https://a.4cdn.org/{board}/thread/{id}.json` – fetch complete post content within a thread.
- `https://a.4cdn.org/{board}/archive.json` – optionally backfill closed or archived threads (when supported).

These endpoints ensure that we can monitor activity across multiple boards without relying on hard-coded values.

2.2 Reddit

We chose Reddit over Bluesky for the following reasons:

- The Subreddits provide built-in topic segmentation, which makes the targeting straightforward.
- The higher and more consistent comment volume supports the required continuous collection and weekly projection plots.
- Direct alignment with grading: the rubric awards points for implementing a Reddit crawler and for dynamic subreddit handling.

We will use the OAuth-based JSON API [2], making raw HTTP requests without third-party libraries. The specific endpoints are the following:

- `https://www.reddit.com/api/v1/access_token` - Obtain an OAuth 2.0 access token.
- `https://oauth.reddit.com/r/{sub}/new` - Continuously discover fresh submissions (parameters such as `limit` and `after` will be used for pagination).
- `https://oauth.reddit.com/r/{sub}/comments` - Retrieve recent comments across multiple posts in a subreddit.

Subreddits will be stored in a configuration file ensuring that they are not hard-coded. Polling will be used for near real-time updates, keeping data fresh and avoiding one-time snapshots.

3 System Design

Our system continuously collects posts from 4chan and Reddit using a crawler service. The crawler has four main components: a scheduler to trigger fetch cycles, an HTTP fetcher (with OAuth for Reddit), a parser/normalizer to unify JSON data, and a checkpointer to prevent duplicates and enable recovery. The Collected data is stored in a PostgreSQL database, with monitoring and logging to track crawler health. This stored data supports downstream analysis, labeling, and visualization for future projects.

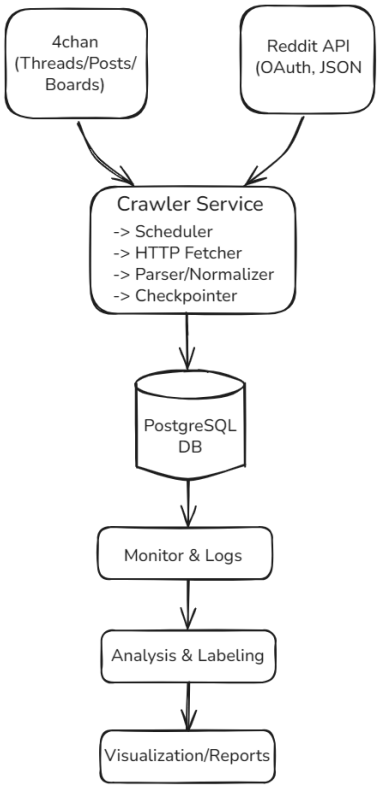

Figure 1: System architecture: data sources (4chan, Reddit) → Crawler Service → PostgreSQL DB → Monitoring/Analysis/Reports.

4 Measurements and Analysis

Once the crawler is running, we will measure data volume, community activity, and engagement to validate continuous collection. These metrics confirm the pipeline's reliability and provide insights into how communities behave and differ across platforms. Below we outline the key ideas:

- **Volume over Time:** Count posts and comments per hour, day, and week. This supports the required "continuous collection" plot.
- **Per-Community Activity:** Track activity per board (4chan) and per subreddit (Reddit) to compare which communities are more active.
- **Engagement / Structure:** Measure reply chains on 4chan and comment tree depth or upvotes on Reddit to study discussion patterns.
- **User Participation:** Count unique authors per day per community to estimate community size and engagement.
- **Growth and Trends:** Detect week-to-week changes, such as spikes in posting activity, to identify emerging events.
- **Cross-Platform Comparisons:** Compare Reddit vs. 4chan in terms of posting speed, text length, and media usage to highlight differences across platforms.

5 Napkin Math: Weekly Estimates

To confirm feasibility, we provide approximate weekly estimates:

- **4chan:** Suppose we configure 8 boards. If each board has around 50 new threads per hour and each thread averages 10 posts, this yields:

  $$8 \times 50 \times 10 = 4{,}000 \text{ posts/hour}.$$

  Because thread activity is uneven, we conservatively approximate this as ~4,000 posts per day, or about 28,000 posts per week.
- **Reddit:** Suppose we target 6 subreddits. If each subreddit averages 150 new comments per hour:

  $$6 \times 150 \times 24 = 21{,}600 \text{ comments/day}.$$

  This corresponds to about 151,200 comments per week.
- **Combined Volume:** Adding both platforms, our crawler is expected to collect on the order of ~180k items per week.

  $$28{,}000 + 151{,}200 \approx 179{,}200 \text{ items/week}.$$

6 Conclusion

In this project, we presented a continuous data collection pipeline for 4chan and Reddit. The system avoids crawler frameworks and prohibited libraries, uses configuration files instead of hard-coded sources, and includes a scheduler, HTTP fetcher, parser, and checkpointer with PostgreSQL for storage and monitoring. Planned measurements cover posting volume, community activity, and cross-platform comparisons. Napkin math estimates suggest ~180,000 items per week (≈215 MB), which is well within infrastructure limits. This pipeline provides a flexible, scalable, and compliant foundation for subsequent projects and ensures continuously updated data for analysis.

References

[1] 4chan. 4chan JSON API. https://github.com/4chan/4chan-API. Accessed 2025-09-22.
[2] Reddit API Documentation. https://www.reddit.com/dev/api/. Accessed 2025-09-22.
[3] luesky PBC. AT Protocol / Bluesky: Firehose & Read Endpoints (Documentation). https://docs.bsky.app/docs/advanced-guides/firehose. Accessed 2025-09-22.
[4] yan Mitchell. 2018. *Web Scraping with Python: Collecting Data from the Modern Web* (2nd ed.). O'Reilly Media, Sebastopol, CA.
[5] oberto Gonzalez-Bailon and Ning Wang. 2016. *Social Science Computer Review*, 34(1), 56–77. SAGE Publications.