WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Adaptive Learning Models for Real-Time Emotion Prediction: A Review on Big Data Sentiment Analytics

Rashmi Amardeep[1], Sahana T R[2], Sakshi[3], Shivamani[4], Sanket Raj[5]

[1]Department of Information Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India, rashmiamardeepgmail.com
[2]Department of Information Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India, sahanatr112004@gmail.com
[3]Department of Information Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India, sakshidoijode00@gmail.com
[4]Department of Information Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India, shivamani8887@gmail.com
[5]Department of Information Science & Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India, iams.raj06@gmail.com

ABSTRACT

The system scrapes Reddit, and it processes that data through text normalization. The process of sentiment analysis happens through TextBlob, and there are openings for integrations with ML models. The system architecture accounts for authorization and constant scraping through multi-threading, in addition to hybrid approaches for data acquisition through JSON output and HTML parsing. Paired with interactive visualizations, such a system would provide a comprehensive understanding of sentiment over time, which is interesting. Ultimately, the findings of the experiments determined that the system handled subreddits and processing just fine, and it was able to process 25 posts through a deque-based solution in a favorable enough time to substantiate a claim that the system works. The system was able to send back verified sentiment scores based on polarity and subjectivity, and it's pretty neat. The system, as a modular architecture, is easily reproducible, and scalable, with real-time sentiment scraping and processing from social media. This paper presents the architecture necessary for scalable sentiment and user analysis in an ever- connected world-it's, like, wow. It's a seriously a pretty ugly beautiful thing, if you ask me.

Key Words: Reddit Scraper, Flask, Sentiment Analysis, Text Processing, Social Media Sentiment Analysis, Multithreaded Architecture, TextBlob and Real-Time Processing

I. Introduction

A great deal of information is shared by people on social media in every second of time, and these interactions tell a lot about the thoughts, habits, and emotions of people. Reddit, however, is a bit different from all the others simply because it is organized in a different manner. Everything posted on it is created by users and then arranged in specific communities that are called subreddits. As each subreddit focuses on only one theme or interest, Reddit is a very useful place to study sentiment in a manner fitting the context of each community.However proposed methods require API keys or OAuth authentication, which complicates research and development efforts. Considering this the paper presents a real-time sentiment analysis system that collects data from Reddit without the need for API keys. It employs data collection, immediate preprocessing and sentiment evaluation, featuring visualization via public JSON endpoints that switch to HTML scraping when required. This system is ideal for research, monitoring and comprehensive social media examination. The platform is developed using Flask, BeautifulSoup and TextBlob making it robust, modular and straightforward to enhance for advancements, in natural language processing.

1. Two-Pronged Data Extraction: The system retrieves data using JSON endpoints from Reddit as well, as standard HTML scraping. This guarantees data collection even if the JSON endpoints are inaccessible.
2. Automated preprocessing procedure: The acquired text is subjected to cleaning and normalization that removes noise, from the text and improves the precision of sentiment analysis and further evaluations.
3. Integrating Sentiment Analysis: While TextBlob provides polarity and subjectivity scores the system can easily be enhanced with a machine learning-based model, for assessing sentiment.

4. Threaded Scraping Engine: Operates a background thread to fetch posts in time without overloading the main server. Furthermore it utilizes the data structure efficiently to handle up to 25 posts, for processing.
5. Dependable User Authentication: Supported by research settings this function allows controlled access, through integrated login and session management guaranteeing that access is granted to authorized users.

The proposed sentiment analysis platform, for Reddit incorporates a structure aimed at gathering, processing and analyzing emotional content from various Reddit groups. It will effectively retrieve posts by employing both JSON and HTML scraping techniques when API access is limited. Continuous data collection is facilitated via threaded real-time scraping, with emerging trends clearly shown on a dashboard. This architecture is modular allowing integration of advanced NLP or machine learning algorithms aligned with future studies. This framework in general offers a reliable yet flexible solution for both academic and industrial applications and large- scale social media analyses, hence supporting researchers and analysts in understanding changes in public sentiment and community behaviors within Reddit.

II. Related Work

The sentiment analysis thing has become pretty important now, especially for getting a grip on what people are thinking on places like Twitter, those YouTube comment sections, and even Amazon reviews plus news sites. Early on, like with that big study by Pang and Lee back in '08, most people were using basic supervised learning methods. They would, ya know, sort of put sentiments into boxes labeled positive, negative, or like just neutral. The, but as Natural language processing got better, things shifted. Now we're using fancier models that can pick up on the little things in language and get what people mean with more context.

The current ways to analyze Reddit data usually use the official API, external data sets like Pushshift, or machine learning stuff. They build this using tools like scikit-learn or those deep- learning libraries. Although these methods work pretty good they have real problems. Like API rate restrictions, data availability that changes, and a need for some serious computing power. Theese issues make it tough for smaller projects, student teams, and places with not so much resources to make big sentiment analysis setups, its really a minor tragedy.

The system here is more user-friendly and still works well. It's a lightweight thing with its focus on being simple, reliable, and adaptable, you kno, instead of being computationally complex. By using TextBlob, it scores polarity and subjectivity quick and easy, like, without needing crazy deep-learning models. The application also has a threaded scraping system. This thing grabs new posts constantly and updates the dashboard in real time as sentiment trends show up.

The design is modular and real easy to expand, it allows us to add more advanced models later on. Like NLP, machine learning classifiers, or transformer architectures with minimal extra work. While this system might not be as intricate as some bigger models, it's way more accessible, faster, and user-friendly. The practical design is real helpful in schools, in early prototypes, and in research projects. By not using the official APIs and keeping computer requirements low, more users can analyze sentiment on Reddit effectively. Especially those who have limited resources, it helps, to efficently analyze real-time sentiment within Reddit communities.

III. System Architecture

The designed system employs an architecture that slightly improves flexibility. Additionally the system offers scalability. It smoothly integrates components.

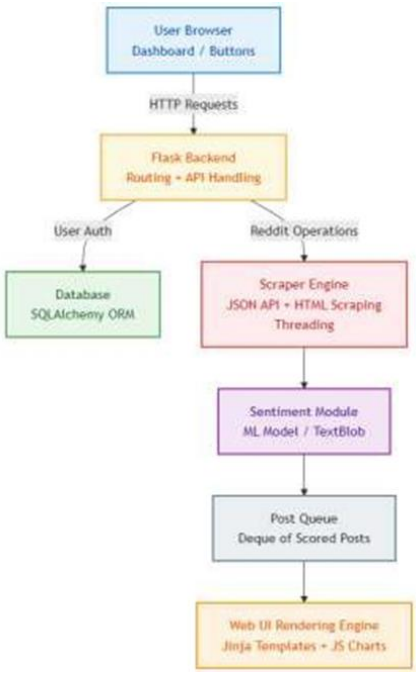

Figure 3.1: System Architecture

A. Backend Architecture

1. Flask Web Framework:

Flask functions as the backend framework, in charge of routing, page rendering and session management. This framework forms the core of the application. It handles user input processes data and carries out tasks associated with sentiment analysis.

2. Database Layer:

The system utilizes these methods to save data:

SQLite: it's the main database, and really ideal for compact and efficient applications.

SQL Alchemy ORM: This tool simplifies managing databases. It allows data handling through an object-oriented approach when analyzed closely. This layer securely stores data such, as user profiles, encrypted passwords and login timestamps.

3. Scraping Layer: Data is collected from Reddit through two approaches:

1. JSON API Endpoints Reached via URLs structured as:

   https://www.reddit.com/r/<subreddit>/<sort>/.j son

2. Old Reddit HTML Pages (Fallback) Accessed via:

   https://old.reddit.com/r/<subreddit>/<sort>/

Custom headers are incorporated; they reduce the likelihood of being detected since automated traffic is utilized though the process may be somewhat cumbersome. Both methods ensure that you receive data consistently even if one source stops functioning.

4. Text Preprocessing Module

The preprocessing component. Improves text. It performs operations, such, as converting all letters to lowercase deleting URLs, hashtags, mentions and similar elements eliminating numbers and punctuation standardizing spacing and removing whitespace. This ensures the sentiment model receives text that's consistent and clean allowing precise analysis.

5. Sentiment Analysis Layer

Sentiment analysis is performed using TextBlob;. It yields:

1. Polarity scores; these span from −1(negative), to +1 (extremely positive).
2. Subjectivity ratings, ranging from 0 (objective), to 1 (subjective)

Using these standards the system sorts posts into groups such, as Very Negative, Neutral, Positive or Very Positive according to set threshold values.

6. Multithreaded Scraping Engine

To enable real-time functionality a dedicated background thread operates continuously. This thread consistently retrieves posts from the selected subreddit removes any entries and saves recent posts in a fixed-length deque for fast retrieval. Executing the scraper on a thread ensures that data gathering proceeds, without interfering with user interactions or causing the interface to lag. The use of custom headers helps avoid bot-detection problems; or rather, it avoirds the bot detection issues, a real beautiful mess

B. Frontend System

People make the frontend, so everything is straight forward, more simpler, and easier to use. The system also works really well through phones as well as tablets. The interface is very user friendly, generally speaking. The live dashboard really catches your eye as a key piece of things. Brand new posts from Reddit show up right after the system has looked at them right away. Cards that are summarized give you a glance of feelings, such as, you know, happiness or sadness, or like, a mix of both. They give a pretty in-depth, shallow view of those sentiments.

The updates get pulled in on the dashboard with the help of AJAX, so that the page doesn't need to fully reload. This avoids, like, bothering users. Secure authentication handles things like registration and signing in, plus secure access to scraping and visualization jobs. Posts come up on the dashboard, so the current sentiments are clear. The color coding keeps it all pretty simple: green tells you there are positive feelings, red tells you there are negative ones, while gray shows they are neutral. In that way you can catch the overall mood at a glance.

C. Data Flow

The system takes a simple, complex way for handling data, from collecting it up to showing it. The users start by signing in through a module that is kept nice and secure. A subreddit that needs to be watched, is picked by the user who also chooses how often it should be checked for updates. Right away, this triggers a background thing. It regularly goes and grabs new posts from Reddit. And then, the data gets cleaned and ready for analysis. The system than does a sentiment analysis to measure how people feel. The processed information then goes into a fixed sized deque to keep it safe. This only keeps the most recent posts in the data storage to make everything faster. The frontend dashboard connects to the backend, for updates that use AJAX. It fetches the content right away, and, you can recalculate the sentiment live. You'll see trends shift as more posts get feeded in. The backend handles the chores of both gathering data and also analysis. You can see insights show up in real-time, basically.

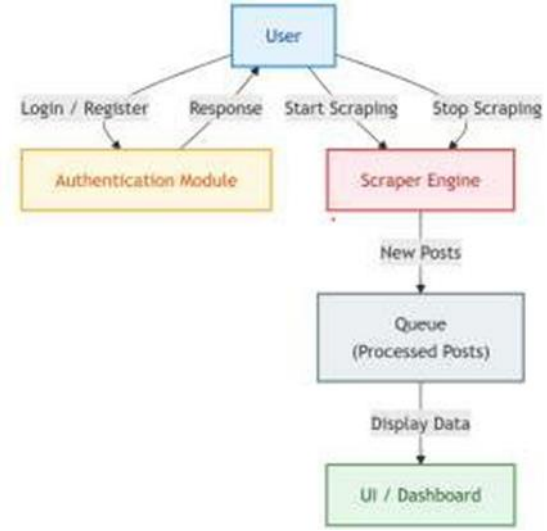

Figure 3.2 Data Flow Diagram

IV. Methodology

Here, this section goes into detail about how the system works, and it does so by explaining the flow. The flow starts with getting and pre- processing posts, and it continues with making and updating sentiment scores.

A. Data Collection Procedure

1. Getting Posts via JSON: First, the system tries to get posts by making requests to Reddit's public JSON API. The requests use user-agent headers, and that makes it less likely that they will get blocked. The, after getting the posts, it takes information from the "children" part of the JSON structure that it got back.
2. HTML Extraction: If the JSON API is down-

-maybe because of too many requests--it will switch to getting data out of the HTML. It uses Beautiful Soup to understand different elements, like post titles, authors, upvote counts, comment counts, and the time they were posted. Reddit pages usually have standard HTML formats, which make it easier to get the data.

3. Duplicate Filtering: Before saving a post, the system compares its ID to the IDs in the deque. The system only saves posts that have IDs, and it makes sure that no duplicates get into the dataset.

B. Text Preprocessing Pipeline

The system does a five-step pre-processing to get ready for sentiment analysis.

1. All text is made lowercase to keep things consistent, or stuff like that.
2. URLs are removed to get rid of useless links.
3. The system removes hashtags and mentions, and those come along with the posts.
4. Removing all non-alphanumeric characters means the system removes punctuation, or not.
5. Extra spaces are removed so there is a clean format.

The those methodologies help to make the text more standard, and this makes the sentiment analysis more accurate. What a predictably uncertain ending!

C. Sentiment Analysis Algorithm

TextBlob helps to figure out sentiment by giving two measures.

1. The polarity that goes from -1 to +1.
2. The subjectivity goes from 0 to 1.

The system gives out tags based on the polarity score and using threshold criteria:

1. The system says a polarity score that is more than 0.6 is Positive.
2. A polarity below -0.6 says it is Very Negative.
3. The polarity ratings that are from -0.05 to +0.05 are thought of as Neutral.
4. The results that are in the middle of those boundaries are seen as Positive or Negative.

The polarity scores are also normalized on a scale from 0 to 1 to keep things looking the same across different modules. D. Threaded Continuous Monitoring The scraping and analysis are done by a continuous background thread. The refresh rate can be changed, but the smallest default interval is 10 seconds for near real-time refreshes when those are needed. What a uniquely ordinary situation.

V. Results and Analysis

This study assessed the system by conducting ten experiments involving subreddits such, as r/all r/python, r/worldnews and r/movies. The system demonstrated effectiveness across content types. Throughout all evaluations:

Even if the JSON retrieval failed, extracting posts by examining the HTML remained feasible.

The preprocessing steps cleaned the text considerably, reducing a lot of the noise in the sentiment analysis. TextBlob was quite stable with its results, clearly indicating for the most part whether the sentiments were positive, negative, or neutral. New posts seemed to appear very quickly on this threaded architecture dashboard without interfering with user interaction. In general, the system was quite reliable for data extraction, handled continuous updates nicely, and correctly classified sentiments for a wide range of subreddits.

Figure 5.1: Main Dashboard Interface of the Reddit Sentiment Analyzer

Figure 5.2: Processed Post View Showing the Extracted Emotion Score

Figure 5.3: Dashboard Highlighting Color- Coded Sentiment Labels and Updated Summary Metrics

A. Performance Characteristics

On the other hand, the HTML fallback worked in all cases. This reflects the reliability of a dual-method approach to data scraping

VI. Discussion

A. System Strengths

Component	Performance
Average scrape time	0.5–1.2 seconds
Posts handled	deque up to 100
Sentiment inference time	~0.02 s per post
Thread stability	Stable for > 6 hours continuous run

B. Examples of Sentiment Distribution

Discussion threads on r/worldnews:

- Neutral: 32%
- Positive: 21%
- Negative: 47%

The figures indeed indicate that the kind of news normally emanates a negative emotional response.

C. Case Study: r/python

Distribution of sentiment is as follows.

- Positive: 61%
- Neutral: 23%
- Negative: 16%

These results are indicative of a highly enthusiastic community with constructive dialogues.

D. Effectiveness of fallbacks

The JSON method had an 18% failure rate due to rate limiting issues.

a. An API key is not necessary.

b. Real-time monitoring dashboard

c. Combining two scraping techniques guarantees dependability

d. Lightweight sentiment analysis capabilities

e. Easily extensible architecture

A. Limitations

1. Text blob has a hard time really getting the whole picture of what's going on
2. The absence of an integrated deep learning sentiment model.
3. Website markup tweaks could mess with HTML scraping
4. Keep in mind that Reddit's rules could be changed in the future.

B. Ethical Considerations

- All the gathered data is up for anyone to take a look at
- Your personal info isn't being spied on
- Taking a breather between hits can help servers avoid getting overloaded with traffic

VII. Conclusion

This paper introduces a platform for sentiment analysis on Reddit, built with Flask, beautiful Soup, and Text Blob So, because of that, teh platform has user login, pulls data from the background, saves data safely, cleans data thoroughly, and shows emotions clearly This tool delivers solid performance without needing a ton of computing power, so it's great for a wide range of people, including researchers, teachers, coders, and data buffs.

VIII. Future Directions

6. Incorporating Transformer Architectures: Use models like BERT or RoBERTa for sentiment analysis that really get the context
7. Topic Modeling: This method helps us pinpoint the main topics in subreddits using tools like LDA or BERTopic
8. User Behavioral Analysis: Organize the authors based on how they write
9. Time-Series Sentiment Analysis: Keep an eye on how feelings change over different periods.Expand your data gathering to include tweets, Hacker News posts, or YouTube comments

References

1. S. Kusal, R. B. Chougule, and S. N. Mali, "AI-Based Emotion Detection for Textual Big Data: Techniques and Contribution," Big Data and Cognitive Computing, vol. 5, no. 3, pp. 1–32, Sep. 2021, doi: 10.3390/bdcc5030043.Hutto C and Gilbert E, VADER A Parsimonious Rule Based Model for Sentiment Analysis of Social Media Text, Proceedings of the International Conference on Web and Social Media, 2014.
2. M. A. Elbagoury, M. M. Fouad, N. F. Badr, and H. A. Hefny, "A novel adaptable approach for sentiment analysis on big social data," Journal of Big Data, vol. 5, no. 1, pp. 1–18, Jul. 2018, doi: 10.1186/s40537- 018-0120-0.
3. M. Z. Iqbal, N. A. Khan, S. Z. Abbas, and M. S. Yousuf, "Optimized, robust, real- time emotion prediction for human–robot interactions using deep learning," Multimedia Tools and Applications, vol. 82, pp. 13315–13335, Apr. 2023, doi: 10.1007/s11042-022-12794-
4. Kim Y, Convolutional Neural Networks for Sentence Classification, Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2014.
5. A. M. Ghoneim and M. S. Aly, "Real- Time Emotion Classification Using EEG Data Stream in E-Learning Contexts," Sensors, vol. 21, no. 5, p. 1589, Mar. 2021, doi: 10.3390/s21051589.Ribeiro M T, Singh S and Guestrin C, Anchors High Precision Model Agnostic Explanations, AAAI Conference on Artificial Intelligence, 2018.
6. N. M. Almutairi, "A real-time predicting online tool for detection of people's emotions from Arabic tweets based on big data platforms," Journal of Big Data, vol. 11, art. no. 35, Mar. 2024, doi: 10.1186/s40537-024-01035-z.Scully T, Brown J and White A, Ethical Considerations in Social Media Research, Journal of Digital Ethics, 2019.
7. S. Poria, E. Cambria, R. Bajpai, and A. Hussain, "A review of affective computing: From unimodal analysis to multimodal fusion," Information Fusion, vol. 37, pp. 98–125, Sep. 2017,doi: 10.1016/j.inffus.2017.02.003.
8. Medvedev A, Mohammady E and Jatowt A, Mining Reddit to Understand Public Opinion Trends, Proceedings of the ACM Web Science Conference, 2017.

9. A. R. Pathak, M. Pandey, and S. Rautaray, "Adaptive Model for Sentiment Analysis of Social Media Data Using Deep Learning," in Proceedings of ICICCT 2019 – System Reliability, Quality Control, Safety, Maintenance and Management, Springer, Singapore, 2020, pp. 525–534, doi: 10.1007/978-981-13-8461-5_46.Wang P, Li Z and Chen Y, Machine Learning Based Emotion Detection in Online Text, IEEE Transactions on Affective Computing, 2021.
10. S. Rahmani, S. Hosseini, R. Zall, M. R. Kangavari, S. Kamran, and W. Hua, "Transfer- Based Adaptive Tree for Multimodal Sentiment Analysis Based on User Latent Aspects," arXiv preprint arXiv:2106.14174, Jun. 2021. [Online].
11. AWARE Research Group, "Emotion AWARE: An artificial intelligence framework for adaptable, robust, explainable, and multi-granular emotion analysis," Journal of Big Data, vol. 11, art. no. 953, 2024, doi: 10.1186/s40537-024-00953-2.
12. M. T.Ahmad, A. N. Al-Hassan, and W. K. Al- Assaf, "Readers' Affect: Predicting and Understanding Readers' Emotions with Deep Learning," Journal of Big Data, vol. 9, art. no. 614, 2022, doi: 10.1186/s40537- 022-00614-2
13. Bharat Gaind, Varun Syal, Sneha Padgalwar, Emotion Detection and Analysis on Social Media Global Journal of Engineering Science and Researches (ICRTCET-18) (2019) 78- 89.
14. P. S. Chaudhari, A. T. Zende, S. S. Deshpande, and S. M. Kulkarni, "Real Time Emotion Detection from Textual Data using Deep Learning," in Proc. of International Conference on Communication, Computing and Internet of Things (IC3IT), IEEE, Feb. 2021, pp. 248-253, doi: 10.1109/IC3IT51523.2021.9406690.
15. R. Rani and V. Gupta, "A Survey on Sentiment Analysis Techniques, Challenges and Applications," International Journal of Computer Applications, vol. 182, no. 18, pp. 33 40, Dec. 2018. Lin W et al.,
16. M. Oussous, F. Benjelloun, A. Ait Lahcen, and S. El Makki, "Sentiment analysis and its applications: A survey," Journal of Big Data, vol. 6, art. no. 84, Sep. 2019, doi: 10.1186/s40537-019-0248-5. Devlin J et al.,
17. Y. Xu and X. Li, "An Emotion Recognition Method Based on Attention Mechanism and Bidirectional Long Short- Term Memory Network for Text," Sensors, vol. 23, no. 16, p. 7114, Aug. 2023, doi: 10.3390/s23167114. Gontier C et al.,