Pipeline Maverick Project 1 Report

Tripti Ramesh Anchan
Department of Computer Science, SUNY Binghamton
Binghamton, New York, USA
tanchan@binghamton.edu

Abstract

This report covers the implementation of a data collection system that continuously gathers posts and comments from 4chan and Reddit about geopolitical events. The goal was to build a pipeline that could run automatically, pulling data from both platforms to see how the same topics get discussed in very different online communities. I built separate crawlers for each platform, stored everything in PostgreSQL with TimescaleDB, and created some basic analysis tools to compare sentiment and activity patterns. This report describes what I built, problems I ran into, what data I collected, and how much more data the system could handle going forward.

1 Introduction

This project looks at how political discussions play out on two very different platforms: **4chan** and **Reddit**. Reddit has structured threads with moderation and upvotes, while 4chan is anonymous, fast-moving, and pretty much unmoderated. By collecting data from both, I wanted to see how the same events get talked about differently depending on where the conversation happens.

The system was designed to run continuously, fetching new data from both platforms and storing it in a unified database for long-term analysis. Compared to the initial proposal, several practical changes were made during implementation:

- Introduced basic modules for **sentiment analysis** and **keyword trend visualization** to better capture tone shifts and topic activity.
- Implemented an **undocumented Reddit .json API method** instead of the OAuth-based approach to reduce setup overhead.
- Focused on sentiment and trend analysis rather than implementing the proposed keyword bootstrapping and activity spike detection, in order to ease out the development and concentrate on cross-platform comparison.

2 Implementation Overview

The system was implemented as a modular, continuously running data pipeline. Each module has a well-defined purpose, allowing the Reddit and 4chan crawlers to operate independently while sharing a common database and analysis framework.

- **Crawlers:** The two core scripts, *chan_crawler.py* and *reddit_crawler.py*, are responsible for fetching and normalizing JSON data. The 4chan crawler uses public board and thread APIs, while the Reddit crawler uses the undocumented *.json* endpoints.
- **Database:** All posts and comments are stored in *TimescaleDB* tables indexed on *created_at*. Tables are automatically deduplicated using unique indexes on thread and post IDs.
- **Filtering:** The *keyword_filter.py* script extracts geopolitically relevant records by matching a set of predefined keywords such as *Ukraine*, *Gaza*, *Taiwan*, and *Election*. Filtered outputs are exported as CSV files.
- **Analysis:** Analytical modules *volume_analysis.py* and *sentiment.py* visualize the collected data. Volume analysis produces time-series plots comparing posting activity and keyword trends between Reddit and 4chan, while sentiment analysis applies VADER to generate comparative distributions and category breakdowns.
- **Validation and Monitoring:** Additional scripts like *check_data.py* periodically inspect database contents, summarize record counts, and ensure that crawler jobs are populating data as expected.

Together, these components create a system capable of continuously collecting, filtering, and analyzing large volumes of online discussion data with less manual intervention.

3 Preliminary Exploration of Data

The final collection phase produced approximately **160,000 posts from 4chan** and **30,000 items from Reddit** (including both posts and comments). Figure 1 illustrates the activity volume over time, where 4chan's data density significantly exceeds Reddit's. The cumulative collection plot confirms stable ingestion, showing that the pipeline continuously fetched and stored data across the observation period without major interruptions.

For the initial exploration, keyword and sentiment analyses were conducted using the filtered datasets (Figures 2 and **??**). The keyword trends reveal clear event-linked bursts around major geopolitical discussions such as *Gaza* and *Ukraine*. Sentiment distribution comparisons show distinct tonal contrasts between the two platforms—Reddit discussions tend to remain neutral to mildly positive, while 4chan posts lean sharply negative and more polarized, reflecting its less moderated and more emotionally charged discourse style.

From a technical perspective, data quality was stable overall, though 4chan occasionally served dead threads due to its rapid content churn. Future runs may benefit from tighter scheduling intervals and lightweight duplicate checks.

4 Projected Data Growth

Based on current ingestion rates:

- 4chan: ~32,000 posts/day $\Rightarrow$ ~224,000 per week.
- Reddit: ~6,000 posts/comments per day $\Rightarrow$ ~42,000 per week.
- Combined weekly growth: ~266,000 records.

At this rate, the system will exceed 1 million records within 6 weeks of continuous operation.

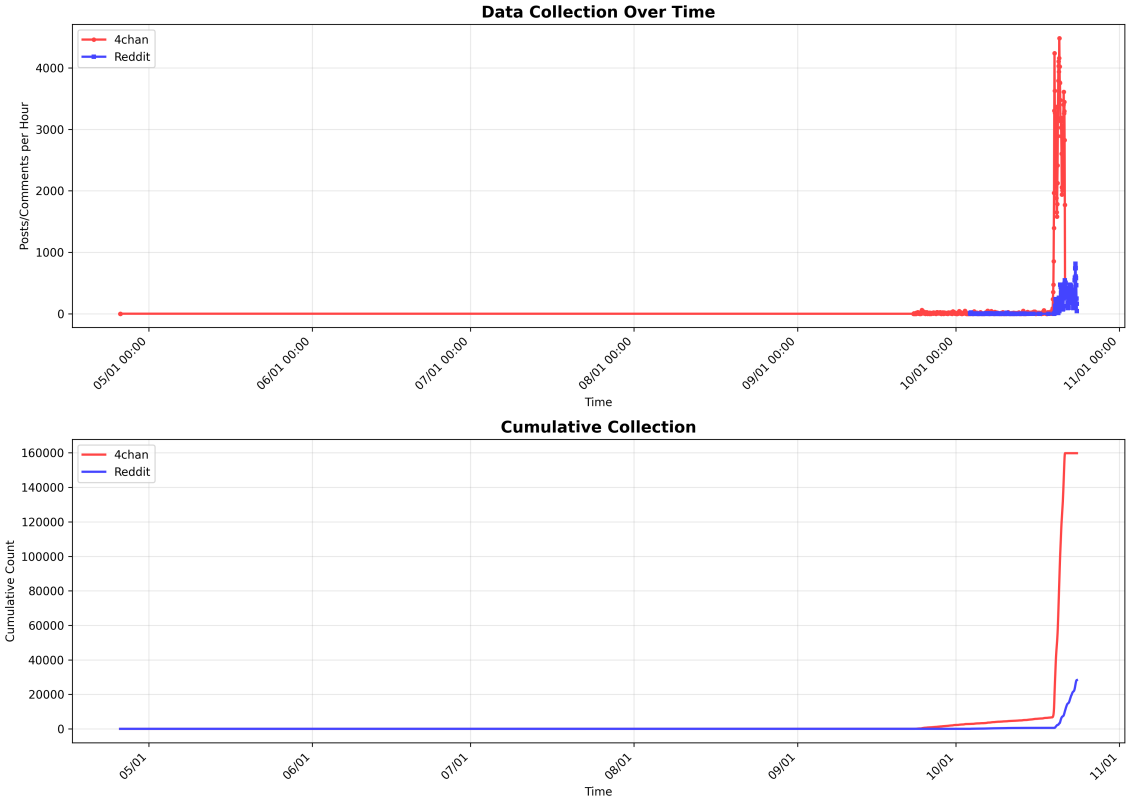

Figure 1: Data Collection Over Time showing hourly post/comment volume for 4chan (red) and Reddit (blue). The top panel displays volume per hour while the bottom panel shows cumulative collection.

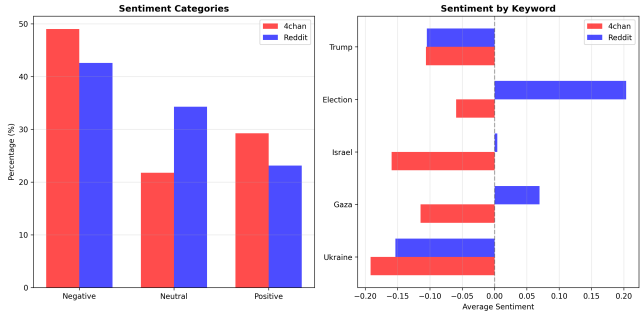

Figure 2: Sentiment Analysis across Reddit (blue) and 4chan (red). It shows overall category breakdown (pos, neg and neutral), and average sentiment per major keyword.

5 Challenges and Adjustments

Throughout the development, several practical challenges surfaced that shaped how the final system evolved:

- **Reddit Rate Limits:** When switching to the undocumented `.json` method for Reddit collection, the crawler frequently encountered `429 Too Many Requests` responses. To mitigate this, random sleep intervals and delayed requeueing in Faktory were introduced, allowing tasks to back off and resume without crashing the worker.
- **Duplicate Job Execution:** During early runs, the Faktory job queue occasionally spawned duplicate jobs for the same thread or post due to overlapping producer pushes. This led to redundant database inserts. The issue was resolved by adding unique composite keys (`created_at`, `subreddit`, `post_id`)
- **Visualization Clarity and Overlaps:** Daily aggregation was essential to make keyword time-series readable; otherwise, the high frequency of 4chan posts overwhelmed the Reddit data, obscuring meaningful patterns

These adjustments collectively improved both the reliability and interpretability of the collected data, ensuring the pipeline remained stable even under continuous load. To verify system stability during prolonged runs, the Faktory job queue dashboard (Figure 4) was monitored too

6 Conclusion

The implemented system successfully demonstrates a continuous, multi-platform data collection pipeline. It captures differences in sentiment and framing between anonymous and moderated online communities.

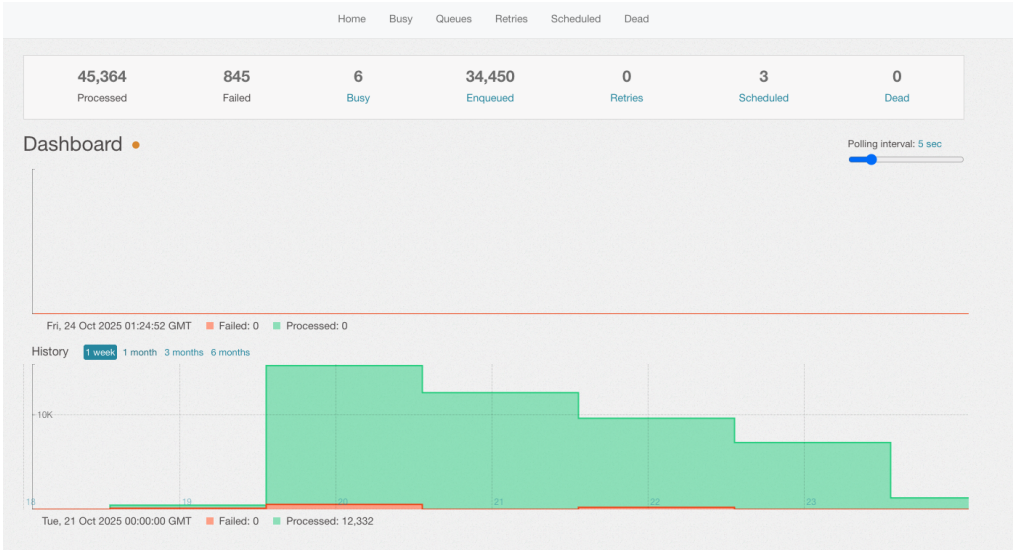

Figure 4: Faktory job queue dashboard showing system throughput and reliability. Over 45,000 jobs were processed with some failures, confirming a working crawler.