Faculty of Mathematics and Information Science
Warsaw University of Technology

Big Data Analytics: Methods and Systems

The Sentinel Project

Marcin Dardziński, Paweł Rzepiński, Ryszard Szymański,
Sebastian Sowik

Contents

1 Introduction

Nowadays, the vast majority of discussions take place on the Internet, especially in social medias. Consumers are now equipped with easily accessible platforms to share and amplify their views. The connected nature of the Internet allows for the rapid spreading of opinions. Companies which value their brand highly must take into account the importance of this communication channel. Online Reputation Management (ORM) systems allow a company to track mentions of itself, their product and staff or even competitors. Automatic trends detection and mentions' reach assessment are invaluable in terms of preventing public relation crises or evaluating the impact of marketing campaigns.

1.1 Project description

The Sentinel project provides a dashboard presenting insights into volume and quality of the conversation towards followed brand, products or services in time-aware manner. The most important features include:

1. Tracking keywords appearances – real-time monitoring of occurrences of selected keywords or their variations on the Internet. Keywords can include, but are not limited to: brand name, key products or events, most important staff members, company-owned domains.

2. Evaluating sentiment expressed in the text – numerical summary of emotions taking into account keywords and evaluated automatically.

3. Assessing author's reach and credibility – taking into account not only the opinion, but also its' possible impact by considering author's reputation and influence e.g. in the number of followers or rating.

4. Listing of mentions – direct links and content of all of the mentions; to allow for quick response from the PR department.

Most of the before-mentioned features are visualised in form of a live dashboard containing:

1. Mentions count in time, grouped by text sentiment

2. Words most commonly associated with followed keywords

3. Current sentiment towards keywords broken into negative, neutral, positive categories

2 System overview

2.1 Data overview

The vast majority of processed data is text-based and features natural language – posts, comments, tweets, news headlines and articles. Additionally, in order to asses the author's credibility and mention reach some numerical values in form of number of comments, likes or rating are taken into account.

Data sources used in this project have different characteristics. Sources differ in terms of data quality, quantity and apart from common features of mention, like text, source url, creation date, have different metadata specific to the given source.

After downloading the data was transformed to a unified format to facilitate further processing. The format of the single unified entry is provided in the Table 1.

Attribute	Description
Text	Content of the comments, tweet or article.
Url	Url of the specific mention.
Origin Date	The data at which the mention was submitted to the specific source.
Download Date	The date at which the mention was downloaded by the system.
Source	The source of the downloaded mention
Metadata	Mention metadata dependent on the specific source.

Table 1: Unified format model

Mention's metadata varied across sources, to include platform specific information. Description of collected metadata is provided in tables 2 - 5.

Attribute	Description
Author	Hacker News ID of an article author
Points	Number of points
Relevancy score	Rank of an article

Table 2: Hacker News mention metadata

Attribute	Description
Author	Article author
News source	Url of the original article

Table 3: Google News mention metadata

Attribute	Description
Redditor	Id of the author
Redditor Comment Karma	Karma (score) of the author related to their comments
Redditor Link Karma	Karma (score) of the author reated to their submissions
Score	Numer of points comment received
Submission	Id of a submission (thread) the comment was added in

Table 4: Reddit mention metadata

Attribute	Description
User Id	Id of the author
Followers	Total author followers
Statutes	Total number of author tweets
Friends	Total author friends
Verified	If the author account is verified (e.g. public persona account)
Listed Count	Total tweet views
Retweets Count	Total retweets

Table 5: Twitter mention metadata

2.2 Data quantity analysis

A difference between the expected and predicted amount of collected mentions has been noticed. This is caused by the fact that free tier access to the exposed APIs is used which comes with several restrictions. For example, in case of Google News a free tier user is only allowed to download a maximum of 100 results, which significantly reduces the amount of available data.

To summarise, Twitter generates the highest number of messages with an average of 1000 tweets per minute, Reddit sends around 130 comments per minute. Hacker News and Google News are sources focused on longer articles have lower frequencies of incoming data with an average of a 5 and 2 articles per minute.

Number of incoming messages from the various sources can be correlated with the length of a single message sent from the given source. Twitter and Reddit have shorter messages with the average length of 100 and 150 characters on average respectively. Articles appearing on Google News and Hacker News are longer, 2000 and 400 characters respectively.

A fluctuation in the quantity of data dependant on time has been observed. In case of Twitter, users tend to be most active between 4pm and 10pm UTC as depicted in Figure 1. This is probably caused by the fact that most of world's population is up during this time interval. Moreover, it is also worth noticing that there is always traffic on Twitter – it never goes below 500 tweets per minute.

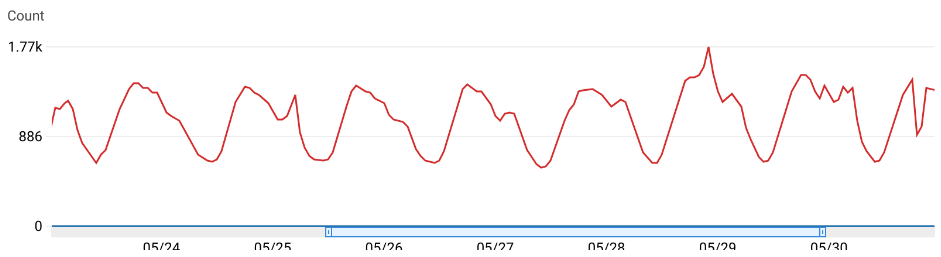

Figure 1: Twitter data quantity over time

Reddit data has a similar characteristic as Twitter, however the trend is not as fixed as in case of Twitter as observed in Figure 2.

When it comes to Hacker News articles, the largest amount of traffic is observed around 4pm. However, the amount of mentions varies as shown in Figure 3, extreme cases occur as well – either 1 mention or 14 mentions per minute. It is also worth to notice that during those extremes also occur in period, for example the traffic was lower

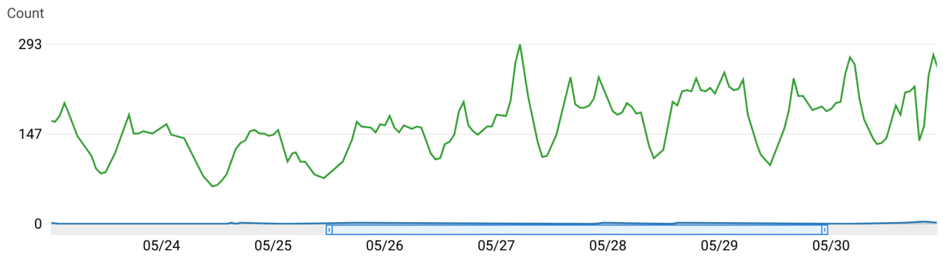

Figure 2: Reddit data quantity over time

between the 25th and 27th of may, but it spiked suddenly between the 28th and 30th of may.

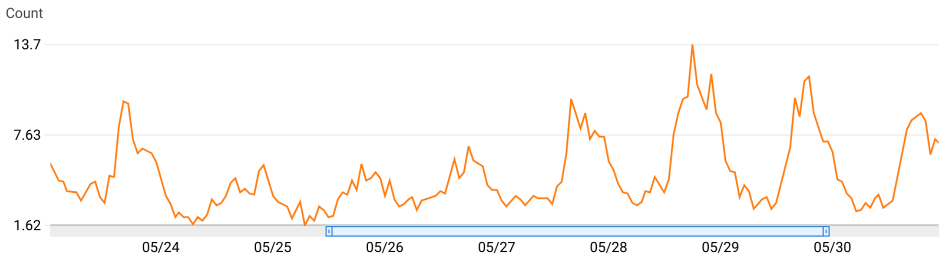

Figure 3: Hacker News data quantity over time

In case of Google News the data quantity over time is the most chaotic as presented in Figure 4. There is no significant trend observed. This is caused by the fact that the amount of articles per minute never exceed 5 and a small difference results in a significant change on the plot.

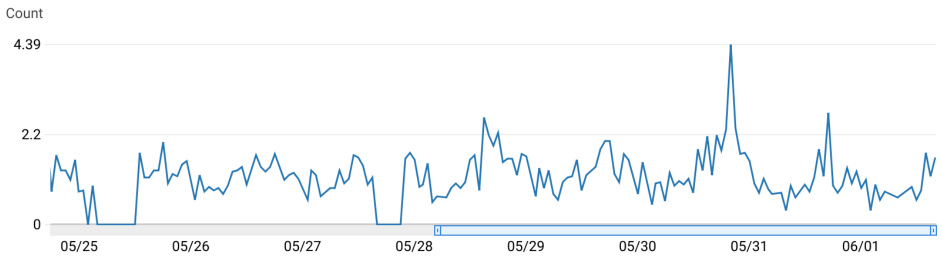

Figure 4: Google News data quantity over time

2.2.1 Sentiment analysis

In order to perform initial data and sentiment analysis sample of 35479, 20151, 712, 160 mentions were collected from Twitter, Reddit, Hacker News and Google News. The preliminary inspection of the text data revealed features of texts specific to each source. Texts from Twitter often include special unicode characters in forms of emojis, words reflecting the subject matter are often preceded with the hash symbol and user names are preceded with the at symbol. Similarly users of Reddit often use emojis, user's mentions in texts have special syntax and the discussion subject is often mentioned as a subreddit name. Other than emojis Hacker News articles contain html tags. Google News texts and texts from all other sources contain url links.

The model is pretrained on movie reviews from TextBlob library[1] was used for sentiment analysis. Histograms of the sentiment value from the collected data are presented on Figures 5, 6, 8 and 7.

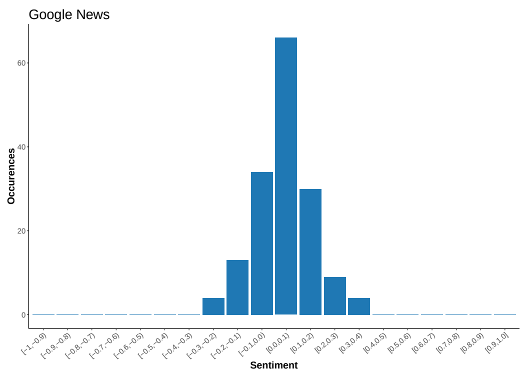

Figure 5: Google News articles sample's sentiment

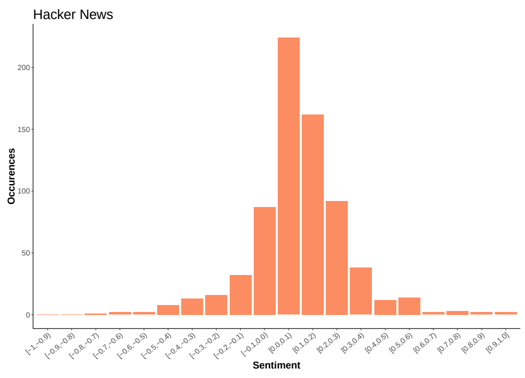

Figure 6: Hacker News articles sample's sentiment

Histograms show that Google News and Hacker News texts sentiment is more balanced and symmetrical, as expected from information articles. Social services have more imbalanced distributions of sentiment, especially tweets have a lot of outliers in higher quintiles of sentiment which is unexpected.

[1] https://textblob.readthedocs.io/en/dev/index.html

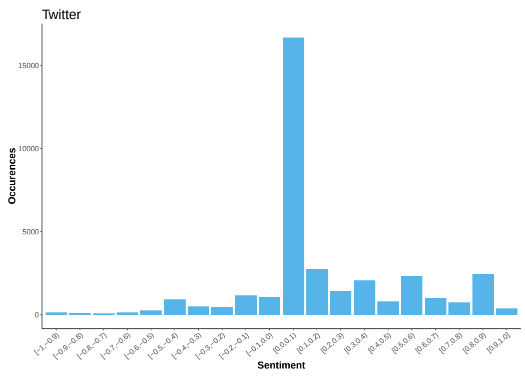

Figure 7: Twitter tweets sample's sentiment

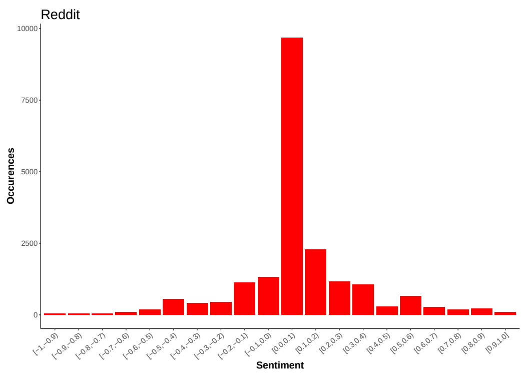

Figure 8: Reddit comments sample's sentiment

2.3 System architecture

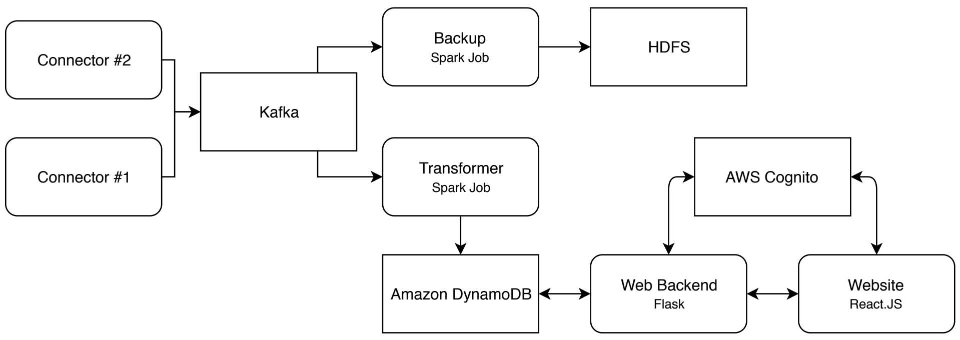

Figure 9: System's architecture overview

In order to provide scalability of the whole system every part of it has to be scalable, preferably horizontally. Connectors can be placed on separate, high end machines(vertical scaling) and multiple instances can consume different part of data source(subreddits in case of Reddit, georegions in Twitter). The Kafka message broker can be placed on the cluster of multiple machines to increase throughput. The Spark jobs are assigned to the specific data source to loosen impact of incoming messages burst on any of them. Moreover, due to built-in partitioning Spark can handle varying volume of data. DynamoDB is a partitioned NoSQL database, thus providing every means necessary to scale read and write operations.

2.3.1 Connectors

Connectors are the modules responsible for collecting the data from the sources. Each of the connectors is listening to the stream of messages – text data feed from it's source. Received texts are compared against the set of observed keywords and detected mentions are converted from json responses to the unified format. During the conversion a validation of the received data is performed e.g. checking if the mention url is constructed properly. This makes sure that all of the collected data is usable and appropriate for further usage. Later, mentions are published then to corresponding Kafka topics for further analysis.

For performance reasons there is no further preprocessing of mentions in the connectors. Moreover, keyword finder detects only first match of any keyword to speed up processing.

The acora[1] package was used for efficient keyword detection – it provides an implementation of the Aho–Corasick algorithm[3]. This algorithm is capable of detecting multiple keywords at once and processing text characters only once without backtracking. The algorithm works by building minimal finite-state machine for a given keywords set. The build automaton highly resembles a triee with additional connections. Use of acora functionality that allows to break algorithm operation on the first occurrence of keyword provided efficient way of detecting interesting texts.

Connectors were developed as Python scripts to provide easier maintenance in case of API changes. As the system performs stream processing it is crucial to make the system immune to errors as breakage of a single component affects the whole pipeline. In order to guarantee the continuance of data streams and resilience to anomalies in the incoming data a rich logging system and robust error handling have been designed.

Connectors are gathering logs in daily-rotating file sinks stored locally on the machine. Entries with higher severity levels (warn and above) are also published to AWS CloudWatch logs. Custom metrics describing incoming data – items and hits counts – are periodically gathered and forwarded to CloudWatch by a service running on another thread to avoid additional load on connectors thread and prevent time drift. Such data is later displayed on the dashboard and used as a trigger for alarms regarding stream health, so appropriate actions can be taken as quickly as possible.

2.3.2 Transformer Spark Job

Spark jobs are responsible for the main processing of the data including text processing, find all keywords in text and converting mentions to database entities.

Texts from mentions objects are processed in separate Spark jobs leveraging source characteristics. Spark job for Twitter and Reddit removes emojis, links, user mentions. Removing hash symbols from twitter texts was abandoned due to providing possibility of following specific hashtags with users specifying keywords in an appropriate form. The processing of Google News and Hacker News sources removes html tags and urls. After text preprocessing mentions are enriched with text sentiment value computed in every Spark job.

Data conversion from JSON to unified format and text cleaning are performed using `map` operations for maximum performance. Downloading and detecting keywords in mentions is performed using `mapPartitions` operation to avoid sending HTTP request for every single data entry.

Keyword occurrence detection was implemented with the nltk package[2] providing support for multi-word expression tokenizer. This solution allows to leverage handling of white characters provided by the tokenizer and find all keywords in given text. Tokenizer allows for finding all single and multi-word keywords in efficient manner. Optimal efficiency is provided by building trie of tokens in initial step of algorithm and then performing fast lookup of keywords.

2.3.3 Backup Spark Job

Possibility of introducing changes to processing pipeline or sentiment analysis models without loosing current application state has been assured with constant asynchronous backup of raw data coming through system. Saved raw data can be processed through new versions of pipeline after implementing improvements to the pipeline or used to perform historical data characteristics analysis (e.g. throughput or the size of single item).

Kafka consumer – Spark Job is responsible for reading data from stream in batches of appropriate sizes and saving it to HDFS.

2.3.4 Dashboard

The Dashboard provides users with the possibility of exploring the collected data by exposing interactive visualisations. The web application is composed of a website written in ReactJS[2] and a backend application developed in Flask[3]. The Flask server exposes endpoints providing summarised data in the data format required by Plotly.js[4] compo-

[2] `https://reactjs.org/`
[3] `http://flask.pocoo.org/`
[4] `https://plot.ly/javascript/`

nents. The backend retrieves data from DynamoDB, performs calculations and then converts the results into the plot data format. Additionally, the web application allows the user to add, delete and edit the set of tracked keywords.

In order to enable a multi-user scenario and guarantee the security of the users Amazon Cognito is used for authorization and access control. Moreover, it is scalable and offers additional features such as multi-factor authentication and integration with social and enterprise identity providers.

3 Case study

In order to test the credibility of the tool an analysis of the retrieved for the keyword *Chernobyl* was made.

Chernobyl is five-part historical drama television miniseries about the nuclear plant disaster which occurred in Soviet Ukraine in April 1986.

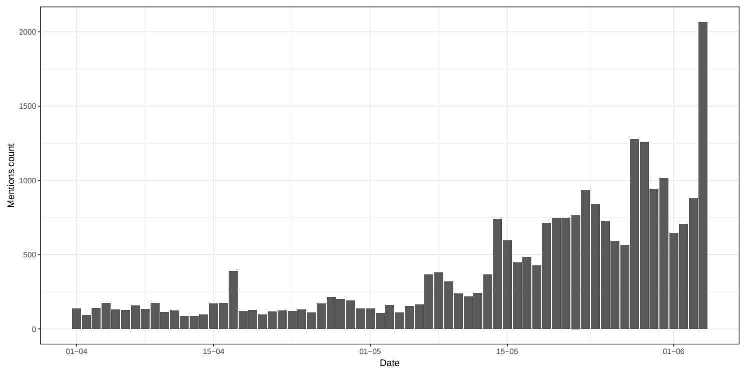

Figure 10: The number of mentions of "Chernobyl" in Reddit over time.

In Figure 10 the number of collected mentions regarding Chernobyl over time is presented. A significant burst in popularity can be observed after the first week of May. This phenomenon was caused by the release of the first episode of the TV series – the 6th of May. Peaks can also be observed in days when consecutive episodes were release – 6th of May, 13th of May, 20th of May, 27th May and the 3rd of June. Moreover, on the 3rd of June the final episode of Chernobyl was emitted, that is when the highest amount of mentions was observed.

The average sentiment per minute for this TV series is presented in Figure 11. The trend is clearly visible – positive emotions were more and more expressed towards the end of the series. Moreover, the most heated debated was also taking place in the end of May where the averages are more extreme.

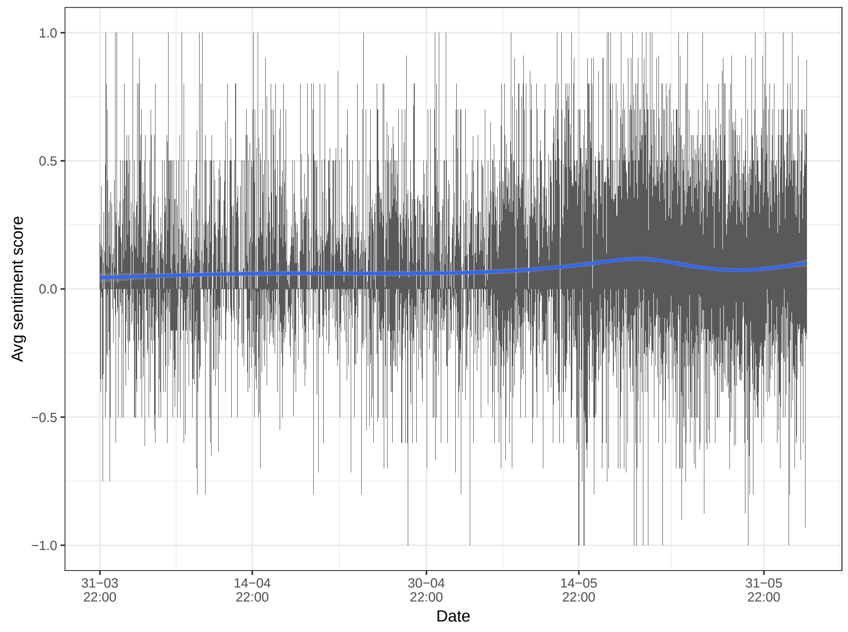

Figure 11: The average sentiment score of mentions regarding "Chernobyl" in Reddit over time.

4 Summary

All components of the system were developed successfully. The main feature – real-time monitoring of dynamic set of keywords was delivered alongside with basic visualisation.

The deployed application was running continuously for several weeks and endured multiple edge-case scenarios. We were able to continuously monitor the application with alerts for periods with no incoming data. Data anomalies such as missing required fields or incorrect data formatting did not lead to any process disruptions. The performance of data processing was satisfying – we have not yet observe a single occurrence of resources strain.

Due to time constraints and characteristics differences between sources we were not able to implement a unified method for assessing author's reach and credibility, although most of the required information is already gathered in metadata column.

Overall, delivered features and reliability of the system already allows for the production usage of the system as online reputation monitoring tool.

A Work division

The work division between team members is listed in Table 6 and Table ??. All programming tasks were peer-reviewed using GitHub platform. The sections in report were also subject to review.

Main contributor	Project
Marcin Dardziński	Reddit API analysis, integration and tests
	Web application development
	Unified format design
	Database schema design
	Amazon DynamoDB & Amazon Cognito integrations
Paweł Rzepiński	Architecture design
	Unified format design
	Natural language processing and sentiment analysis
	Integration of Kafka & Spark to the main pipeline
	Integration tests
Sebastian Sowik	Twitter API analysis, integration and tests
	Keywords search algorithms
	HDFS integration
	Natural language processing and sentiment analysis
	Integration of Kafka & Spark to the main pipeline
Ryszard Szymański	Hacker News API analysis, integration and tests
	Google News API analysis, integration and tests
	DevOps: CI pipelines
	Web application development

Table 6: The work division in the project.

Main contributor	Report
Marcin Dardziński	System Architecture
	Web backend & Website development
	Data overview
Paweł Rzepiński	Introduction
	Project description
	Connectors
	System overview
	Transformer Spark Job
	Summary
Sebastian Sowik	Data overview
	Data analysis
	Sentiment Analysis
	Connectors
	Backup Spark Job
Ryszard Szymański	System Architecture
	Web backend & Website development
	Data quantity analysis
	Dashboard

Table 7: The work division in report.

References

1. S. Behnel. *acora package*. URL: https://pypi.org/project/acora/.
2. J. Nothman. *nltk tokenize package*. URL: http://www.nltk.org/api/nltk.tokenize.html#module-nltk.tokenize.mwe.
3. J. Rejeb and M. Srinivasan. “Extension of Aho-Corasick Algorithm to Detect Injection Attacks”. In: *SCSS*. 2007.