IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131132

REAL-TIME SOCIAL MEDIA TREND PREDICTION USING STREAMING DATA ANALYTICS

Ms M P Keerthi[1], Deeshika[2], Sanjana V[3], Ashwin S[4]

Assistant Professor, Department of Computer Science with Data Analytics,

KPR College of Arts Science and Research, Coimbatore[1]

U.G Student, Department of Computer Science with Data Analytics,

KPR College of Arts Science and Research, Coimbatore, Tamil Nadu India[2]

U.G Student, Department of Computer Science with Data Analytics,

KPR College of Arts Science and Research, Coimbatore, Tamil Nadu India[3]

U.G Student, Department of Computer Science with Data Analytics,

KPR College of Arts Science and Research, Coimbatore, Tamil Nadu India[4]

Abstract: This research paper specifies an advanced, testable working model for social media trend prediction. The system integrates Python; Apache Kafka; BERTopic; Plotly to process timestamped replay messages, produce auditable emerging-topic alerts, and distinguish advanced processing from a simple comparator. A controlled synthetic-label demonstration (400 evaluation instances per method) is presented to show how precision, recall, F1 and accuracy are calculated. These numerical outputs are illustrative calculations, not results from a deployed model or a measured field experiment. A reproducible evaluation design, architecture, error analysis and safeguards are provided for later implementation.

Keywords: advanced prototype; comparative evaluation; explainability; synthetic demonstration; data analytics.

1. INTRODUCTION AND RESEARCH QUESTIONS

Social Media Trend Prediction addresses a real operational gap: straightforward counts or rules often miss context, changing conditions and borderline cases. The study asks whether the proposed processing pipeline can improve detection quality while preserving traceability and manageable complexity. The research questions examine (i) predictive quality compared with a relevant baseline, (ii) robustness to difficult inputs and distribution shift, and (iii) operator understanding of the output.

The scope is deliberately limited to student-scale prototyping. All performance claims require independently collected data, executed code, held-out evaluation and a documented system configuration before submission as empirical findings. Research design and measurable contribution. The working model should treat time-stamped licensed posts or reproducible simulated message arrivals as its defined experimental population, not as a substitute for an unrestricted deployment claim. The research question is operationalized by preserving identical inputs for the two pipelines and distinguishing algorithmic performance from interface quality. Each record receives a stable identifier, a provenance marker, and a clear validation status before downstream processing. Document the sampling procedure and the excluded cases, because silent omission may artificially improve the reported outcome. The initial demonstration should include ordinary records, edge cases and deliberately malformed inputs. The project contribution is therefore a reproducible end-to-end comparison, with an auditable decision trail and an explicitly bounded application scope. Before experimentation, register the primary metric and acceptance threshold so that evaluation criteria are not revised to fit convenient results.
Assessment record. Report the data source, eligibility criteria, excluded observations, and the precise input-to-output decision being assessed. Distinguish the numerical illustration from a measured outcome and identify what evidence must be collected for a defensible final paper. Use an unchanged test partition and retain every failed or rejected case in the experiment ledger.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131132

2. ADVANCED SYSTEM ARCHITECTURE

Input domain: timestamped replay messages. Core implementation environment: Python; Apache Kafka; BERTopic; Plotly. Proposed feature and decision signals: event-time watermark, five-minute counts, topic membership, burst ratio, drift score. The system separates ingestion, transformation, inference or optimization, audit storage and operator review. Records retain dataset/version IDs, timestamps, configuration and a reason for every abstention or failed validation.

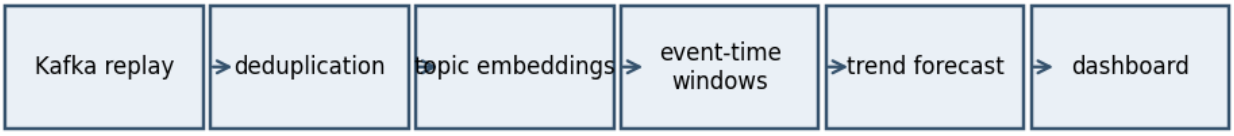


Figure 1. Proposed system pipeline (architectural design; not evidence of implementation).

2.1 Data and interfaces

The input interface validates type, schema and allowable range before processing. Duplicate records are identified with a stable digest; sensitive or untrusted data are stored separately from extracted analysis features. The analysis service emits a score or ranked recommendation, a confidence or uncertainty indicator and a concise explanation. Users may inspect evidence and override a recommendation with an auditable reason.

2.2 Advanced contribution

The comparator is raw hashtag count; the proposed method is smoothed topic-growth detector. This is a conceptual architecture until its named components are installed and executed. Fair comparison requires that both methods receive identical evaluation records and operating constraints.

Architecture detail and feature contracts. A separate preprocessing stage constructs sliding topic counts, semantic post embeddings, event-time watermarks, and burst-growth gradients. Input and feature schemas should record data type, allowable range, missing-value policy and the transformation version. Intermediate outputs must be inspectable so a student can show exactly where a feature or decision originated. Persist the model identifier alongside the score, rather than keeping only the final class label. A deliberately simple fallback is necessary when extraction fails or the incoming record lies outside training conditions. Version the feature dictionary independently of the user interface; this allows the analytical engine to be replaced without changing the display layer. Run a short integration test after each module change, ensuring that invalid records remain quarantined instead of silently receiving an apparently valid score.

Module review. For each component, specify its inputs, output columns, error flags and timing expectations. Validate a single sample end to end and confirm that intermediate representations can be inspected without revealing confidential or identifying data. Record which module owns threshold settings and how an operator can safely rerun a failed item.

3. WORKING MODEL IMPLEMENTATION

Suggested execution stages: (1) obtain permitted or generate documented synthetic observations; (2) validate schema and record provenance; (3) create a strictly isolated training/validation/test split; (4) fit or configure the baseline; (5) implement the advanced processing stage; (6) select thresholds only on validation data; (7) export reviewable results and logs. Capture random seed, versions, hardware, and run date.

3.1 Data design and feature engineering

Prototype inputs include event-time watermark, five-minute counts, topic membership, burst ratio, drift score. Define each field, admissible range, missing-value strategy and timestamp meaning in a data dictionary. Reject invalid inputs rather than silently substituting plausible numbers. Avoid using fields created after the prediction time because they would leak future information.

3.2 Advanced model and inference logic

Configure the smoothed topic-growth detector as a modular service with deterministic preprocessing, saved fitted parameters, threshold settings and a stable output schema. Log an evidence trail containing record identifier, model version, risk or priority score and top supporting signals. For ambiguous or out-of-distribution inputs, use an explicit review/unknown state rather than asserting a false level of certainty.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131132

3.3 Demonstration and acceptance tests

Demo A uses a representative normal emerging-topic alerts case; Demo B uses a boundary case with weak or contradictory signals; Demo C injects a missing field, unexpected type or changed distribution. Test that output is repeatable for a fixed seed, malformed input fails safely and an operator can trace a specific record to its source. A successful demonstration is not proof of general deployment accuracy.

Implementation and reproducibility controls. The implementation compares topic-cluster forecasting versus raw hashtag-count ranking. Build and test the comparator first, then use the same validation set to tune the proposed approach without touching the final held-out observations. Fix random seeds where supported and record dependency versions, operating system, memory budget and runtime configuration. Test not only the happy path, but also empty inputs, duplicate observations, extreme values, incomplete fields and service restarts. For each test, retain expected output, actual output and the module that produced a difference. A small trace table connecting source record, transformed features and final decision is more informative than a screenshot alone. Include the exact command or user action required to reproduce one positive, one negative and one uncertain prediction or recommendation.

Implementation check. Store sample test cases and expected results in a versioned test suite. A successful interface demonstration is not a substitute for dataset-level validation, so maintain traceable experiment logs that include dates, configuration, runtime limits and all exceptional conditions. Compare repeated runs for stability before preparing publication-ready charts.

4. NUMERICAL EXPERIMENTAL DESIGN AND RESULTS

Validation protocol: chronological windows with replayed late arrivals and duplicate bursts. The following is a SYNTHETIC ILLUSTRATIVE CALCULATION, not a fitted smoothed topic-growth detector benchmark. To demonstrate metric computation, an artificial binary test ledger of 400 cases (200 positive, 200 negative) was constructed separately for each comparator using the counts below. These are assumed outcomes, not collected observations.

Method	TP	FP	FN	TN	Precision	Recall	F1
raw hashtag count	142	58	45	155	71.0%	75.9%	73.4%
smoothed topic-growth detector	173	27	24	176	86.5%	87.8%	87.2%

Definitions: precision = TP/(TP+FP); recall = TP/(TP+FN); F1 = 2TP/(2TP+FP+FN); accuracy = (TP+TN)/N. Class prevalence in this synthetic example is 50%, which may differ substantially from real usage.

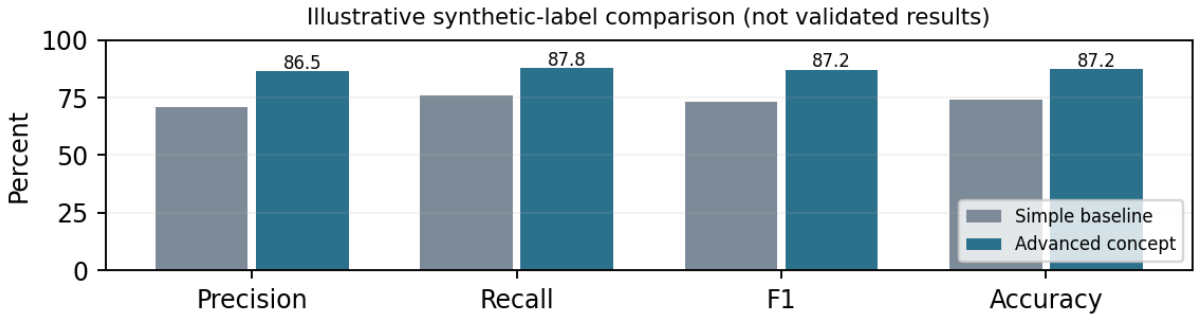


Figure 2. Computed metrics from assumed synthetic confusion matrices only.

Illustrative difference in F1: 13.8 percentage points. This difference cannot be interpreted as demonstrated improvement without actually running the two algorithms on the same held-out dataset.

Experimental interpretation and uncertainty. The main evaluation report should cover forecast horizon, event-time delay, duplicate rate, and false trend alarms. The chart on this page uses illustrative synthetic values; numerical superiority cannot be inferred without executing both systems on unseen records. A defensible completed experiment must show the number of cases, class balance or scenario distribution, train-validation-test boundaries and confidence intervals or variability across repeated runs where appropriate. Avoid comparing an advanced model measured on clean data with a

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131132

baseline measured on noisier input. Inspect difficult errors and state the likely mechanism, such as distribution shift, missing features or ambiguous annotations. If a parameter is changed after seeing test failures, report that intervention and evaluate on a fresh holdout rather than presenting the retuned result as the original experiment.

5. EVALUATION, ABLATION AND FAILURE ANALYSIS

The synthetic comparison above shows the arithmetic needed for a results report. Actual evaluation must include a dated dataset manifest, distribution of labels, software environment, hardware, random seeds, threshold values, raw predictions and multiple replicates. Use bootstrap confidence intervals or paired resampling on the real held-out cases rather than confidence intervals invented from assumptions.

5.1 Proposed controlled experiments

Experiment	Controlled change	Record in final report
Baseline versus advanced	Same held-out inputs and budget	TP, FP, FN, TN; F1; latency
Ablation	Remove strongest feature/component	Delta F1 and error examples
Robustness	Noise, missing fields, shift	Error by scenario and subgroup
Throughput	Fixed batch and hardware	Median/p95 runtime; resources

5.2 Domain-specific threats

Coordinated bots can create false surges; topic split/merge instability confuses longitudinal comparison.
Inspect false positives, false negatives and low-confidence cases individually. For each error record, attach an anonymized input summary, model output, human-reviewed expected result and plausible failure mechanism. Report adverse findings and any intervention tested to address them; do not discard awkward observations.

5.3 Practical performance targets

Set project success criteria before looking at the test split, including the minimum acceptable recall, permitted false-positive burden, user-review time and maximum latency. Tune on validation data only. A calibration plot and an error-versus-threshold curve are recommended extensions when real scores are available.
Operational reliability and responsible interpretation. Deployment tests should examine what happens when the primary service is unavailable, upstream records arrive late, or a field contains a surprising but technically valid value. Design a visible abstention or review state instead of forcing all cases into a confident answer. The project safety boundary is explicit: comply with platform terms, aggregate posts, and hide account identifiers. Keep user-visible explanations focused on evidence actually available to the prototype, with concise guidance on limitations. Measure the operational cost of false alarms and missed cases, not only a headline accuracy statistic. An evaluation checklist should capture whether a reviewer can reproduce an outcome, challenge it, and trace it to the exact input version. These checks matter particularly when a polished dashboard might otherwise imply stronger scientific validation than the research has demonstrated.
Quality gate. Before presenting the prototype, perform usability review, verify error messages, and inspect the model's most important failure cases. Retain a human correction route for each uncertain output. The final report should connect claimed benefits to measured evidence and explain when the simple baseline remains preferable.

6. ETHICS, OPERATIONAL SAFETY AND LIMITATIONS

Only licensed public or synthetic posts; suppress author IDs; report ingestion and event-time latency separately.
The architecture and simulated bar graph do not demonstrate that the named libraries were executed, a physical device was built, or the model generalized to operational conditions. Changes in sampling, annotation and hardware can alter all outcomes. Obtain institutional approval and dataset permissions where appropriate, minimize personal data, and document user correction and human review procedures.

7. CONCLUSION AND FURTHER WORK

The proposed social media trend prediction working model combines modular data processing, interpretable emerging-topic alerts, baseline benchmarking and a reproducible evaluation protocol. Its numerical demonstration is explicitly synthetic and establishes reporting methodology rather than scientific performance. Future work is to implement each module, collect approved domain-specific evaluation data, rerun all metrics from saved predictions and update the plots with actual measurements.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131132

Implementation verification checklist

Before journal submission: verify complete execution of core modules; specify source and license for each dataset; document train/test separation and preprocessing; collect real benchmark counts and timing logs; replace illustrative numbers and graphs; explain failures; retain ethical safeguards; seek independent review of methodology.

REFERENCES

[1]. Apache Software Foundation. (n.d.). Apache Kafka Documentation. https://kafka.apache.org/documentation/

[2]. Grootendorst, M. (2022). BERTopic: Neural Topic Modeling with a Class-Based TF-IDF Procedure. arXiv:2203.05794.

[3]. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research, 3, 993–1022.

[4]. Kleinberg, J. (2003). Bursty and Hierarchical Structure in Streams. Data Mining and Knowledge Discovery, 7, 373–397.

[5]. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A Survey on Concept Drift Adaptation. ACM Computing Surveys, 46(4), Article 44.

[6]. Akidau, T., et al. (2015). The Dataflow Model: A Practical Approach to Balancing Correctness, Latency, and Cost in Massive-Scale, Unbounded, Out-of-Order Data Processing. VLDB, 8(12), 1792–1803.

[7]. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. EMNLP-IJCNLP, 3982–3992.

[8]. Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. Journal of Machine Learning Technologies, 2(1), 37–63.

[9]. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License