ICCS 2025

25th International Conference on Computational Science

MAVS: An Ensemble-Based Multi-Agent Framework for Fake News Detection

PAPER ID : 380

AUTHORS:

Dhruv Tyagi, Anurag Singh, Hocine Cherifi

National Institute of Technology Delhi,

University de Bourgogne Europe

PRESENTED BY:

Dhruv Tyagi

National Institute of Technology Delhi

Table of Content

Introduction and Motivation

- In the digital age, the rapid spread of fake news has emerged as a significant challenge, affecting public trust, influencing democratic processes, and endangering public health[1].

- Social media and online platforms facilitate the widespread dissemination of misinformation at an unprecedented speed, making it difficult to distinguish between credible and false information[2].

[1] Pennycook, Gordon and David G. Rand. “The Psychology of Fake News.” Trends in Cognitive Sciences 25 (2021): 388-402.
[2] Allcott, Hunt & Gentzkow, Matthew. (2017). Social Media and Fake News in the 2016 Election. Journal of Economic Perspectives. 31. 211-236. 10.1257/jep.31.2.211.

There is an urgent need for an automated, efficient, and multidimensional system to effectively detect fake news.

Problem Statement

- Automated: Detects fake news in real time without human intervention.
- Efficient: Scales to handle vast amounts of data with minimal resources.
- Multidimensional:
  - Analyzes text, sentiment, and stance.
  - Checks facts using knowledge bases and AI models.
  - Tracks spread patterns and user behavior.

Comparative Analysis of Fake News Detection Techniques

Table 1. The comparison of various fake news detection techniques [3-8]

Approach	Strengths	Limitations	Why Improvements Needed
Fact-Checking	Verifies with external knowledge	Data dependency	Difficult with breaking news
	Provides explainability	-	Access to up-to-date datasets required
NLP-Based Methods	Language-based analysis of text	Limited to textual features	Adversarial manipulation of text
	Text style and readability analysis	Lack of contextual understanding	Ignored propagation patterns
Deep Learning Models	Contextual understanding of text	Data-intensive	Large labeled datasets required
	Multimodal learning (text + images)	Black box nature	Lack of explainability
	-	Still text-focused	Ignored social dynamics
GNN-Based Methods	Captures propagation patterns	Complex to scale	Needs more efficient algorithms
	Multi-hop reasoning in graphs	Vulnerable to adversarial attacks	Better adversarial defenses needed
	Models node relationships	Hard to interpret results	Improve interpretability

Feature Comparison

Table 2. MAVS Framework vs. Other Fake News Detection Approaches, where x repesents “no”, ✓ represents “yes” and △ represents “partially yes” [4-9]

Feature	GNN	Fact Checker	Stance Checker	Sentiment Checker	MAVS (Overall)
Propagation-Based Analysis	✓	✗	✗	✗	✓
Textual Analysis	✗	✓	✓	✓	✓
Credibility Checking	✓	✓	✗	✗	✓
Context Understanding	✗	✓	✓	✗	✓
Graph-Based Analysis	✓	✗	✗	✗	✓
Stance Detection	✗	✗	✓	✗	✓
Sentiment Analysis	✗	✗	✗	✓	✓
Fact Verification	✗	✓	✗	✗	✓
Source Reliability Check	✓	✓	✗	✗	✓
Claim-Based Evaluation	✗	✓	✓	✗	✓
Prone to Adversarial Attacks	✓	✓	✓	✓	△

Proposed Methodology

Approach to solve the problem

MAVS Framework

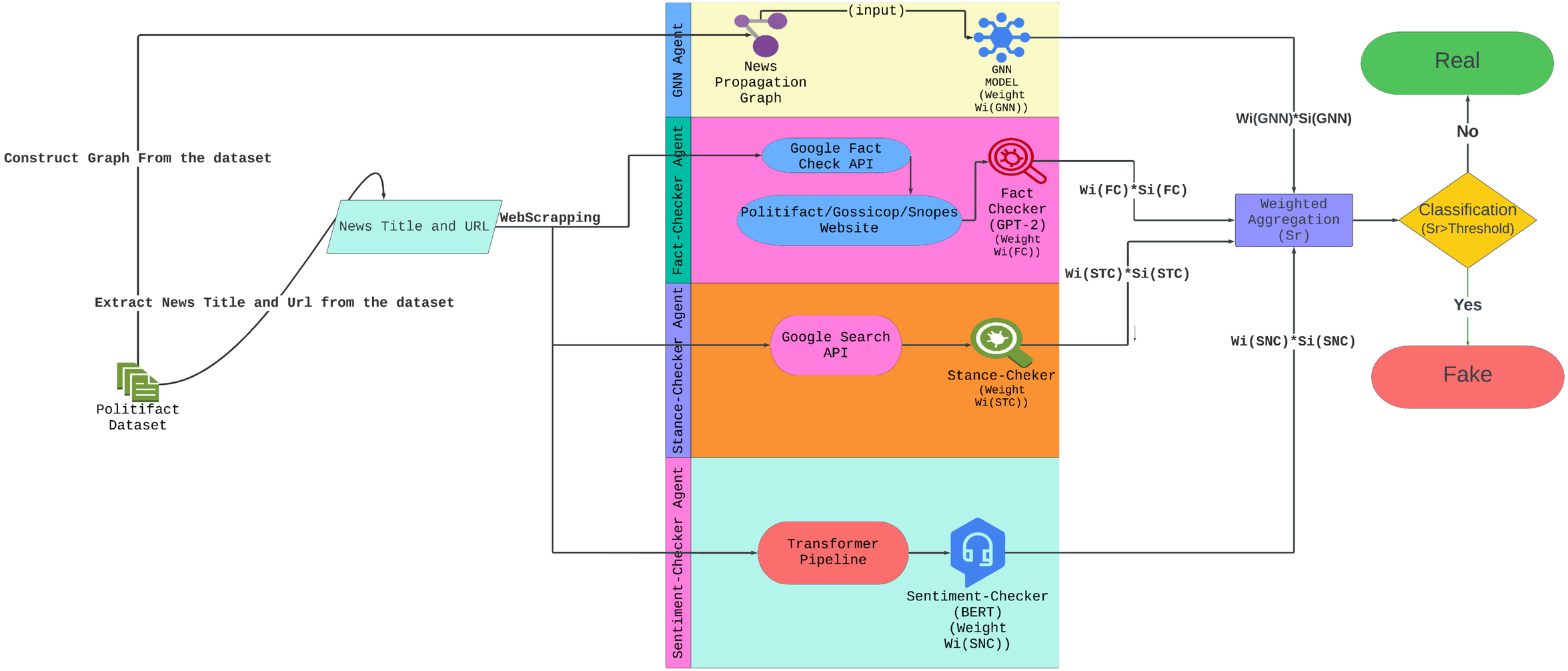
The proposed solution introduces MAVS (Multi-Agent Verification System), a framework that employs an ensemble-based approach, integrating multiple specialized agents working in parallel and independently, each responsible for a distinct aspect of fake news detection which are as follows:

- GNN (Graph Neural Network): Propagation-based analysis.
- Fact-Checker: Conducts credibility verification.
- Stance-Checker: Perform crowd sourcing.
- Sentiment-Checker: Analyzes the polarity.

The final classification of news as real or fake is determined through a weighted aggregation of these agents' outputs, where Stochastic Gradient Descent (SGD)-based Logistic Regression is used to learn the optimal weights, ensuring a robust and adaptive multi-perspective evaluation.

Architecture of MAVS

Fig 1. Architecture of MAVS Framework for Fake News Detection



AI Agents

Algorithm 1. Sentiment Score Computation for News Articles

Sentiment-Checker

- The sentiment-checker agent leverages the BERT Multilingual model to assess the emotional tone of the content retweeted by users.
- It categorizes the sentiment as 1 star, 2 stars, 3 stars, 4 stars, or 5 stars, helping in identifying emotional manipulation or polarizing content.

Input : T: News title, URL: Article URL.

Output: S_{final}: Sentiment score, Sentiment Label: Positive, Neutral, or Negative.

Step 1

Initialize model: Model $\leftarrow$ pipeline("sentiment-analysis", model = "bert-multilingual")

Step 2

Compute sentiment score for title: $(L_T, S_T) \leftarrow \text{Model}(T)$ **if** L_T *is "4 stars" or "5 stars"* **then**

  $S_T \leftarrow -S_T$

else if L_T *is "3 stars"* **then**

  $S_T \leftarrow 0$

Step 3

Extract and summarize article content: $C \leftarrow \text{ExtractText(URL)}$, $S \leftarrow C[:200]$

Compute sentiment score: $(L_C, S_C) \leftarrow \text{Model}(S)$ **if** L_C *is "4 stars" or "5 stars"* **then**

  $S_C \leftarrow -S_C$

else if L_C *is "3 stars"* **then**

  $S_C \leftarrow 0$

Step 4

Compute weighted sentiment score: $S_{\text{final}} = 0.3 \cdot S_T + 0.7 \cdot S_C$

Step 5

if $S_{final} \geq 0$ **then**

  Assign label as Positive.

else

  Assign label as Negative.

return S_{final}, Sentiment Label

Stance-Checker

AI Agents

Algorithm 2. Stance Analysis Process Using Zero-Shot Classification

Input : T: News title, URL: Article URL.

Output: L_{final}: Final stance label, S_{final}: Stance score.

Step 1

Initialize stance detection model: Model $\leftarrow$ pipeline("zero-shot-classification", model = "bart")

Step 2

Extract article content and summary: $C \leftarrow \text{ExtractText}(\text{URL}), \quad S \leftarrow C[:300]$

Step 3

Compute stance of title w.r.t. content: $S_T \leftarrow w_i \cdot p(L_i \mid T, S)$

Step 4

Perform Google search for related URLs.

foreach *related URL i* **do**

  Extract content H_i (first 200 words), compute stance score: $S_{R_i} \leftarrow w_i \cdot p(L_i \mid T, H_i)$ Append to stance list.

Compute average stance score: $S_R = \frac{1}{n} \sum_{i=1}^{n} S_{R_i}$

Step 5

Compute weighted final stance score: $S_{\text{final}} = 0.3 \cdot S_T + 0.7 \cdot S_R$

Step 6

if $0.7 \cdot S_C > 0.3 \cdot S_T$ **then**

  $L_{\text{final}} \leftarrow L_C$

else

  $L_{\text{final}} \leftarrow L_T$

Step 7

if L_{final} *is "supports"* **then**

  $S_{\text{adjusted}} \leftarrow -S_{\text{final}}$

else if L_{final} *is "neutral"* **then**

  $S_{\text{adjusted}} \leftarrow 0$

else

  $S_{\text{adjusted}} \leftarrow S_{\text{final}}$

return $L_{\text{final}}, S_{\text{final}}$

AI Agents

Fact-Checker

- The fact-checker leverages the Google Fact Check API, and GPT-2 to evaluate the accuracy of statements.
- The process includes retrieving related fact-checked claims and analyzing them using a language model. The final score indicates the likelihood that a given statement is true or false

Algorithm 3. Fact-Checking Process

Input : S: Statement to verify, API_Key: API key for fact-checking.

Output: S_{weighted}: Final weighted score, Generated_Text: Explanation from GPT-2.

Step 1

Initialize tokenizer and text generator: Tokenizer $\leftarrow$ GPT2Tokenizer('gpt2'), Generator $\leftarrow$ pipeline('text-generation', model = 'gpt2')

Step 2

Fetch results: $F \leftarrow$ API_Request(S, API_Key) **if** *status_code* $\neq 200$ **then**
  return Error

Extract claims: $C \leftarrow F$['claims']

Step 3

for *each claim* $C_i \in C$ **do**

Retrieve verdict V_i and compute score: $S_i = \begin{cases} -1, & V_i \in \{\text{"true"}, \text{"mostly true"}, \text{"half true"}\} \\ 1, & V_i \in \{\text{"false"}, \text{"mostly false"}, \text{"pants on fire"}\} \\ 0, & \text{otherwise} \end{cases}$ Accumulate: $S \leftarrow S + S_i$

Step 4

$S_{\text{weighted}} \leftarrow \frac{S}{|C|}$

Step 5

Construct prompt: Prompt $\leftarrow$ "Given the statement 'S' and the fact-check results:"

Generate explanation: Generated_Text $\leftarrow$ Generator(Prompt)

return S_{weighted}, Generated_Text

Algorithm Used in MAVS

Input : Feature matrix X containing agent scores, Binary labels y (0 = Fake, 1 = Real).

Output: Trained SGD Logistic Regression Model, Classification result for new instances.

Step 1

Construct feature vectors: $X = [S_{i,\text{GNN}}, S_{i,\text{FC}}, S_{i,\text{STC}}, S_{i,\text{SNC}}]$ Assign labels and split dataset: $(X_{\text{train}}, y_{\text{train}}), (X_{\text{test}}, y_{\text{test}})$

Step 2

Initialize and train SGD Logistic Regression Model: model = SGDClassifier(loss = 'log_loss', max_iter = 1000, tol = $1e^{-3}$) model.fit($X_{\text{train}}, y_{\text{train}}$)

Step 3

Predict and compute accuracy: y_{pred} = model.predict(X_{test}), accuracy = $\frac{\text{Correct Predictions}}{\text{Total Predictions}}$ Extract feature weights: $w_1, w_2, w_3, w_4 =$

else

return Real News (0)

$$P(\text{Real News}) = \frac{1}{1 + e^{-S_{\text{r}}}}$$

$$\text{Classify } n_i = \begin{cases} \text{Fake}, & \text{if } P \geq 0.5, \\ \text{Real}, & \text{otherwise.} \end{cases}$$

The threshold P ≥ 0.5 for classification as the label encoding assigns 0 to real news and 1 to fake news, a
lower sigmoid output (closer to 0) indicates a stronger belief in news being real.

Algorithm 4. MAVS Score-Based Fake News Classification Using SGD Logistic Regression

Experimental Setup for MAVS Framework

System Configuration

- Intel Core i5, 16 GB RAM
- Tools: Python 3.8+, PyTorch 1.10+, Torch Geometric 2.0+, Hugging Face Transformers
- Web Scraping: Selenium, BeautifulSoup

Dataset

- UPFD Politifact Dataset **[10]**
  - News propagation graphs: Input for GNN
  - Labels: Fake (1), Real (0)
  - Split: 70% Training, 20% Testing, 10% Validation

GNN Model Architecture

- GNN with 3 GATConv layers
  - Input: 310, Hidden: 128, Output: 1
- Learning Rate: 0.01, Optimizer: Adam

Additional Components

- Fact-Checking: GPT-2, Google Fact Check API
- Stance-Checking: Zero-Shot Classifier
- Sentiment-Checking: BERT Multilingual Model
- Adversarial Attacks: MARL Framework,(HR,HF,MF)

Column Name	Description	Example Value
id	Unique identifier for the news article	”politifact4190”
news_url	URL of the source news article	`http://www.c.gov/doc.pdf`
title	Headline or title of the news article	”Budget and Economic Outlook”
tweet_ids	List of related tweet IDs that mention or retweet the news article	”1102113056 1102113348 ...”
label	Binary label indicating whether the news is real (0) or fake (1)	1

Table 3. Dataset Columns and Their Descriptions

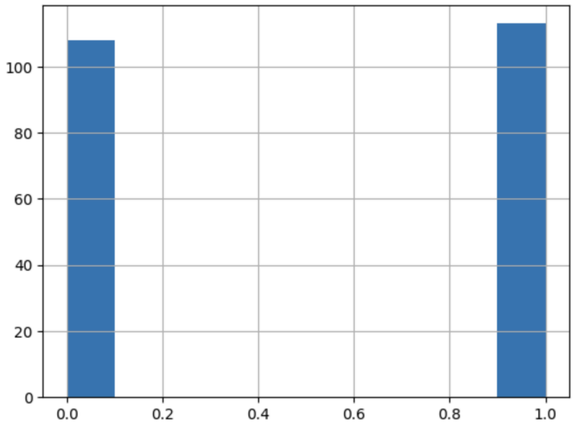


Fig 2. Dataset Labeling where x-axis shows labels (0: fake, 1: real), y-axis shows number of samples.

[10] Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. 2021. User Preference-aware Fake News Detection. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21). Association for Computing Machinery, New York, NY, USA, 2051–2055. https://doi.org/10.1145/3404835.3462990

Description of GNN

Node : Each news article is represented as a node. Users who retweeted the news are the leaf nodes

Edge: Users are connected to each other if one retweeted the other.

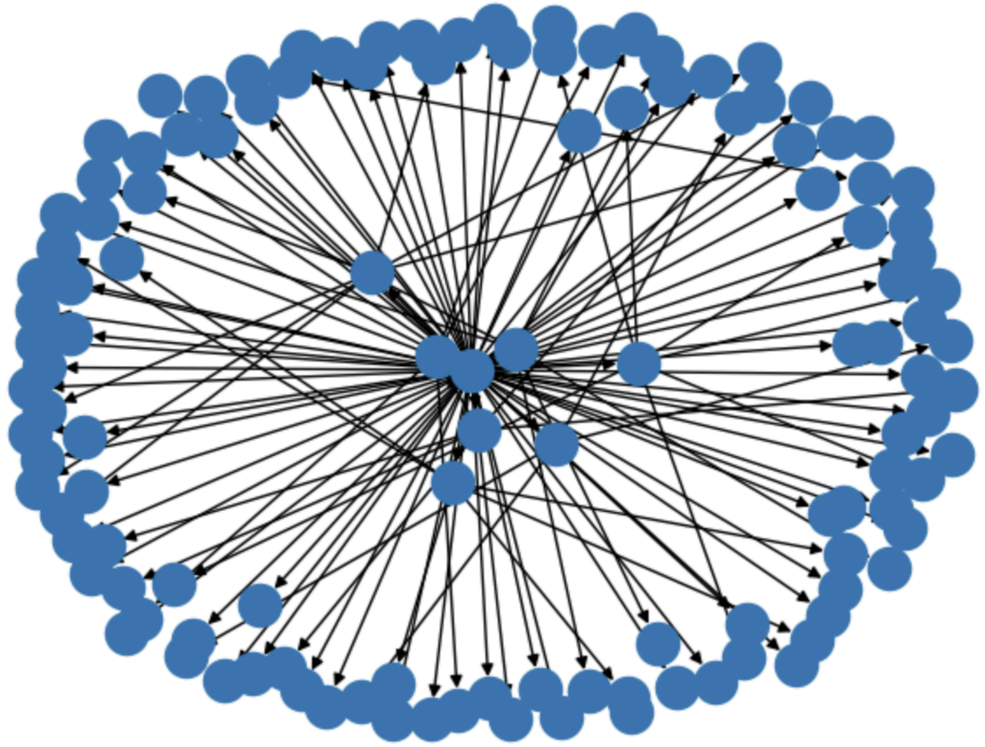


Fig 3. Node representing news and leaf nodes representing users

Training of GNN

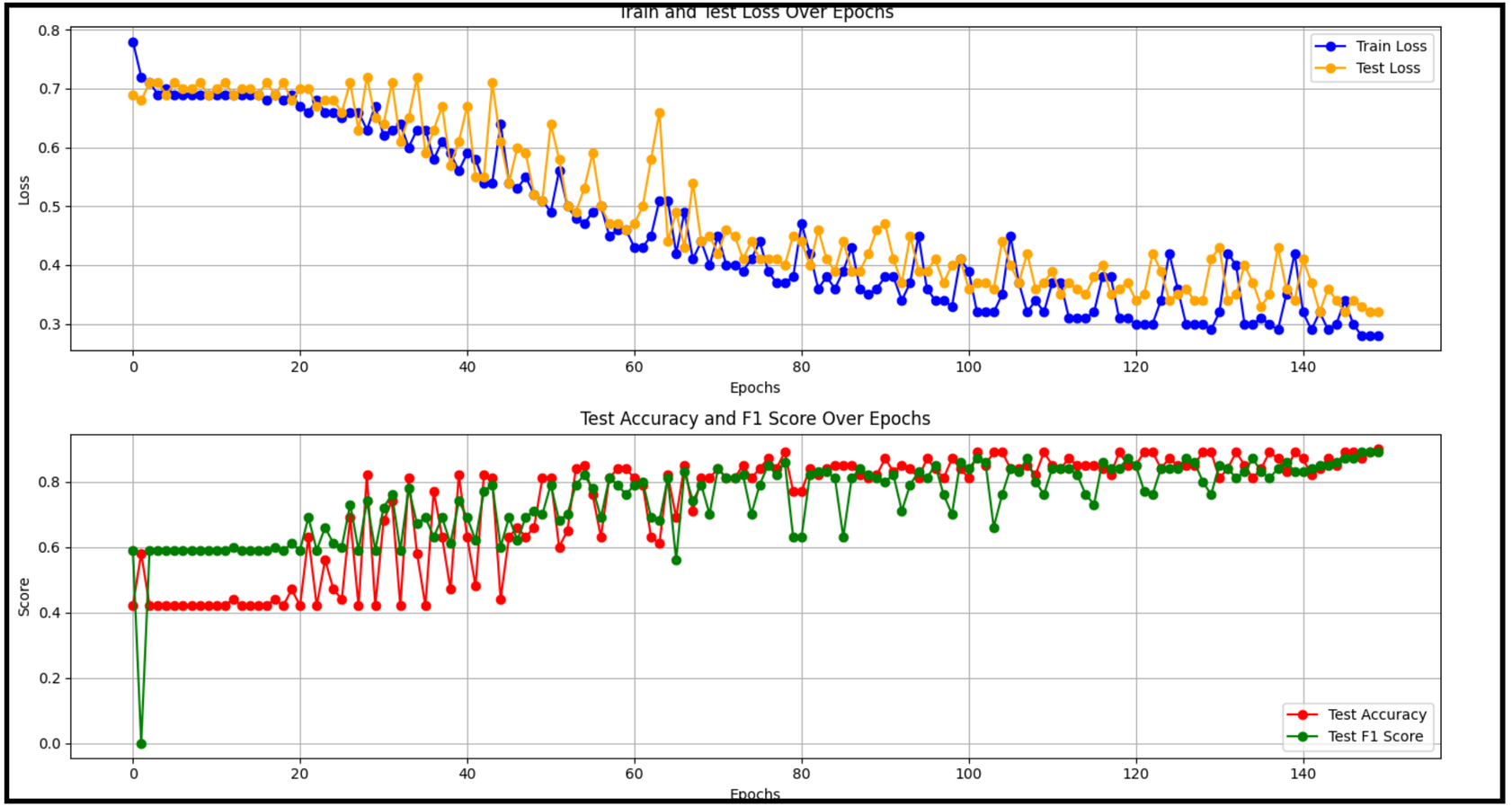


Fig 4. The test accuracy trend shows that the GNN model performs well under normal conditions with accuracy stabilizing at 90% and the loss curves further emphasize this trend, where both training and test losses converge smoothly during training.

Attack on GNN

The attack here stands for inserting nodes and edges in the graph(assumed to be static) to change the structure to decrease GNN's efficiency.

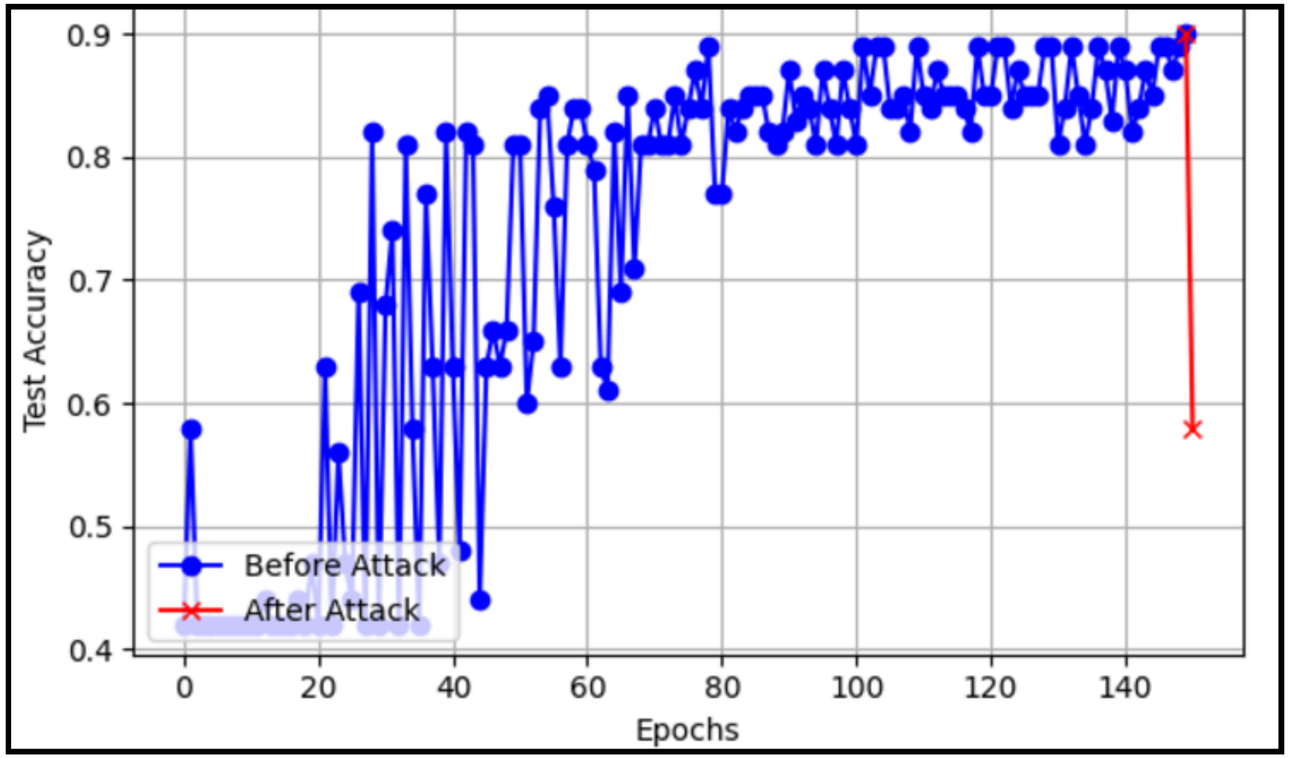


Fig 5. Test Accuracy Before and After Attack

Results Obtained

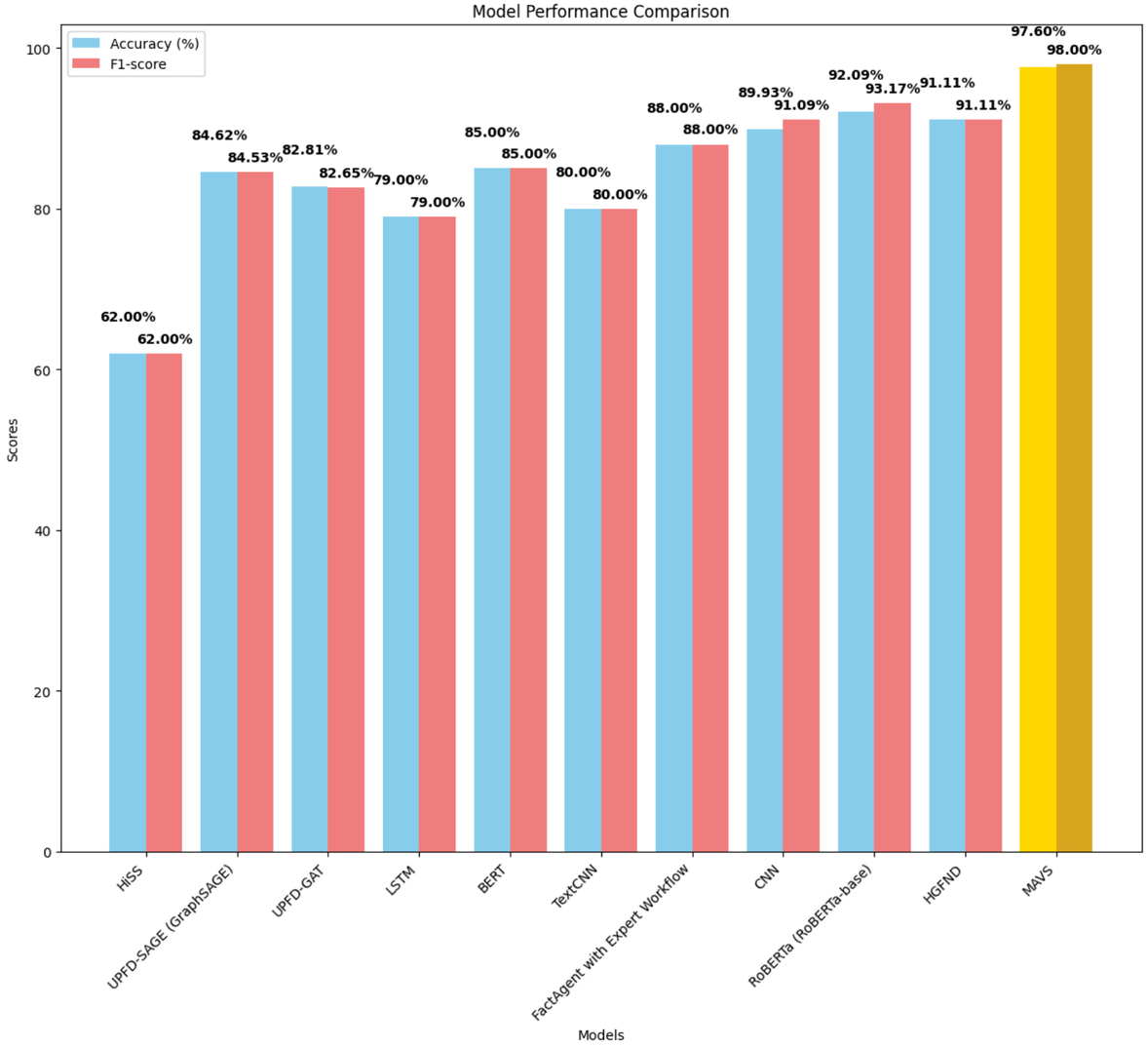


Fig 6. Accuracy and F1-score comparison of baseline models and MAVS.

Table 4. Comparison of Accuracy and F1-score between baseline models and MAVS [11-13]

Model	Accuracy (%)	F1-score(%)
HiSS	62	62
UPFD-SAGE (GraphSAGE)	84.62	84.53
UPFD-GAT	82.81	82.65
LSTM	79	79
BERT	85	85
TextCNN	80	80
FactAgent with Expert Workflow	88	88
CNN	89.93	91.09
RoBERTa (RoBERTa-base)	92.09	93.17
HGFND	91.11	91.11
MAVS	**97.6**	**98**

Results Obtained

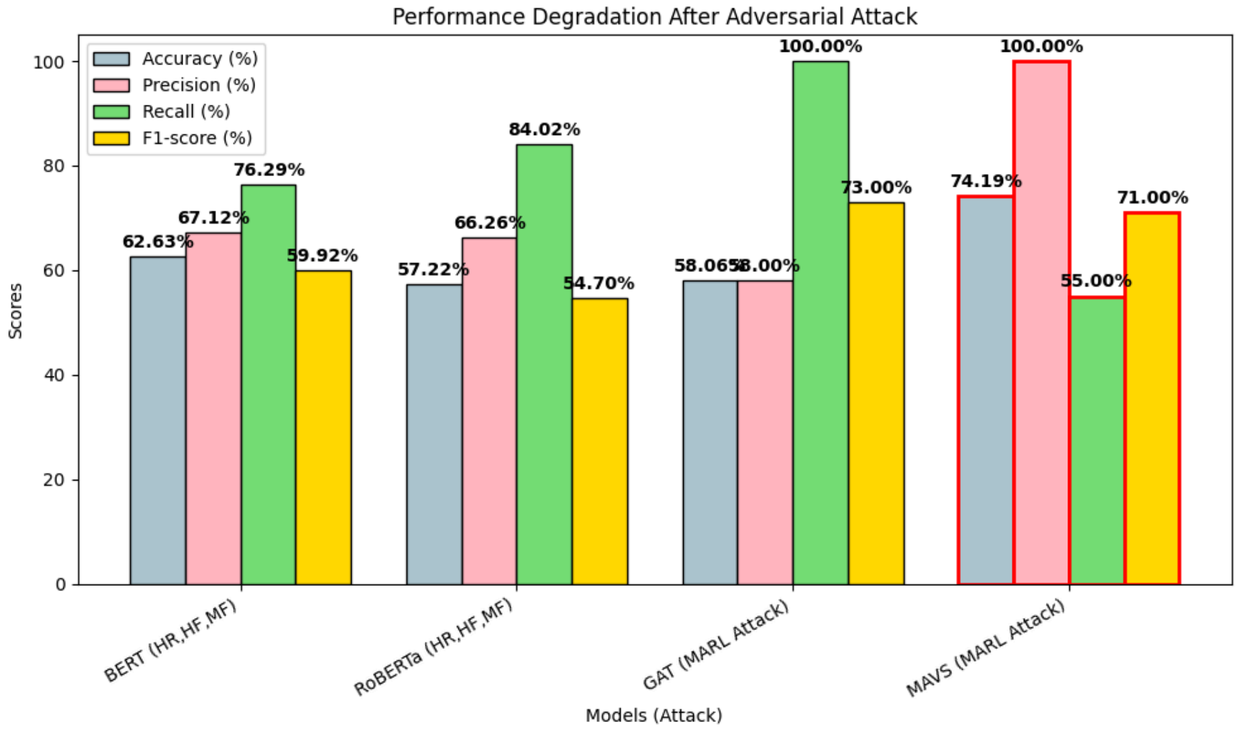


Fig 7. Performance comparison after adversarial attack for BERT, RoBERTa, GraphSAGE, and MAVS.

Table 5. Performance degradation after adversarial attack [10,14]

Model(Attack)	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
BERT (HR,HF,MF)	62.63	67.12	76.29	59.92
RoBERTa (HR,HF,MF)	57.22	66.26	84.02	54.70
GAT (MARL Attack)	58.06	100	58	73
MAVS (MARL Attack)	**74.19**	**55**	**100**	**71**

- BERT and RoBERTa experience a substantial drop, particularly in Recall and F1-score.
- GAT, despite maintaining a high precision of 100%, suffers from a considerable recall drop, suggesting that it misses a large number of true positives..
- MAVS demonstrates better robustness with an Accuracy of 74.19% and a balanced F1-score of 71%, and recall score of 100% for MAVS suggests that it avoids false negatives.

Results Obtained

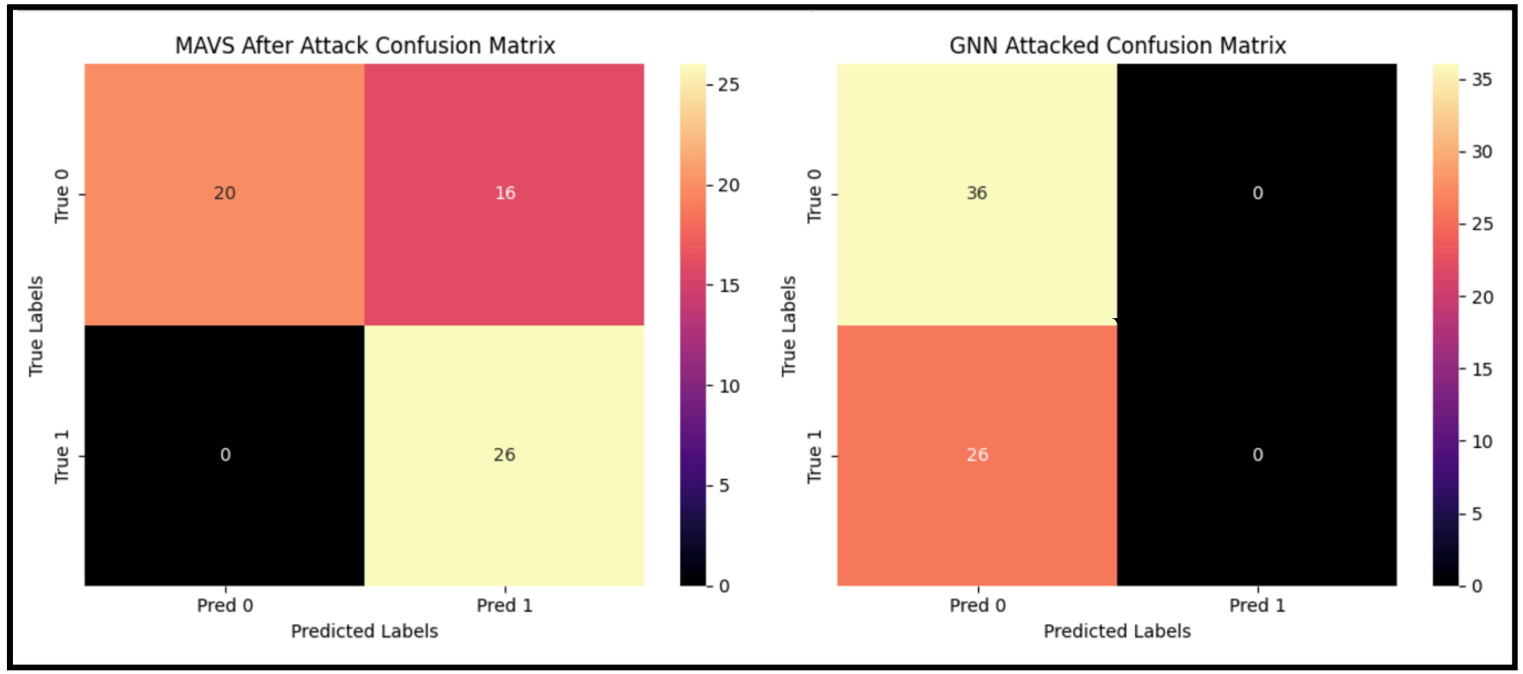


Fig 8. The comparison of GNN and MAVS performance post-attack.

- After the adversarial attack, MAVS maintains a balance between true positives (20) and false negatives (16) due to its fact-checking agent, indicating partial resilience to adversarial interference.
- Conversely, GNNs demonstrate a severe decline, completely failing to classify ”Fake News” instances, as all predictions default to ”Real.”

Results Obtained

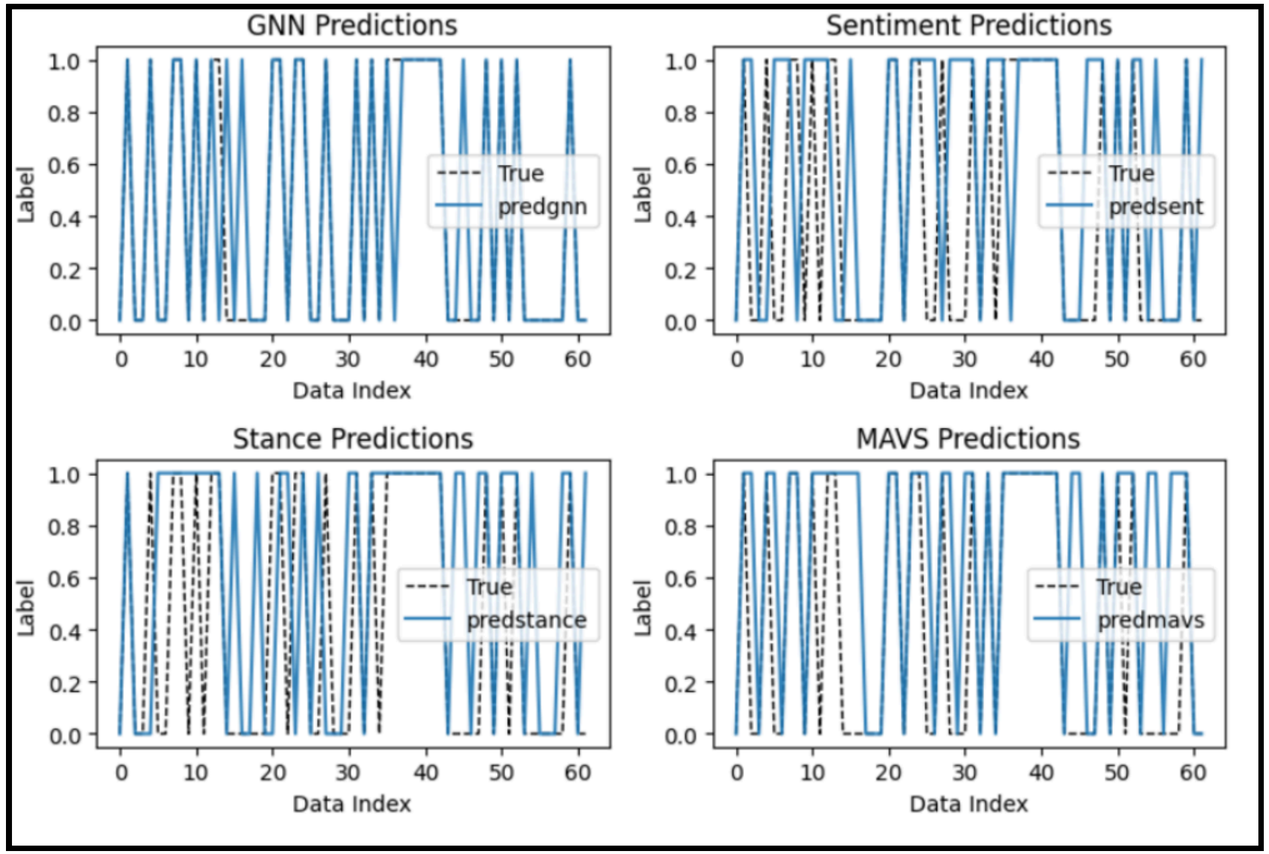


Fig 10. Comparison of True Values with Predictions for GNN, Sentiment, Stance, and MAVS Models

Agent	**Training Time Complexity**	**Inference Time Complexity**	**Space Complexity**
GNN	$O(EF + NF^2)$	$O(NF)$	$O(NF)$
Fact-Checking Agent	$O(NC)$	$O(C)$	$O(C)$
Stance-Checking Agent	$O(NT)$	$O(T)$	$O(T)$
Sentiment-Checking Agent	$O(NT)$	$O(T)$	$O(T)$
Overall MAVS Model	$O(EF + NF^2)$	$O(NF)$	$O(NF + C + T)$

Table 7. Time Complexity Analysis of Agents vs MAVS where E = Edges, N = Nodes, F = Features, C = Number of Claims , T = Token Count

The overall **time complexity** remains equivalent to that of the base GNN model.

*

Ablation Study

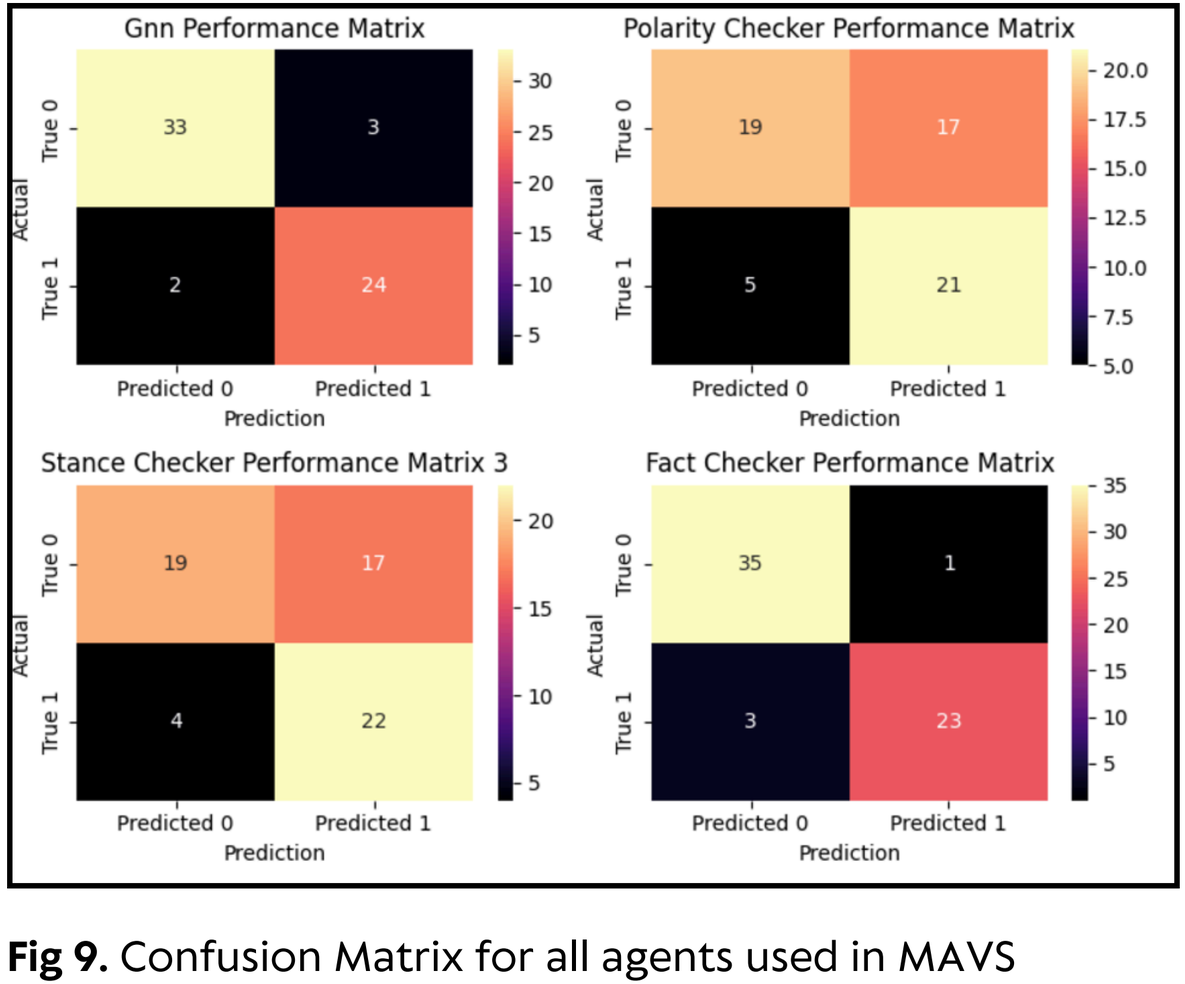


Fig 9. Confusion Matrix for all agents used in MAVS

Table 6. Performance Metrics comparison of agents used in MAVS

Agent	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
GNN	91.94	94.29	91.67	92.96
Sentiment Checker	64.52	79.17	52.78	63.33
Stance Checker	66.13	82.61	52.78	64.41
Fact Checker	93.55	92.11	97.22	94.59

- The results indicate that the GNN achieves the highest precision (94.29%), ensuring minimal false positives.
- The Fact Checker exhibits the highest recall (97.22%), making it the most effective at detecting fake news instances.
- The Sentiment Checker and Stance Checker, while weaker in precision, provide valuable complementary information.

Future Works

Future work will prioritize to study the spread of misinformation, test intervention strategies, and evaluate their effect on public opinion dynamics.

The robustness of MAVS will be systematically evaluated under a broader range of adversarial scenarios.

To include reinforcement learning for dynamic adjustment of agent weights improving adaptability and decision robustness in real-time.

References

Peer-reviewed sources

[1] Pennycook, Gordon and David G. Rand. “The Psychology of Fake News.” Trends in Cognitive Sciences 25 (2021): 388-402.

[2] Allcott, Hunt & Gentzkow, Matthew. (2017). Social Media and Fake News in the 2016 Election. Journal of Economic Perspectives. 31. 211-236. 10.1257/jep.31.2.211.

[3] Haoran Wang, Yingtong Dou, Canyu Chen, Lichao Sun, Philip S. Yu, and Kai Shu. 2023. Attacking Fake News Detectors via Manipulating News Social Engagement. In Proceedings of the ACM Web Conference 2023 (WWW '23). Association for Computing Machinery, New York, NY, USA, 3978–3986. https://doi.org/10.1145/3543507.3583868

[4] Xinyi Zhou and Reza Zafarani. 2020. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. ACM Comput. Surv. 53, 5, Article 109 (September 2021), 40 pages. https://doi.org/10.1145/3395046

[5] Alonso, Miguel A., David Vilares, Carlos Gómez-Rodríguez, and Jesús Vilares. 2021. "Sentiment Analysis for Fake News Detection" Electronics 10, no. 11: 1348. https://doi.org/10.3390/electronics10111348

[[6] Riedel, Benjamin & Augenstein, Isabelle & Spithourakis, Georgios & Riedel, Sebastian. (2017). A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. 10.48550/arXiv.1707.03264.

[7] Su, Xing & Xue, Shan & Liu, Fanzhen & Wu, Jia & Yang, Jian & Zhou, Chuan & Hu, Wenbin & Paris, Cécile & Nepal, Surya & Jin, Di & Sheng, Quan & Yu, Philip. (2022). A Comprehensive Survey on Community Detection With Deep Learning. IEEE Transactions on Neural Networks and Learning Systems. PP. 1-21. 10.1109/TNNLS.2021.3137396.

[[8] FaGANet: An Evidence-Based Fact-Checking Model with Integrated Encoder Leveraging Contextual Information] (https://aclanthology.org/2024.lrec-main.621/) (Luo et al., LREC-COLING 2024)

[9] Ahmed, Hadeer & Traore, Issa & Saad, Sherif. (2017). Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. 127-138. 10.1007/978-3-319-69155-8_9.

[10] Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. 2021. User Preference-aware Fake News Detection. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21). Association for Computing Machinery, New York, NY, USA, 2051–2055. https://doi.org/10.1145/3404835.3462990

[11] J. Alghamdi, Y. Lin, and S. Luo, “A comparative study of machine learning and deep learning techniques for fake news detection,” Information, vol. 13, p. 576, 2022.

[12] U. Jeong, K. Ding, L. Cheng, R. Guo, K. Shu, and H. Liu, “Nothing stands alone: Relational fake news detection with hypergraph neural network,”2022

[13] Li, Y. Zhang, and E. C. Malthouse, “Large language model agent for fake news detection,” 2024.

[14] J. Su, T. Y. Zhuo, J. Mansurov, D. Wang, and P. Nakov, “Fake news detectors are biased against texts generated by large language models,” 2023.

Thank you!