Project - Data Stream Processing
Reddit Real Time Sentiment Analysis

Rémi Khoury - Morgane Brossard

23th January 2025

Abstract

Capturing trends on social media is a challenge difficult to address because of the difficulty to obtain real time data and to manage to process them efficiently. Reddit is a popular social media platform where users can share, discuss, and vote on content organized into topic-based communities called "subreddits". This project consists in the real-time streaming of comments on the social media Reddit, their sentiment analysis, and the implementation of a dashboard of several visualizations in an interface. The goal is to produce an intuitive platform for users to explore and capture the overall sentiment tendencies in Reddit comments in real time. Our method relies on an Apache Kafka pipeline, using the official Reddit API to get real-time comments on various topics. Each comment is processed through this pipeline to obtain both sentiment and topic information. This project implements a pre-trained topic classification model (from hugging-face), a pre-trained sentiment analysis model (from hugging-face too) and a sentiment analysis model created from scratch.

The code for this project is available at https://github.com/remik354/Reddit-real-time-sentiment-analysis.git.

Contents

1 Global pipeline

As the goal of the project is to perform real-time sentiment analysis, we used Apache Kafka to build a robust and scalable data streaming pipeline. Indeed, Kafka enables collection, processing and transfer of data in real time through *kafka topics*.

Our pipeline works as follows:
- get data from the official Reddit API site through the subreddit 'all', where all the comments posted are accessible
- send them into a first *kafka topic* where they are directed into the NLP processing unit
- after NLP processing, the modified comments are sent into another *kafka topics*, while an archive file is fulfilled
- the dash board uses both the archive file and *kafka topic* to plot the real-time data stream

The figure below summarizes the project pipeline.

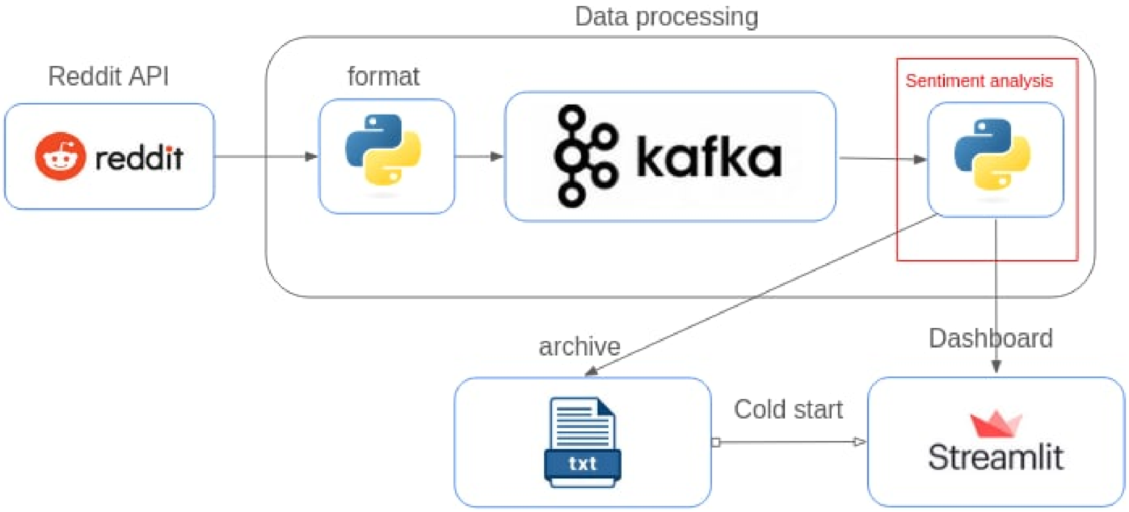


Figure 1: Project pipeline

During the project we worked with a shared Github repository, to allow to collaborate on code with asynchronous contributions and manage different versions of the project. We created a virtual environment to facilitate the set-up of necessary libraries for any user to be able to quickly compile the project (all information in the Readme file in the Github).

2 Real time streaming of Reddits comments in Kafka

Before going further let's discuss how Reddit works. Reddit is in fact a collection of forums, each called a subreddit, dedicated to specific topics (from news and politics to video games or fruit harvesting). Users can post either a 'Topic' (where they ask a question, state an opinion or make a joke) or a 'comment' (posted under Topics to react to the original subject). A topic is posted under a 'subreddit' which usually deals with a specific subject. For example, the subreddit 'r/politics' contains a lot of topics on political opinion that themselves are commented by users.

2.1 Reddit API

We initially wanted to stream Tweets with the Twitter API, however we could only stream very few tweets at a time, that did not allow for the real-time aspect that is a key element of this project. Therefore, we decided to use the Reddit API that had much less limitations in the quantity of content to stream.
Indeed, the Reddit API is very accessible and has its own python library to help people use it ($'praw'$). With just an API key, we are able to send 1 request every second (meaning 60 per minutes) with very few limitations on it. We can get specific data from a given subreddit, with all the information needed (number of up/down votes, timestamp, username, topic, ...). This makes this API very interesting and easy to work on.

2.2 Kafka pipeline

Let's dive a little more into our kafka pipeline. The data we get from the Reddit API is the comments themselves, from the subreddit 'all'. The comments are formatted into a json with these following information :
- "author": the username of the author
- "body": the comment itself
- "subreddit": the name of the subreddit the comment was posted on
- "created-at": the datetime of the post
- "timestamp": the timestamp of the post
- "topic-title": the topic title, composed of the name of the subreddit and the name of the topic where the comment is posted

These comments are then sent into a first kafka topic: reddit-topic.

From here, the comments are treated by the NLP pipeline and two keys are added to the json of each comment :
- "sentiment": the sentiment score of the comment
- "category": the topic category the commment belongs to

After being treated, the comment is sent to another topic, reddit-transformed and then used for the dashboard and the archive file.

3 Natural Language Processing

We used two approaches for the Sentiment Analysis part : Using a pre-trained open-source model (producing state-of-the-art performance), and implementing our own model, to compare results. Both models were trained on tweets, as more extensive databases are available than for Reddit comments. In addition to sentiment analysis, we also implemented a pretrained topic classification model. This model enabled us to classify each comment into one of 5 categories.

3.1 Pre-trained state-of-the-art model

For the topic classification, the BART Large MNLI model from Hugging Face was used. This model is transformer-based and designed for natural language inference (NLI), enabling it to classify or infer relationships between text inputs. It outputs confidence scores for multiple categories, making it versatile for tasks like zero-shot classification and text understanding. We defined 5 categories in which every comment is put into. They are: Politics, Technology, Entertainment, Finance and Health.

For the pre-trained approach, the Cardiff NLP "twitter-roberta-base-sentiment" model was used. This model is transformer-based and specifically designed for sentiment analysis on text data, and outputs a confidence score for each sentiment classification.

3.2 Manually-crafted model

The sentiment analysis model from scratch was developed based on the sentiment140 dataset. We used a representaiton learning approach, using Singular Value Decomposition (SVD). Several steps where necessary to build such a model :

- **Training dataset**: The sentiment analysis was performed using an extract of the Sentiment140 dataset (100.000 labeled tweets). The data was pre-processed with lowercase conversion, removing special characters, URLs and stopwords, and tokenizing the text
- **Vocabulary and co-occurence matrix**: A vocabulary of unique words was built from the data and a co-occurence matrix was created based on a context-window of 5 words around each token
- **Positive Pointwise Mutual Information (PPMI)**: The co-occurence matrix was transformed into a PPMI matrix to emphasize meaningful word associations
- **Dimensionality reduction with Singular Value Decomposition (SVD)**: To reduce complexity, we applied SVD to the PPMI matrix obtained to extract embbeddings of words
- **Sentence representation**: Each sentence in tweets wad represented as the average of its word embeddings, providing a numerical representation of the text
- **Classification mode**: We used Logistic Regression on the SVD-transformed word embeddings

The table below presents the accuracy results of the model on sentiment140 validation data, which was around 40000 twitter posts.

Sentiment	**Precision**	**Recall**	**F1-Score**
Negative	0.74	0.73	0.73
Positive	0.73	0.75	0.74

Table 1: Classification report of the model on Sentiment140 validation data.

4 Visualization dashboard

We chose to add a visualization aspect to this project, that is very interesting to combine with real-time streaming, to create an auto-updating dashboard.

4.1 Implementation with Streamlit and Kafka

The Streamlit library allows to implement customizable dashboards with simple syntax, no front-end programming, and supports dynamic updates of data. Moreover, it easily integrates with Kafka. We created a Kafka consumer that listens to a topic and retrieves new comments after their sentiment analysis.

4.2 Structure and content of the dashboard

The dashboard contains several plots and tables, auto-updating with the comments stream received from a Kafka topic:

– **Comments table** displaying the comments with their metadata (figure 3)

– **Sentiment chart** displaying the evolution of average sentiment in each category over time (figure 2)

– **Bar chart** of the current average sentiment for each category (figure 4)

- **Keyword searchbar** allowing the user to limit the display to comments containing a specific keyword, adapting the different plots and the table (figure 5)

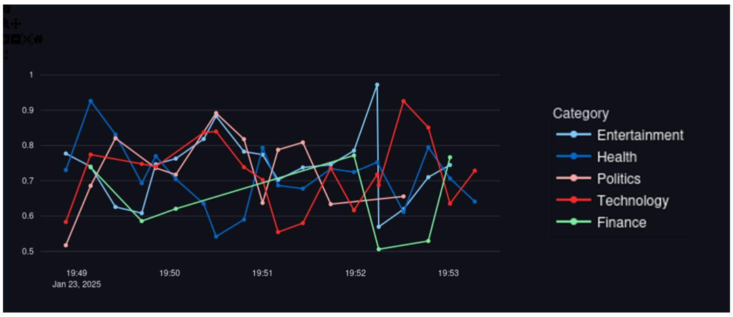
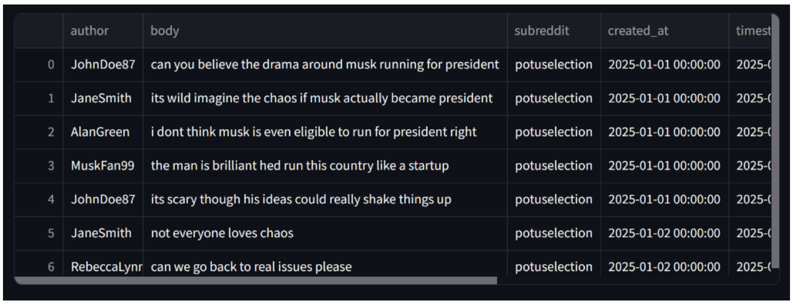


Figure 2: Sentiment over time for categories



	author	body	subreddit	created_at	timest
0	JohnDoe87	can you believe the drama around musk running for president	potuselection	2025-01-01 00:00:00	2025-(
1	JaneSmith	its wild imagine the chaos if musk actually became president	potuselection	2025-01-01 00:00:00	2025-(
2	AlanGreen	i dont think musk is even eligible to run for president right	potuselection	2025-01-01 00:00:00	2025-(
3	MuskFan99	the man is brilliant hed run this country like a startup	potuselection	2025-01-01 00:00:00	2025-(
4	JohnDoe87	its scary though his ideas could really shake things up	potuselection	2025-01-01 00:00:00	2025-(
5	JaneSmith	not everyone loves chaos	potuselection	2025-01-02 00:00:00	2025-(
6	RebeccaLynr	can we go back to real issues please	potuselection	2025-01-02 00:00:00	2025-(

Figure 3: Comments table

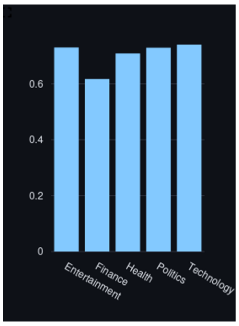
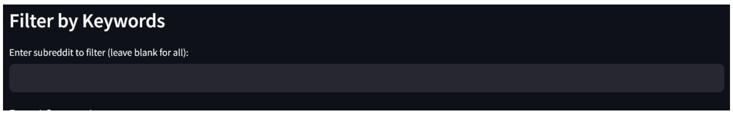


Figure 4: Current sentiment for categories



Filter by Keywords

Enter subreddit to filter (leave blank for all):

Figure 5: Keywords searchbar

5 Conclusion and discussion

5.1 Overall conclusion

In this project, we successfully implemented a real-time sentiment analysis on Reddit comments, with a display in an interactive dashboard. The Reddit API and our Apache Kafka pipeline provide a steady data flow that enables real-time insights based on our sentiment model in the Streamlit dashboard. This work demonstrates the potential of real-time analysis to capture social media dynamics.

5.2 Ideas for improvement

Sentiment analysis:

For the training of our own model for sentiment analysis, the next step would be to include a testing phase to validate the model's performance. For the overall analysis approach, an improvement would be to implement a trend analysis for more complex insights into the sentiment over time. Finally, adding Topic Detection to the pipeline would complement the sentiment analysis to provide even more insights on the Reddit trends.

Dashboard:
The immediate improvements easy to implement with more time would be to add other charts, such as a radar chart for the visualization of overall sentiment in categories in a more intuitive way than a simple bar chart. The current dashboard allows for keyword search to filter comments, a next step would be to add an advanced search functionality to limit the display to comments belonging to a specific subreddit. If topic detection is implemented, an improvement would be to display a word cloud of the main topics discussed in a subreddit.

6 References

[1] Apache kafka: https://kafka.apache.org/

[2] Reddit Api: https://www.reddit.com/dev/api/

[3] Twitter-roBERTa-base for Sentiment Analysis: https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment

[4] bart-large-mnli: https://huggingface.co/facebook/bart-large-mnli

[5] Sentiment analysis using Singular Value Decomposition, Veena Dubey and Dharmendra Lal Gupta: https://inpressco.com/sentiment-analysis-using-singular-value-decomposition/