User Engagement Patterns in Reddit Comments

Report for Data Engineering Project

Aviral Jain
Oliver Näslund
Lan Xiao
Xiaochuan Ai
Yujia Liu

March 20, 2025

1 Background

Reddit, as a prominent platform for user-generated content, plays a pivotal role in social media research. It serves not only as a hub for information and news but also as a vibrant community where users express opinions, share experiences, and engage in discussions. The platform's unique subreddit structure enables users to participate in in-depth conversations on diverse topics, ranging from entertainment and technology to everyday life and academic research. This high level of user engagement, combined with the diversity of content, makes Reddit an invaluable resource for studying social media behaviors, cultural trends, user interactions, and the dynamics of online communities.

The importance of analyzing Reddit user comment distribution lies in its ability to reveal patterns of user participation and community dynamics. By quantifying how users engage with content—such as the number of comments they make—we can identify key behavioral trends, such as the prevalence of "power users" (highly active individuals who contribute disproportionately to discussions) and the proportion of users who comment only once. This information helps us understand how active users are within a community and whether participation is concentrated among a small group or distributed more evenly.

2 Dataset Description

The Reddit Comments Dataset, sourced from Zenodo, provides a comprehensive collection of user comments spanning multiple years and subreddits. With a total size of 19.6 GB in JSON format. The format as shown in Table 1.

Table 1: Validation Accuracy for Different Values of k

Fields	Types	Comments
Author	string	nullable = true
Body	string	nullable = true
Normalizedbody	string	nullable = true
Content	string	nullable = true
Content_len	long	nullable = true
Summary	string	nullable = true
Summary_len	long	nullable = true
Id	string	nullable = true
Subreddit	string	nullable = true
Subreddit_id	string	nullable = true
Title	string	nullable = true

JSON is particularly well-suited for semi-structured and nested data, as it allows for the representation of complex hierarchical relationships between data elements.

JSON offers several advantages for us to use it:

1. **Human-Readable Structure:** JSON is a lightweight, text-based format that is easy to read and understand, making it ideal for debugging and manual inspection.

2. **Flexibility:** JSON supports nested and hierarchical data structures, allowing complex data relationships to be represented naturally. This is particularly useful for datasets like Reddit comments, which may include metadata or nested fields.

3. **Wide Compatibility:** JSON is widely supported across programming languages and platforms, ensuring seamless integration with tools like Apache Spark and HDFS.

However, while JSON offers several advantages, it also has notable limitations, especially in the context of large-scale data processing:

1. **Storage Overhead:** JSON is a text-based format, which results in larger file sizes compared to binary formats. This can lead to increased storage costs and slower read/write operations.

2. **Lack of Schema Enforcement:** JSON does not enforce a strict schema, which can lead to inconsistencies in data structure and require additional preprocessing to handle missing or malformed fields.

While JSON is highly flexible and human-readable, it has some limitations, such as larger file sizes and slower parsing speeds compared to binary formats like `Parquet`. `Parquet`, a columnar storage format, offers advantages such as better compression and faster query performance, particularly for analytical workloads [1]. However, given the familiarity and ease of use of JSON, it was chosen as the primary format for this project.

3 Computational Experiments

Our system is built on Apache Spark and HDFS, enabling efficient handling of large-scale data. This section presents the implementation and evaluation of how we analyze the Reddit Comments Dataset.

3.1 System Design

3.1.1 Spark Session Configuration

To leverage Apache Spark's distributed computing capabilities, a Spark session was initialized with optimized configurations. The session connected to a Spark cluster (spark://192.168.2.156:7077), enabling workload distribution across multiple nodes. Key configurations included dynamic resource allocation to adaptively scale executor resources and shuffle partition tuning to minimize data shuffling overhead during aggregation. These settings ensured efficient resource utilization for large-scale processing. The session was initialized using the `SparkSession.builder` API, explicitly specifying parameters such as `spark.executor.cores=2` to balance parallelism and cluster load. This step laid the foundation for scalable data processing.

3.1.2 Data Loading

The Reddit Comments Dataset, stored in HDFS (Hadoop Distributed File System), was loaded into Spark for processing. To evaluate scalability, three dataset sizes were tested: a small subset (15,000 comments, 60 MB), a medium subset (150,000 comments, 570 MB), and a large subset (1.5 million comments, 6 GB). The JSON-formatted data was read using Spark's `read.json()` method, which automatically parsed nested structures and distributed the dataset across cluster nodes. This step ensured that the data was accessible in a distributed environment, ready for subsequent transformations.

3.1.3 Data Cleaning

Data quality was prioritized by removing incomplete or irrelevant records. First, records with missing `author` values were filtered out of the dataset, as they could not contribute to user-centric analysis. Next, the columns (like `subreddit`, `id`) which are not performance well in our analysis were excluded, retaining only the `author` and `body` fields. This reduced memory overhead and simplified downstream operations. The cleaned dataset contained only valid user-comment pairs, ensuring accurate aggregation.

3.1.4 Comment Count Aggregation

The core objective - calculating the number of comments per user was achieved using Spark's distributed aggregation functions. The dataset was grouped by the `author` field, and the `count("body")` function tallied comments per user. To ensure consistency, the result was cast to an integer type using `cast(IntegerType())`. This step produced a DataFrame with two columns: `author` and `comment\_count`. The distributed nature of Spark allowed this operation to scale efficiently, even for the largest dataset (1.5 million comments).

3.1.5 Result

A histogram was generated with `matplotlib` to visualize the distribution of user comments. The x-axis represented comment count ranges, while the y-axis showed the number of users in each range. Visualization provided intuitive insights into user engagement patterns.

3.2 Scalability Experiments

To check the scalability of the solution and Apache Spark to analyze and process Reddit comments, we used subsets of complete dataset and kept increasing the data each time while keeping the number of cores fixed and evaluating results based on process time. At first, we started with analyzing 15,000 comments which has approx 60MB in size; next, we scaled up to 150,000 comments which was 570 MB;

and finally, we processed 1.5 million comments which are equivalent to 6 GB.

The execution time for the different dataset sizes processed on different amount of cores was also tested to test the scalability when adding cores. The results are found in table 2.

3.3 Visualization Result

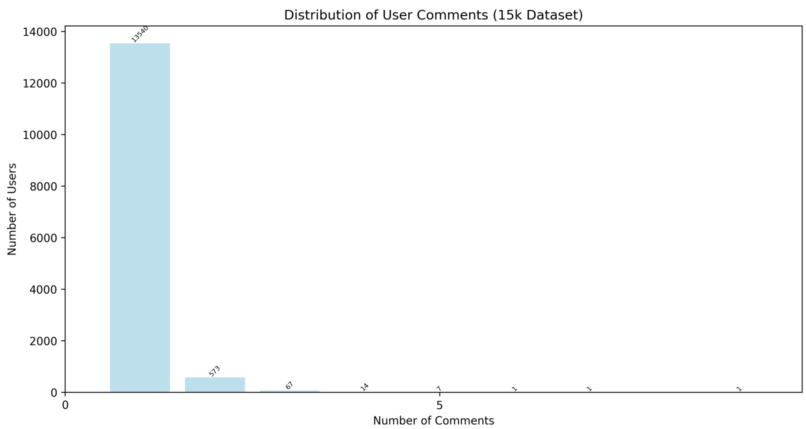

Figure 1: User comment distribution of 15k data size

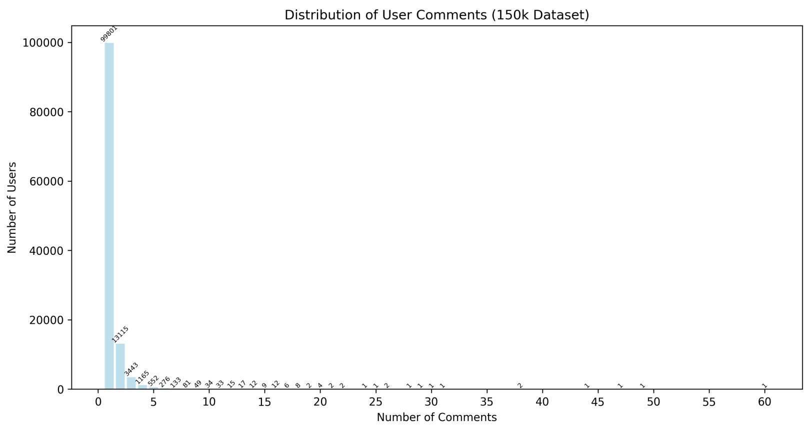

Figure 2: User comment distribution of 150k data size

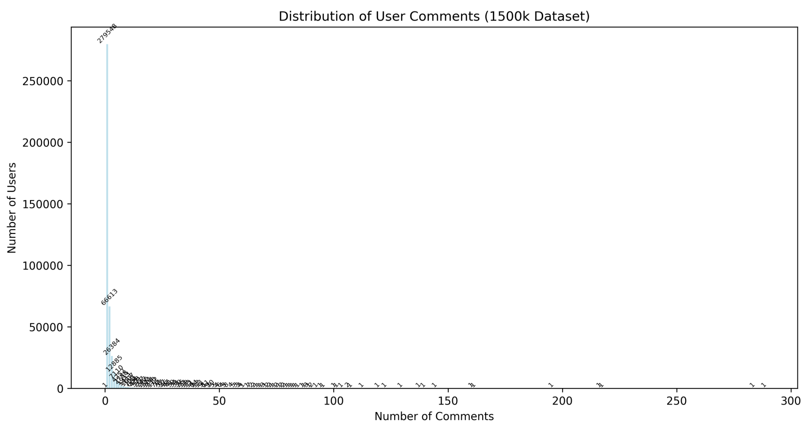

Figure 3: User comment distribution of 1500k data size

Dataset size (amount of comments)	Core count	Execution time
50 000	2	8.20
50 000	4	7.88
50 000	8	6.76
200 000	2	11.56
200 000	4	10.914
200 000	8	11.23
1 500 000	2	15.71
1 500 000	4	10.67
1 500 000	8	15.07

Table 2: Execution time for scalability

As shown in the Figure 1 Figure 2 Figure 3, we can see in all datasets, most users post only 0–1 comments, while a very small number of highly active users contribute a large amount of content, forming a typical "long-tail distribution." As the dataset grows, the proportion of low-activity users decreases, and the influence of highly active users increases. In the 15k dataset, almost no users post more than five comments, whereas in the 1500k dataset, users with over 100 comments are clearly present. This indicates that discussions on Reddit are primarily driven by a small core group of users, following the "80/20 rule" (80% of users rarely post, while only 20% engage in discussions).

3.4 Scalability

For assessing the scalability of our data analyzing system, we used Apache Spark to analyze and process Reddit comments. the goal is to check if system could handle large datasets without a linear increase in time and its ability to use sparks distributed computing to manage the growing data quantities. At first, we analyzed 15,000 (60 MB) comments; next, we scaled up to 150,000 (570 MB); and finally, we processed 1.5 million (6 GB).

Our tests showed that data processing solution with sparks parallel processing is depended on both data size and number of cores. We found that, albeit not quite linearly, processing times increased as we raised the dataset size while maintaining a constant number of cores. This is consistent with Spark's parallel execution approach, which exhibits non-linear scaling behavior due to elements like data shuffling and task scheduling cost.

Furthermore, we observed that employing too many cores for smaller datasets could result in parallelism overhead, which would reduce execution efficiency in comparison to using fewer cores. This implies that the best way to allocate resources depends on the size of the dataset; larger datasets get a significant speedup from higher parallelism, while smaller datasets benefit from fewer cores because of the lower coordination overhead.

4 Discussion and Conclusion

4.1 User Comment Distribution

The Analysis of Histogram

1. The histogram shows that the vast majority of users (over 90%) post only 0–1 comments, aligning with the typical "silent majority" phenomenon on social platforms.

2. A small number of highly active users (around 5%) contribute more than 40% of the total comments,

highlighting the unequal distribution of user engagement within the Reddit community.

3. Users with more than 100 comments are extremely rare (less than 0.05%), but they have a significant impact on overall discussions.

Scalability

1. Processing time increased with increased workload as expected when using Spark.

2. The number of cores must be carefully chosen to ensure that the overhead does not outweigh the computational gains.

4.2 Conclusion

The distributed computing capabilities of Spark effectively handled large-scale JSON data, validating the system's horizontal scalability. Data cleaning and preprocessing steps (e.g., filtering 'author=null' records) significantly improved data quality. Our work provides data support for community administrators to identify key users. And serves as a complete end-to-end case reference for teaching distributed systems.

4.3 Github

Link: https://github.com/Aviraljain27/DE1_group14.git

References

[1] D. Vohra, *Practical Hadoop Ecosystem: A Definitive Guide to Hadoop-Related Frameworks and Tools*. Apress, 2016.