A Comprehensive Approach to Fake News Detection with Adversarial Training and Model Explainability

Vlad-Constantin CRISTESCU and Stelian SPÎNU

***Abstract*—The proliferation of fake news has become a significant challenge in the digital era, with misinformation spreading rapidly across online platforms and social media. This paper explores the application of natural language processing and machine learning techniques to automatically detect and classify fake news articles. Multiple classification models, including traditional algorithms and modern deep learning architectures, were trained and evaluated on a dedicated dataset. To analyze the robustness of these models, adversarial attacks were applied using the TextAttack framework. Such attacks simulate subtle modifications in the input text, exposing potential vulnerabilities that may lead to misclassifications. In addition, explainability techniques such as LIME were employed to interpret model predictions and to better understand the factors influencing decision-making. A web-based application was developed to integrate the trained models into an interactive platform. The system allows users to analyze news articles by either providing the full text or submitting a URL, while administrators have access to model explanations, database management, and retraining functionalities.**

***Index Terms*—adversarial attacks, explainable AI, fake news detection, machine learning, natural language processing.**

I. Introduction

In the academic literature, the term *fake news* is used to refer to materials that mimic the formal characteristics of authentic news—such as headlines, structure, and journalistic style—but deliberately convey false information with the intent to influence public opinion. A growing body of recent studies emphasizes the intentional dimension of this phenomenon: fake news is not limited to mere inaccuracies or factual errors, but rather consists of fabricated content explicitly designed to misinform, deceive, or manipulate the perceptions of its audience [1].

In recent years, fake news has become increasingly present in our daily lives, largely due to the rapid growth of the internet and social media. In a digital environment where information spreads at an incredible speed, users are often left with limited means to verify the authenticity of what they read. As a result, fake news not only spreads quickly, but also has the potential to significantly influence the way people think and make decisions. Moreover, its impact extends beyond the individual level, affecting social stability, undermining economic trust, and eroding confidence in public institutions [2-4].

The harmful effects of misinformation can be observed across various domains. In the economic sphere, companies may face considerable losses due to false rumors, intentional defamation, or market manipulation. On a social and political level, the uncontrolled spread of false information can influence elections, deepen societal divisions, and contribute to the loss of credibility of official sources and the media. In times of crisis, such as pandemics or armed conflicts, the consequences can be even more severe—fueling confusion, panic, and misguided public responses.

What makes fake news especially difficult to counter is its deceptive nature. Many such articles are crafted using half-truths or are presented in a way that seems objective, making them harder to detect. This increases the challenge of fighting disinformation and highlights the need for automated tools that can reliably distinguish between real and fake content.

In this context, automatic fake news detection has emerged as a major area of interest in the fields of artificial intelligence (AI) and natural language processing (NLP). Modern analytical techniques can support informed decision-making, help reduce the negative impact of disinformation, and contribute to building a safer and more trustworthy information space. Our work aligns with this broader effort by proposing a content-based approach to analyzing online news articles.

The main contributions of this work are as follows:

1. Extensive model comparison: We implemented and evaluated 23 fake news detection models, including traditional classifiers (e.g., SVM, Logistic Regression), deep learning architectures (CNN, LSTM, CNN-LSTM), and Transformer-based models (BERT, RoBERTa). This allowed us to analyze the strengths and limitations of various approaches under a unified framework.

2. Explainability: We applied the LIME (Local Interpretable Model-agnostic Explanations) technique [5] to both correct and incorrect predictions, offering insights into the key linguistic features that influenced model decisions. We analyzed two representative examples using the LIME explainability technique, in order to better understand how

V. C. CRISTESCU is with the Military Technical Academy "Ferdinand I", Bucharest, Romania (e-mail: cristescuvlad1@gmail.com).

S. SPÎNU is with the Military Technical Academy "Ferdinand I", Bucharest, Romania (e-mail: stelian.spinu@mta.ro).

Digital Object Identifier 10.32754/JMT.2026.1.09

the models interpret and classify fake news content.

3. Adversarial robustness evaluation: We employed the TextAttack framework [6] to generate adversarial examples using TextFooler, TextBugger, and PWWS algorithms. Our experiments showed that even high-performing models like RoBERTa can be deceived by subtle word-level perturbations.

4. Adversarial training for robustness enhancement: We fine-tuned RoBERTa on an augmented dataset containing both original and adversarially perturbed texts. This significantly improved the model's resilience to attacks, while maintaining high accuracy on unaltered data (F1 score of 99%).

The source code of the implementation of the methods described in this paper is publicly available on GitHub at https://github.com/VladBeX2/RemoteServerAI.

II. Related Work

Automatic fake news detection has become a major topic of interest in recent research, especially in the context of the overwhelming flow of information on social media platforms. At the same time, the safety and robustness of NLP models are being intensely studied, particularly in terms of their susceptibility to manipulation or deception. In what follows, we present three relevant contributions from the literature, each addressing the problem from a different perspective.

A first significant contribution is the work that introduced the Combined Corpus dataset [7]. In this study, Khan et al. conducted a series of comparative experiments, evaluating various traditional models such as Support Vector Machines, Naïve Bayes, and Logistic Regression, as well as neural architectures based on convolutional neural networks and Transformer architectures. Their analysis revealed that Transformer-based models achieve significantly better performance than traditional approaches.

In another study, Dua et al. [8] proposed I-FLASH, an explainable system for fake news detection that not only classifies articles as fake or real but also provides insights into the reasoning behind its predictions. To evaluate the system, the authors created two new datasets—FactCheck and FactCheck2—collected from verified media sources and covering topics such as education, crime, and technology. A variety of models were tested, ranging from Logistic Regression with TF-IDF to LSTM with GloVe embeddings and pre-trained BERT. The results demonstrated competitive performance across several datasets, including LIAR and COVID-19. Moreover, the use of explainability methods such as LIME and SHAP (SHapley Additive exPlanations) allowed the identification of key words that most influenced the classification process, adding a layer of transparency to the system.

The third contribution [6] introduces TextAttack, a flexible platform designed to test the vulnerability of NLP models to textual perturbations. The authors provide standardized implementations of various adversarial attacks found in the literature, compatible with a wide range of models and NLP tasks, including BERT and RoBERTa. Furthermore, the platform supports the use of these attacks not only for testing but also for generating augmented datasets or training with modified text inputs, aiming to improve model robustness. The user-friendly nature of the framework makes it suitable for both research and practical applications.

III. Methods

This section outlines the methodology used to develop, train, and evaluate the fake news detection models in our study. We employed a diverse set of approaches, from traditional machine learning classifiers to deep learning architectures and Transformer-based models. Additionally, we analyzed model robustness through adversarial attacks and explored model explainability with LIME.

A. Dataset

For this study, we used the Combined Corpus dataset [7], which contains approximately 85,000 news articles published between 2015 and 2017. The articles cover a wide range of topics, including politics, health, economics, sports, and more. The dataset is relatively balanced, with 52% of the articles labeled as real and 48% as fake. The median article length is 511 words.

B. Data Preprocessing

To prepare the textual data for training, we applied different preprocessing pipelines depending on the type of model. For traditional machine learning and deep learning models, we removed irrelevant elements that could add noise or increase vocabulary size unnecessarily. This pipeline included:

• lowercasing all text,

• removing punctuation, digits, HTML tags, URLs, and special characters,

• eliminating stopwords (e.g., "the", "is", "and"),

• lemmatizing tokens using WordNet Lemmatizer from the Natural Language Toolkit [9].

For Transformer-based models (BERT [10] and RoBERTa [11]), only minimal preprocessing was applied. These models are pre-trained on raw text and rely on their internal tokenizers. We retained most of the original structure, replacing URLs with the "[URL]" token and reducing excessive whitespace.

C. Text Representation

We used three main strategies to convert text into numerical form:

• Bag of Words (BoW): We extracted the top 20,000 n-grams (unigrams, bigrams, and trigrams) to create sparse frequency vectors.

• TF-IDF (Term Frequency-Inverse Document Frequency): A statistical measure capturing the importance of terms relative to documents, also using the top 20,000 features.

• GloVe embeddings [12]: Each word was mapped to a

 Digital Object Identifier 10.32754/JMT.2026.1.09

300-dimensional vector. Article vectors were computed as the mean of all word vectors.

For deep learning models, we also trained supervised FastText embeddings [13] on our dataset using label supervision, capturing domain-specific semantics. These 300-dimensional embeddings were used in CNN, LSTM, and hybrid CNN-LSTM architectures.

D. Model Training

1) Traditional Classifiers

We initially trained a set of traditional classifiers—including Random Forest, Logistic Regression, Naïve Bayes, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN)—using the Combined Corpus dataset and three types of text representations: Bag of Words, TF-IDF, and GloVe embeddings.

To ensure data quality, we excluded articles shorter than 30 words. We then applied standard preprocessing steps: lowercasing, removing punctuation, numbers, HTML tags, URLs, and stopwords, followed by lemmatization. The dataset was split into 80% training and 20% testing, maintaining class balance. Additionally, 5-fold cross-validation was performed on the training set to improve evaluation robustness.

The three vectorization methods were configured as follows:

• BoW and TF-IDF: top 20,000 most frequent unigrams, bigrams, and trigrams.

• GloVe embeddings: 300-dimensional pre-trained vectors; each article represented as the mean of its token embeddings.

Each classifier was adapted accordingly:

• Random Forest: 100 decision trees.

• Logistic Regression: up to 1000 iterations for convergence.

• Naïve Bayes: Multinomial for sparse inputs (BoW, TF-IDF), Gaussian for dense embeddings (GloVe).

• SVM and KNN ($k = 5$): applied to both sparse and dense inputs, with Euclidean distance used for KNN.

2) Deep Learning Models

To evaluate the effectiveness of neural networks in fake news detection, we trained three deep learning architectures—CNN (Convolutional Neural Network), LSTM (Long Short-Term Memory), and a hybrid CNN-LSTM—using two types of word embeddings: pretrained GloVe vectors and supervised FastText embeddings.

For GloVe-based models, we applied the same preprocessing pipeline as in previous experiments. Texts were tokenized using Keras [14], restricted to the top 20,000 words, and padded to a fixed sequence length of 100. The embedding matrix was initialized using the *glove.6B.300d* vectors [12]. Words not found in the GloVe file were mapped to zero vectors.

The three architectures were structured as follows:

• LSTM: Bidirectional LSTM with 128 units, dropout 0.2, and a sigmoid output layer.

• CNN: 1D convolution (128 filters, kernel size 5), max-pooling, and two dense layers.

• CNN-LSTM: Conv1D (128 filters, kernel size 3), max-pooling, followed by an LSTM layer (64 units) and a sigmoid output layer.

In the second stage, we generated supervised FastText embeddings by training a FastText model directly on the fake/real classification task. The model was trained for 50 epochs with a learning rate of 0.01 and bigram features. Embeddings were 300-dimensional and tailored to the dataset vocabulary.

The same three architectures were reused, replacing the GloVe embeddings with FastText ones. Input sequences were fixed to a length of 100 tokens. The dataset was split into 70% train, 15% validation, and 15% test, preserving label distribution.

All models were trained for a maximum of 20 epochs using the Adam optimizer [15] and binary crossentropy loss. We applied early stopping with a patience of 3 epochs based on validation accuracy to avoid overfitting. Model performance was monitored on the validation set after each epoch.

3) Transformer Models

To evaluate the performance of Transformer-based models, we selected BERT and RoBERTa, both known for their outstanding results in text classification tasks. Fine-tuning was carried out using the Hugging Face Transformers library [16].

Unlike traditional and deep learning models, which require extensive preprocessing—such as removing stopwords or applying lemmatization—Transformer models do not need such steps. They include built-in tokenizers capable of handling punctuation, special characters, and other textual nuances automatically. Therefore, we opted for minimal cleaning, keeping the input text as close as possible to its original form. The only preprocessing steps applied were the removal of excessive whitespace and any remaining HTML characters.

For BERT, we used the bert-base-uncased pre-trained model, while for RoBERTa we used roberta-base. Both models were fine-tuned using the following training configuration:

• Training was performed for 5 epochs;

• A distributed training strategy was used across 3 GPUs, with a batch size of 16 per device;

• To simulate a larger batch, gradient accumulation was applied every 2 steps, resulting in an effective batch size of 96;

• A learning rate of 1e-5 was used;

• The training process included a warmup phase of 500 steps;

• Model performance was evaluated at the end of each epoch.

E. Adversarial Training

To assess model robustness, we generated adversarial examples using TextAttack [6], employing the TextFooler, TextBugger, and PWWS (Probability Weighted Word Saliency) algorithms. These attacks made minor modifications to words (e.g., synonyms, spelling changes) to mislead the model.

We created augmented training datasets combining both unaltered and adversarially perturbed examples, retraining the RoBERTa model on this hybrid data. The new dataset included 11,000 examples balanced across original and adversarial texts. Fine-tuning was repeated using the same hyperparameters. This improved model robustness without degrading performance on the unmodified inputs.

F. Explainability

We employed LIME (Local Interpretable Model-agnostic Explanations) [5] to interpret both correct and incorrect predictions. For each test instance, LIME provided a ranked list of words with positive or negative influence on the final decision.

To analyze common model weaknesses, we aggregated the most influential terms across all misclassified examples. This helped identify which types of words (e.g., political names, ambiguous verbs) were most likely to cause confusion, offering insights for future improvements.

IV. RESULTS

This section presents a comparative analysis of the performance of all models trained in our study, focusing on their ability to detect fake news accurately. We report the F1 score for the "Fake" class as the primary evaluation metric, as it balances precision and recall and is especially relevant in real-world applications where mislabeling fake news as real is critical.

A. General Model Performance

We evaluated 23 models across various architectures and vectorization techniques. Fig. 1 depicts a comparison of the F1 score for the "Fake" class across model categories.

The RoBERTa model achieved the highest F1 score of 99.5%, followed closely by BERT at 99.2%. These results highlight the strength of Transformer-based models pre-trained on large corpora in understanding nuanced language patterns typical of fake news.

Deep learning models also performed strongly. The best among them was the LSTM model trained with supervised FastText embeddings, reaching an F1 score of 96.22%. Hybrid architectures such as CNN-LSTM trained with either GloVe or FastText showed stable and robust behavior, achieving over 94% F1 score.

Traditional classifiers exhibited a wide range of results. Models such as SVM, Logistic Regression, and Random Forest, when used with TF-IDF or BoW, outperformed some neural models, with F1 scores nearing 98%. In contrast, KNN and Naïve Bayes, particularly with dense GloVe embeddings, struggled to exceed the 70–80% threshold, likely due to their limited ability to model context and semantics.

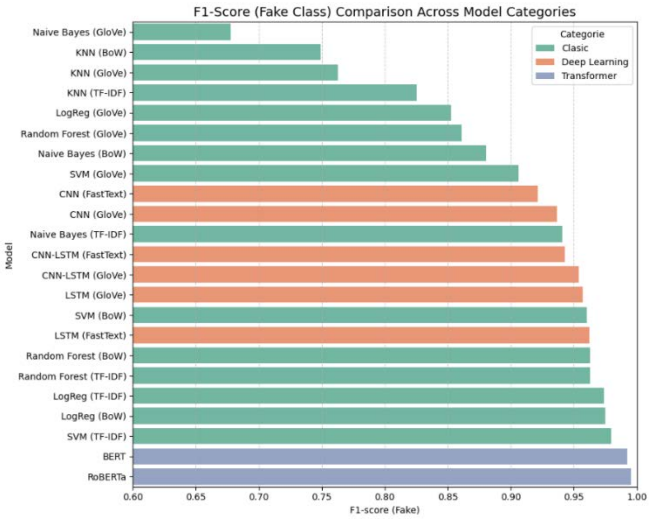

Figure 1. F1 score ("Fake" class) comparison across model categories

These results confirm that while deep architectures offer excellent performance, well-tuned traditional models with proper vectorization remain highly competitive. The choice of text representation was also shown to significantly impact the results, especially for traditional approaches.

B. Training Behavior

To better understand the learning dynamics, we monitored training and validation loss for selected models.

RoBERTa, for instance, as shown in Fig. 2, exhibited rapid convergence, with training and validation loss decreasing steadily and remaining low across all epochs. No signs of overfitting were observed, indicating strong generalization ability.

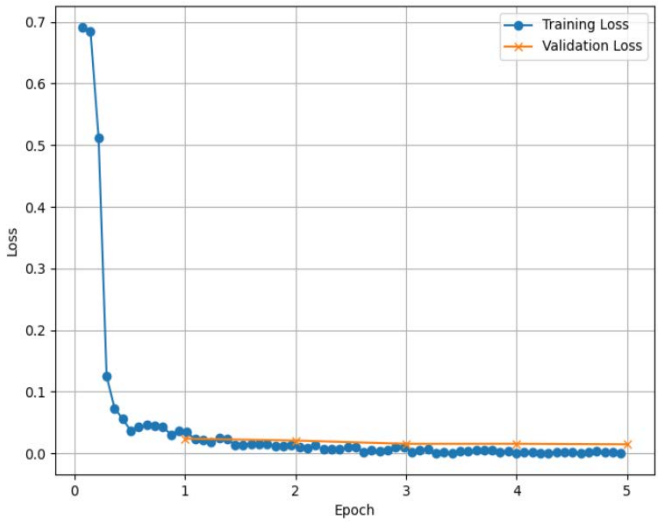

Figure 2. Training and validation loss for RoBERTa

As illustrated in Fig. 3, the CNN-LSTM model trained with FastText embeddings also demonstrated a healthy learning curve. Loss decreased smoothly across train, validation, and test sets, and accuracy increased steadily before plateauing. The relatively small differences between the curves indicate consistent performance across seen and unseen data.

 Digital Object Identifier 10.32754/JMT.2026.1.09

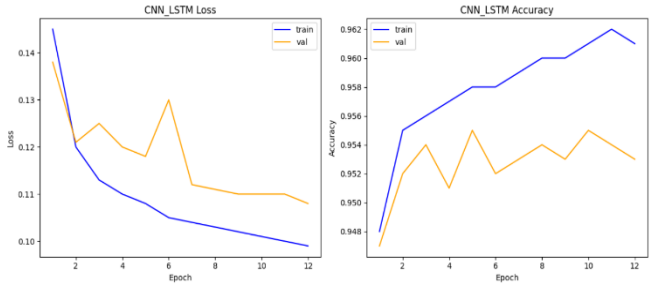

Figure 3. Learning curves for CNN-LSTM with FastText Embeddings

C. Adversarial Robustness with TextAttack

At this stage, we aimed to improve the robustness of the RoBERTa model against attacks involving intentional perturbations of input texts designed to mislead the classifier. To achieve this, we applied three adversarial attack methods—TextFooler, TextBugger, and PWWS—using the TextAttack library.

We retained the same data split used in the original training of RoBERTa to ensure a consistent comparison baseline. Thus, the Combined Corpus was divided into training, validation, and test sets, maintaining the same proportions and class distribution.

This setup allowed us to directly compare the performance of the original model with that of the retrained version using manipulated data. To generate adversarial examples, we used samples from each subset and preserved the original versions of the articles alongside the modified ones. In the final stage, both the unaltered and adversarially modified texts were combined to create a balanced training dataset.

Table I presents detailed statistics on the number of attacks generated for each subset, how many were successful, and the class (fake or real) of the targeted articles. It can be observed that all three attack methods significantly disrupted the predictions of the original model, with a higher success rate in the case of *fake* articles.

TABLE I. ADVERSARIAL ATTACK SUCCESS RATES ON ROBERTA

Attack	Subset	No. of articles	Successful	Fake → Real	Real → Fake
PWWS	Train	3000	2349	1439	910
	Val	800	623	405	218
	Test	800	644	415	229
TextBugger	Train	3000	1971	1227	744
	Val	800	537	347	190
	Test	800	540	319	221
TextFooler	Train	3000	2486	1585	901
	Val	800	681	439	242
	Test	800	686	423	263

Building on these results, we constructed a new training set that included both the original articles and their adversarially modified counterparts that had successfully deceived the model. For each modified article, we also included its original, correctly labeled version in the dataset, allowing the model to learn the subtle differences between the two.

The resulting training set contained 11,000 articles in total, consisting of 6,000 fake and 5,000 real samples, evenly balanced between original and perturbed texts. The validation and test sets were built similarly, each containing 1,200 real and 2,000 fake articles.

After preparing the augmented dataset, we performed an additional fine-tuning phase on the previously trained RoBERTa model. The goal was for the model to learn to recognize adversarial perturbations and enhance its resistance to them, without compromising performance on the unmodified inputs.

Following this process, the model achieved an F1 score of 99% on both the adversarially augmented set and the original dataset, demonstrating that training with such data effectively improved robustness without negatively impacting overall accuracy.

After generating perturbed examples using the PWWS, TextBugger, and TextFooler methods, we aimed to better understand how these attacks manage to disrupt RoBERTa's predictions. To this end, we analyzed the most frequently replaced words and the grammatical categories targeted by the attacks.

On average, each modified article underwent approximately 33.91 word substitutions. Despite the high number of alterations, the attacks were crafted in such a way that the overall meaning of the article remained largely intact.

Fig. 4 highlights the parts of speech most commonly affected by the attacks. Nouns and verbs dominate this distribution, which is expected given their central role in constructing the meaning of sentences. These are followed by adjectives and adverbs, which provide nuance, while other grammatical categories were targeted less frequently.

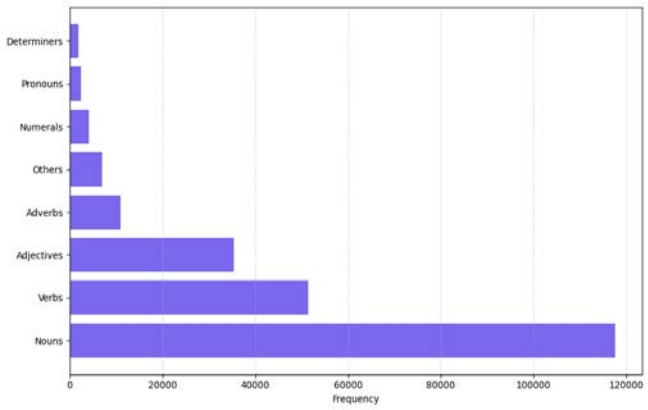

Figure 4. Frequency of word categories replaced during adversarial attacks

Fig. 5 shows the most frequently replaced verbs, along with the top three substitute forms for each. For instance, the verb "*said*" was replaced over 700 times, primarily with more formal or technical alternatives such as "*aforesaid*" or "*stated*". Other verbs like "*think*", "*know*", or "*make*" were also commonly substituted with less typical or more stylized synonyms. These subtle changes can significantly impact the model's prediction, especially when they occur frequently within an article.

Similarly, in Fig. 6, we analyzed the most frequently modified nouns. It was observed that articles were often

altered by replacing common terms such as "*government*", "*support*", or "*president*" with semantic alternatives like "*council*", "*patronise*", or "*chairwoman*". Although these words may appear similar, they can carry different connotations or appear in distinct contexts, thereby affecting the model's ability to accurately interpret linguistic subtleties.

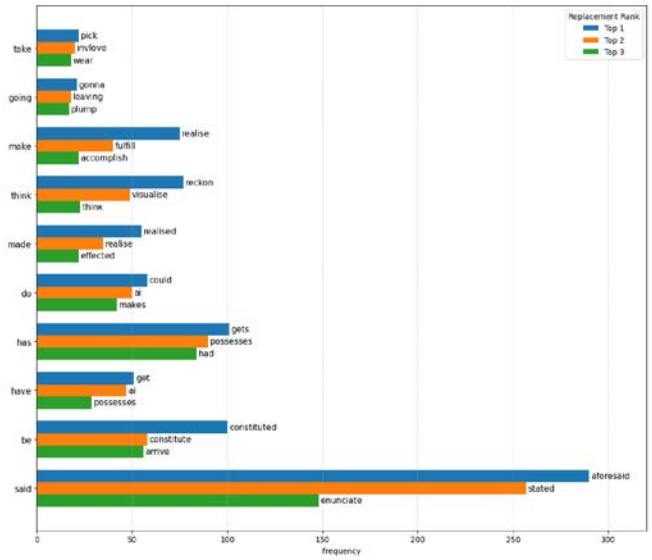

Figure 5. Most frequently replaced verbs and their substitutes in adversarial attacks

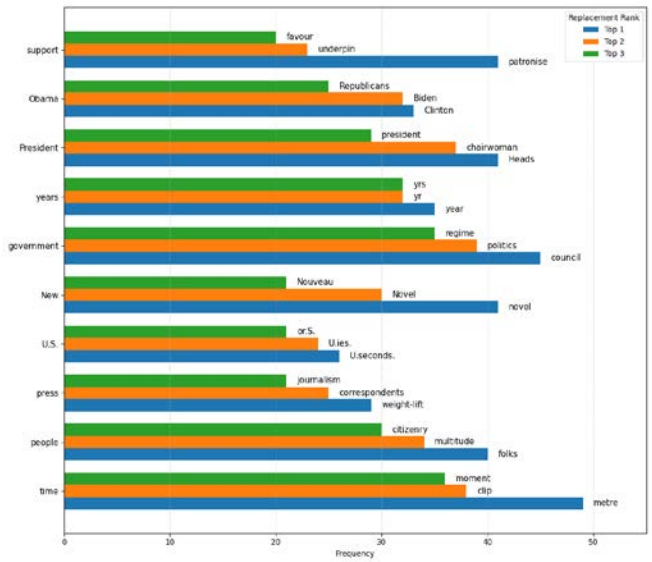

Figure 6. Most frequently replaced nouns in adversarial texts and their substitutes

D. Explainability with LIME

We present below two representative examples that illustrate how the LIME method can support the interpretation of decisions made by classification models. The first example highlights a misclassification case made by a traditional model, offering insights into the text features that contributed to the error. The second example shows a correct prediction made by the RoBERTa model, emphasizing how explanatory analysis helps reveal the elements the model considers relevant in its decision-making process. These visualizations contribute to increasing model transparency and to a deeper understanding of their behavior when processing texts of varying complexity.

The text analyzed in Fig. 7 was artificially created as part of the project and was designed as a fake news article to test model performance. The article describes a supposed proposal by the European Parliament to modify Schengen zone regulations, presenting unverified claims about potential negotiations and official positions. Despite this, the Logistic Regression model, trained on TF-IDF representations, misclassified the article as real. To understand the reasons behind this error, we applied the LIME method, which highlights the contribution of individual words to the model's decision.

It can be observed that terms such as "*European*", "*Parliament*", and "*declined*" had a strong positive influence on the classification toward the Real class. The model likely learned to associate these terms with authentic articles, as they frequently appear in legitimate news. In contrast, only a few words, such as "*sources*" or "*reportedly*", were identified as indicators for the Fake class. This suggests that the model gives greater weight to terms that convey formality and credibility, without truly assessing whether the information presented is logically sound.

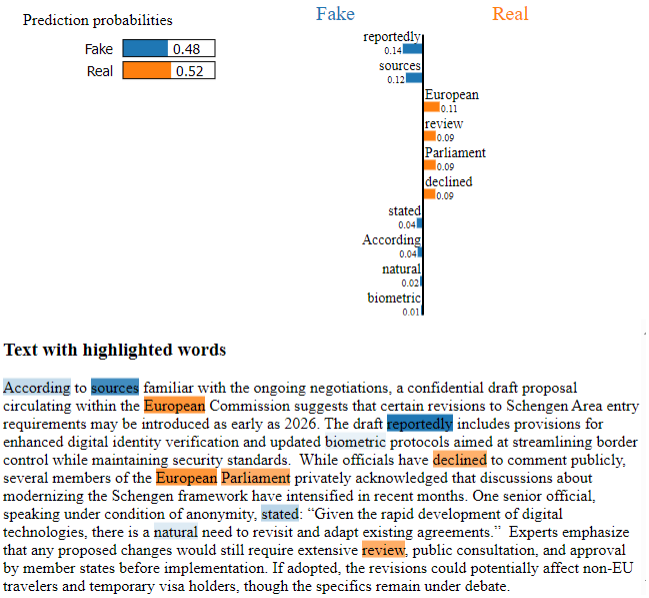

Figure 7. LIME explanation for misclassified fake news article by Logistic Regression model trained on TF-IDF representations

The limitation becomes clear when considering that TF-IDF-based models cannot capture semantic relationships or the broader context of the article. Through LIME analysis, we can uncover the elements that led to the misclassification, offering valuable insight into the vulnerability of simple models that rely solely on term frequency and can be easily misled by text that mimics journalistic style.

Fig. 8 presents an article that was artificially created as part of this project to test the model's behavior. The article describes a supposed discovery of beneficial effects of coffee in preventing cancer. The text was evaluated by the

 Digital Object Identifier 10.32754/JMT.2026.1.09

RoBERTa model, which correctly classified it as fake, with a confidence of 91%. Although the article appears at first glance to follow a journalistic tone and includes scientific terminology, the LIME analysis reveals the underlying factors that influenced the model's decision.

Notably, terms such as "*study*", "*researchers*", and "*participants*" had a significant impact on the model's classification toward the Fake label. These elements are often found in sensational or clickbait-style articles that use vague or exaggerated scientific claims to mislead readers. The model appears to have learned to associate such linguistic patterns with a higher likelihood of misinformation.

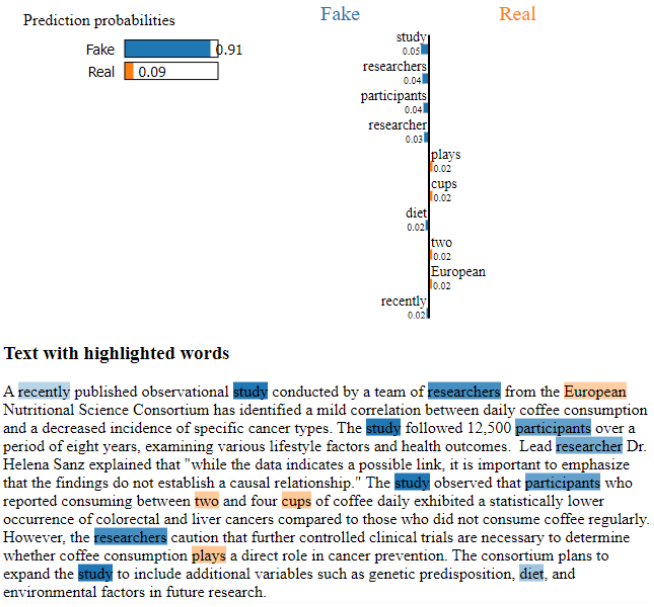

Figure 8. LIME explanation for correctly classified fake news article by RoBERTa

This example illustrates how the model has learned to identify specific language cues that are frequently associated with fake news. By applying LIME, we can clearly observe which elements of the text the model considered most relevant in making an accurate prediction.

V. Conclusion

This project focused on developing a fake news detection system using modern techniques from natural language processing and machine learning. We explored and compared the performance of various classification models, including traditional algorithms, neural networks, and Transformer-based architectures, trained on a dedicated dataset for misinformation detection.

To assess model robustness, we incorporated adversarial training with perturbed versions of news articles, concentrating in particular on RoBERTa. Using the TextAttack platform, we applied multiple well-known attack strategies, which revealed vulnerabilities in NLP models when facing subtle textual manipulations. This process enabled us to fine-tune more resilient models. In parallel, we employed the LIME explainability method to analyze and interpret individual model predictions, gaining insight into the linguistic features influencing classification outcomes.

In addition to model development, we built a web application that allows users to test potentially misleading articles. Platform administrators also have access to model explanations, dataset management, and the ability to retrain models on new data.

Looking ahead, the analysis of model vulnerabilities could be extended by exploring more advanced perturbation strategies, including adaptive or targeted attacks that better reflect real-world information manipulation scenarios. At the same time, the integration of additional explainability tools—such as SHAP, attention distribution analysis in Transformer models, or the combination of local and global interpretability techniques—could provide deeper insights into how these models process and classify information. Such investigations would contribute to a better understanding of model behavior and current limitations, offering clear directions for improving robustness and transparency in future research.

References

[1] 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, Sep. 2020. doi:10.1145/3395046

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

[3] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," in *27th International Conference on Computational Linguistics (COLING)*, 2018, pp. 3391–3401.

[4] N. Capuano, G. Fenza, V. Loia, and F. D. Nota, "Content-based fake news detection with machine and deep learning: A systematic review," *Neurocomputing*, vol. 530, no. C, pp. 91–103, Apr. 2023. doi:10.1016/j.neucom.2023.02.005

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144. doi:10.1145/2939672.2939778

[6] J. X. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP," *arXiv [cs.CL]*, 2020. doi:10.48550/arXiv.2005.05909

[7] J. Y. Khan, M. T. I. Khondaker, S. Afroz, G. Uddin, and A. Iqbal, "A benchmark study of machine learning models for online fake news detection," *Mach. Learn. Appl.*, vol. 4, p. 100032, Jun. 2021. doi:10.1016/j.mlwa.2021.100032

[8] V. Dua, A. Rajpal, S. Rajpal, M. Agarwal, and N. Kumar, "I-FLASH: Interpretable fake news detector using LIME and SHAP," *Wirel. Pers. Commun.*, vol. 131, no. 4, pp. 2841–2874, Jul. 2023. doi:10.1007/s11277-023-10582-2

[9] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python*. Sebastopol, CA: O'Reilly Media, 2009.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional Transformers for language understanding," in *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423

[11] Y. Liu *et al.*, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv [cs.CL]*, 2019. doi:10.48550/arXiv.1907.11692

[12] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. doi:10.3115/v1/D14-1162

[13] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *15th Conference of the European*

Chapter of the Association for Computational Linguistics (EACL), 2017, pp. 427–431.
[14] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.
[15] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015. doi:10.48550/arXiv.1412.6980
[16] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45. doi:10.18653/v1/2020.emnlp-demos.6