Real-Time Social Media Spike Detection & Topic Spike Analyzer

Matheus Rama Amorim Michael Rivera Felipe Villegas

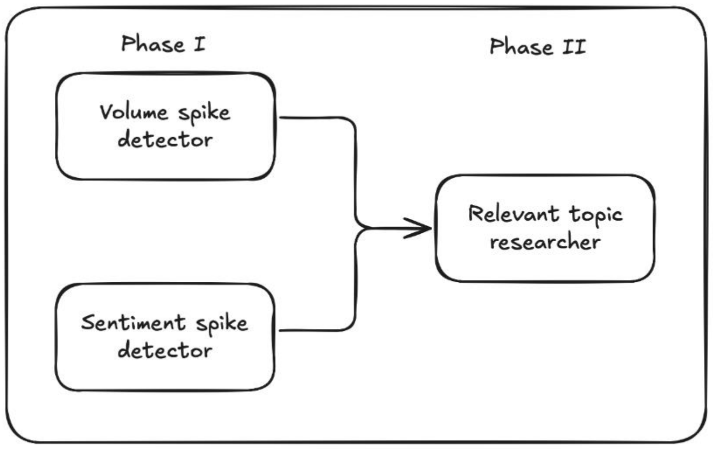
Abstract— This report introduces a unified system for real-time detection and explanation of unexpected spikes in social media interactions. The system combines a volume spike detector, a negative-sentiment spike detector, and a topic spike analyzer to identify sudden changes in user interactions and uncover the themes driving them. Using Prophet-based time-series models and proxy labeling strategies, we evaluate performance through recall, precision, and multi-fold validation in the absence of ground-truth labels. A historically trained BERTopic model provides interpretable, event-aligned explanations for each spike. Together, these components offer marketing teams timely alerts and actionable insights, with recommendations for improving responsiveness and future integration.[1]

1 INTRODUCTION

Digital presence plays a central role in marketing operations for many companies around the world. An integral part of digital marketing efforts is monitoring social media for indication of potential spikes in user interactions. Companies that can react timely to these crises have an advantage over their competitors, as they can reduce the reputational damage from those crises and get the most out of the opportunities derived from attention booms.

[1] This project was developed as part of a graduate practicum in collaboration with a social media management platform. The version shared here was approved for academic submission and does not include proprietary or confidential information.

Our system is intended to help marketing teams detect spikes from the moment they emerge and equip decision makers with the right information to react appropriately. This system is composed of a detection phase in which two models are used to identify unexpected changes in the interaction with the account, and



research phase in which a model is used to identify the key topics that definethe spike.

Figure 1—System Structure

Sections 2 to 4 will explore in depth the definition and results of each subcomponent, and section 5 will explore some recommendations and future work.

2 VOLUME SPIKE DETECTOR

2.1 Model Description

The volume spike detector subcomponent attempts to detect unusual patterns in the volume of interaction[2] with the account, this detection involves multiple steps that will be enunciated in this section, and the evaluation criteria and results will be exposed in sections 2.2 and 2.3 respectively.

2.1.1 *Temporal Granularity*

Capturing the unexpected patterns as soon as possible is critical for marketing teams to react appropriately; this requirement has direct consequences on the time granularity used in model training and its further predictions in production.

As part of the pre-processing step, interaction volumes are grouped into 15-minute windows. This resolution captures changes more quickly than hourly windows while still being aggregated enough to reveal clearer patterns than per-minute windows.

2.1.2 *Training Strategy*

The component architecture implies training a model to predict a baseline for the expected volume of interactions and compare the observations against it to identify unexpected spikes.

Interaction patterns evolve over time—for example, the Atlanta Falcons may experience different interaction levels during the season compared to the off-season. To account for these shifts, the baseline prediction for any point of time within week t is generated using a model trained on data from the rolling weeks $t-5$ to $t-1$.

The expected output of the model for any 15-minute window is the predicted volume ($\hat{y}$) and a confidence interval $\hat{y}_{\text{upper}}$ and $\hat{y}_{\text{lower}}$ for the upper and lower values of the interval, respectively.

[2]Interaction is defined as the number of direct messages, posts mentioning the account and shares of the account's posts

2.1.3 *Model*

The model is configured using Prophet with weekly and daily seasonality enabled, while yearly seasonality is intentionally disabled to avoid patterns that are irrelevant at this time scale. In addition to Prophet's built-in components, a custom 15-minute seasonal cycle is introduced to capture short-term fluctuations in interaction volume. This component is defined with a period of one day, split into 96 quarter-hour segments, and modeled using a Fourier order of 8 to provide sufficient flexibility. The additive mode ensures that this seasonal pattern contributes directly to the baseline level, and the prior scale of 10 allows the model to express this variation without being overly constrained. The interval width parameter is set to 0.8

2.1.4 *Alert Trigger*

There are two types of alerts that are triggered based on a set of rules:

- Orange Aler: Triggered when the volume observed (y) in at least two out of the last four windows is greater than $\hat{y}_{upper.}$
- Red Alert: The criteria for orange alert is met and: $(y - \hat{y}_{upper}) > 2 \cdot (\hat{y}_{upper} - \hat{y})$

2.2 Evaluation Strategy

Evaluating anomaly detection algorithms is inherently challenging due to the lack of ground-truth labels. In such cases, common alternatives include stability metrics under controlled perturbations, reconstruction error for encoder-based models, or drift-sensitivity assessments. However, these approaches are only indirect proxies for the core objective of identifying anomalous behavior. To mitigate this limitation, we constructed a proxy variable to enable pseudo-labeling. Specifically, we calculated the 99th percentile within rolling five-week windows and designated interaction volumes exceeding this threshold as unexpected spikes.

This pseudo-labeling strategy enabled us to conduct two complementary analyses to assess the model's performance.

Recall Analysis

The purpose of the recall analysis is to quantify the proportion of spikes that the model successfully captures. This metric is particularly important for marketing teams, as missing relevant spikes may delay response efforts and negatively impact brand reputation. For this analysis, recall is defined as:

$$Recall = \frac{(\alpha \cap \sigma)}{\sigma}$$

Where:

$$\alpha = Number\ of\ alerts$$

$\sigma = Number\ of\ data\ points\ above\ 99th\ percentile$ [3]

This definition can be applied consistently to both orange-level and red-level alerts.

Precision Analysis

The objective of the precision analysis is to determine the proportion of alerts that correspond to an actual spike. This metric provides insight into the level of noise introduced by the model's predictions. High precision is essential for effective adoption, as an overly sensitive model may generate excessive false positives and undermine stakeholder confidence in its outputs. Precision is defined as:

$$precision = \frac{(\alpha \cap \xi)}{\alpha}$$

[3] α & σ are defined as number of data points in 15-minute windows as exposed in section 2.1

It's important to note that the model should not only detect all spikes accurately but also detect them as quickly as possible. By defining σ as in the recall analysis, the evaluation penalizes the model for detecting the spikes too early. We introduce a new parameter ξ, defined as the number of data points that lie above $\hat{y}_{\text{upper}}$ and are connected by continuity to an unexpected spike (according to the definition of points exceeding the 99th percentile). This concept is illustrated in Figure 2.

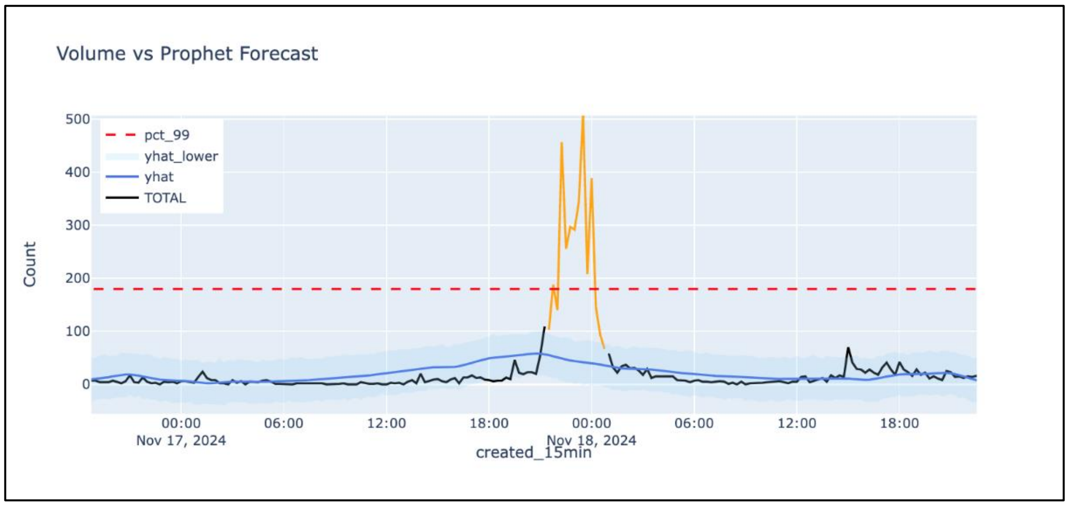


Figure 2—Spike definition

This strategy enables the measurement of precision without penalizing the model for timely spike detection. This definition can be applied consistently to both orange-level and red-level alerts.

2.3 Results

2.3.1 *Recall*

Table 1 illustrates the results of the recall analysis under the conditions exposed in section 2.2. Waffle house showed the best performance consistently across the orange and the red alerts with 78.6% and 67.10%, in the other hand, MARTA had the lowest scores with 59.5% for the orange alerts and 56.2% for the red ones.

Table 1 —Recall

Dataset	Orange Alerts	Red Alerts	Orange -1 timelapse
Falcons	72.60%	55.20%	83.17%
Marta	59.50%	56.20%	78.14%
Waffle-House	78.60%	67.10%	87.00%

The *orange −1 timelapse* column isolates the impact of late alerts by treating them as correct detections. It does so by recalculating recall under the assumption that each alert was triggered one time-lapse earlier than it occurred. Under this assumption, the average recall increases by 12 percentage points, with the largest improvement observed for **Marta**, at 18.6 percentage points.

The magnitude of this effect indicates that interaction volumes often rise from normal levels to above the 99th percentile very rapidly. This highlights an opportunity to improve the algorithm's responsiveness and reduce delays in detection.

2.3.2 *Precision*

Table 2 presents the model's precision scores across the three datasets. The orange-alert precision values are relatively low, which is not inherently negative; these alerts may be capturing less severe spikes that do not exceed the 99th percentile. In this sense, the model behaves conservatively, flagging a broader range of activity as potential spikes.

It is noteworthy that precision improves substantially when considering only red alerts, with an average increase of 41.6 percentage points.

Despite the limitations of the evaluation approach used, the results provide a clear sense of the expected noise level in this implementation. Most alerts—particularly the red alerts—are associated with genuinely critical spikes.

Table 2 — Precision

Dataset	Orange Alerts	Red Alerts
Falcons	54.58%	82.05%
Marta	11.97%	59.17%
Waffle-House	27.88%	78.18%

3 NEGATIVE SENTIMENT SPIKE DETECTOR

3.1 Model Description

The negative sentiment spike detector model aims to bring a second layer of crisis detection to the system. Its goal is to identify crises that might not necessarily increase the volume of tweets but shift negatively to the sentiment expected towards a brand. In that way, our system can identify crises even if they don't come in the form of more people talking about a given brand, but in the form of people talking more negatively about a given brand.

The proxy used for sentiment is the proportion of negative tweets, using the sentiment label provided in the listening data. A time decomposition model was used to create a baseline of expected negative sentiment at any given time. That baseline is then compared to actual observed values, and alerts are triggered when there is a big discrepancy.

3.1.1 *Data Preparation*

Similarly to the volume spiker model, the data was aggregated on time windows for the negative sentiment model, using 1-hour windows instead of the 15-minute windows used in the volume model. The larger window is needed for the negative sentiment detector as it uses a derivative metric from total tweets and therefore needs more tweets in the time window to identify reliable patterns and yield good results.

An additional step on the data processing was created for this model, in which time windows were filtered based on whether they had enough tweets. This was determined by a parametrized threshold as well as a dynamic threshold calculated per hour of week. Any hour that had zero tweets were also removed from this data. This is a very important step so that the model only learns from data points that represent a general sentiment from a relevant number of tweets.

3.1.2 *Model, training and alert trigger*

The model used for the time decomposition was Prophet with weekly and daily seasonality enabled (in a multiplicative setting, which works better for proportions), and yearly seasonality disabled. No additional regressors were added. The output of the model is the expected percentage of negative tweets for a given hour, as well as a confidence interval, in a similar format to the training outputs of the volume spike model.

The model was trained using the whole time series. In the evaluation section, we will discuss why that was the case and not using a rolling window, similar to how the volume spike detection model was trained.

After training the model, it was used to construct the baseline of expected share of negative tweets for any given hour, and such baseline was compared to the actual share to identify anomalies. To be classified as an anomaly, a given hour needs to:

- Have been considered valid for the model, i.e. has enough tweets so that the share of negative ones is a good proxy for the overall sentiment on that window
- Have a z-score higher than 2.5, i.e. the actual observed share is deviated from the predicted baseline by 2.5 standard deviations.
- The standard deviation is calculated using the confidence interval outputted by the Prophet model, dividing the width of the interval by four (95% interval)

Anomalies are then identified as yellow alerts. Yellow alerts on their own are not raised to a crisis. To be considered a crisis, the system needs to observe a configurable number of yellow alerts in a configurable number of windows – for the three datasets the configuration was the same, three yellow alerts in a five-hour span. An orange alert can then be raised to a red alert if any of the z-scores in the orange alert window spike to over a configurable number (four in our test runs) or if there are more yellow alerts (configured for four in our test runs) in a configurable number of windows (configured for 6 in our test runs). Finally, a red alert can be triggered solely based on a z-score spike coming from an yellow alert.

3.2 Evaluation Strategy

Since there are no ground-truth labels in the data for what is a crisis and what is not, we had to find alternative methods of evaluating the model. If we assume that any anomaly found by our alert trigger process is an actual anomaly, we are then able to clean the data from any outliers and evaluate how the model performs on the remaining data points, which would be a proxy for normal behavior. We then tested the model using different processes as described below.

3.2.1 *Single fold, single training evaluation*

The first evaluation process was a single fold, single training evaluation. This means that we trained the model on the first 80% valid time windows, predicted the final 20%, marked anomalies on the test window, and evaluated the model on non-anomalous data points on the test window. The below table shows the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), the coverage (% of non-anomalous observations in the test set covered by the confidence interval of the model) and the mean absolute z-score. The results of this process were encouraging, with the model performing well on the Atlanta Falcons and Waffle House datasets. The MARTA dataset proved to be a challenge, because of the low volume: only 22% of the time windows were deemed valid for the model, while for the other two datasets that figure was between 57 and 77% for the Atlanta Falcons and Waffle House, respectively.

Table 3 — Evaluation results on single fold, single training sentiment spike detection

Dataset	RMSE	MAE	Coverage	Mean Absolute Z-Score
Atlanta Falcons	0.1342	0.1080	93.55%	0.8560
Waffle House	0.0995	0.0788	96.55%	0.7387
MARTA	0.2911	0.2457	97.27%	0.9107

3.2.2 *Single fold, iterative training evaluation*

The next approach that was tested was to iteratively remove anomalies from the training dataset and retrain the model. The process would work as follows:

1. Train the model on the first 80% of the valid time windows
2. Evaluate the model on the 20% remaining valid time windows, and annotate the model's results
3. Identify anomalies on the train dataset, remove those anomalies, and retrain the model
4. Repeat evaluation on the same test set, and compare results
5. Iterate steps 3 and 4 for a configurable number of times

In all three datasets, the iteration process was run 5 times, and the model improved its error figures, without losing coverage considerably. The below table shows the results. The Atlanta Falcons dataset's results improved the most, while the MARTA dataset's results improved only marginally on errors, while losing a lot of coverage and increase mean absolute z-score. This shows that some brands would benefit from this approach, while others might not, thus signaling that this process is not an absolute must-do for all brands.

Table 4 — Evaluation results on single fold, iterative training sentiment spike detection

Dataset	Final RMSE	Final MAE	Final Coverage	Final Mean Absolute Z-Score
Atlanta Falcons	0.1248	0.1001	93.48%	0.8497
Waffle House	0.0975	0.0776	95.31%	0.7873
MARTA	0.2904	0.2450	95.89%	0.9236

3.2.3 *Multiple fold, single training evaluation*

Finally, we also wished to understand if the model would be consistent through different slices of the timeframe, and how the model would perform with an expanding versus rolling training window. For this process, we did not apply any iterative training, as it was not shown to be an absolute truth that such technique would improve the model for every brand. We tested 20 folds for each dataset. The process starts by training the model on the valid observations for the first 12 weeks and predicting the baseline for the following week (week 13). Then, errors and coverage are annotated for week 13. The model then "slide" one week, using data from week 1 to 13 to train in the training expanding window case, or week 2 to 13 in the rolling window case, and both are used to predict the baseline of week 14 and evaluated on that.

Table 5 — Evaluation results on multi fold, expanding training window sentiment spike detection

Dataset	RMSE	MAE	Coverage

Atlanta Falcons	0.1355 ± 0.0253	0.1122 ± 0.0242	91.0% ± 7.6%
Waffle House	0.1059 ± 0.0075	0.0853 ± 0.0069	95.5% ± 2.1%
MARTA	0.2794 ± 0.0359	0.2352 ± 0.0355	95.4% ± 3.9%

Table 6 — Evaluation results on multi fold, rolling training window sentiment spike detection

Dataset	RMSE	MAE	Coverage
Atlanta Falcons	0.1381 ± 0.0296	0.1141 ± 0.0271	91.4% ± 7.1%
Waffle House	0.1073 ± 0.0085	0.0865 ± 0.0079	95.5% ± 2.1%
MARTA	0.2841 ± 0.0375	0.2371 ± 0.0351	94.8% ± 4.6%

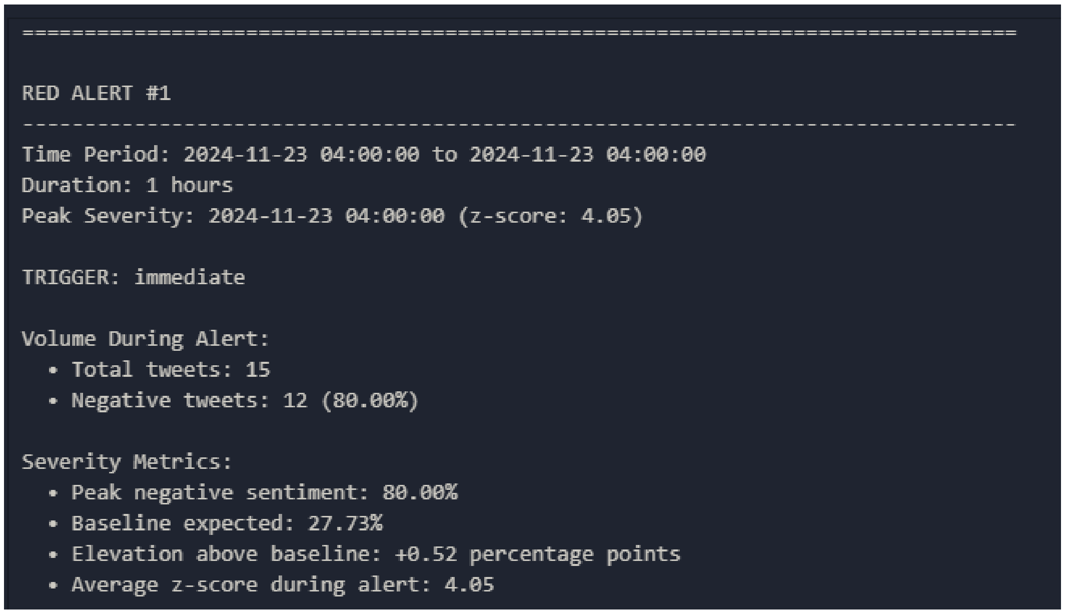
As can be seen in Table 3 and Table 4, the models are fairly consistent, except for coverage for the Atlanta Falcons. Still, the errors are not high for that dataset. Finally, comparing the expanding and rolling training windows, it can be observed that expanding windows perform slightly better, and because of that we decided to keep the full dataset when training the final model to showcase the anomalies and alerts that would be raised.

3.3 Model Outputs

The model was designed to create three outputs: a crisis report executive summary, a report per orange/red alerts, and a plot that shows how a dashboard could look like. In the sections below we will discuss what they are and how we envision they would be used in practice. For all of them, a mechanic was created to escalate orange alerts to red alerts and de-duplicate these alerts; this means that if a red alert happened in an orange alert window, that whole window will be classified and reported as a red alert, and that window will not show as an orange alert as well.

3.3.1 *Crisis report executive summary*

This executive summary is a simple report that states, for the time window analyzed, how many alerts happened. The below figure shows what this alert would look like. As the name suggests, this would be an alert fired in an email or through a messaging app once a week or once a month, for executives to have a general overview of crises during that time window.

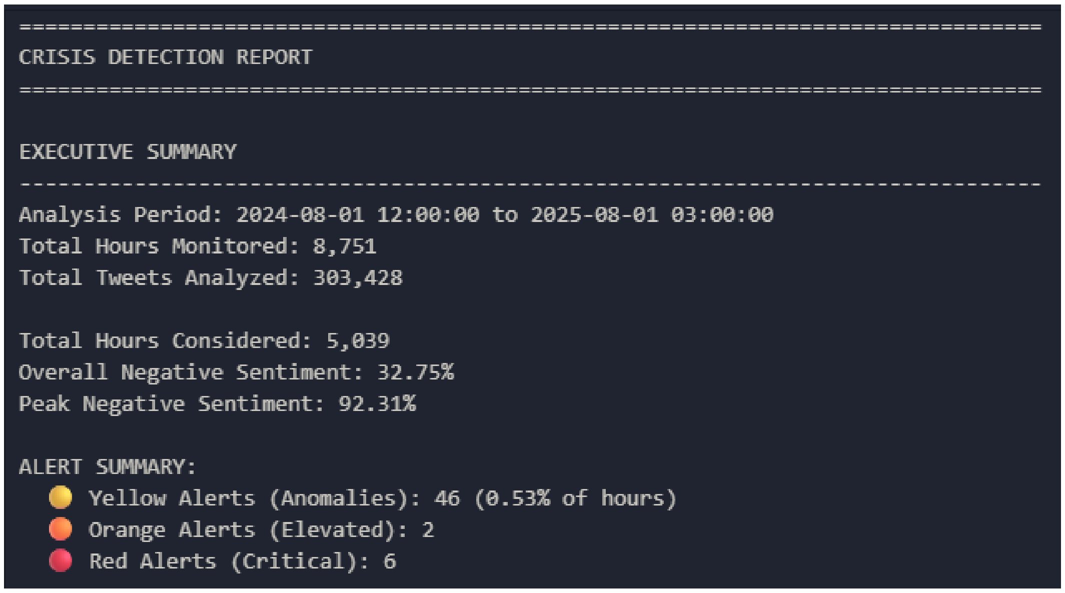


```
================================================================================

RED ALERT #1
--------------------------------------------------------------------------------
Time Period: 2024-11-23 04:00:00 to 2024-11-23 04:00:00
Duration: 1 hours
Peak Severity: 2024-11-23 04:00:00 (z-score: 4.05)

TRIGGER: immediate

Volume During Alert:
  • Total tweets: 15
  • Negative tweets: 12 (80.00%)

Severity Metrics:
  • Peak negative sentiment: 80.00%
  • Baseline expected: 27.73%
  • Elevation above baseline: +0.52 percentage points
  • Average z-score during alert: 4.05
```

Figure 3—Sentiment Spike Detector crisis report executive summary example

3.3.2 *Detailed alert report*

The detailed alert report will go more in depth on a given alert, either orange or red. This could be sent every time an alert is raised, maybe to the marketing team assigned contact, so that the team can act quickly in mitigating the crisis. Figure 2 examplifies what this alert could look like.



```
================================================================================
CRISIS DETECTION REPORT
================================================================================

EXECUTIVE SUMMARY
--------------------------------------------------------------------------------
Analysis Period: 2024-08-01 12:00:00 to 2025-08-01 03:00:00
Total Hours Monitored: 8,751
Total Tweets Analyzed: 303,428

Total Hours Considered: 5,039
Overall Negative Sentiment: 32.75%
Peak Negative Sentiment: 92.31%

ALERT SUMMARY:
  🟡 Yellow Alerts (Anomalies): 46 (0.53% of hours)
  🟠 Orange Alerts (Elevated): 2
  🔴 Red Alerts (Critical): 6
```

Figure 4—Sentiment Spike Detector detailed alert report example

3.4 Plots

The final output of the model's report method is a couple of plots that show the alerts identified in the period, and the time series decomposition. This could be very usefull to be provided in a dashboard, so marketing teams and executives could monitor crises as well as gain more insight into how people interact with their brands on social media. The two figures below show the two charts.

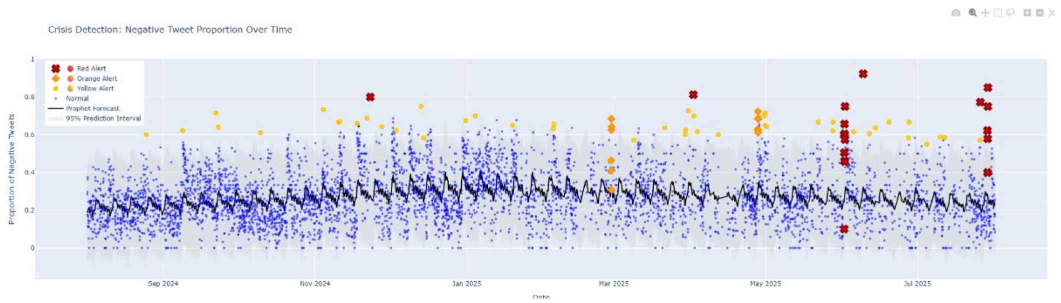


Figure 5—Plot showcasing yellow/orange/red alerts

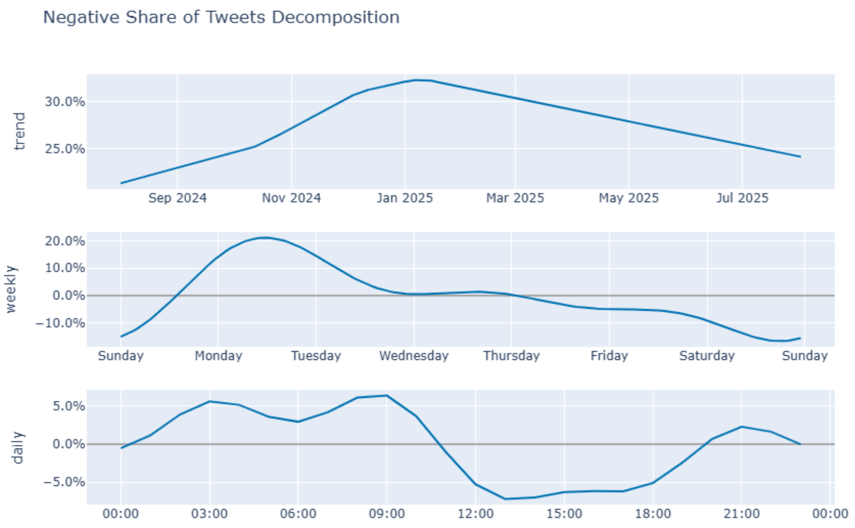


Figure 6—Time series decomposition plot

4 TOPIC SPIKE ANALYZER

4.1 Model Description

The topic spike analyzer is the final major component of the system. While the volume and sentiment detector models identify when unexpected activity emerges, this module explains why it occurs by highlighting the themes, complaints, or conversations driving the anomaly. The goal is to give marketing and operations teams an immediate, interpretable summary of the semantic factors behind a spike.

The approach combines a fixed BERTopic model trained on a multi-month historical dataset with a targeted analysis of the anomaly window. By contrasting historical topic frequencies with the distribution observed during a spike, the system isolates meaningful thematic deviations instead of simply listing the topics present in the window. This mirrors the design philosophy of the detectors: a stable baseline model, followed by situational analysis.

4.1.1 *Text Preparation and Granularity*

Before topic modeling, all tweets undergo a unified preprocessing pipeline:

- Removal of placeholders (<SCREEN_NAME>, <URL>).
- Normalization of whitespace, punctuation, and Unicode characters.
- Optional preservation of hashtags as semantic cues.
- A stopword list combining English stopwords, de-identification tokens, and common social noise.

This reduces noise and ensures that the clusters reflect actual user language rather than structural artifacts.

4.1.2 *Historical Training Strategy*

A key architectural decision is to train BERTopic once on a large historical dataset and reuse that model for all future windows. This avoids the instability of re-fitting BERTopic on small, shifting batches of tweets and ensures that topic identities remain stable across the entire system.

To determine a robust historical topic vocabulary:

- Train BERTopic on the full historical period (excluding one holdout month; see Section 4.3).
- Compute the historical share of each topic.
- Remove the long tail of sparse topics representing one-off events.

This provides a stable semantic baseline against which future spikes can be evaluated.

4.1.3 *Embedding and model configuration*

The system uses *all-MiniLM-L6-v2* as the sentence representation model due to its strong performance on short social media text and its low latency. BERTopic is configured with a min_topic_size tied to the dataset scale (typically 1% or ≥40 tweets) to balance stability with efficiency.

4.2 Window Analysis (Spike Investigation)

After a spike is detected by the volume or sentiment models, the tweets from that window are inferred through the historical model using:

```
window_model = BERTopic.load("bertopic_model")

new_topics, new_probs = window_model.transform(docs_window)
```

This guarantees a direct mapping between window and historical topics, allowing the system to measure how strongly each theme deviates from its long-term baseline.

4.2.1 *Weighted topic comparison*

A simple difference between window share and historical share can misrepresent topics that are absent in the window. To correct this, the system computes:

- hist_share: topic frequency across historical data
- win_share: frequency within the anomaly window
- hist_share_weighted: historical shares renormalized only across topics that appear in the window
- delta_weighted = win_share – hist_share_weighted

This adjustment highlights true thematic surges rather than artifacts of rare topics. The top five topics by weighted delta are used as the primary explanations of the spike.

4.2.2 *Outlier topic remodeling*

Tweets labeled as -1 by BERTopic represent conversations not captured by the historical model. These are not discarded, instead, a small BERTopic model is trained solely on these window outliers. This secondary clustering surfaces new or emerging themes that were absent from the baseline, but that may be critical during a crisis.

4.2.3 *Topic evolution visualization*

The system produces topics-over-time visualizations for both the top weighted-topic comparison and the window outliers, using a consistent 15-minute granularity to keep all components aligned, showing when each theme began rising, how often it appeared over time, whether multiple topics overlapped to create compound spikes, and how thematic acceleration relates to the volume anomaly, with all figures generated in Plotly using customized sizing and legends for clarity.

4.3 Evaluation Strategy

Manual annotation is not feasible at the scale of social media data, and topic identity is inherently subjective. Because traditional accuracy metrics do not apply to unsupervised NLP, we adopted an evaluation approach grounded in external validity.

To test whether the historical model could generalize, we used a holdout-month evaluation:

6. Historical training: The BERTopic model was trained on the entire dataset except one month.
7. Identification of real world events: For the holdout month, we manually searched for news events or external drivers associated with the observed spikes (e.g., service outages, viral roster changes, viral incidents).
8. Window extraction: We selected windows around those spikes and ran the Topic Spike Analyzer using the historical model.
9. Qualitative validation: We examined whether the top delta-weighted topics and the outlier clusters aligned with the known real-world cause of each event.

This approach evaluates the model's ability to surface the correct themes in a time period it never saw during training, providing evidence of robustness and generalization.

The model consistently highlighted relevant topics for spikes in the holdout month, despite not having been trained on that period, confirming that the historical vocabulary was sufficiently broad and that the delta-weighted mechanism successfully detected thematic changes.

4.4 Topic Analyzer Results

Using the holdout-month setup described, we evaluated how well the Topic Spike Analyzer recognized the themes associated with real spikes across the MARTA, Falcons, and Waffle House datasets. For each dataset, we identified one or more spikes in the excluded month, selected the corresponding time windows, and generated the outputs of the system during real world events.

4.4.1 *Dataset summary*

For each dataset, we report the holdout month, the spike window evaluated, and the real-world event used as reference (full link provided in the appendix)

MARTA

- Holdout month: January 2025
- Spike window: January 03, 2025 - January 05, 2025
- Reference event: MARTA Bus driver shot at Decatur Station [1]

FALCONS

- Holdout month: January, 2025
- Spike window: January 18, 2025 - January 20, 2025
- Reference event: Jeff Ulbrich new defensive coordinator [2]

WAFFLE HOUSE

- Holdout month: February
- Spike window: February 03, 2025 - February 05, 2025
- Reference event: Surcharge of 50 cents on every egg due to bird flu [3]

4.4.2 *Weighted delta output*

For each spike window, we report the top three topics with the highest weighted-delta values, which represent the topics with the highest percentage-point increase relative to their historical baseline. In the tables below, topics that relate to the real world event used as reference are highlighted in blue.

Table 7 —MARTA Top delta weighted topics

Name	hist_share	win_share	hist_share_weighted	delta_weighted
6_shooting_police_station_marta station	2.44%	18.62%	3.01%	+15.62pp
37_driver_marta bus_hit_bus	0.40%	1.62%	0.50%	+1.12pp
19_line_schedules_lines_red	0.92%	1.62%	1.13%	+0.48pp

Table 8 — FALCONS Top delta weighted topics

Name	hist_share	win_share	hist_share_weighted	delta_weighted
16_defense_offense_defense defense_play defense	0.89%	3.56%	0.91%	+2.65pp
14_ulbrich_sanders_prank_shedeur	0.92%	3.42%	0.94%	+2.48pp
33_coach_coaching_coaches_head coach	0.45%	2.56%	0.46%	+2.10pp

Table 9 — WAFFLE HOUSE Top delta weighted topics

Name	hist_share	win_share	hist_share_weighted	delta_weighted
26_egg_surcharge_egg surcharge_prices	0.51%	45.83%	0.52%	+45.31pp
53_eggs_house eggs_eggs waffle_scrambled	0.26%	1.71%	0.26%	+1.44pp
20_prices_price_expensive_house prices	0.73%	1.52%	0.74%	+0.78pp

4.4.3 *Outlier cluster output*

The tables below present the clusters identified within the spike window that had no representation in the historical BERTopic model. These outlier groups often correspond to emerging or incident-specific themes. Topics that relate to the real-world event used as reference are highlighted in blue.

Table 10 — MARTA Top outlier clusters

Topic	Representation	Count
0	['marta', 'bus', 'driver', 'marta bus', 'bus driver', 'fare', '50', 'killed', 'just', 'shot']	122
1	['carter', 'center', 'king', 'jan', 'carter center', 'king memorial', 'memorial', 'station', 'shuttles']	19

Table 11 — FALCONS Top outlier clusters

Topic	Representation	Count
0	['fuck', 'luck', 'good luck', 'don', 'guys', 'yall', 'good', 'fucking', 'hell', 'care']	279
1	['hire', 'hiring', 'interview', 'terrible hire', 'great hire', 'good hire', 'interviewing', 'just', 'kidscut']	109

2	['franchise', 'joke', 'unserious', 'joke franchise', 'unserious franchise', 'fucking', 'year', 'franchises']	61

Table 12 — WAFFLE HOUSE Top outlier clusters

Topic	Representation	Count
0	['waffle', 'house', 'waffle house', 'like', 'don', 'qt', 'house waffle', 'just', 'know', 'food']	807
1	['gaza', 'waffle', 'house gaza', 'waffle house', 'house', 'gaza strip', 'strip', 'location', 'gazans', 'lets']	18
2	['car', 'asleep', 'fell asleep', 'fell', 'inside', 'driver', 'car woke', 'inside waffle', 'police fell', 'woke']	17

4.4.4 *Visualizations*

Two topics-over-time figures are provided for each window analysis, one for historical topic variance, and one for outlier clusters. Both visualizations use the same 15-minute granularity as the volume detector to maintain alignment across components. The figure below shows an example of the topics-over-time output generated by the model.

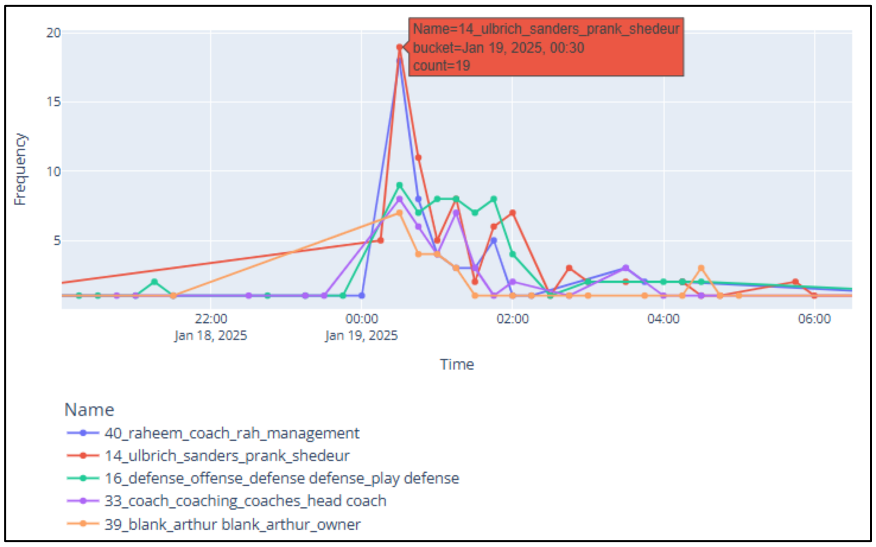


Figure 7 — delta-weighted visualization for FALCONS spike window

5 CONCLUSION

This project delivered a working prototype of a crisis detection system that combines three parts: a volume spike detector, a negative sentiment spike detector, and a topic spike analyzer. The two time series components use Prophet to learn a baseline and trigger alerts when behavior deviates from what is normally expected for each brand. The topic component then explains those alerts by surfacing the themes that dominate during spike windows.

For volume, we modeled 15-minute interaction counts with a rolling training window and evaluated performance using a 99th percentile proxy for extreme spikes. Orange alerts showed solid recall for all brands, and the "one step earlier" analysis showed that many misses are mainly timing issues. Precision was weak at the orange level but improved strongly at the red level, which supports the design where orange acts as a noisy early signal and red represents a more actionable escalation.

For negative sentiment, we modeled the hourly proportion of negative tweets on valid high-volume windows, enforced by dynamic thresholds. Across Falcons and Waffle House, the model achieved low errors and high coverage, and iterative training improved performance further without breaking coverage. MARTA exposed a clear limitation for low-volume brands, where relatively few valid windows and higher noise reduce the benefits of more complex training schemes. Still, across datasets, the approach provided a stable baseline for z-score based sentiment anomalies and enabled a structured yellow/orange/red alerting logic.

The Topic Spike Analyzer closed the loop by connecting spikes to interpretable topics. A single BERTopic model trained on historical data, combined with a delta-weighted comparison and an outlier pass, consistently highlighted themes that aligned with real world events in a holdout month for all three brands. Together, these components show that the company can use time series modeling and topic analysis to detect unusual behavior, rank its severity, and explain its drivers in a way that fits existing workflows for executives and social media teams, while also making clear where additional tuning and labeling would be required before a full production rollout.

6 FUTURE STEPS AND IMPROVEMENTS

In the short term, this is what we see as future steps for improvement of the current system:

- Integration of the topic model with the spike detectors for real-time automated explanations.
- Tune thresholds and configuration per brand, including minimum volume filters, z-score cutoffs, and escalation rules, and define a small set of "conservative / balanced / aggressive" templates.
- Run broad backtesting and set up internal human review of alerts so customer success and social teams can tag alerts as useful, minor, or noise, then adjust rules based on that feedback.
- Prototype the executive summary, detailed alert report, and topic explanations inside existing dashboards, and wire alerts to current channels such as email or Slack to confirm they fit real workflows.

After the system is running, this is what we believe to be the next steps to ensure smooth operation and good user experience:

- Log all alerts and user actions on them, build a labeled crisis vs non-crisis dataset, and use it to periodically recalibrate thresholds, evaluate recall and precision, and justify changes to the models.
- Define clear handling for low volume brands, including minimum supported volume, fallback behavior when data is too sparse, and options to pool information across similar accounts.
- Establish a regular maintenance schedule for the sentiment and topic models, including quarterly BERTopic refreshes to handle vocabulary drift, and continuous monitoring of coverage, error, and alert rates as part of a simple SLA.

7 REFERENCES

10. FOX 5 Atlanta. (2025, January 5). *MARTA bus driver shot at Decatur Station*. https://www.fox5atlanta.com/news/decatur-marta-station-shooting-police
11. Reuters. (2025, January 19). *Falcons tab Jeff Ulbrich defensive coordinator.* https://www.reuters.com/sports/falcons-tab-jeff-ulbrich-defensive-coordinator-2025-01-19/

12. The New York Times. (2025, February 4). *Waffle House adds 50-cent surcharge on eggs as bird flu pressures supply*. https://www.nytimes.com/2025/02/04/business/waffle-house-egg-surcharge.html