International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Fake News Detection Using Machine Learning: A Survey of Approaches and a TF-IDF–Naive Bayes Implementation with Live Evidence Checking

K. Nandini, Monika M. S, Prajna H Gouda, Prof. Jasma Bhavani B (Assistant Professor)
Department of Computer Science and Engineering (Data Science)
SJB Institute of Technology, Bengaluru, Karnataka, India

Abstract—Online misinformation can circulate rapidly through social platforms and news aggregators, making manual verification difficult to apply at scale. This survey reviews major computational approaches to fake-news detection, including rule-based and linguistic techniques, conventional machine-learning classifiers, deep neural networks, and transformer models. It discusses text preparation, feature representation, classification methods, benchmark datasets, evaluation measures, and recurring limitations. The paper also describes our two-part prototype. The first component is a Java text classifier built with TF-IDF features and Multinomial Naive Bayes in Weka. On the reported 80/20 split of the ISOT dataset (44,889 articles), it achieved 95.23% accuracy, 96.34% precision, 94.61% recall, and a 95.47% F1-score. These figures describe performance on that dataset and should not be interpreted as evidence of equivalent performance on arbitrary current claims. The second component gathers contextual material from Google News RSS, NewsAPI, and Wikipedia, then sends the claim and retrieved material to the Gemini API for a REAL, FAKE, or UNVERIFIED assessment. This live component has been checked manually; no formal accuracy score is currently available. The survey concludes by considering dataset bias and ageing, adversarial wording, evidence quality, and the trade-off between model transparency and predictive performance.

***Index Terms*—Fake news detection, Natural Language Processing, TF-IDF, Naive Bayes, Text Classification, Evidence Retrieval, Wikipedia, Survey.**

I. Introduction

News verification was once carried out mainly by editors and established news organisations. Digital publishing has changed the scale of the problem: material that resembles a news report can now be created and shared widely within minutes. Research has documented that false stories can spread rapidly on social networks [1], making misinformation a concern that spans computing, journalism, and the social sciences [2]. Automated systems can help prioritise claims for review, but they should not be treated as a replacement for careful human judgement.

The phrase “fake news” is used for several different phenomena, so a detection task needs a clear operational definition. In this paper, it refers to news-like material containing fabricated or materially misleading claims, often presented in a way that may deceive readers [3], [4]. Satire is usually written for humour rather than deception; propaganda seeks to influence attitudes and may use selective or false claims; misinformation can be shared without an intention to deceive, whereas disinformation is deliberately circulated. These categories can overlap, but intent and factual accuracy are not identical. This distinction matters because a text classifier learns from the labels and patterns in its training examples, not from direct access to truth.

A common computational formulation treats detection as a classification problem: a system assigns a label to an article or claim using features extracted from its text or context. This simplifies evaluation, but it can also hide a key weakness. A model may learn recurring vocabulary, formatting, or publisher-specific cues in a dataset rather than learn whether a claim is factually supported. A strong score on a fixed benchmark therefore does not, by itself, establish reliable verification of new events.

This survey reviews the main stages used in automated fake-news detection, from text normalisation and feature construction to model selection, dataset design, and evaluation. It then places our prototype within this landscape. The prototype has two distinct components: a dataset-trained TF-IDF and Naive Bayes classifier, and a live evidence-checking workflow that retrieves external information and asks a language model to assess the claim against it. The review is narrative rather than systematic; its comparisons are intended to describe broad methodological patterns, not to rank every published method.

II. Background and Taxonomy

Several related concepts should be distinguished. Satirical writing uses irony or exaggeration and is not necessarily intended to mislead. Propaganda aims to shape public opinion and may combine selective presentation with inaccurate statements. Misinformation describes false or misleading material shared without necessarily knowing it is false; disinformation refers to material spread with deceptive intent. In practice, automated detectors often cannot infer the author's intent from text alone. They instead examine observable signals and, where possible, compare claims with external evidence.

Detection methods can be grouped by the information they examine (Fig. 1). Content-focused systems analyse wording, style, and the claims made in an article; some also consult external knowledge to check those claims. Context-focused methods consider how content is shared, including user activity, source reputation, and patterns in a social network. Prior surveys describe related dimensions such as factual knowledge, linguistic cues, propagation behaviour, and source credibility [3], [4]. Systems that combine content and contextual signals are commonly described as hybrid approaches.

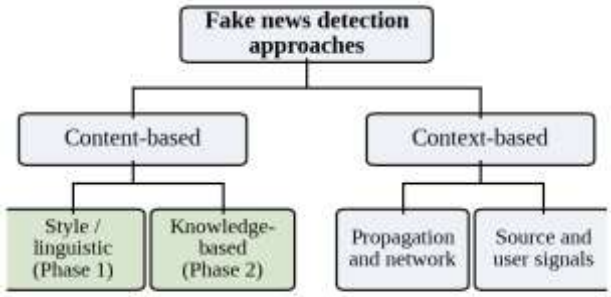


Fig. 1. Taxonomy of automated fake news detection approaches.

Early work explored lexical and rule-based cues, including differences in vocabulary and style across reliable reporting, satire, hoaxes, and propaganda. Rashkin et al. reported linguistic differences among these categories [5]. However, such signals may not transfer well when the topics or sources in the test data differ from those in the training set; this distribution mismatch has been identified as a source of reduced performance [6].

The first component of our prototype is content-based: it represents article text with TF-IDF features and classifies it using the learned labels. The second component is evidence-oriented: it retrieves information related to a submitted claim and asks a language model to judge the claim in light of that material. Neither component currently models how a story propagates through a social network or uses individual user profiles.

The two components answer related but different questions. The dataset-trained classifier estimates whether a text resembles examples labelled fake or real in the ISOT training data; its label is not proof that the article's claims are true or false. The live component instead compares a claim with retrieved material. Its reliability depends on the relevance and quality of the search results and on how accurately the language model interprets them.

III. The NLP Pipeline

Many content-based pipelines follow three broad stages (Fig. 2): prepare the text, convert it into numerical features, and pass those features to a classifier. Decisions made during preprocessing and representation can influence results as much as the choice of learning algorithm.

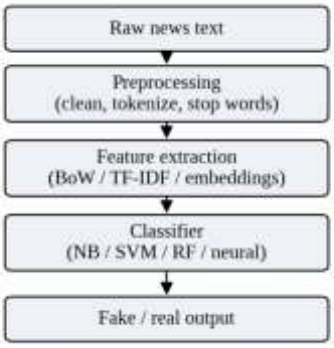


Fig. 2. General NLP pipeline for content based fake news detection.

A. Text Preprocessing

Text preparation reduces inconsistencies before feature extraction. Common operations include converting text to lowercase, splitting text into tokens, removing selected punctuation or non-letter characters, filtering common stop words, and sometimes applying stemming. Stemming maps related word forms to a shared stem; for example, “announced” and “announcing” may be reduced to a common form. Preprocessing must be chosen carefully: removing punctuation or capitalisation can discard emphasis or other useful cues, while retaining every token may introduce noise. The appropriate sequence depends on the dataset and task.

B. Feature Extraction

A classifier requires a numerical representation of text. A bag-of-words representation records the terms present in a document, often using their counts, but common words may then receive similar treatment to more informative terms. TF-IDF adjusts term weights according to both within-document frequency and how widely a term occurs across the corpus. The idea is associated with work on statistical term specificity [7]. For term t in document d within corpus D, containing N documents, the following expression gives a common form of the weighting scheme:

$$TF\text{-}IDF(t,d,D) = TF(t,d)\log(N/DF(t)) \quad (1)$$

Here, TF(t,d) denotes the frequency of term t in document d, while DF(t) is the number of corpus documents containing that term. A term that occurs repeatedly in one article but appears in relatively few other documents receives a larger weight; a term found almost everywhere receives a smaller one. TF-IDF has been used with n-gram features and conventional classifiers in fake-news research, including work associated with the ISOT dataset [8]. Since features remain linked to terms, the representation can also support limited inspection of which words influence a prediction.

Word embeddings provide an alternative to sparse count-based vectors. Word2Vec [9] and GloVe [10] learn dense representations in which terms used in similar contexts tend to occupy nearby positions. Contextual encoders such as BERT [11] additionally vary a word's representation according to its surrounding sentence. These representations can capture semantic and contextual relationships that simple counts miss, but they usually require greater computational resources and can be harder to interpret.

IV. Classification Methods

Multinomial Naive Bayes estimates class probabilities using Bayes' theorem and a simplifying conditional-independence assumption about features [12]. Although words in natural language are not truly independent, the method is computationally efficient and often serves as a useful text-classification baseline. Support Vector Machines seek a separating boundary with a large margin and can perform well with high-dimensional, sparse TF-IDF features, though training costs depend on the implementation and dataset. Logistic regression is another common baseline. Random forests combine decision trees, while their suitability depends on the feature representation and computational budget.

Models that process sequences can use word order and longer patterns that bag-of-words features do not explicitly represent. Long Short-Term Memory networks (LSTMs) were developed to learn dependencies across sequences [13]. Transformer-based architectures, including BERT, have since become widely used for language tasks [11]. Such models can capture richer context, but fine-tuning and deployment may require more labelled examples and computing resources than traditional classifiers.

Comparative studies have examined conventional classifiers such as Multinomial Naive Bayes, logistic regression, and linear SVM with TF-IDF features [14]. Other work has combined TF-IDF with LSTM-based architectures

and optimisation procedures [15]. Directly comparing reported scores across papers can be misleading when datasets, split strategies, preprocessing, and hyperparameters differ. For this reason, Table I presents qualitative tendencies rather than a numerical leaderboard; the ratings should be read as broad observations, not universal guarantees.

TABLE I
GENERAL TENDENCIES OF DETECTION APPROACHES

Approach	Accuracy	Interpretability	Data / compute
Naive Bayes + TF-IDF	Moderate	High	Low
SVM	Moderate to high	Medium	Medium
Random forest	High	Low to medium	Medium
LSTM / transformers	High	Low	High
Evidence + LLM (ours)	Not yet measured	Medium	External API

Note: The ratings are qualitative judgments, not measured values. They depend on the dataset, implementation and experimental setup, and do not imply that any particular model will reach a given level of accuracy under all conditions.

No classifier is best for every dataset and deployment setting. More complex architectures may model richer patterns, but any gains need to be weighed against training cost, latency, data requirements, and the difficulty of explaining a prediction. Performance may also fall when the source, topic, or time period changes between training and testing. These issues are particularly relevant when a system must run with limited hardware or be maintained over time.

A benefit of TF-IDF with Naive Bayes is that its learned term statistics can be inspected more easily than the internal representations of many deep models. This provides some clues about which words contribute to a class score, although it is not a complete explanation of a decision. If highly weighted terms mainly identify a publisher, a dateline, or a formatting convention, the classifier may be relying on source-specific shortcuts rather than factual evidence.

V. DATASETS

Dataset design affects what a detector can learn and how broadly its results apply (Table II). ISOT contains full-length articles labelled as fake or real; its source composition is an important consideration when interpreting results [8]. LIAR contains short claims drawn from fact-checking material and uses a multi-level truthfulness label scheme [16]. FakeNewsNet includes article information alongside social-context data, supporting research beyond text alone [17]. WELFake combines material from existing collections and contains roughly 72,000 articles according to the cited work [18]. Differences in unit of analysis and label design mean these resources are not interchangeable.

TABLE II
REPRESENTATIVE DATASETS

Dataset	Content	Labels
ISOT	Full articles (44,889 in our copy)	Binary
LIAR	Short statements (about 12,800)	Six levels
FakeNewsNet	Articles plus social context	Binary, with propagation
WELFake	Merged articles (about 72,000)	Binary

Benchmark datasets can reflect the period, topics, publishers, and annotation practices from which they were assembled. A detector may therefore fail on later events or unfamiliar subjects. If fake and real examples come from different sources, a model can learn publisher style or formatting instead of truth-related signals; source-specific patterns in ISOT are one reason to interpret benchmark scores cautiously. Labels can also contain errors or reflect source-level assumptions. Consequently, strong performance on one collection does not guarantee comparable results on independently collected news.

VI. Evidence-Based Verification

Text classification alone can struggle with a carefully written false claim because fluent style is not evidence of accuracy. Evidence-based verification instead searches for material that supports or contradicts a claim. The FEVER benchmark contains claims paired with evidence and uses three outcomes: Supported, Refuted, and Not Enough Information [19]. The third outcome is important in practical systems because the available material may be insufficient to justify a confident true/false decision.

Language models can help interpret retrieved passages, but their outputs are only as dependable as the evidence and reasoning supplied to them. A model may misunderstand a passage, over-trust a weak source, or produce a confident response despite missing evidence. Wikipedia can provide useful background, but it is collaboratively edited and may be incomplete or outdated; news feeds are valuable for recent reporting but may not cover obscure claims or breaking events. Evidence-based verification should therefore expose its sources and allow an UNVERIFIED outcome when support is inadequate.

VII. METRICS

Accuracy measures the share of evaluated items classified correctly. It is calculated from true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN):

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \times 100\% \quad (2)$$

This can be misleading if there is an imbalance in classes. This is why studies will often also report precision (P), recall (R) and F1-score:

$$Precision = TP / (TP + FP) \quad (3)$$

$$Recall = TP / (TP + FN) \quad (4)$$

$$F1 = 2PR / (P + R) \quad (5)$$

Equations (3)–(5) express precision, recall, and F1 as proportions; they are reported as percentages for our model in Section VIII. A confusion matrix separates the types of correct and incorrect predictions and can reveal whether one class is favoured. Evaluation on a different source or time period provides a stronger test of generalisation than repeating a random split from the same dataset, although such external evaluations are not always reported.

VIII. OUR SYSTEM

A. Phase 1 (TF-IDF, Naive Bayes)

We implemented the first component in Java using the Weka library [20]. The data were obtained from the ISOT collection distributed through Kaggle [8], with 44,889 articles in our copy: 23,472 labelled fake and 21,417 labelled real. After combining the data and applying the text preparation steps described in Section III-A, TF-IDF features were built using the training vocabulary. We used an 80/20 train-test split, giving 35,900 training examples and 8,980 test examples. TF-IDF and Multinomial Naive Bayes were

selected because they are relatively lightweight and their behaviour is easier to inspect than that of many deep models. The approach remains limited by the coverage and source characteristics of the training data.

B. Results

On the reported ISOT test split, the model achieved 95.23% accuracy, 96.34% precision, 94.61% recall, and a 95.47% F1-score (Table III and Fig. 3). These aggregate metrics are useful but do not show the error distribution for each class. Per-class scores and a confusion matrix would provide a more complete account of the model's behaviour.

TABLE III
PHASE 1 RESULTS ON ISOT (80/20 SPLIT)

Metric	Value
Accuracy	**95.23%**
Precision	**96.34%**
Recall	**94.61%**
F1-score	**95.47%**

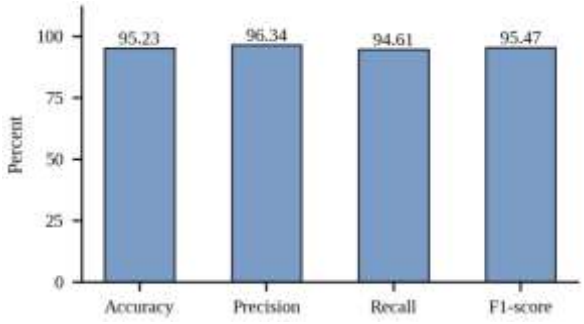


Fig. 3. Phase 1 test-set metrics on ISOT.

These measurements describe only the held-out portion of the ISOT dataset. They should not be taken as evidence that the model can reliably assess arbitrary claims or breaking news. The result demonstrates performance on examples drawn from the same collection; broader claims would require testing on other datasets and newer material.

C. Phase 2 (Live Evidence)

The live component assesses a submitted claim using retrieved context rather than treating the dataset classifier's label as ground truth. The backend searches Google News RSS for recent headlines and snippets, uses NewsAPI to find related reporting, and queries the Wikipedia API for background information about relevant people, places, or events. The collected passages are assembled with the original claim and sent to the Gemini API. The intended response is REAL when the evidence supports the claim, FAKE when reliable evidence contradicts it, and UNVERIFIED when the available material is insufficient. This live workflow has been manually tested; its accuracy has not yet been established through a formal labelled evaluation.

Negation can complicate retrieval. For instance, the claims "X happened" and "X did not happen" share most of their keywords, so a search based only on overlapping terms may return similar results for claims with opposite meanings. Query construction and evidence interpretation must therefore account for negation and context rather than relying on keyword overlap alone.

D. Implementation

The two components run behind a Java server on port 8080, while the user interface is built with React and Vite (Fig. 4). The interface includes a dashboard, browser-based speech input and output through the Web Speech API, a live news ticker, and an About page. The prototype was tested in Chrome. Since browser support and speech-recognition behaviour can differ, these features may not work identically in every browser. Table IV summarises the main tools and services.

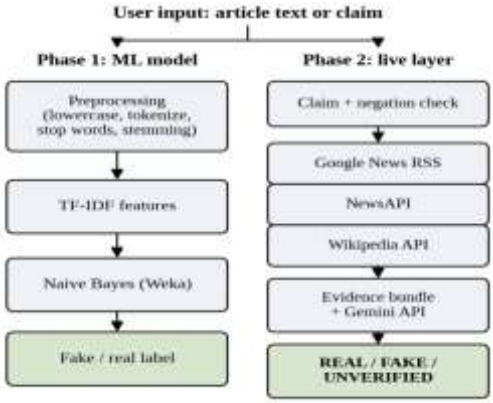


Fig. 4. Architecture of the implemented system.

TABLE IV
TOOLS AND SERVICES USED

Layer	Tools / services
ML backend	Java (Eclipse), Weka
Frontend	React + Vite, CSS, Web Speech API
Data	ISOT dataset (Kaggle)
Live layer	Google News RSS, NewsAPI, Wikipedia API, Gemini API
Development	VS Code, npm, Chrome

E. Comparison of Phase 1 & 2

Table V contrasts the dataset-based classifier with the live evidence workflow. The former categorises text according to patterns learned from labelled examples, whereas the latter assesses a claim using information retrieved from external services. Their inputs, response times, explanations, and failure modes are therefore different.

TABLE V
PHASE 1 VERSUS PHASE 2

Aspect	Phase 1 (ML)	Phase 2 (live)
Input it suits	Articles resembling ISOT	Short claims with retrievable evidence
External services	None	News, Wikipedia and Gemini APIs
Speed	Fast	Slower: retrieval plus an API call
Explanation	Word-level weights	Supporting evidence snippets
Typical failure	Learns source style	Weak or missing evidence
Measured accuracy	95.23% on ISOT	Not measured

F. Evaluation

The live component has been checked manually with example claims, but no formal accuracy estimate is available. A more reliable evaluation would use a documented set of recent claims labelled by trusted fact-checking sources. The test protocol should specify the number and selection of claims, the criteria for assigning labels, and the treatment of cases with insufficient evidence. Accuracy, precision, recall, F1-score, and the frequency of UNVERIFIED outcomes could then be reported. Until this is done, the live component should not be described as quantitatively validated.

IX. ETHICAL & PRACTICAL ISSUES

Incorrect verdicts can affect public understanding and may harm people or organisations. A false FAKE label can unfairly discredit accurate reporting, while an incorrect REAL label can lend credibility to misinformation. The interface should communicate uncertainty clearly and present results as decision support, not as a final authority. Retaining UNVERIFIED is important when the retrieved material does not justify either conclusion.

The live workflow sends user-submitted text to external services. Users should be advised not to submit confidential or sensitive material, and the system should handle API credentials securely. The training corpus is English-language and has a particular mix of topics and sources, so performance on other languages or domains has not been established. Evaluating Indian-language claims and improving multilingual evidence retrieval are useful directions for future work.

X. Discussion and Challenges

The reviewed approaches show a recurring trade-off. Traditional text classifiers are relatively inexpensive and can score well on a fixed benchmark, but they may learn patterns tied to the dataset's period or publishers. External evidence can add factual context, yet the benefit depends on retrieving relevant, credible material and interpreting it correctly.

Dataset ageing: vocabulary, events, and misinformation tactics change over time, so models require periodic evaluation and potentially updated training data. Adversarial wording: a fabricated article written in conventional news style may evade a classifier that depends heavily on lexical patterns. Interpretability: complex models may capture richer signals but can be harder to explain, which matters when users need to understand why a claim was flagged.

Evidence quality: retrieved sources may be irrelevant, incomplete, or biased, and a language model can misread the material. External-service dependence: the live verdict relies on third-party APIs whose availability, rate limits, and behaviour may change. Missing social context: a text-only workflow does not model how a claim spreads, although resources such as FakeNewsNet support research into social signals [17]. Evaluation remains a key need, especially testing across datasets and time periods and creating a labelled benchmark for live verification.

The reported ML results come from one 80/20 split. Without repeated runs or cross-validation, their variability is unknown. The paper also does not provide a confusion matrix, class-specific metrics, or an evaluation on an independent dataset. Reproducibility would improve if the Weka classifier configuration, TF-IDF settings, preprocessing choices, and random seed were documented. The live component has only been tested informally, so its reliability remains unquantified.

This is a narrative survey rather than a systematic review: no formal search protocol or replication procedure was applied. Comparisons across methods should therefore be treated as descriptive trends. Future work should document the literature search, evaluate the ML model on independent datasets, report class-specific results and a confusion matrix, and build a labelled evaluation set for live claims. The live evaluation should record source quality, evidence shown to the user, and the rate of UNVERIFIED responses.

XI. Conclusion

This paper reviewed text-based and evidence-assisted approaches to fake-news detection and described a prototype with two complementary components. The first uses TF-IDF and Multinomial Naive Bayes and achieved 95.23% accuracy on the reported ISOT test split. That score reflects classification within the dataset and does not establish factual correctness for unseen news. The second component retrieves material from Google News RSS, NewsAPI, and Wikipedia and asks Gemini to assess a claim as REAL, FAKE, or UNVERIFIED. This component has been manually tested but has not yet undergone formal accuracy evaluation. The main priorities are to test generalisation across sources and time periods, measure live-verification performance with labelled claims, improve reproducibility, and communicate uncertainty responsibly. These limitations reflect broader challenges involving accuracy, interpretability, evidence quality, and computational cost.

References

[1] S. Vosoughi, D. Roy, and S. Aral, “The spread of true and false news online,” *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018, doi: 10.1126/science.aap9559.

[2] D. M. J. Lazer *et al.*, “The science of fake news,” *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018, doi: 10.1126/science.aao2998.

[3] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, “Fake news detection on social media: A data mining perspective,” *ACM SIGKDD Explor. Newsl.*, vol. 19, no. 1, pp. 22–36, 2017, doi: 10.1145/3137597.3137600.

[4] X. Zhou and R. Zafarani, “A survey of fake news: Fundamental theories, detection methods, and opportunities,” *ACM Comput. Surv.*, vol. 53, no. 5, pp. 1–40, 2020, doi: 10.1145/3395046.

[5] H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi, “Truth of varying shades: Analyzing language in fake news and political fact-checking,” in *Proc. EMNLP*, 2017, pp. 2931–2937, doi: 10.18653/v1/D17-1317.

[6] V. Pérez-Rosas *et al.*, “Automatic detection of fake news,” in *Proc. COLING*, 2018.

[7] K. Spärck Jones, “A statistical interpretation of term specificity and its application in retrieval,” *J. Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[8] H. Ahmed, I. Traore, and S. Saad, “Detection of online fake news using n-gram analysis and machine learning techniques,” in *Proc. Int. Conf. Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments (ISDDC)*, 2017, pp. 127–138, doi: 10.1007/978-3-319-69155-8_9.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” arXiv:1301.3781, 2013.

[10] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation,” in *Proc. EMNLP*, 2014, pp. 1532–1543.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL-HLT*, 2019, pp. 4171–4186, doi: 10.18653/v1/N19-1423.

[12] A. McCallum and K. Nigam, “A comparison of event models for naive Bayes text classification,” in *Proc. AAAI-98 Workshop Learning for Text Categorization*, 1998, pp. 41–48.

[13] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[14] Piyush, S. Kaur, and R. Kumar, “Fake news detection using machine learning: A comparative study of Naive Bayes, Logistic Regression, and Linear Support Vector Machine with TF-IDF features,” *Int. J. Res. Innov. Appl. Sci.*, vol. 11, no. 6, pp. 263–271, 2026, doi: 10.51584/IJRIAS.2026.11060027.

[15] R. K. Ayyasamy *et al.*, "A hybrid deep learning framework for fake news detection using LSTM-CGPNN and metaheuristic optimization," *Sci. Rep.*, vol. 15, Art. no. 41522, 2025.

[16] W. Y. Wang, "'Liar, liar pants on fire': A new benchmark dataset for fake news detection," in *Proc. ACL*, 2017, pp. 422–426, doi: 10.18653/v1/P17-2067.

[17] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "FakeNewsNet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media," *Big Data*, vol. 8, no. 3, pp. 171–188, 2020, doi: 10.1089/big.2020.0062.

[18] P. K. Verma *et al.*, "WELFake: Word embedding over linguistic features for fake news detection," *IEEE Trans. Comput. Social Syst.*, vol. 8, no. 4, 2021.

[19] J. Thorne *et al.*, "FEVER: A large-scale dataset for fact extraction and VERification," in *Proc. NAACL-HLT*, 2018.

[20] M. Hall *et al.*, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newsl.*, vol. 11, no. 1, 2009.

[21] NewsAPI, "NewsAPI Documentation." [Online]. Available: https://newsapi.org/. [Accessed: Oct. 2026].

[22] Wikimedia Foundation, "Wikipedia API Documentation." [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:API. [Accessed: Oct. 2026].

[23] Google, "Gemini API Documentation." [Online]. Available: https://ai.google.dev/gemini-api. [Accessed: Oct. 2026].