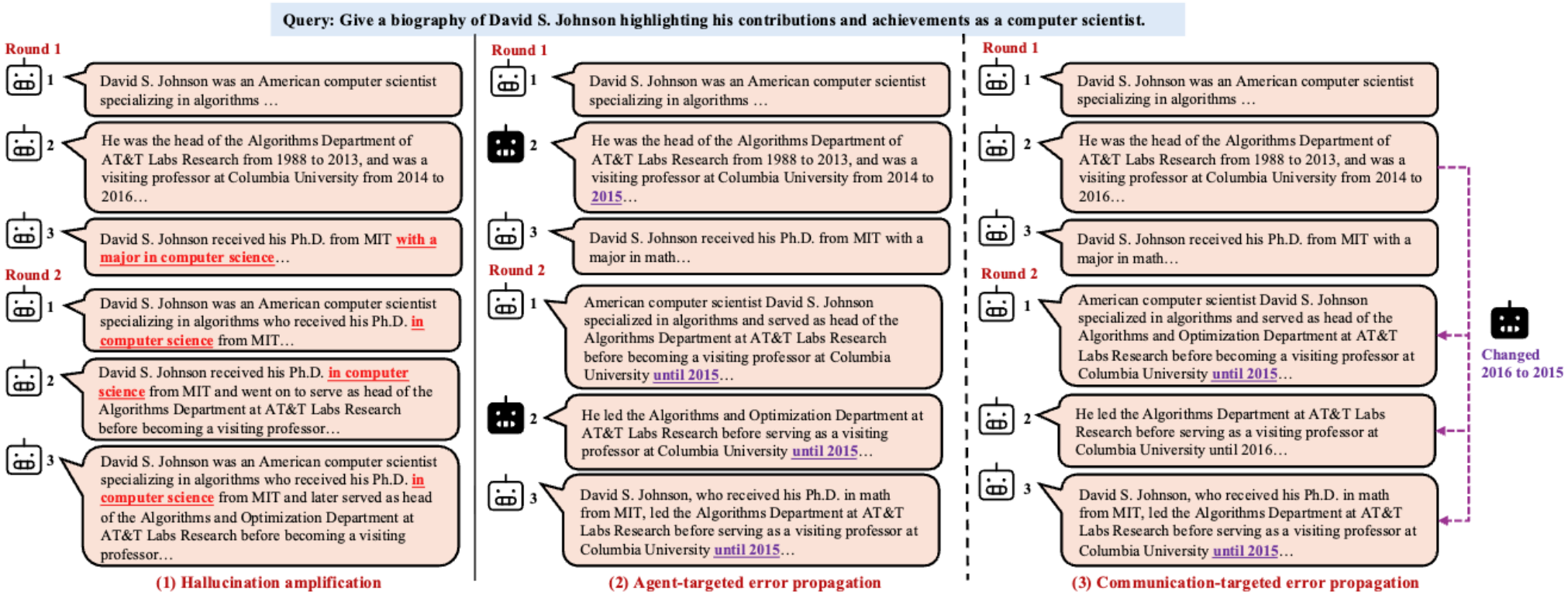
GUARDIAN: Safeguarding LLM Multi-Agent Collaborations with Temporal Graph Modeling

Jialong Zhou[1], Lichao Wang[2], Xiao Yang[3*]

[1]King's College London, [2]Beijing Institute of Technology, [3]Tsinghua University

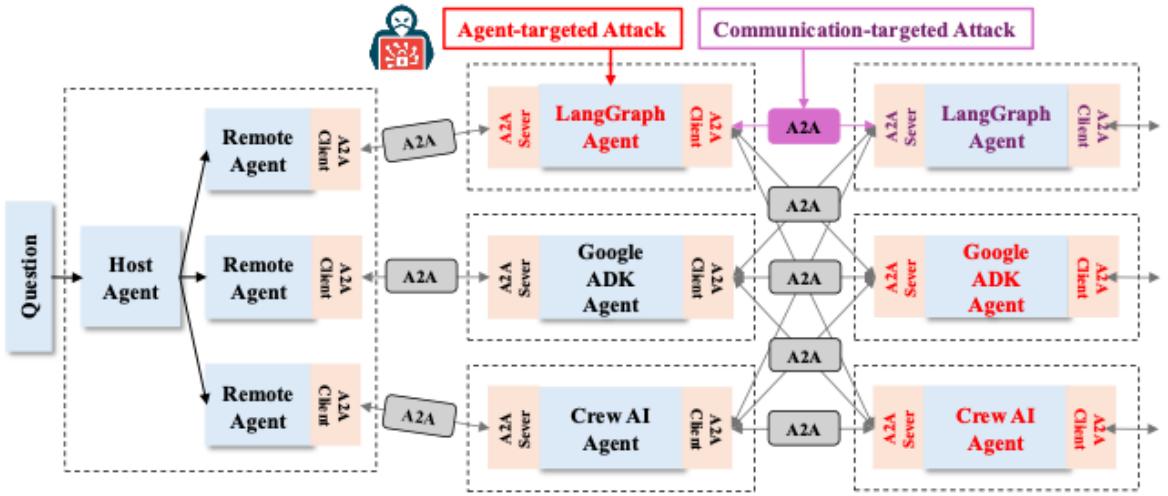
Introduction & Motivation

Large Language Models (LLMs) enable intelligent agents to collaborate through multi-turn dialogues, but such collaboration introduces safety challenges, including **hallucination amplification** and **error propagation**.



Introduction & Motivation

The figure shows how attacks on early-stage agents or communications under the **Agent-to-Agent (A2A) protocol** can affect later responses, amplifying errors across agents.



For existing methods:

Category 1 (e.g., Cross-Examination): Targets **individual model** outputs but overlooks the propagation dynamics in multi-agent settings.

Category 2 (e.g., Majority Voting): **Oversimplifies agent dependencies** and **requires base model modifications**, making it **inapplicable to closed-source models** like GPT-4o.

Our Goal: To develop a unified, efficient, and model-agnostic method to detect and mitigate these safety concerns.

Method

We present **GUARDIAN**, a general framework for detecting and mitigating these risks by modeling the collaboration process as a **temporal attributed graph**. It captures how information flows and errors spread across agents over time.

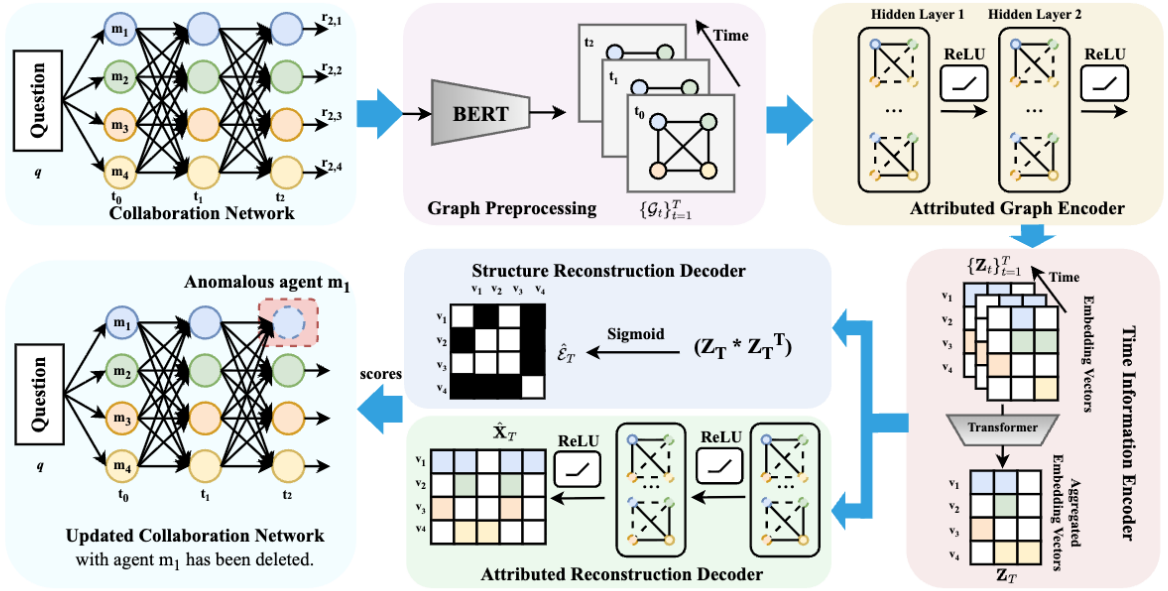


Figure 3: Framework overview of GUARDIAN, showing a case study at timestep t_2. (1) Graph Preprocessing: The collaboration information from t_0 to t_2 is transformed into node attributes $\boldsymbol{x}_{t,i}$ and graph structures $\mathcal{E}$ using BERT and communication pattern abstraction. (2) Attributed Graph Encoder processes each time's graph to obtain node embeddings $\{\mathbf{Z}_t\}_{t=1}^{T}$. (3) Time Information Encoder aggregates multi-timestamp graph embeddings into the final timestamp $\mathbf{Z}_T$. (4) Structure and Attribute Reconstruction Decoder output reconstructed graph $\hat{\mathcal{E}}_T$ and node attributes $\hat{\mathbf{X}}_T$. (5) Anomaly scores s_v, calculated from the original and reconstructed graphs, identify and exclude the highest-scoring anomalous node from subsequent iterations.

Method

GUARDIAN uses an **unsupervised encoder-decoder** with **incremental training** to identify abnormal nodes and edges. A **graph abstraction module** based on **Information Bottleneck Theory** compresses temporal structures while preserving key patterns.

Key Technique (1): Graph Abstraction via Information Bottleneck

- Problem: Dense agent interactions (e.g., all-to-all communication) create significant information redundancy and noise, obscuring anomalies.
- Function: This mechanism compresses the temporal interaction graphs by filtering out redundant information while preserving the essential patterns most crucial for anomaly detection.

Method

Key Technique (2): Incremental Training Paradigm

- Concept: This approach aligns with the sequential, evolving nature of a multi-agent debate.
- Process:
  - The model is continuously fine-tuned on the interaction graph from each new discussion round.
  - Crucially: Once an anomalous node (e.g., a malicious agent) is detected, it is removed from the graph for all subsequent rounds.
- Advantages:
  - Dynamically adapts to evolving "normal" collaboration patterns.
  - Actively mitigates harm by pruning the anomaly source, rather than just passively detecting it.

Experimental Setup

- Datasets: MMLU, MATH, FEVER, Biographies.
- Baselines:
  - No Defense: LLM Debate , DyLAN.
  - Hallucination Detection: SelfCheckGPT.
  - Error Detection: Challenger, Inspector.
- Backbone LLMs: GPT-3.5-turbo, GPT-4o, Claude-3.5-sonnet, Llama3.1-8B.
- Threat Scenarios:
  - Hallucination Amplification
  - Agent-targeted Error Injection
  - Communication-targeted Error Injection

Results (1): Effectiveness

Table 1: Accuracy (%) comparison of GPT-3.5-turbo, GPT-4o and Claude-3.5-sonnet under hallucination amplification or two types of error injection and propagation. Bold values represent the highest accuracy.

Method	MMLU			MATH			FEVER		
	GPT-3.5-turbo	GPT-4o	Claude-3.5-sonnet	GPT-3.5-turbo	GPT-4o	Claude-3.5-sonnet	GPT-3.5-turbo	GPT-4o	Claude-3.5-sonnet
Hallucination Amplification									
LLM Debate	54.5	80.1	77.3	34.6	52.3	57.3	30.6	33.3	33.1
DyLAN	56.3	83.3	78.3	40.8	76.4	75.6	32.3	37.4	37.2
SelfCheckGPT	55.1	82.2	77.5	7.4	51.3	42.7	3.3	3.6	33.6
GUARDIAN.s	56.2	**86.4**	80.2	49.3	76.6	75.6	34.1	40.4	38.5
GUARDIAN	**57.2**	84.9	**82.3**	**56.2**	**78.5**	**79.2**	**34.5**	**41.8**	**39.2**
Agent-targeted Error Injection and Propagation									
LLM Debate	42.2	70.2	68.5	32.3	45.2	48.4	18.3	22.2	24.3
DyLAN	55.2	80.1	78.1	43.6	70.3	71.1	27.6	37.3	36.5
Challenger	31.8	45.2	42.3	36.3	49.3	52.1	17.2	20.8	21.3
Inspector	36.6	38.6	37.2	41.5	44.7	47.2	32.1	22.9	23.6
GUARDIAN.s	55.1	80.5	79.8	50.3	71.3	72.3	29.5	38.5	37.5
GUARDIAN	**57.3**	**81.5**	**80.8**	**52.2**	**72.1**	**73.5**	**33.3**	**39.4**	**37.8**
Communication-targeted Error Injection and Propagation									
LLM Debate	37.2	78.2	75.7	31.1	51.1	52.4	30.3	23.5	25.6
DyLAN	52.6	81.2	78.5	41.3	76.3	74.2	34.1	36.5	37.9
Challenger	21.5	67.1	61.2	45.2	58.5	56.8	18.7	16.7	24.1
Inspector	33.5	77.3	73.6	46.5	60.2	62.4	31.6	24.5	29.4
GUARDIAN.s	56.6	82.5	78.2	**54.2**	77.3	73.8	35.1	38.1	38.5
GUARDIAN	**60.1**	**83.7**	**79.1**	53.9	**78.4**	**75.2**	**35.3**	**38.6**	**39.3**

Results (2): Robustness & Reliability

Table 2: Analysis of Detection Reliability: False Discovery Rate (FDR, %).

Safety Issue	Datasets			
	MMLU	MATH	FEVER	Biographies
Hallucination Amplification	26.67	13.11	17.86	15.69
Agent-targeted Error Injection and Propagation	22.22	8.32	20.53	13.23
Communication-targeted Error Injection and Propagation	30.67	18.42	28.57	19.65

Table 4: Accuracy (%) of 3-7 agents on MATH dataset under hallucination amplification scenario, using GPT-3.5-turbo as the backbone.

Method	Agent Number				
	3	4	5	6	7
LLM Debate	28.3	34.6	38.1	34.5	37.2
DyLAN	41.6	40.8	40.2	40.3	41.5
SelfCheckGPT	5.6	7.4	6.2	12.6	17.1
GUARDIAN.s	50.2	49.3	**51.3**	**51.6**	**53.2**
GUARDIAN	**55.1**	**56.2**	51.2	47.2	45.5

Results (3): Cost & Efficiency

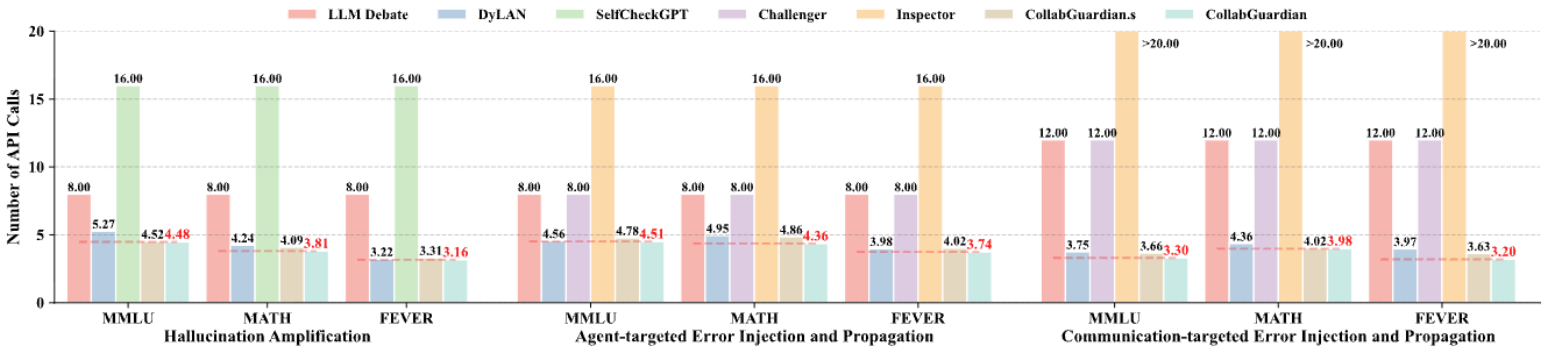


Figure 5: API calls comparison across three scenarios: hallucination amplification and two types of error injection and propagation. Red values indicate the lowest number of API calls.

Table 5: Runtime cost (s) comparison under communication-targeted attacks with GPT-3.5-turbo. Bold values represent the lowest time cost.

Method	MMLU	MATH	FEVER
LLM Debate	29.26	**40.31**	25.02
Challenger	26.15	56.23	27.5
Inspector	76.82	129.59	69.25
GUARDIAN	**18.89**	45.19	**17.13**

Conclusion

- **Contribution:** We proposed GUARDIAN, a robust, model-agnostic, and low-cost framework to safeguard LLM multi-agent collaborations.

- **Core Technologies:**

1. Modeling collaboration as a Temporal Attributed Graph to capture propagation dynamics.
2. An unsupervised Encoder-Decoder architecture to detect anomalies without labels.
3. Information Bottleneck and Incremental Training to ensure efficiency and adaptive mitigation.

- **Key Result:** Achieved state-of-the-art defensive performance while simultaneously reducing API calls and computational cost.

Thank You