

Multilingual Fake News Detection using a Machine Learning Approach

Nilam Honmane¹, Mayuri Bais², Shraddha Bhilare³, Komal Bhosale⁴, Vaishnavi Chikhale⁵

¹Professor, ^{2,3,4,5}Student

Department of Computer Engineering

Zeal College of Engineering and Research Pune, INDIA

Abstract: Misinformation is the intentional dissemination of false or misleading information, often shared as news through social media platforms. In today's digital era, people across the globe increasingly depend on online sources for their news and information. While this convenience allows for easy access to data at any moment, it also presents challenges, particularly concerning the risks associated with information overload when untrue information is presented to the public. The phenomenon of fake news poses a significant threat due to its adverse effects on public perceptions. To address this issue, this discussion focuses on comparing computational techniques for detecting fake news in real-time articles. The process involves data mining, feature extraction, model training, and evaluation, utilizing various datasets. The methodology encompasses data preprocessing, feature extraction, and the application of model training and assessment utilizing multiple metrics, including accuracy, precision, recall, and F1-score.

Keywords: Multilingual Fake News Detection, Machine Learning, Machine Learning algorithms, Natural Language Processing, Misinformation.

I. INTRODUCTION

Various news platforms provide users with up-to-the-minute information on current events. However, these platforms can also be hotspots for misinformation, often presenting itself as fake news. This type of disinformation can encompass everything from political falsehoods to erroneous health solutions, typically leading to serious consequences. A notable instance of this was the dissemination of misleading information related to vaccines and false cures during a significant global health emergency.

Machine literacy utilizes algorithms that learn from labelled data, enabling models to distinguish between licit and fraudulent newspapers. By being trained on exemplifications from both orders, these models can directly prognosticate the authenticity of new, unseen papers. In the realm of natural language processing (NLP), ML algorithms exceed at processing expansive textual data and relating patterns reflective of false or deceiving information.

still, detecting fake news poses lesser challenges in multilingual surroundings. A model developed for one language may not effectively transfer to another due to differences in verbal structure, vocabulary, and writing styles. This necessitates the development of advanced styles that are able of performing across different languages and formats.

The Fake News Discovery System is developed using Python, which is well-regarded for its comprehensive suite of machine literacy, NLP, and web development libraries. The readability and maintainability of Python's syntax make it a favourable choice for rendering that's adaptable across platforms. For erecting the web operation, the Flask web operation frame is chosen for its featherlight design, versatility, and strong integration with Python tools, along with its erected-in development garçon that streamlines original testing and replication.

On the backend, PostgreSQL is employed as the database operation system due to its enterprise-position trustability, native support for JSON, and capacity to manage large, multilingual datasets. Its adherence to ACID principles and advanced indexing capabilities make it suitable for executing complex textbook queries fleetly and directly.

Core machine literacy models, similar as Support Vector Machine (SVM), Random Forest, Decision Tree, Naive Bayes and Gradient Boosting are enforced using Scikit-learn. This library provides expansive algorithm support and erected-in testing tools, making it ideal for constructing bracket models. Tokenization, stemming, and stopword junking are carried out with the Natural Language Toolkit (NLTK), which is complete in colourful languages and generally employed in academic and exploration settings. also, NumPy and Pandas are employed for effective data operation, offering optimized data structures and high-performance operations that integrate seamlessly with the machine literacy process.

Every technology element in the system armature has been designedly named to ensure the platform is reliable, scalable, and secure. The armature is designed to support ongoing updates and rigidity in response to evolving intimation strategies, making the system well-suited for real-world operation and ongoing improvement

II. LITERATURE REVIEW

The field of detecting false news has significantly accelerated in recent years. Before this advancement, most research utilized basic text classification and sentiment analysis methods, relying on word frequency and presuming that words are independent an assumption that may not effectively capture the subtleties of natural language.

In their comprehensive review, Mridha et al. (2021) examined various deep learning models used for identifying fake news, categorizing these methods into three groups: those based on Natural Language Processing (NLP), feature-based approaches, and hybrid models. They also highlighted the growing significance of attention mechanisms and transfer learning techniques, which are

becoming increasingly popular in this area.

Almarashy et al. (2023) introduced an innovative framework that captures global features using TF-IDF, spatial features through CNN, and temporal features via BiLSTM. Their final classifier, known as the Fast Learning Network (FLN), notably enhanced both accuracy and convergence speed. Their model was evaluated on the ISOT and FAKES datasets, providing a dependable basis for their findings.

At a more fundamental level, Krishna and Adimoolam (2022) explored Decision Tree and SVM algorithms, discovering that Decision Trees had a slight edge in both accuracy and processing speed. Meanwhile, Narkhede et al. (2023) concentrated on preprocessing techniques, showing how effective text cleaning and tokenization can lead to improved model outcomes. Their research also compared CountVectorizer and TF-IDF vectorization methods.

Adopting a different strategy, Yadav and Rao (2023) employed the Naive Bayes classifier along with two vectorization techniques CountVectorizer and TF-IDF and assessed their accuracy. They determined that TF-IDF performed marginally better than CountVectorizer in terms of accuracy.

Apurva Narkhede and colleagues (2023) emphasized supervised learning algorithms, utilizing NLP for both preprocessing and feature extraction through Count Vectorizer and TF-IDF. Despite the effectiveness of this approach, their study indicated that it might not encompass all forms of misinformation beyond textual content.

N. Leela Siva Rama Krishna and M. Adimoolam (2022) evaluated decision trees (DT) and support vector machines (SVM) for assessing the accuracy of fake news based on a dataset of 311 instances. While their approach is useful, it is constrained by the relatively small dataset and specific social media contexts they investigated.

M.F. Mridha et al. (2021) provided a thorough overview of fake news detection methods, with a specific focus on NLP and deep learning. Their survey organizes existing literature, discusses evaluation metrics, and identifies areas for future exploration, emphasizing the need for improved data quality assurance processes that are often overlooked.

Overall, the literature indicates a shift from traditional machine learning methods to more sophisticated deep learning approaches. Each model has its advantages and drawbacks; while machine learning models are simpler to implement and quicker to train, deep learning models tend to deliver superior accuracy and performance.

III. METHODOLOGY

Developing a system for detecting fake news using machine literacy involves several essential way gathering data, preprocessing it, rooting features, opting and training a model, assessing its performance, and planting it. Each of these way plays a pivotal part in creating a system able of directly relating newspapers as either genuine or false.

The process begins with data gathering, where a labelled dataset conforming of both authentic and fake newspapers is collected. fresh data can be sourced to enhance this dataset through web scraping or by exercising APIs from online news outlets and social media platforms.

Once the data is collected, preprocessing is done to clean the textual data for farther analysis. This involves removing HTML markers, special characters, punctuation, and common stop words. The textbook is also regularized by converting it to lowercase, and stemming or lemmatization is applied to reduce words to their root forms. Tokenization is also performed, breaking the textbook into individual rulings or words, transubstantiating it into a structured format suitable for machine literacy algorithms.

Following preprocessing, point birth takes place, where textual data is converted into numerical features interpretable by algorithms. Common styles include the Bag- of-Words model, which analyzes word frequentness, and Term frequency- Inverse Document frequency (TF- IDF), which measures the significance of words relative to the entire dataset. These styles allow for the effective processing of textbook data by machine literacy algorithms.

The coming phase centers on model selection and training, where colourful machine learning algorithms are estimated to determine the most effective for this task. Popular algorithms employed to establish birth performance include Decision Trees, Naive Bayes, Support Vector Machines (SVM), Random Forest, and grade Boosting. The models suffer training with a designated dataset, and cross-validation ways are employed to corroborate their capability to generalize to new data.

After training, the models are assessed using a variety of performance criteria, similar as delicacy, perfection, recall, and F1- score. These criteria indicate how well the models distinguish between real and fake news. Confusion matrices are also used to dissect the types of crimes made by the models, including false cons and false negatives, which assists in perfecting model performance.

The final stage is deployment, wherein the trained model is integrated into a practical application. This process involves developing a backend service capable of processing user input, making predictions based on the trained model, and returning the results. Additionally, a basic front-end interface is created, allowing users to submit news articles for evaluation. To keep the system current, it can be routinely updated with new data and user feedback.

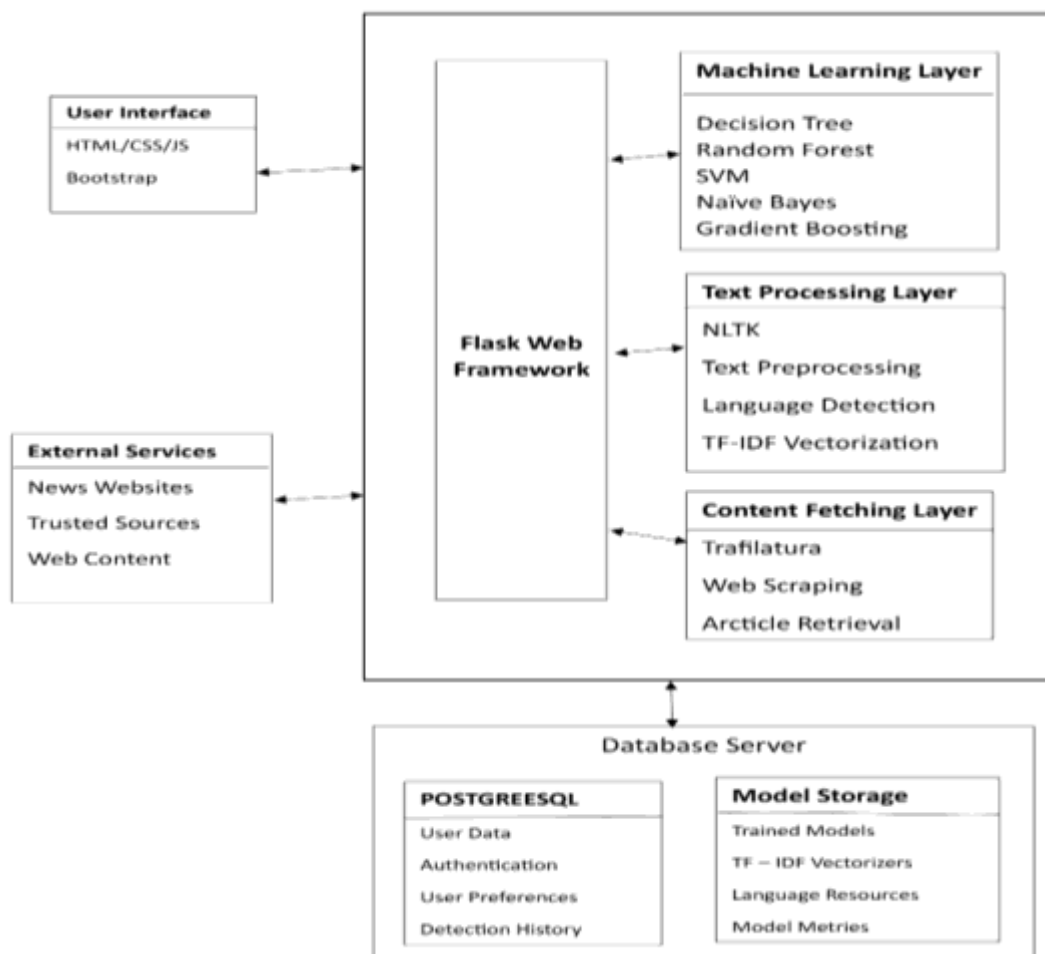


Fig. 1 System Design

In system design, developing a classification-based machine learning model follows a systematic process that starts with gathering a dataset. This dataset serves as the fundamental input for training and assessing the model. Typically, it is divided into two parts: a training set for the model's learning phase and a test set for measuring how well it generalizes.

Once the data is divided, the preprocessing stage takes charge of cleaning and modifying the data. This stage addresses missing values, normalizes numerical data, and converts categorical features into numerical forms to ensure uniformity and compatibility with machine learning algorithms.

Following preprocessing, a feature extraction phase is employed to identify and select the most relevant features, thus reducing data dimensionality and improving the model's performance. The selected features are then provided to the model training component, where a classification algorithm identifies patterns and relationships within the training data.

After the model has been trained, it is integrated into the inference pipeline. During inference, when fresh input data is received, it undergoes preprocessing and conversion similar to the training data. The model yields two outcomes: a probability score that reflects confidence in the prediction, and a binary classification label that indicates the final decision (e.g., true or false).

This structure ensures that the classification system operates efficiently, remains scalable, and can reliably produce accurate predictions for previously unseen data, forming a vital part of intelligent decision-making systems.

IV. IMPLIMENTATION

The core operational logic of the application resides in the `app.py` and `main.py` files, which handle the global configuration and routing for the web application. These files act as the main control hub for executing the Flask app, establishing the database connection, and overseeing user authentication. They outline routes for every webpage, manage incoming requests, and produce appropriate responses. Additionally, they set up the application environment and manage sensitive data such as API keys and other confidential information, ensuring that the application operates securely and efficiently.

In parallel, the `models.py` file is responsible for defining data structures and facilitating database interactions. It encompasses critical models such as `User`, which retains essential account information like username, email address, and hashed passwords; `UserPreference`, which stores individual user input and output language preferences; and `NewsHistory`, which records each user's attempts to verify

news content and the outcomes of those attempts. These models are interconnected through defined relationships: a one-to-one association between the User and UserPreference models ensures that each user has a distinct set of preferences, while a one-to-many relationship links User to NewsHistory, allowing users to maintain multiple verification records. This architecture establishes a strong basis for tailored and traceable user engagement throughout the application.

The `ml_models.py` module focuses on identifying fake news using various machine learning algorithms. It includes several important functions that aid in training, deployment, and model assessment. The `train_models()` function trains distinct classification models on a labeled dataset, while `load_models()` loads pre-existing models from the disk or begins the training process if no saved models are available. For real-time predictions, the `predict_fake_news()` function utilizes an ensemble method to provide results by combining outputs from multiple models. The `save_models_to_disk()` function stores the trained models locally for convenience during future launches. Additionally, `get_precision_metrics()` gathers performance indicators to evaluate the accuracy and efficiency of each model, facilitating performance visualization. Together, these functions create a reliable workflow for detecting fake news through machine learning.

Conversely, the `text_processor.py` module deals with all processes related to preparing and modifying text for suitable machine learning input. Its main function is to clean, normalize, and enhance text data for precise and consistent analysis. The `preprocess_text()` function removes noise from the text and normalizes it, preparing it for the models. The `extract_keywords()` function pinpoints the most significant words in the text, leveraging them to find related news articles. To support various languages, the `translate_text()` function ensures seamless translation of text across different languages. Furthermore, there is a Stemmer class that trims words down to their root forms, tailored for the languages involved. This comprehensive processing pipeline guarantees that the input data is clean and meaningful, thereby enhancing machine learning model performance.

The `news_scraper.py` module is dedicated to collecting relevant news articles from reliable sources to aid in the fake news detection process. It contains several key functions designed to efficiently extract, identify, and summarize online content. For instance, the `get_website_text_content()` method retrieves and extracts readable text from URLs, ensuring only the most relevant content is processed. The `fetch_article_links()` method employs keyword-based logic to locate news articles that correspond to key terms derived from the input text. Lastly, `fetch_related_articles()` detects and summarizes genuine news stories linked to the original content, providing context and points of comparison for assessing authenticity. This module is crucial for enhancing model accuracy by supporting predictions with real-world, evidence-backed information.

The `utils.py` module acts as a handy toolkit filled with helper functions that support the main logic of the application and promote code reusability. Functions like `get_language_code()` and `get_language_name()` translate full names of languages into their respective ISO codes, making language management straightforward across different features. The `calculate_keyword_match_percentage()` function measures how closely retrieved articles align with the keywords provided by the user, while `format_percentage()` ensures that percentage figures are displayed consistently throughout the user interface.

On the client side, the application employs a templating system to create a cohesive and user-friendly interface. The `base.html` template sets up a general layout including the navigation bar and footer, while individual templates like `home.html` serve as an inviting entry point to familiarize users with the app. The `detector.html` page is where the authentication of news occurs, alongside `login.html` and `register.html` for user authentication processes. Additional pages such as `about.html` and `feedback.html` provide useful information and avenues for user expression. Static assets enhance the visual appeal and functionality of the frontend - `style.css` contributes a retro-themed design, `main.js` manages client-side events and form actions, and `chart.js` visualizes machine learning performance data. SVG resources comprise the application's logo and UI icons, providing a cohesive visual experience.

The data flow architecture of the application is designed for accuracy and an improved user experience. The process begins with user input, which then undergoes stages of text preprocessing and language identification. Once the language is identified, the appropriate model is chosen, and ensemble prediction is utilized to produce results. These results are further enriched by searching for similar verified news articles, and the comprehensive output is then presented to the user.

V. ALGORITHM DETAILS

The fake news detection system utilizes an ensemble approach that integrates five distinct machine learning models, each adding unique strengths to enhance classification effectiveness. The Decision Tree Classifier, created with `sklearn.tree.DecisionTreeClassifier`, is fine-tuned with parameters such as `max_depth=10`, `min_samples_split=5`, and `random_state=42`. This model plays a vital role in uncovering hierarchical decision rules within text features. Building on this, the Random Forest Classifier (`sklearn.ensemble.RandomForestClassifier`) computes the average of 100 decision trees (`n_estimators=100`, `max_depth=20`) to minimize the risk of overfitting. To draw a clear line between counterfeit and genuine news, we employ the Support Vector Machine (`sklearn.svm.SVC`) featuring a linear kernel and probability estimates, allowing for the identification of the best hyperplane for classification.

Another significant participant in this ensemble is the Gradient Boosting Classifier (`sklearn.ensemble.GradientBoostingClassifier`), which operates with 100 estimators and a learning rate of 0.1. This model boosts prediction accuracy by constructing trees sequentially, where each tree addresses the errors of its predecessor. Lastly, the Naive Bayes Classifier (`sklearn.naive_bayes.MultinomialNB`), using a smoothing parameter of `alpha=0.1`, imparts a probabilistic dimension to the classification based on Bayes' theorem. By combining these diverse models, the ensemble system gains a comprehensive perspective in identifying false news, resulting in stronger and more dependable predictions.

Feature Extraction

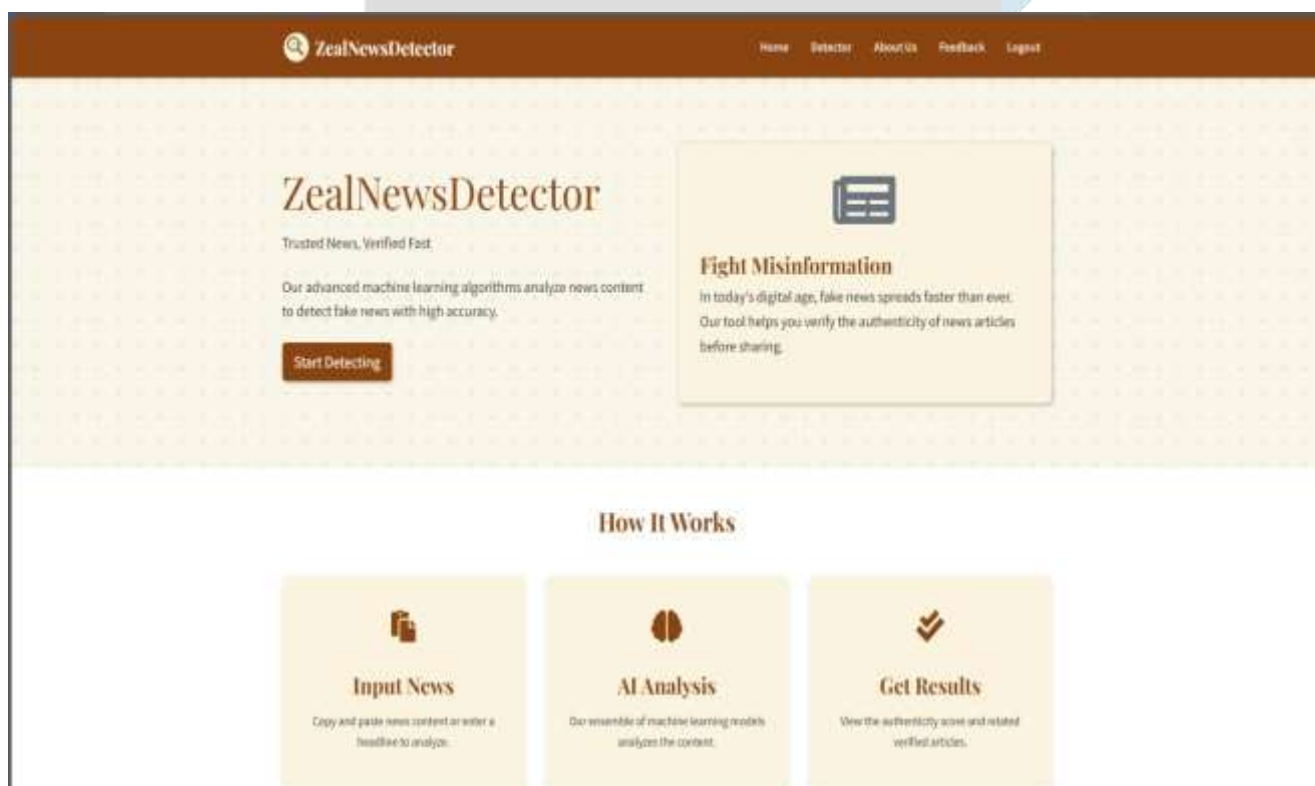
To transform unprocessed text into meaningful numerical formats suitable for machine learning, we utilize a technique called Term Frequency-Inverse Document Frequency (TF-IDF) for feature extraction. This is done with `sklearn.feature_extraction.text.TfidfVectorizer`, which is configured with several key parameters: `max_features=5000` to prevent

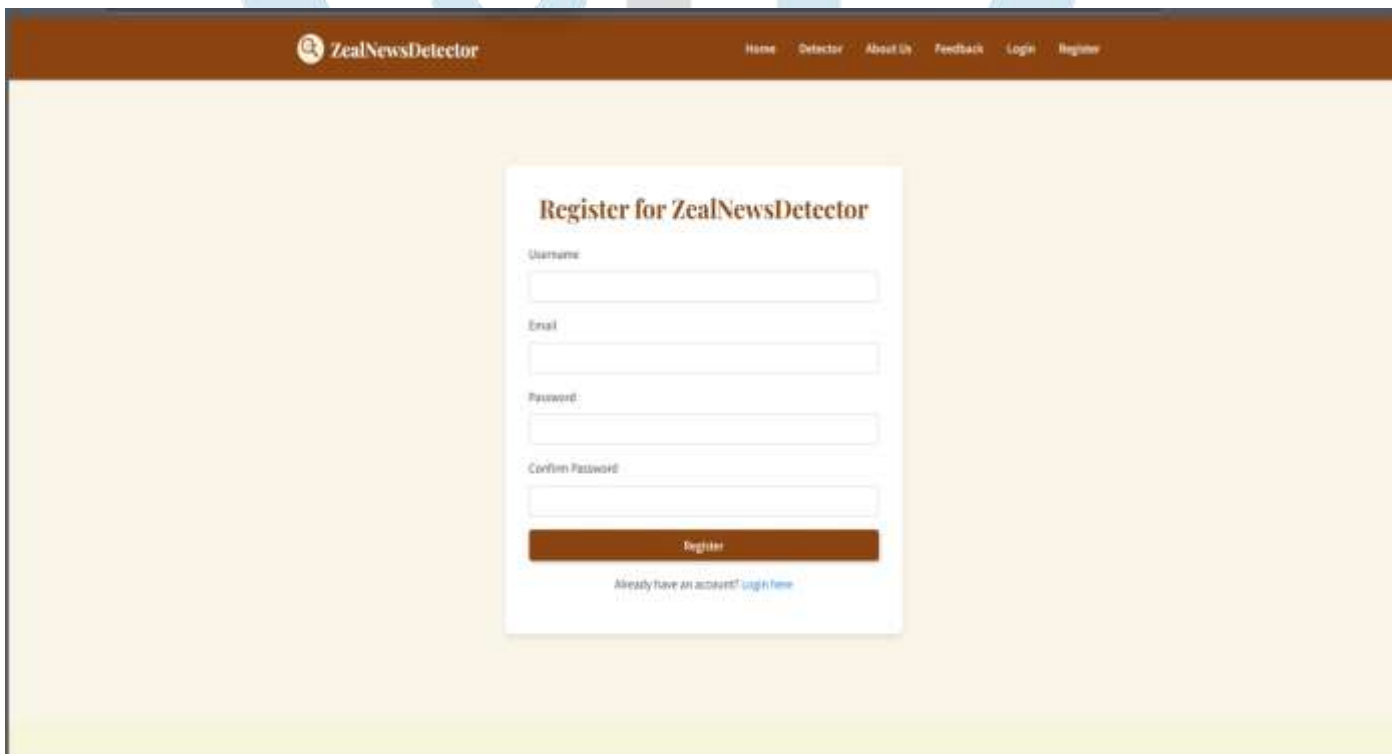
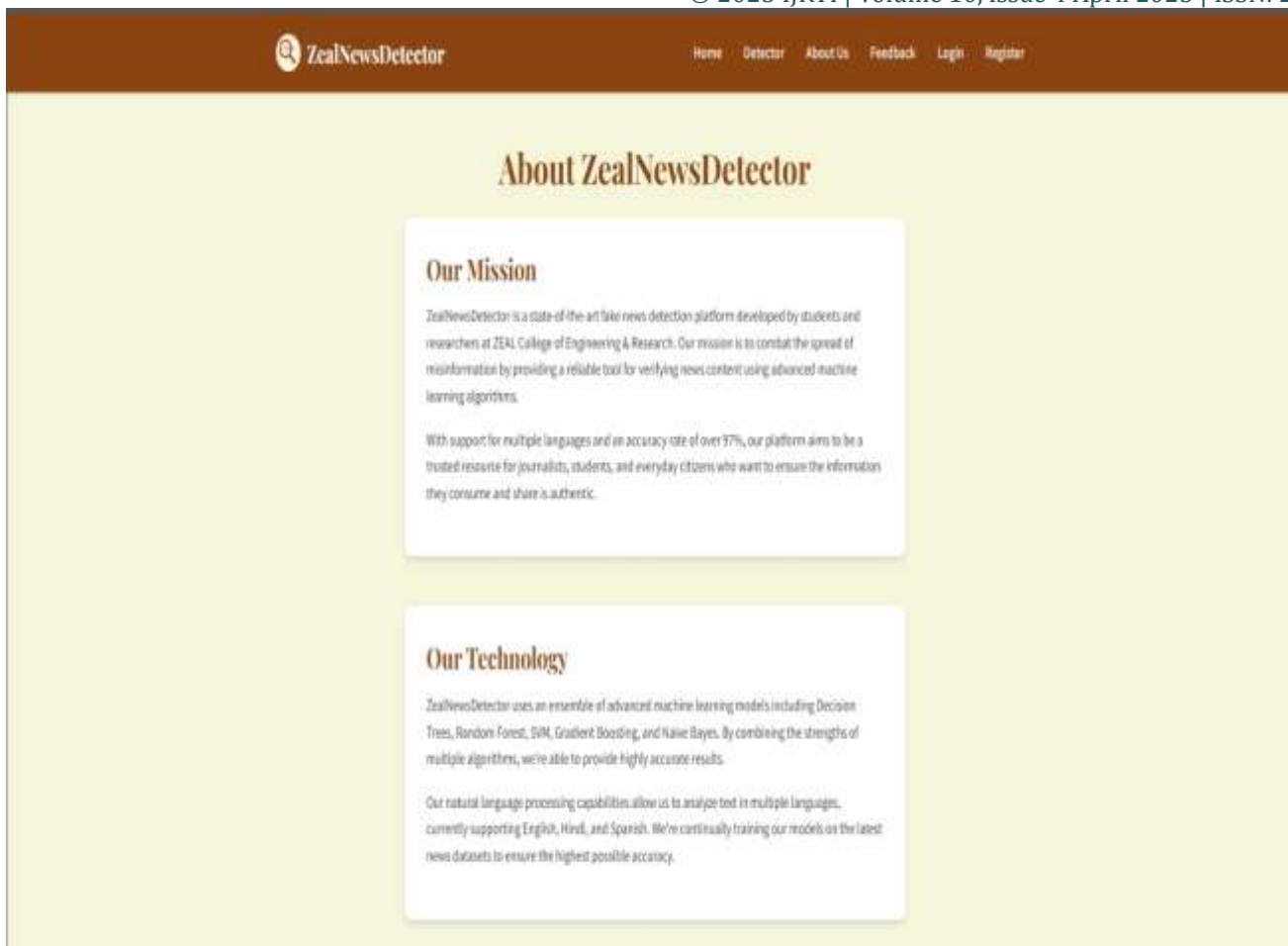



excessive vocabulary expansion, $\text{min_df}=5$ to remove very rare words, language-specific `stop_words` to cut down on noise, and `ngram_range=(1, 2)` to encompass both unigrams and bigrams.

Feature extraction is a multi-step process. Initially, we tokenize and normalize the input text to standardize word forms. Next, we remove stopwords based on the identified language to avoid including unnecessary terms in our dataset. Following that, the vectorizer generates the TF-IDF matrix, which assesses the significance of each term across the dataset. Finally, we convert this matrix into feature vectors that are ready for input into machine learning models. This thorough preprocessing captures the essential linguistic patterns necessary for accurately detecting fake news.

VI. SNAPSHOTS OF SYSTEM WORKING





 ZealNewsDetector
 [Home](#)
[Detector](#)
[About Us](#)
[Feedback](#)
[Login](#)
[Register](#)

Login to ZealNewsDetector


Username:

Password:

☐ Remember me

Login

(Don't have an account?) [Register here](#)

 ZealNewsDetector
 [Home](#)
[Detector](#)
[About Us](#)
[Feedback](#)
[Login](#)
[Register](#)

Feedback & Rating

How would you rate your experience?

☆☆☆☆☆

Type of Feedback

Select a feedback type


Your Feedback

Please share your thoughts here...

Email (optional)

Your email address (we'll only use this to follow up if needed)

Submit Feedback

 ZealNewsDetector
 [Contact Us](#)
 © 2025 ZealNewsDetector. All rights reserved.

Trusted News, Verified Fast
 [chikhalevaishar@gmail.com](#)
[About Us](#)
[Feedback](#)

 ZealNewsDetector
 [Home](#)
[Detector](#)
[About Us](#)
[Feedback](#)
[Logout](#)

Fake News Detector

Enter the news content to verify its authenticity

Input Language
English

Output Language
English

News Content:

Put the news article on headline here...

Verify News

ZealNewsDetector
Trusted News, Verified Fast

Contact Us
✉ ch4halevidatar@gmail.com
☎ +91 8767095186

© 2025 ZealNewsDetector. All rights reserved.
[About Us](#) | [Feedback](#)

 ZealNewsDetector
 [Home](#)
[Detector](#)
[About Us](#)
[Feedback](#)
[Logout](#)

Fake News Detector

Enter the news content to verify its authenticity

Input Language
English

Output Language
English

News Content:

Rohingya terror attack: India decides to keep Indian Water Treaty in abeyance, close Rital border post.

Verify News

ZealNewsDetector
Trusted News, Verified Fast

Contact Us
✉ ch4halevidatar@gmail.com
☎ +91 8767095186

© 2025 ZealNewsDetector. All rights reserved.
[About Us](#) | [Feedback](#)

[Verify News](#)

Verification Result

REAL

Probability of being real news: 48.76%

Individual Model Predictions

Decision Tree

FAKE

11.48%

Random Forest

FAKE

82.89%

SVM

REAL

41.33%

Gradient Boosting

REAL

0%

Naive Bayes

FAKE

82.89%

Related Verified Articles

Rage and despair after brazen attack kills 26 in Kashmir families are...

2025-04-23

[Read More](#)

Gold surges to new record high after Trump's repeated attacks on U.S...

2025-04-23

[Read More](#)

India

2025-04-23

[Read More](#)

Rage and despair after brazen attack kills 26 in Kashmir families are...

2025-04-23

[Read More](#)
ZealNewsDetector

Trusted News, Verified Fast

Contact Us
chikhalevaistan@gmail.com

+91 8767065180

ZEAL College of Engineering & Research, Pune - 411041

© 2025 ZealNewsDetector. All rights reserved.

[About Us](#) | [Feedback](#)

VII. RESULTS

Table 1: Accuracy metrics across different ML Models

Algorithm	Accuracy(%)	Precision(%)	Recall(%)	F1 Score(%)
Decision Tree	92.8	93.2	92.5	92.8
Random Forest	96.3	96.8	95.9	96.3
SVM (Linear)	93.5	94.1	92.8	93.4
Gradient Boosting	97.1	97.4	96.8	97.1
Naïve Bayes	91.2	92.7	89.5	91.0
Ensemble (Combined)	98.2	98.5	97.9	98.2

Table 2: Confusion Matrix Analysis

	Fake News (Predicted)	Real News (Predicted)
Fake News (Actual)	9312	142
Real News (Actual)	158	9246

VIII. CONCLUSIONS

The rising prevalence of false information, especially via online platforms, highlights the critical necessity for dependable systems to detect fake news. This study investigated the use of machine learning and natural language processing methods to create an effective means of classifying news articles as either genuine or fraudulent. A systematic process was established that included data collection, preprocessing, feature extraction utilizing TF-IDF, and the training of several classification models.

To enhance prediction accuracy, a combination of machine learning models Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting, and Naïve Bayes was employed through a weighted voting approach. Evaluating performance with metrics such as accuracy, precision, recall, and F1-score confirmed the proposed system's reliability and robustness. Additionally, the integration into a web-based platform demonstrated the system's capability for real-time, user-friendly application.

This research contributes to the ongoing efforts to combat disinformation by providing a scalable and practical framework for detection. Future improvements could incorporate deep learning models, more comprehensive datasets, and social media context analysis to enhance detection capabilities further.

REFERENCES

- [1] Abhinandan, Y., & Devaraju Venkata, R. (2023). A Comparative Study on Fake News Detection Utilizing Naive Bayes Classifier. This research examines the effectiveness of models created by Naive Bayes using Count Vectorizer and TF-IDF Vectorizer to detect fake news.
- [2] Altheneyan, A., & Alhaldag, A. (2023). Distributed Learning for Fake News Detection Based on Big Data and Machine Learning. They established a machine learning model employing a stacked ensemble approach alongside the FNC1 dataset, incorporating methods such as Ngrams and Hashing TF-IDF.
- [3] Almarashy, A. H. J., FeiziDerakhshi, M. R., & Salehpour, P. (2023). Improving Fake News Detection Through Multi-feature Classification. This work aims to enhance accuracy using CNN, BiLSTM, and TF-IDF within the framework of a neural network.
- [4] Adam, A. Y., Bhat, S. N., Lakshith M. Y., Shashank K. C., Shoaib Kamal, & Tripti Rao. (2023). Recent Machine Learning Techniques for Fake News Detection. They explored classifiers such as Random Forest, Naive Bayes, and Passive Aggressive in the context of fake news detection.
- [5] Narkhede, A., Patharkar, D., Chavan, N., Agrawal, R., & Dhule, C. (2023). Employing Machine Learning Algorithms for Fake News Detection. This study uses NLP techniques along with supervised algorithms for detection using Count Vectorizer and TF-IDF.
- [6] Park, M., & Chai, S. (2023). A User-Centered Model for Fake News Detection through Machine Learning. They utilize XGBoost and various models like SVM and RF to assess feature importance and evaluate the model.
- [7] Kong, J. T. H., Wong, W. K., Juwo, F. H., & Apriono, C. (2023). Developing a Fake News Detection Model through a Two-Stage Evolutionary Method. This approach employs Genetic Programming to understand feature correlations in tweet-based datasets.
- [8] Krishna, N. L. S. R., & Adimoolam, M. (2022). A Decision Tree Algorithm and SVM for Fake News Detection System. This study analyzes the accuracy of Decision Tree and SVM in identifying fake news.
- [9] Shahbazi, Z., & Byun, Y.-C. (2021). Detecting Fake Media Using NLP and Blockchain Techniques. They combine NLP methods with blockchain frameworks to identify fake media content on social media platforms.
- [10] Mridha, M. F., et al. (2021). An Extensive Review of Fake News Detection Using Deep Learning. This review focuses on various methods of detecting fake news, emphasizing NLP, deep learning, and metrics for evaluation.