International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

Vibeshield – Multimodal Misinformation Detection In Social Media

Ms Sameera Begum[1],S.Pranavi[2],V.Sahithi[3],L.Srihitha[4]

[1]Assistant Professor; Department Of Computer Science And Engineering(Ai&Ml) Bhoj Reddy Engineering College For Women Hyderabad India.

[2,3,4]B.Tech Students; Department Of Computer Science And Engineering(Ai&Ml) Bhoj Reddy Engineering College For Women Hyderabad India.

Abstract

The rapid growth of social media has led to an unprecedented spread of misinformation in the form of text, images, and videos, creating significant challenges for public awareness and online content moderation. Detecting such deceptive content is particularly difficult when it involves multiple types of data simultaneously. This study presents a multimodal misinformation detection system that analyzes both textual and visual information to improve accuracy. The framework utilizes advanced models for understanding text and extracting image features, which are then combined and processed by a neural network to classify content as real or fake. In addition to providing predictions, the system offers interpretability by using techniques that visually highlight influential regions in images and focus areas in text, helping users understand the reasoning behind the model's decisions. The system is implemented using a user-friendly interface and a backend processing framework, supported by modern deep learning and computer vision libraries. It can handle text-only, image-only, and combined inputs, offering flexibility and robustness. By integrating multimodal analysis with explainable outputs, the proposed approach provides an effective and transparent solution for tasks such as social media monitoring, fact-checking, and content moderation.

Keywords*: This study focuses on multimodal misinformation detection by combining text and image analysis to classify digital content as real or fake. The framework employs deep learning techniques, including neural networks and feature fusion, and incorporates explainable artificial intelligence (XAI) methods to provide interpretability for users. Applications of the system are particularly relevant for social media monitoring, content moderation, and fact-checking, offering a flexible and transparent approach to detecting deceptive online information.*

Introduction

The widespread adoption of social media has significantly accelerated the circulation of false and misleading information. Content that is inaccurate or deceptive often spreads more rapidly than verified information, potentially causing social, political, and economic consequences. Misinformation appears in a variety of formats, including text posts, images, memes, and videos, which poses a challenge for traditional detection systems that typically focus on a single type of content. To address this issue, this research proposes a system capable of analyzing both textual and visual information simultaneously. By combining advanced language understanding models with image feature extraction techniques, the system can detect misleading content more effectively than approaches that rely on only text or images. A key aspect of this system is its ability to provide explanations for its predictions. Using techniques that highlight important features in both text and images, the system allows users to interpret why a particular piece of content was classified as misleading. This level of transparency increases trust and usability for real-world applications. The system is designed to support multiple content formats, including text, images, and video, providing a comprehensive approach to misinformation detection across modern digital platforms.Existing misinformation detection approaches are mostly limited to analyzing a single modality. Text-based methods use natural language processing techniques to identify fake news, but they ignore visual content that may convey deception. Image-based methods employ convolutional networks to detect manipulated images without considering accompanying text. Manual fact-checking relies on human verification, which is time-consuming and not scalable, while keyword-based filtering systems use rule-based approaches that often produce false positives. These limitations highlight the need for a unified multimodal system that can process both textual and visual information while providing interpretable results, enabling more accurate and trustworthy detection of misinformation.

Literature Survey

Research on misinformation detection has progressed considerably with advances in deep learning, natural language processing, and computer vision. Early efforts primarily focused on analyzing textual content. Kumar and Singh (2021) provided a comprehensive survey on text-based detection methods, highlighting the effectiveness of transformer models such as BERT and RoBERTa in capturing deep contextual information. While these models perform well in identifying textual misinformation, they are limited in detecting content

that relies on misleading or manipulated visual media, highlighting the need for multimodal approaches. Zhang et al. (2022) reviewed various techniques for fake news detection, categorizing them into text-only, image-only, and multimodal systems. Their findings suggest that models combining convolutional neural networks for images and transformer-based models for text generally outperform unimodal approaches. However, challenges remain, including the scarcity of large-scale multimodal datasets and poor generalization across different content domains.Patel and Roy (2023) introduced a framework that leverages CLIP (Contrastive Language–Image Pre-training) to align textual and visual representations, demonstrating that such embeddings are effective in identifying inconsistencies between images and associated text. Despite these advantages, CLIP alone may not detect subtle image manipulations or deepfake content, which require specialized vision-focused models. In the context of manipulated media, Chen et al. (2023) developed a deepfake detection system based on spatio-temporal convolutional neural networks, focusing on temporal inconsistencies such as unnatural facial movements, blinking patterns, and lighting variations in videos. Their work underscores the importance of incorporating temporal features for video-based misinformation detection.Explainable artificial intelligence has emerged as a crucial component in modern detection systems. Sharma and Huang (2023) explored the use of techniques such as LIME and SHAP to interpret model predictions, showing that explainable models enhance user trust by highlighting key textual and visual features influencing decisions. Nevertheless, many existing systems still lack sufficient transparency, which limits their practical applicability. More recently, Banerjee et al. (2024) proposed a multimodal fusion architecture that integrates text features from BERT, image features from ResNet or CLIP, and metadata within a unified transformer framework. Their results demonstrate that this approach improves both accuracy and robustness, particularly for content where misinformation is contextually subtle rather than overtly false.Overall, while significant progress has been made in multimodal misinformation detection, several gaps remain. Many systems focus on limited modalities, lack effective explainability mechanisms, or are not designed for real-time deployment. To overcome these limitations, the proposed system integrates transformer-based text analysis using RoBERTa, deep convolutional image processing with ResNet-50, feature fusion strategies, and explainability techniques such as Grad-CAM and attention visualization. This combined approach aims to enhance detection accuracy, transparency, and scalability, providing a reliable solution for real-world misinformation detection tasks.

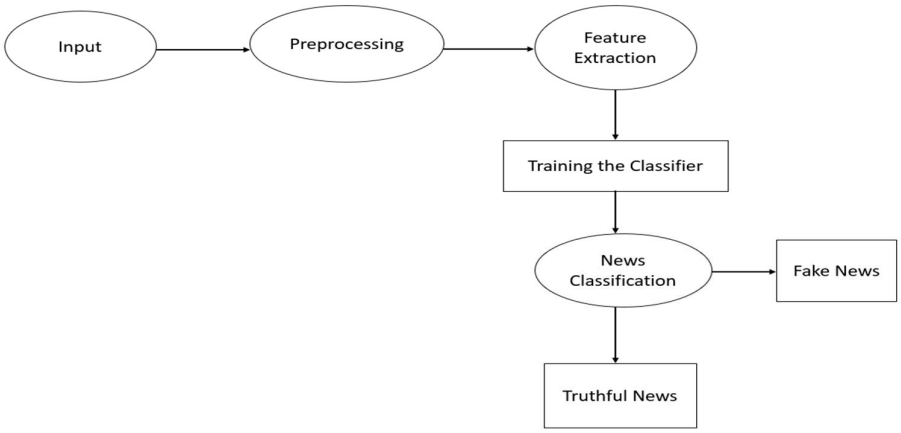
Methodology

The methodology of the proposed system is designed to address both the functional and non-functional requirements necessary for effective multimodal misinformation detection. Functionally, the system allows users to upload text, image, and video content for analysis. Uploaded content is processed to generate predictions regarding its authenticity, and the system provides visual explanations that help users interpret the results. On the backend, the system extracts features from text using transformer-based models and from images using deep convolutional networks. These features are then combined through a feature fusion approach and processed by a classifier to determine whether the content is genuine or misleading. In addition to generating predictions, the system outputs explainability visualizations and prediction confidence scores to ensure transparency and user trust.Non-functional requirements are equally important in guiding the design of the system. The framework is optimized for high performance, enabling real-time detection with minimal delay. It is scalable, supporting large data streams from multiple sources simultaneously, and incorporates secure data processing to protect user privacy. Reliability is ensured through consistent outputs even under high system load, while usability is addressed by providing a simple and intuitive interface. The modular architecture allows for easy maintenance and integration of updates, ensuring the system remains adaptable to evolving misinformation patterns.From a computational perspective, the system relies on widely available software and hardware resources. It can operate on Windows or Linux platforms, and is implemented primarily in Python (version 3.10) using frameworks such as PyTorch for deep learning, Transformers for text analysis, Streamlit for the user interface, and OpenCV for image and video processing. Hardware requirements include a processor equivalent to Intel i5 or higher, a minimum of 8 GB of RAM, and at least 10 GB of storage, with high-speed internet connectivity recommended for handling live data streams efficiently. This combination of functional, non-functional, and computational design considerations ensures that the proposed system is robust, scalable, and capable of providing accurate and explainable misinformation detection across multiple content types.

Design

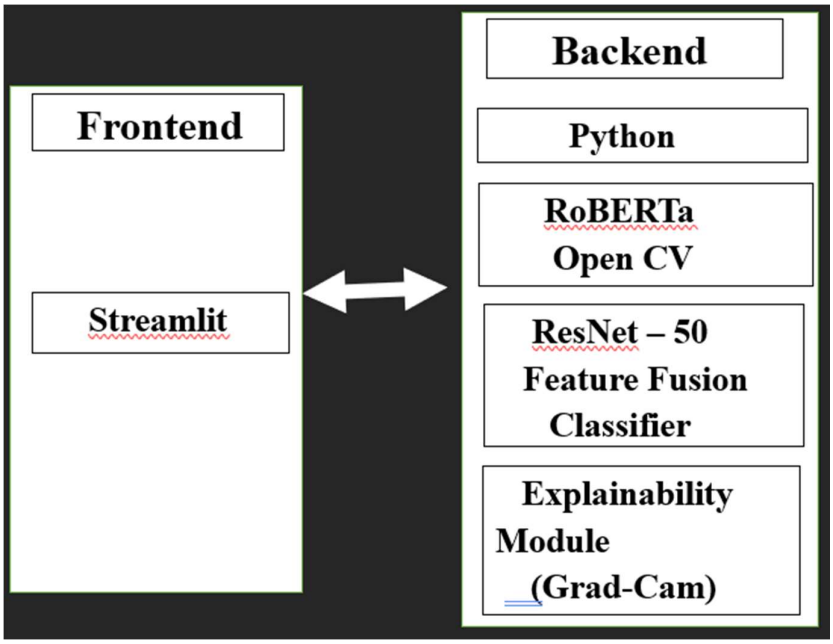
Software Architecture

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026



The software architecture defines the high-level structure of the system, specifying the software components, their responsibilities, and interactions. In this project, a modular architecture is adopted to enhance maintainability and scalability. The system is divided into distinct modules, each responsible for a specific functionality. This separation of concerns ensures that changes in one module do not significantly impact others, improving system reliability.The architecture follows a **layered approach**, consisting of the presentation layer, business logic layer, and data access layer. The presentation layer handles user interactions and input validation, the business logic layer processes the core functionalities, and the data access layer manages interactions with the database. This design supports flexibility, allowing for future integration with additional services or components.

Technical Architecture



The technical architecture specifies the hardware, software, and network infrastructure required to support the system. The system is designed to run on a client-server model, where the server hosts the application logic and database, and the client interfaces provide user interaction.The technical architecture ensures optimal performance, data security, and system reliability, supporting concurrent users efficiently.

Implementation

Technologies

The proposed system is implemented using Python-based tools and modern deep learning frameworks, enabling efficient processing of multimodal data, feature extraction, classification, and generation of explainable outputs. The system is designed to handle text, images, and video inputs through a web-based interface, supporting both single- and multimodal analysis.The development environment integrates key Python libraries including PyTorch

for deep learning model implementation, Transformers for natural language processing, OpenCV for image and video handling, and Streamlit and Flask for web deployment. Users interact with the system via a Streamlit interface, which allows for uploading text, images, and video files. OpenCV facilitates frame extraction from video inputs and image preprocessing.Data preprocessing is tailored to each modality. For text, the pipeline includes cleaning, normalization, tokenization using the RoBERTa tokenizer, and the generation of padded sequences with attention masks. For images, preprocessing involves resizing to 224 × 224 pixels, RGB conversion, and normalization based on ImageNet standards. These steps ensure that inputs are compatible with the feature extraction models.Feature extraction employs pretrained models to capture deep semantic and visual representations. Text embeddings are generated using RoBERTa, producing 768-dimensional vectors, while ResNet-50 extracts 2048-dimensional visual features from images. The system then performs multimodal feature fusion through concatenation, creating unified representations that integrate textual and visual information.For classification, a fully connected neural network processes the fused features and predicts whether the input corresponds to real or fake content. To enhance transparency and trust, an explainability module leverages Grad-CAM to highlight important regions in images and attention-based visualization to indicate critical words in text. The results, including prediction labels, confidence scores, and visual explanations, are presented interactively through the Streamlit interface.

Preprocessing Steps

The preprocessing workflow ensures high-quality inputs for model training and inference. Text preprocessing includes removing extra spaces, normalizing characters, tokenization using byte pair encoding (BPE), padding sequences to 128 tokens, and generating attention masks. Image preprocessing converts images to RGB format, resizes them to 224 × 224 pixels, applies data augmentation, converts them to tensors, and normalizes them using ImageNet statistics. These steps standardize inputs across modalities, enhancing the robustness of the system.

System Workflow and Pseudocode

The overall system workflow is controlled by a main application function that handles user authentication, interface rendering, model loading, input processing, and output display. The application provides separate modes for text-only, image-only, multimodal, and video analysis.For multimodal inputs, the system collects text and image data, applies preprocessing, extracts features using RoBERTa and ResNet-50, fuses them into a unified feature vector, and performs classification. Video inputs are processed by extracting frames via OpenCV and analyzing each frame in conjunction with textual metadata. The user interface provides interactive tabs for each input type and displays the results with visual explanations for both text and images.User authentication is integrated into the interface to control access. The sidebar displays user information and provides logout functionality. The architecture ensures that models are loaded efficiently using caching mechanisms, and predictions are computed in real-time to support interactive analysis.

Software Testing

Software testing is a systematic process aimed at evaluating the functionality, performance, and reliability of a software application to ensure that it meets specified requirements and operates without defects. In the context of intelligent systems and artificial intelligence applications, such as misinformation detection platforms, testing becomes particularly critical. These systems are increasingly relied upon to analyze online content and assist users in identifying false or misleading information.Accurate operation of such systems is essential, as incorrect predictions or processing errors can lead to the dissemination of misinformation or the misclassification of legitimate content. This can compromise user trust and diminish system credibility. Consequently, rigorous testing is vital to ensure that the software performs reliably, efficiently, and securely under various conditions. In this project, the system is evaluated across multiple input types, including text-only, image-only, and multimodal data, to verify the correctness of preprocessing, model predictions, feature fusion, and explainability outputs.The key benefits of testing in this context include accuracy assurance, reliability across diverse input types, efficient performance, protection of user data, and overall user satisfaction. Accurate predictions and clear visual explanations enhance user confidence in the system while maintaining high-quality operational standards.

Test Objectives

The primary objective of testing in this project is to ensure that the system functions correctly, efficiently, and according to the specified requirements. Testing aims to validate each module, including input handling, data preprocessing, feature extraction, classification, and output display.Specific testing goals include: verifying system functionality across all modules; ensuring high model accuracy in distinguishing real and fake content; validating preprocessing steps for both text and images; confirming effective multimodal feature fusion; assessing the correctness of explainability outputs such as Grad-CAM and attention visualizations; evaluating performance in real-time or near real-time scenarios; ensuring robustness under different input conditions; and detecting and resolving any software defects. These

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

objectives collectively ensure a reliable and user-friendly system.

Stages of Testing

Software testing is conducted through multiple stages, each focusing on different levels of the application.

Unit Testing represents the initial phase, in which individual components or functions are tested to ensure correct behavior. Typically employing white-box testing methods, unit tests verify that each element of the system operates as intended. This phase allows developers to identify and fix issues early, promoting more efficient integration in subsequent stages.

Integration Testing evaluates the interactions among combined units or modules to detect interface defects and ensure cohesive operation. Despite individual units functioning correctly, improper integration can compromise overall system performance. Various testing approaches may be applied depending on module definitions and interdependencies.

System Testing involves assessing the complete application as a unified entity. Conducted in an environment that closely resembles production, this stage confirms that the system meets functional, technical, and business requirements. Independent testers perform system testing to maintain objectivity and verify compliance with quality standards.

Acceptance Testing, or User Acceptance Testing, is the final stage conducted to determine whether the system is ready for deployment. Users evaluate whether the application fulfills intended business needs and operational requirements. Successful completion of acceptance testing indicates that the software is ready for production release, ensuring higher reliability and user satisfaction.

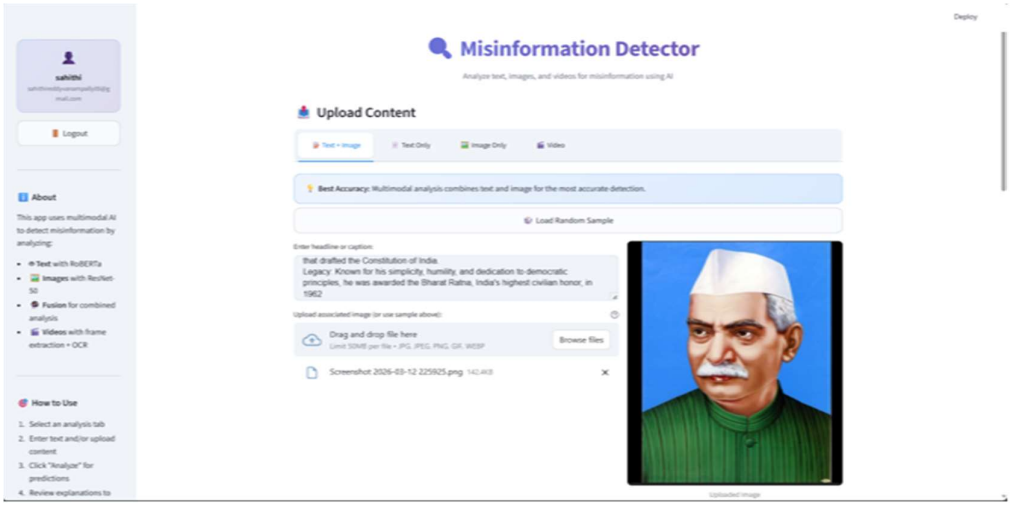
Types of Testing

Two primary testing methodologies are employed to ensure comprehensive evaluation:

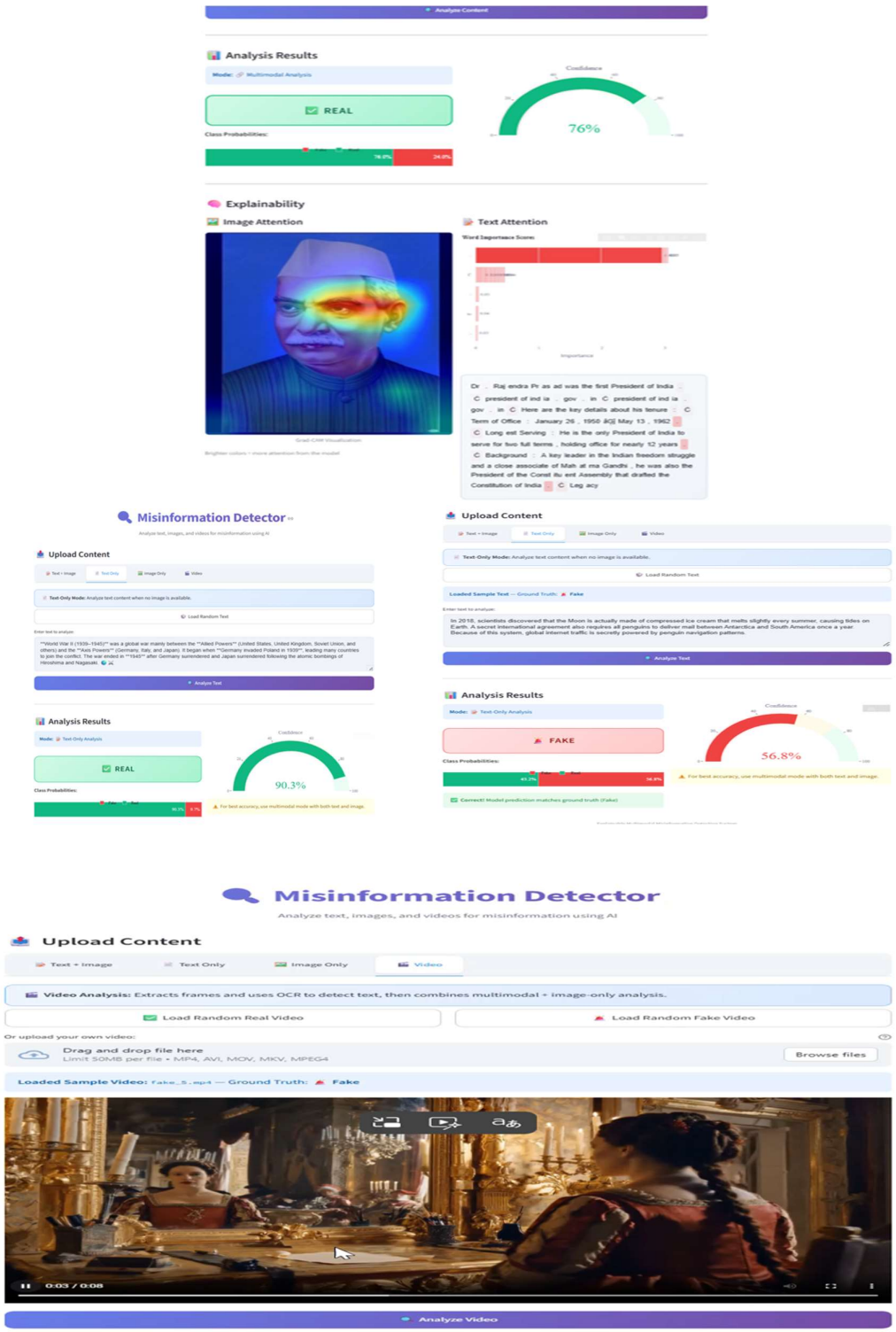
Black Box Testing assesses software functionality without reference to internal code structure. This approach, also known as behavioral, specification-based, or input-output testing, can be applied at all levels, including unit, integration, system, and acceptance testing. Black box testing focuses on validating outputs against expected results based on given inputs.

White Box Testing examines the internal code structure and logic of the application, also referred to as glass-box or structural testing. It relies on programming knowledge to design test cases and is typically performed at the unit level. Common white-box testing techniques include statement coverage, branch coverage, and path coverage, ensuring that all possible execution paths are validated for correctness.

RESULTS



International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026



Conclusion and Future Scope

This study presents an Explainable Multimodal Misinformation Detection Classifier that effectively identifies misleading digital content by leveraging the combined strengths of textual and visual analysis. By employing advanced deep learning architectures such as RoBERTa for text and ResNet-50 for images, the system achieves high detection accuracy while capturing complex relationships

between modalities that traditional approaches often overlook.

The integration of multimodal feature fusion enhances the system's ability to jointly interpret textual and visual information, improving classification performance. Moreover, the explainability component, incorporating methods like **Grad-CAM** and attention-based visualization, provides transparent insights into the model's predictions, fostering trust among users and researchers. Experimental evaluations demonstrate that the proposed framework achieves robust generalization and reliable classification across diverse inputs.

For future development, the system can be extended to support **multilingual content**, improving global applicability. Additional enhancements may include **adversarial robustness** to resist malicious manipulation and **scalable deployment** for real-time large-scale applications. Overall, this project contributes to the development of intelligent tools that help mitigate the widespread challenge of misinformation in digital environments, supporting informed decision-making and enhancing the reliability of online content.

References

1. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach.* arXiv preprint arXiv:1907.11692.
2. He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778.
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention Is All You Need.* In Advances in Neural Information Processing Systems (NeurIPS), 5998–6008.
4. Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). *Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization.* In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 618–626.
5. Kiela, D., Rethmeier, J., & Williams, M. (2021). *Fakeddit: A New Multimodal Dataset for Fine-Grained Fake News Detection.* arXiv preprint arXiv:2012.01229.
6. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.
7. Chollet, F. (2018). *Deep Learning with Python.* Manning Publications.