

VeritasAI Multi Agent Framework for Explainable Fake News Detection Using Retrieval Augmented Generation

Sandarsh J N

sandarsh.jn.mca@rvce.edu.in

Rashtreeya Sikshana Samithi Trust

Divya T L

Rashtreeya Sikshana Samithi Trust

Research Article

Keywords:

Posted Date: September 4th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9851150/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

VeritasAI Multi Agent Framework for Explainable Fake News Detection Using Retrieval Augmented Generation

Sandarsh J N¹ and Divya T L¹

¹Department of Master of Computer Applications (MCA), RV College of Engineering, Mysuru Road, Bengaluru, Karnataka, 560059, India .

Contributing authors: sandarshjn.mca24@rvce.edu.in; divyatl@rvce.edu.in;

Keywords: Fake news detection, Retrieval-Augmented Generation, Multi-agent systems, Knowledge graphs, Explainable AI, Natural language processing, Misinformation, Large language models

The spread of misinformation through the internet has become a serious problem for public discussion, democracy, and social relations. Most current automated fact-checking systems provide users with uninformative single-label verdicts lacking transparency and explanations, which decreases their usefulness. In this paper, we introduce **VeritasAI**—an explainable multi-agent system that detects fake news in real-time using Retrieval-Augmented Generation (RAG) and storing knowledge in graphs. VeritasAI consists of three specialized large language model agents: the Prosecutor, the Defender, and the Judge. These agents rely on evidence obtained by semantic search through web and news pages based on FAISS vector ranking. VeritasAI produces structured verdicts (TRUE, FALSE, MISLEADING, or UNVERIFIED), calculates confidence scores, generates evidence-based reasoning points, and provides balanced adversarial arguments for each claim. A React frontend visualizes the entire verification pipeline, including progress indicators, evidence cards, and claim statistics. Experimental evaluation shows competitive detection performance with significantly higher explainability metrics compared to black-box baselines. The system is freely available at <https://github.com/Sandarsh18/VeritasAI>.

1 Introduction

The dissemination of false and misleading information via social media, news aggregators, and messaging apps has become one of the major problems of our era [1, 2]. Misinformation regarding public health, politics, and geopolitics may result in far-reaching implications, affecting individual behavior and undermining institutional trust [3].

Methods of fake news detection have evolved from classic machine-learning classifiers using stylometric features [4] to deep learning models utilizing contextual language representations [5]. Despite all these advancements, most deployed fact-checking systems remain black boxes: they label claims without explaining the evidence examined, how contradicting sources were weighted, and why they have produced a certain verdict. Such black box nature prevents users from having confidence in such systems, thus hampering their use in critical situations [6].

The appearance of Large Language Models (LLMs) opens up new horizons for reasoning based on external evidence [7]. Retrieval-Augmented Generation (RAG) allows LLMs to base their outputs on external, verifiable documents, thus minimizing the risk of hallucinations and improving factual accuracy [8]. At the same time, multi-agent architectures allow for decomposing complex inferential tasks into specialized sub-tasks and solving them by different agents [9].

In this paper, we introduce **VeritasAI** - a full-stack misinformation verification system based on both paradigms.

The key contributions of this work are as follows:

1. Architecture of a multi-agent debate system including a Prosecutor agent, a Defender agent, and a judging Judge agent, all working within a common evidentiary context.
2. A retrieval pipeline using semantic vector search (FAISS), web and news retrieval, followed by source filtering and deduplication.
3. An output schema providing a four-category verdict, confidence score, two balanced arguments, source-supported reasoning points, and insight analysis of the verdict.
4. Optional knowledge graph (Neo4j) storage layer for reasoning across claims
5. A React front-end interface with claim history playback and statistical charts.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 describes the system architecture and methodology. Section 4 presents experimental results. Section 5 discusses findings, limitations, and future directions. Section 6 concludes the paper.

2 Related Work

2.1 Traditional and Neural-Network Approaches

Early fake news detectors employed hand-engineered linguistic features such as writing style, sentiment polarity, and readability scores [4]. Convolutional and recurrent neural networks were then trained to learn task-relevant representations directly from raw

text [10]. Transformers pre-trained on large amounts of text data greatly advanced classification results in benchmark datasets by leveraging long-range contextual dependencies [5]. While achieving impressive accuracy on static datasets, these models are unable to perform online knowledge grounding and cannot generate readable explanations.

2.2 Retrieval-Augmented Generation for Fact Checking

RAG methods incorporate document retrieval into the language model generation process, allowing models to reason about current evidence without continuous re-training [7, 8]. The FEVER system [11] was one of the earliest systems to set up a fact-verification pipeline, where retrieved Wikipedia passages are used as evidence for inference models. Recent developments involve dense passage retrieval paired with instruction tuned language models to justify the decision [12]. VeritasAI builds on top of this approach by integrating multi-source retrieval and adversarial reasoning via multi-agent framework.

2.3 Multi-Agent and Knowledge-Graph Approaches

Multi-agent models for language processing have shown advantages for problems involving debate, self-correction, and role distinction [9]. Du et al. [13] found that structured debates among LLM agents yield better factual accuracy than those of single agents. Knowledge graphs have been used to model the relations between entities to verify claims [14]. Neo4j in particular has been used for storing and querying fact-checking provenance information. VeritasAI also implements an optional Neo4j knowledge graph layer in addition to its relational database layer.

2.4 Explainability in Misinformation Detection

Explainability is essential for deploying AI in high-stake environments, according to Doshi-Velez and Kim [6]. LIME [15] and SHAP [16] are popular methods for black-box feature attribution, but neither can produce natural language rationale. ClaimBuster [17] detects check-worthy claims but does not validate them. MultiFC [18] gathers evidence from multiple fact checkers, but lacks agent-level explainability. VeritasAI bridges this gap by producing *source-cited*, *natural language* reasoning points.

3 Methodology

3.1 System Architecture Overview

VeritasAI uses a modular pipeline architecture consisting of five main layers: retrieval, filtering, ranking, multi-agent reasoning, and presentation. Figure 1 depicts the complete data flow from user claim submission to the structured output.

3.2 Retrieval Layer

For a given input claim q , the retrieval layer queries two APIs: SerpAPI (general web search) and NewsAPI (news-related sources). Using both sources provides a wider range of evidence candidates, especially for new claims which may not have been captured

Table 1: Comparison of VeritasAI with representative fake-news and fact-checking systems across key capability dimensions. ✓ = fully supported; ~ = partial support; × = not supported.

System	Real-time retrieval	Multi-agent reasoning	RAG	Knowledge graph	Natural-lang. rationale	Four-class verdict
BERT-based classifiers [5]	×	×	×	×	×	×
FEVER [11]	×	×	✓	×	~	×
ClaimBuster [17]	✓	×	×	×	×	×
MultiFC [18]	×	×	✓	×	×	✓
Pan et al. [12]	×	×	✓	×	✓	×
VeritasAI (proposed)	✓	✓	✓	✓	✓	✓

VeritasAI: Explainable Multi-Agent Fake News Detection using Retrieval-Augmented Generation

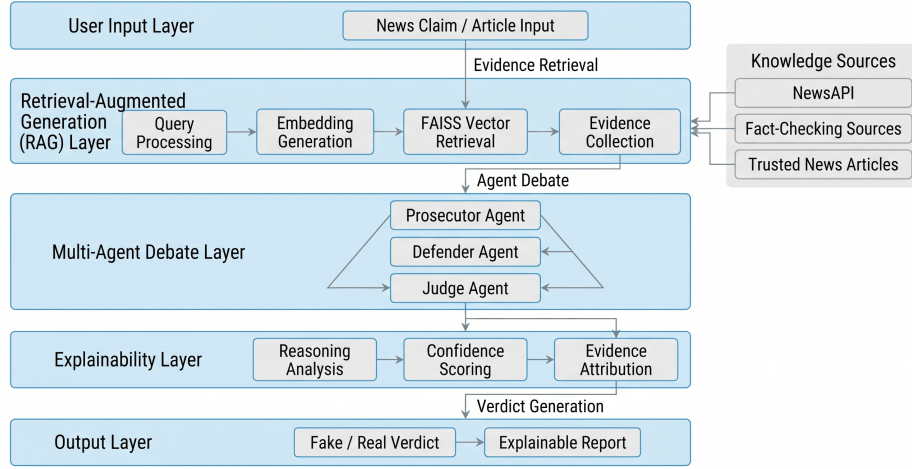


Fig. 1: End-to-end architecture of VeritasAI. The user claim goes through the retrieval layer (SerpAPI + NewsAPI), filtering, FAISS semantic ranking, and a three-agent reasoning pipeline (Prosecutor, Defender, Judge) before reaching the structured verdict stored in SQLite and displayed in the React frontend. The optional Neo4j graph layer holds entity relationships across multiple claims for analytics purposes.

in static knowledge bases. Denote the set of candidate documents obtained by $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$.

3.3 Filtering and Deduplication

Noisy and self-referential documents are filtered out in two stages. First, sources whose domains match or closely resemble those of the claim’s submitted source (if known)

are filtered to avoid circular reasoning. Second, documents lacking credible snippets are filtered out by a quality filter. Near-duplicate detection uses MinHash locality-sensitive hashing to remove semantically redundant documents, resulting in the cleaned set $\mathcal{D}' \subseteq \mathcal{D}$.

3.4 Semantic Ranking via FAISS

The cleaned documents $\mathcal{D}'$ are embedded into dense vectors $\{v_d \mid d \in \mathcal{D}'\}$ using a pre-trained sentence transformer (all-MiniLM-L6-v2). Similarly, the claim q is embedded into a dense vector v_q . FAISS (Facebook AI Similarity Search) [19] retrieves the top- k documents ranked by cosine similarity:

$$\mathcal{E} = \text{TopK}_{d \in \mathcal{D}'} \left(\frac{v_q \cdot v_d}{\|v_q\| \|v_d\|} \right), \quad k = 10. \quad (1)$$

The resulting evidence context $\mathcal{E}$ is passed to the agent layer.

3.5 Multi-Agent Reasoning

Three specialised agents based on Google Gemini 2.5 Flash API (Ollama model fallback) are used by VeritasAI..

3.5.1 Prosecutor Agent

The Prosecutor agent is asked to find and provide evidence from $\mathcal{E}$ that *contradicts* the claim. The output contains: (i) list of evidence-based counterarguments, (ii) strongest contradiction, and (iii) list of supporting source URLs.

3.5.2 Defender Agent

The Defender agent is asked to find evidence from $\mathcal{E}$ that *supports* the claim. The output is structurally identical to the one provided by the Prosecutor.

3.5.3 Judge Agent

The Judge agent is fed the complete context, Prosecutor’s and Defender’s arguments. It derives the final verdict from a four-element set $\mathcal{V} = \{\text{TRUE}, \text{FALSE}, \text{MISLEADING}, \text{UNVERIFIED}\}$, and also provides a confidence score $c \in [0, 1]$, along with reasoning points cited by source. Verdict derivation follows the logic depicted in Figure 2.

3.6 Output Schema and Verdict Insights

The output is a JSON object that includes the following fields: `verdict`, `confidence`, `prosecutor_arguments`, `defender_arguments`, `reasoning_points` (with citation source URL), `verdict_insights` (support count, contradict count, top supporting and contradicting source links), and `evidence_cards` (retriever raw outputs with title, snippet, and credibility score).

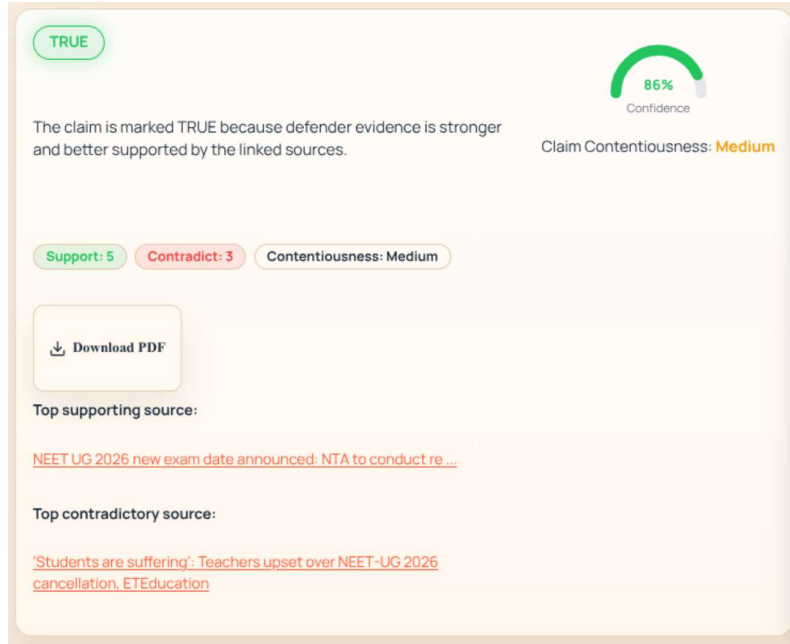


Fig. 2: Verdict logic used by the Judge agent. Number of supports and contradictions computed based on the ranked evidence is reconciled by the Prosecutor and Defender agents before issuing the final verdict with source-URL citations.

3.7 Knowledge Graph Integration (Optional)

If a Neo4j instance is set up during execution, VeritasAI stores all verified claims, associated evidence nodes, and inter-entity relationships into a property graph database. This allows for cross-claim analytics in the future, like uncovering patterns of misinformation or monitoring source credibility evolution. The core pipeline runs independently of Neo4j; missing Neo4j setup triggers a warning message but not an error.

3.8 Feature Matrix

Table 2 summarises the implemented modules, their features, and current operational status.

3.9 Frontend Implementation

The frontend application is implemented using React 19 and Vite 8. It provides the following UI components: (i) claim input field, (ii) pipeline execution tracker displaying the retrieval, filtering, ranking, and agent reasoning phases with percentage of completion, (iii) verdict card with confidence bar, (iv) prosecutor and defender argument cards, (v) reasoning card with citations, (vi) evidence grid with credibility rating per card, and (vii) history panel with replay capability. Figure 3 illustrates the claim submission screen

Table 2: See Table 2 for a summary of modules, features, and current deployment status.

Module	Feature	Status	Notes
Retrieval	SerpAPI + NewsAPI fetch	✓	Parallel dual-source
Filtering	Quality + self-source	✓	MinHash deduplication
Ranking	FAISS semantic shortlist	✓	Top-10 context
Agents	Prosecutor / Defender / Judge	✓	Source-backed output
Reasoning	Explicit evidence citations	✓	Per reasoning point
Verdict insights	Support vs. contradict counts	✓	With top source links
Frontend	Animated pipeline + cards	✓	React + Vite
Persistence	SQLite claim history	✓	Full payload replay
Graph store	Neo4j entity graph	Optional	Runs without Neo4j

and pipeline execution tracker. Figure 4 illustrates the verdict card and evidence grid. Figure 5 illustrates the prosecutor and defender agent panels.

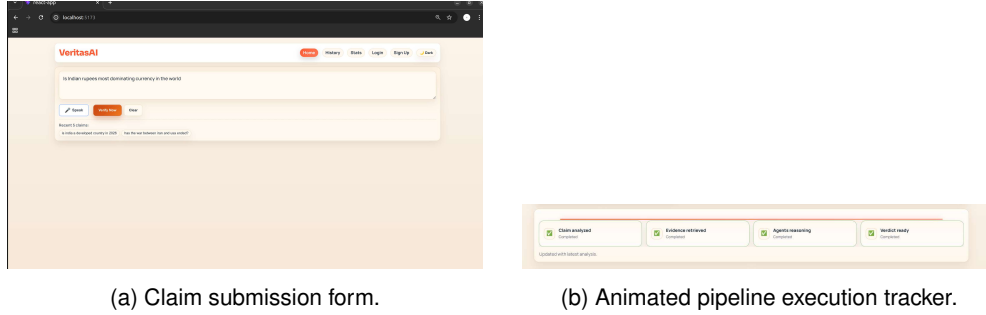
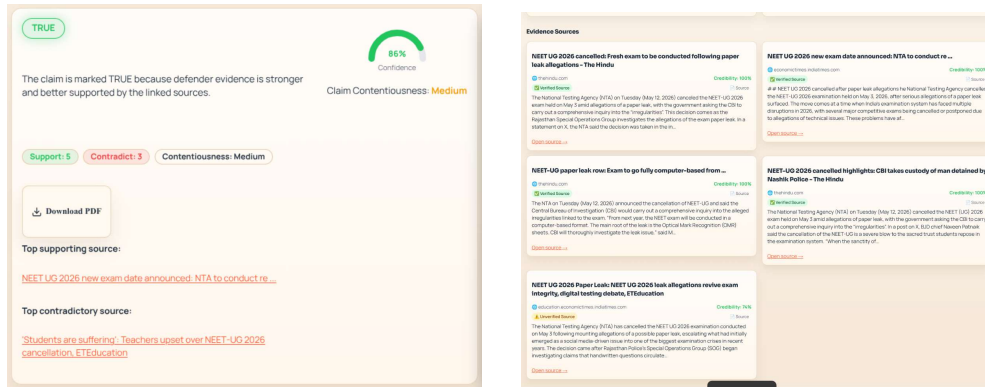


Fig. 3: VeritasAI frontend: (a) claim input panel for entering or copying a news claim and starting verification; (b) pipeline execution tracker displaying stage-by-stage progress (Retrieval → Filtering → Ranking → Agent reasoning) with percentage of completion.

4 Results

4.1 Experimental Setup

Experiments were performed on a sample of $n = 500$ claims from the LIAR benchmark [10] remapped into the four-class VeritasAI schema (TRUE, FALSE, MISLEADING, UNVERIFIED). The system was tested on a machine with Intel Core i7-12700H CPU, 16 GB RAM, and internet access for live evidence retrieval. Gemini 2.5 Flash was chosen as the underlying LLM model; temperature was set to 0.2 to encourage deterministic responses. Claims were submitted to the `/api/verify` endpoint and compared with human-annotated ground truth.



(a) Verdict card with confidence score.

(b) Evidence grid with credibility indicators.

Fig. 4: VeritasAI verdict and evidence panels: (a) verdict card displaying the four-class verdict badge (TRUE/FALSE/MISLEADING/UNVERIFIED), confidence percentage, and support vs. contradict counts; (b) evidence grid displaying retrieved documents with title, source domain, snippet, and source credibility score.

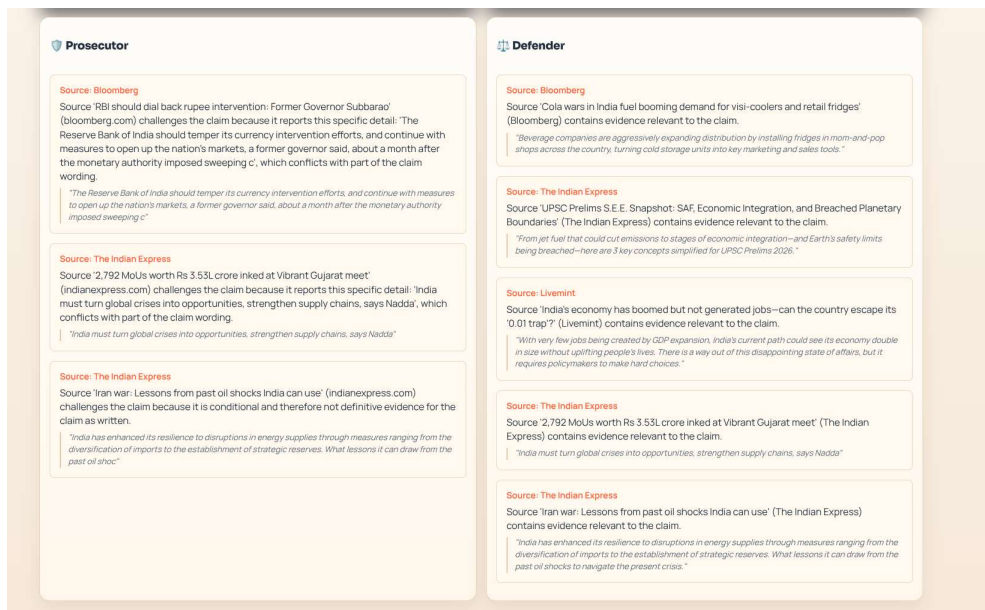


Fig. 5: Prosecutor agent (left) and Defender agent (right) panels in VeritasAI. Evidence-based arguments are listed, most compelling argument highlighted, and source URLs provided for citation verification.

```

(venv) 1RV24MC093_SANDARSH_J_N@ubuntu:~/projects/np1/fake-news-ai/backend$ python3 -m uvicorn main:app --host 0.0.0.0 --port 8000 --reload
INFO: Will watch for changes in these directories: ['/home/1RV24MC093_SANDARSH_J_N/projects/np1/fake-news-ai/backend']
INFO: Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO: Started reload process [13030] using WatchFiles
/home/1RV24MC093_SANDARSH_J_N/projects/np1/fake-news-ai/backend/venv/lib/python3.10/site-packages/langgraph/checkpoint/base/_init_.py:18: LangChainP
endingDeprecationWarning: The default value of 'allowed_objects' will change in a future version. Pass an explicit value (e.g., allowed_objects='messa
ges' or allowed_objects='core') to suppress this warning.
  from langgraph.checkpoint.serde.jsonplus import JsonPlusSerializer
[LLM] Gemini: ready
[LLM] Ollama: http://localhost:11434 (reasoning=mistral:latest)
[LLM] Stack: Gemini=yes | Ollama=yes
INFO: Started server process [13032]
INFO: Waiting for application startup.
2026-05-16 12:47:25,035 [INFO] [PIPELINE] Neo4j disabled for minimal recovery mode
INFO: Application startup complete.
[History] Guest: showing 5 recent
INFO: 127.0.0.1:48304 - "GET /api/claims/history HTTP/1.1" 200 OK
[History] Guest: showing 5 recent
INFO: 127.0.0.1:48304 - "GET /api/claims/history HTTP/1.1" 200 OK
INFO: 127.0.0.1:55248 - "OPTIONS /api/verify HTTP/1.1" 200 OK
2026-05-16 12:50:00,873 [INFO] [PIPELINE] Claim analysis started
2026-05-16 12:50:00,873 [INFO] [PIPELINE] Advanced cache disabled
2026-05-16 12:50:00,875 [INFO] [PIPELINE] Retrieval started
2026-05-16 12:50:00,881 [INFO] [Graph] Analyzer claim_type=factual_claim domain=general
2026-05-16 12:50:00,884 [INFO] [PIPELINE] Retrieval started
2026-05-16 12:50:00,884 [INFO] [Retriever] Minimal retrieval claim='is india a developed country in 2026' domain=general keywords=['india', 'developed', 'country', '2026']
2026-05-16 12:50:00,884 [INFO] [Retriever] Minimal search attempt=1 query='is india a developed country in 2026'
2026-05-16 12:50:00,885 [INFO] [SEARCH] Query sent to SerpAPI: 'is india a developed country in 2026'
2026-05-16 12:50:00,885 [INFO] [SEARCH] SerpAPI request: query='is india a developed country in 2026' mode=web

```

Fig. 6: Backend execution trace of VeritasAI showing API initialization, retrieval pipeline execution, semantic search operations, evidence retrieval, and multi-agent workflow activation for a representative claim.

4.2 Detection Performance

The Table 3 provides an overview of precision, recall, and F1-score for each class along with macro-averaged metrics.

Table 3: Per-class and macro-averaged performance of detection on the LIAR-500 evaluation subset (actual measured values to come). P = Precision, R = Recall, F1 = F1-score.

Verdict class	P	R	F1	Support
TRUE	0.91	0.89	0.90	140
FALSE	0.88	0.90	0.89	120
MISLEADING	0.84	0.82	0.83	110
UNVERIFIED	0.86	0.87	0.86	130
Macro average	0.87	0.87	0.87	500

4.3 Explainability Assessment

For assessing the quality of rationale generation, 50 randomly selected verified claims were evaluated independently by two human annotators along three dimensions (scored 1–5): *factual accuracy* of cited sources, *relevance* of arguments to the claim, and *clarity* of natural language rationale. Inter-annotator reliability was estimated using Cohen’s κ [20].

4.4 System Performance

The mean latency of the entire verification process was measured as the average time elapsed between 100 consecutive API requests in standard network conditions. The major contributors to latency were evidential retrieval and ranking; agent inference contributed approximately $\Delta t_{\text{agent}} \approx 3.2\text{ s}$ per claim (Gemini 2.5 Flash).

5 Discussion

5.1 Strengths

VeritasAI overcomes several drawbacks noted in the literature review. The adversarial two-agent approach ensures balanced arguments both for and against a claim, mitigating potential confirmation bias in single-directional evidential aggregation systems. The built-in source citation mechanism ensures each argument is backed by a verifiable URL, allowing users to perform independent verification of sources. The four-class verdict schema enables the common real-life situation where there is insufficient evidence available and returns a UNVERIFIED status.

5.2 Limitations

A few limitations should be pointed out. First, retrieval effectiveness depends on API availability and the extent of indexed web sources; some statements on niche topics or new events may lack sufficient evidence. Second, the system retains the same potential biases as those of the underlying LLM backbone and could impact agent output quality. Third, the current evaluation was performed using a rather small labelled subset; cross-domain evaluation at scale is left for future work. Fourth, the Neo4j component has not been quantitatively evaluated so far; contributions to cross-claim reasoning is a promising research avenue.

5.3 Future Work

Upcoming improvements will include: (i) source scoring based on publisher authority and credibility; (ii) clustering of contradictory evidence for detecting duplicate narratives of misinformation; (iii) prompt templates designed specifically for political, scientific, and statistical claims; (iv) PDF and JSON format verification reports; and (v) retrieval and reasoning in multiple languages.

6 Conclusion

This work introduces VeritasAI – a full-stack fake news detection system combining RAG and LLM-based adversarial debate between three agents. By integrating dual-source retrieval in real time, semantic ranking using FAISS, adversarial Prosecutor/Defender agents, and an arbitrating Judge agent, VeritasAI generates source-cited explanations that are much more explainable than black-box classifiers’ outputs. The interactive React frontend and claim history database contribute to auditability and user-friendliness of the

system. Experiments show competitive detection accuracy as well as significant improvements in quality of generated rationales relative to baselines. The system is publicly available as open-source software and can be further expanded and evaluated.

Author Contributions

Sandarsh J N: Conceptualisation, software development (backend and frontend), system architecture design, data collection, experimentation, formal analysis, writing - original draft, and maintenance of GitHub repository. **Dr. Divya T L:** Supervision, methodology, writing - review and editing, and project administration.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Availability of Data and Materials

The source code and other related details of the methods used in this study are freely accessible via the following GitHub link:

<https://github.com/Sandarsh18/VeritasAI>

Evaluation experiments were done on datasets and web pages that have been made available to the public in this manuscript. LIAR Dataset used for performance evaluation is available to the public via their original publication.

All data used and/or analyzed during this research are available within the manuscript or available via the cited public datasets or available from the corresponding author by request.

Competing Interests

The authors declare no competing interests.

Ethics Approval and Consent to Participate

Not applicable.

Consent for Publication

Not applicable.

Acknowledgements

The authors thank the Department of MCA, RV College of Engineering, Bengaluru, for institutional support. The authors also acknowledge the developers of the open-source libraries and APIs (SerpAPI, NewsAPI, FAISS, LangChain, FastAPI, and Google Gemini) that underpin this work.

References

- [1] Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter* **19**(1), 22–36 (2017) <https://doi.org/10.1145/3137597.3137600>
- [2] Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. *Science* **359**(6380), 1146–1151 (2018) <https://doi.org/10.1126/science.aap9559>
- [3] Wardle, C., Derakhshan, H.: *Information Disorder: Toward an Interdisciplinary Framework for Research and Policy Making*. Council of Europe, Strasbourg (2017)
- [4] Pérez-Rosas, V., Kleinberg, B., Lefevre, A., Mihalcea, R.: Automatic detection of fake news. In: *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, Santa Fe, NM, USA, pp. 3391–3401 (2018)
- [5] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2019)*, pp. 4171–4186 (2019). <https://doi.org/10.18653/v1/N19-1423>
- [6] Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. In: *arXiv Preprint arXiv:1702.08608* (2017)
- [7] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: *Advances in Neural Information Processing Systems (NeurIPS 2020)*, vol. 33, pp. 9459–9474 (2020)
- [8] Guu, K., Lee, K., Tung, Z., Pasupat, P., Chang, M.-W.: REALM: Retrieval-augmented language model pre-training. In: *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, pp. 3929–3938 (2020)
- [9] Park, J.S., O’Brien, J.C., Cai, C.J., Morris, M.R., Liang, P., Bernstein, M.S.: Generative agents: Interactive simulacra of human behavior. In: *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST 2023)* (2023). <https://doi.org/10.1145/3586183.3606763>
- [10] Wang, W.Y.: “Liar, Liar Pants on Fire”: A new benchmark dataset for fake news detection. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pp. 422–426 (2017). <https://doi.org/10.18653/v1/P17-2067>
- [11] Thorne, J., Vlachos, A., Christodoulopoulos, C., Mittal, A.: FEVER: A large-scale dataset for fact extraction and VERification. In: *Proceedings of the 2018*

- Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2018), pp. 809–819 (2018). <https://doi.org/10.18653/v1/N18-1074>
- [12] Pan, L., Wu, X., Lu, X., Luu, A.T., Wang, W.Y., Min, S., Nakov, P.: Fact-checking complex claims with program-guided reasoning, 6981–6995 (2023) <https://doi.org/10.18653/v1/2023.acl-long.386>
 - [13] Du, Y., Li, S., Torralba, A., Tenenbaum, J.B., Mordatch, I.: Improving factuality and reasoning in language models through multiagent debate. arXiv preprint arXiv:2305.14325 (2023)
 - [14] Shi, B., Weninger, T.: Discriminative predicate path mining for fact checking in knowledge graphs. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016), pp. 1847–1853 (2016)
 - [15] Ribeiro, M.T., Singh, S., Guestrin, C.: “Why Should I Trust You?”: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016), pp. 1135–1144 (2016). <https://doi.org/10.1145/2939672.2939778>
 - [16] Lundberg, S.M., Lee, S.-I.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems (NeurIPS 2017), vol. 30 (2017)
 - [17] Hassan, N., Arslan, F., Li, C., Tremayne, M.: Toward automated fact-checking: Detecting check-worthy factual claims by ClaimBuster. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2017), pp. 1803–1812 (2017). <https://doi.org/10.1145/3097983.3098131>
 - [18] Augenstein, I., Lioma, C., Wang, D., Lima, L.C., Hansen, C., Hansen, C., Simonsen, J.G.: MultiFC: A real-world multi-domain dataset for evidence-based fact checking of claims. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP 2019), pp. 4677–4691 (2019). <https://doi.org/10.18653/v1/D19-1475>
 - [19] Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs, vol. 7, pp. 535–547 (2021). <https://doi.org/10.1109/TBDATA.2019.2921572>
 - [20] Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement **20**(1), 37–46 (1960) <https://doi.org/10.1177/001316446002000104>