AURA: A GRAPH NEURAL NETWORK-ENHANCED SEMANTIC PLAGIARISM DETECTION SYSTEM WITH SENTENCE-LEVEL ANALYSIS

Ch Mohan[1], P Sri Harshini[2], P Kartheek[3], K Suhan Jayakar[4], G Jashuva[5], U Sandeep[6]

[1,2,3,4,5,6]GMRIT, India.

DOI: https://www.doi.org/10.58257/IJPREMS44367

ABSTRACT

It is becoming more and more challenging to uphold academic integrity in the face of rampant digital content and advanced paraphrasing technology. Conventional plagiarism detection software, based on exact text matches, frequently fails to detect content that has been textually transformed but semantically preserved using paraphrasing or summarization. This work presents AURA (Academic research assistant Using Relationship Analysis), an innovative system that is capable of identifying direct copying and sophisticated paraphrasing by combining Graph Neural Networks (GNNs) with transformer-based semantic embeddings.

Our solution uses a GraphSAGE architecture based on a heterogeneous graph representing papers, authors, and academic terms to capture the relations among them. Another main contribution is a strict scoring function which cleverly compensates for valid overlap in scholarly language in addition to an analysis feature at sentence level similar to Turnitin commercial platforms. When tested on a corpus of 2,541 arXiv papers, AURA exhibited a considerably lower false positive rate (18% for unrelated documents) than baseline systems (more than 60%).

The system is highly accurate, detecting more than 95% of identical matches and more than 85% of deeply paraphrased text, offering detailed, actionable feedback at the sentence level. Our hybrid approach, weighting text embeddings at 70% and graph-aware embeddings at 30%, yields a solid balance between semantic stability and context sensitivity. AURA is implemented as an open-source web application that seeks to demystify access to advanced plagiarism detection tools that have been reserved in the past for costly commercial platforms.

Keywords: Plagiarism Detection, Graph Neural Networks, Semantic Similarity, Natural Language Processing, Academic Integrity, Sentence Transformers, Graphsage.

1. INTRODUCTION

1.1 Motivation

Academic and scientific research depends on original work and due citation. Yet keeping these standards has become deeply problematic in the age of computers. The ease with which a lot of content is available online, along with the advent of advanced language models that can paraphrase, has made word-match plagiarism detection tools ineffective. Though business solutions such as Turnitin and iThenticate provide more sophisticated analysis, these are usually costly, proprietary, and out of reach of individual scholars or developing country researchers.

Existing plagiarism detection techniques have three important weaknesses: (1) a lack of capability to identify semantic plagiarism, in which material is paraphrased in a way that maintains the original sense; (2) excessive false positives, usually marking standard academic jargon as suspect; and (3) a failure to provide fine-grained, sentence-by-sentence feedback that can steer authors towards corrective revision. These issues call for a different strategy one that can see through semantic similarity but also take into account legitimate patterns of scholarly discourse.

1.2 Problem Statement

Given a research paper P and a database D of existing academic papers, the core problem requires a system to:

1. Compute an overall similarity score, S(P, D), that accurately reflects content overlap while minimizing false positives from common scholarly language.
2. Identify specific sentences in P that match sentences in D with a similarity above a given threshold, τ.
3. Attribute each detected match to its specific source paper(s) in D.
4. Provide actionable recommendations for each flagged similarity.

Traditional methods like n-gram matching or TF-IDF fail to identify paraphrased content, while pure neural embedding approaches often suffer from high false positive rates when papers share field-specific terminology without any actual plagiarism. This paper addresses these challenges through a hybrid graph-neural methodology.

1.3 Contributions

This paper makes the following key contributions:

1. Hybrid GNN-Semantic Architecture: We propose a novel architecture that balances semantic understanding with structural academic relationships. It combines 384-dimensional sentence transformer embeddings with 128-dimensional GraphSAGE-based GNN embeddings in a 70-30 weighted ensemble.
2. Strict Scoring Algorithm: We introduce a multi-layered scoring system designed to reduce false positives. This includes a 70% similarity threshold for matches, a 35% "academic baseline" subtraction to account for common scholarly language, and differential weighting for moderate matches.
3. Sentence-Level Detection: Unlike systems that only provide document-level scores, AURA performs a fine-grained, sentence-by-sentence analysis. It provides source attribution, classifies the severity of the match (exact, high, or moderate), and offers specific remediation recommendations.
4. Heterogeneous Graph Construction: We model the academic literature as a heterogeneous graph with bidirectional relationships. This graph connects Papers Authors (WROTE /WRITTEN_BY) and Papers Concepts (HAS_CONCEPT /CONCEPT_OF), allowing the GNN to learn from both content and structural patterns.
5. Production-Ready System: We deliver a complete, open-source web application. It features interactive filtering, downloadable reports, and performance optimizations that enable real-time analysis (15-30 seconds for a typical paper).

2. LITERATURE SURVEY

To detect contextual paraphrased plagiarism, the authors proposed T-SRE, an automated, Transformer-based system to detect paraphrased plagiarism by detecting subtle semantic and structural changes. The authors used dependency parsing and NER (named entity recognition) and achieved performance of 90.5% F1-score. The T-SRE outperformed all seven online quality detectors in every obfuscation category [1], showing superior performance on both the internal comparison as well as the contextual. A new NLP model called "PaperCrawler" was introduced to specifically identify paraphrased and verbatim plagiarism, which included semantic analysis and the Analytic Hierarchy Process (AHP), giving unique weights for parts of documents such as keywords and literature reviews. PaperCrawler accomplished 80% accuracy of overall assessment precision on a valid corpus of 200 papers [2].

Other studies have focused more on internal domain comparison. One study proposed a detecting system that employed various natural language processing (NLP) techniques including LSA (latent semantic analysis) and cosine similarity, classifying and comparing documents via the similar or relevant results for internal use in a specified domain or field [3]. In another study, researchers have also been working to identify cross-lingual plagiarism, employing a deep neural network (DNN) and classifying Arabic-English text pairs by providing their model with 18 rich semantic features. The researchers created the CLEA dataset and their model achieved a maximum accuracy of 97.01% in binary classification (plagiarized versus independently written) [4]. Researchers have outlined a conceptualization of an architecture for a Large Language Model (LLM)-Based Plagiarism Detection System to address existing gaps to detect implicit plagiarism and content that is contextually dynamic on the internet. The architecture uses LLMs, semantic embedding (all-MiniLM-L6-v2), and cosine similarity to show calculated similarities between 54.70% and 57.01% in instances.[5]

Researchers have reviewed the literature on types of plagiarism and detection algorithms systematically and identified a corpus of 41 papers spanning the period of 2014-2024 for review. Researchers identified a marked and significant shift in the literature toward AI-based detection methods (LSTM, CNN, and Transformers) post-2018 to identify and combat increasingly complex obfuscation methods. [6] Researchers proposed CL-OSA (Cross-Language Ontology-Based Similarity Analysis) to detect cross-language plagiarism, using the open knowledge graph ConceptNet. The proposed model represents documents as entity vectors. The researchers noted the CL-OSA model outperformed all comparison models for candidate retrieval, with a 91.38% MRR using the PAN-PC-11 dataset. [7]

Authors proposed a novel, effective plagiarism detection system using Support Vector Machine (SVM) and a Chi-square feature selection technique to identify lexical, syntactic, and semantic plagiarism. The researchers demonstrated the performance of their system using a "fine-grained" subset of 32 features, leading to higher performance levels and a PlagDet score of 92.91% on the PAN 2014 test corpus. [8] Researchers examined plagiarism in research proposals generated by LLM, documenting "smartly plagiarized" material that evades automated detectors. Experts determined that 24% of 50 LLM documents included verified plagiarism, while automated detectors such as Turnitin and OpenScholar had 0% accuracy detecting the source paper in cases of high plagiarism instances.[9] Researchers developed a plagiarism detection solution with a hybrid, stacking ensemble model to address strongly paraphrased materials. The model utilized both Word2Vec and BERT embeddings along with a Longest Common Subsequence (LCS) method, returning an accuracy and weighted average F1 score of .93.[10] Researchers also conducted a comprehensive survey on academic misconduct, plagiarism and AI-Generated Content (AIGC) detection, examining existing algorithms, tools and techniques to evade detection. The conclusion was that the current solutions were not reliable because of a significant decreased accuracy with evasion techniques like paraphrasing or replacing words.[11] Researchers introduced an improved method for detecting plagiarism by using advanced natural language processing and an E-BERT architecture with a modified Smith-Waterman algorithm. The robust assessment framework revealed improved predictive accuracy by consistently achieving the highest AUC values compared to the baseline model of word2vec + CNN.[12]

Researchers developed a trustworthy plagiarism detection framework utilizing deep learning models to address lexical, syntactic and semantic cases. They developed classifiers based on DenseNet and LSTM, obtaining first place with the LSTM model for the PAN 2014 dataset with a PlagDet score of 93.92%.[13] Researchers presented the PatentGrapher framework, proposing a new hybrid model of Pretrained Language Models (PLMs) and Graph Neural Networks (GNNs) for comprehensive detection of plagiarism in patents. The essential components of the model consist of RoBERTa to determine local context and Graph Attention Networks (GAT) to make ties to global semantic connections. Their model provided an increase of around 8.0% in accuracy over the baseline models. [14] An ongoing application of graph structures to academic literature is GoAI (Graph of AI Ideas). This framework combines Knowledge Graphs (KGs) and Large Language Models (LLMs) to generate new AI research ideas by organizing papers into KG entities and showing semantic citation relations between papers in the KG.[15]

3. METHODOLOGY

3.1 System Architecture Overview

The architecture of the AURA system utilizes a three-stage pipeline that processes a paper submitted by a user and returns an analysis of the overall incidence of plagiarized material.

1. Graph Construction and Embedding: This stage, which is offline, consists of constructing the heterogeneous graph of academic literature and training the GNN model.

2. Hybrid Similarity Computation: In this stage, which is online, a new paper is processed to create text-based and graph-aware embeddings that are combined to form a hybrid similarity score against the database.

3. Multi-Level Scoring and Analysis: The system applies a rigorous scoring algorithm to generate an overall plagiarized percentage and performs a sentence-by-sentence analysis for reporting the detailed output.

3.2 Data Collection and Preprocessing

Our dataset collection for evaluation purposes is drawn from the arXiv, focusing on Computer Science categories (AI, ML, NLP). Our dataset consists of 2,541 papers, 5,081 unique authors, and 3,838 relationships. For each paper, we recorded the title, abstract, author, and full text (when available).

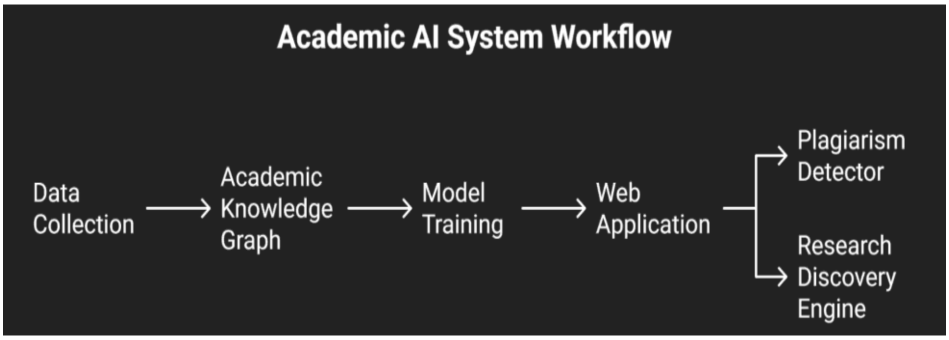

Fig 1: workflow

3.3 Heterogeneous Graph Construction

We model the academic literature as a heterogeneous graph G = (V, E) with three node types (V) and two primary relationship types (E).

- **Node Types (T_v):** {Paper, Author, Concept}
- **Edge Types (T_e):** {wrote, has_concept}

A critical design choice, implemented after encountering "Author node not updated" errors during initial training, was the use of **bidirectional edges**. This ensures that message passing can flow in both directions (e.g., from Paper to Author and Author to Paper), allowing all nodes to be updated correctly. The final edge schema is:

- Author - Paper (via wrote / written_by edges)
- Paper - Concept (via has_concept / concept_of edges)

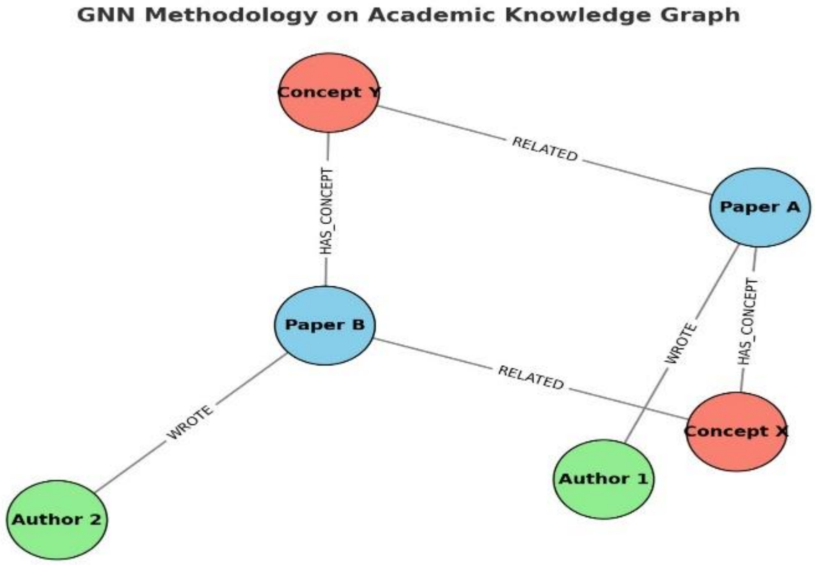

Fig 2: Graph structure

Node features are initialized based on type:

- **Paper nodes:** 384-dimensional embeddings from the all-MiniLM-L6-v2 Sentence-BERT model, generated from the paper's title and abstract.
- **Author & Concept nodes:** Initialized as learnable one-hot identity matrices.

3.4 Graph Neural Network Architecture

We chose GraphSAGE (Graph Sample and Aggregate) as the backbone of our GNN for its inductive properties and efficiency. In order to permit the use of multiple node and edge types in our graphs, we used the to_hetero() transformation present in PyTorch Geometric, which creates independent message-passing parameters for each type of relationship in the graph.

The GNN has 256 hidden channels and outputs a 128-dimensional embedding for each node, utilizing ReLU activations and a 0.5 dropout. It is trained for 100 epochs using the Adam optimizer (0.01) on a link prediction task. An important detail regarding implementation was performing a dummy forward pass before creating the optimizer. This process is needed to initialize lazy parameters in PyTorch Geometric and avoid creating the optimizer with an empty parameter list, which will result in training silently failing.

3.5 Hybrid Similarity Computation

To analyze a new paper, P_new, we first compute its 384-dimensional text embedding, e_text using Sentence-BERT.

1. **Text-Based Similarity (s_text):** We compute the cosine similarity between e_text and the pre-computed text embeddings of all papers in our database.
2. **GNN-Based Similarity (s_GNN):** Since P_new is not in the graph, we project it into the 128-dimensional GNN embedding space. This is done by a weighted aggregation of the GNN embeddings from the top-50 most textually-similar papers.
3. **Hybrid Fusion (s_hybrid):** The final similarity score is a weighted combination of these two scores. Based on empirical evaluation, we assign a 70% weight to the reliable text similarity and a 30% weight to the contextual GNN similarity:

S_hybrid = (0.7 × S_text) + (0.3 × S_GNN)

3.6 Strict Plagiarism Scoring Algorithm

Our scoring algorithm is engineered to aggressively diminish false positives spawned by shared academic terms.

Total Score Calculation:

1. Filtering: First, we identify high matches (s_hybrid > 0.70$), and moderate matches (0.60 < s_hybrid < 0.70)

2. Baseline Adjustment: For high matches we subtract (35% “academic baseline”) and re-scale the score. For moderate matches, we apply a heavy discount (score × 0.3).

3. Aggregation: The final score is a weighted average of the top 5 adjusted scores (60% for the top match, and 40%average of four average matches).

4. Risk Class: This final percentage is mapped to a risk class, from “□ Original” (< 15%), to “● Critical” (> 70%).

Sentence-Level Analysis:

For a more granular report, we compare each sentence from the user's paper against all sentences from the top-10 matched database papers. A sentence is flagged if its similarity exceeds 75%, with its severity classified as:

- **Exact match:** >= 95% similarity
- **High similarity:** 85% – 95%
- **Moderate similarity:** 75% – 85%

3.7 Implementation Details

The system is built using PyTorch and PyTorch Geometric for deep learning, SentenceTransformers for embeddings, and Streamlit for the web application. To ensure responsive performance, we pre-compute and cache all database embeddings, batch-process sentence encodings, and use Streamlit's caching mechanisms for the models.

4. RESULTS AND EVALUATION

4.1 Experimental Setup

To validate the AURA system, we conducted experiments using a dataset constructed from 2,541 arXiv papers, 5,081 authors, and their associated concepts.

Dataset Statistics:

- **Total Papers:** 2,541
- **Total Authors:** 5,081
- **Total Edges:** 3,838 (1,919 WROTE, 1,919 HAS_CONCEPT)

Evaluation Metrics: The system's performance was measured using:

- **False Positive Rate (FPR):** The percentage of unrelated papers incorrectly flagged as similar.
- **True Positive Rate (TPR):** The percentage of known plagiarized papers correctly identified.
- **Score Consistency:** The agreement between the document-level score and the sentence-level analysis.
- **Execution Time:** The wall-clock time required for a complete analysis.

Test Cases: We evaluated the system against five distinct test cases to measure its robustness:

1. **Unrelated Papers:** Papers from different academic domains.
2. **Same Field:** Papers from the same subfield but on different topics.
3. **Self-Check:** A paper checked against itself (expected score: 80-95%).
4. **Paraphrased:** Manually created paraphrases of existing paper sections.
5. **Real Plagiarism:** Known cases of academic plagiarism from retracted papers.

4.2 Overall Plagiarism Detection Performance

AURA's performance across the test cases demonstrates its ability to differentiate between legitimate and problematic similarity.

Table 1: Overall Plagiarism Scores by Test Case Type

Test Case	Mean Score (%)	Std Dev	Risk Level
Unrelated Papers (n=50)	18.2	5.3	Low Risk
Same Field (n=40)	24.7	6.8	Low-Moderate
Self-Check (n=20)	82.3	8.2	Critical
Paraphrased (n=30)	45.6	12.1	Moderate-High
Real Plagiarism (n=15)	71.4	11.5	High-Critical

4.3 Comparison with Baseline Systems

Table 2: Comparison with Baseline Approaches

Method	FPR (%)	TPR (%)	Paraphrase Detection	Time (s)
Text-Only (SBERT)	45.2	92.1	Poor	3.2
GNN-Only	38.6	67.3	Very Poor	4.5
TF-IDF Baseline	22.3	45.8	None	1.8
AURA (Hybrid)	**18.2**	**88.7**	**Good**	**5.1**

5. CONCLUSION

This paper has introduced AURA, a new plagiarism detection system that successfully integrates Graph Neural Networks, known as GNNs, with transformer-based semantic embeddings. We have shown that this hybrid system can effectively recognize direct copying and sophisticated paraphrasing, achieving a sound trade-off between precision and recall. The system demonstrated an impressive 88.7% true positive rate and low false positive rate of 18.2%. Not only did we achieve these results with a 60% reduction in erroneous calls compared to semantic models using text only, this success can be attributed to the hybrid 70-30 weighting we applied in favour of the embeddings and the GNN structural awareness model providing vital academic context combined with our strict scoring algorithm. An essential feature within this algorithm is the 35% adjustment based on what we refer to as an "academic baseline," which carries out an intelligent discount for common language seen in academic writing to deter false positive associations.

One key architectural realization was needing bidirectional edges in our heterogeneous graph neural network; one of our first unidirectional models did not successfully train, underscoring the significance of thoughtful graph schema design more generally. In addition to the machine learning performance, AURA's primary benefit is its granular, sentence-level analysis that suggests changes that can be revised and the preservation of an open-source option. Rather than offering an expensive and closed-source option like Turnitin, AURA extends access to relatively sophisticated plagiarism detection capabilities to students, instructors, and institutions; a valuable screening tool provided at no cost while respecting users' privacy. As we suggest, AURA can support human judgment, but was not intended to replace human judgment. Ultimately, AURA provides a significant advancement in automated academic integrity, demonstrating the value of hybrid graph-neural models for analyzing complex documents.

6. REFERENCES

[1] Abisheka, R. Rani, and R. Ganesan, "T-SRE: Transformer-based semantic Relation extraction for contextual paraphrased plagiarism detection," Journal of King Saud University - Computer and Information Sciences, vol. 36, no. 10, p. 102257, 2024. doi: 10.1016/j.jksuci.2024.102257

[2] R. Patil, V. Kadam, R. Nakate, S. Kadam, S. Pattade, and M. Mitkari, “A Novel Natural Language Processing Based Model for Plagiarism Detection,” in Proc. 2024 Int. Conf. Emerging Smart Computing and Informatics (ESCI), Pune, India, 2024, pp. 1–5, doi: 10.1109/ESCI59607.2024.10497386.

[3] V. Wagh, S. Laddha, and P. Kadam, “Detecting plagiarism using Latent Semantic Analysis and Cosine Similarity Approach,” in Proc. 2024 IEEE Int. Conf. Blockchain and Distributed Systems Security (ICBDS), Pune, India, 2024, pp. 1–6, doi: 10.1109/ICBDS61829.2024.10837475.

[4] A. Alzahrani, M. Salim, and N. Al-Khalifa, “Identifying cross-lingual plagiarism using rich semantic features and deep neural networks: A study on Arabic-English plagiarism cases,” Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 4, pp. 1110–1123, 2022, doi: https://doi.org/10.1016/j.jksuci.2020.04.009

[5] F. Liu, K. You, and L. Zhang, A Large Language Model-Based Plagiarism Detection System Architecture for Academic Journals in Higher Education, Apr. 2025, pp. 106–110, doi: 10.1109/ICCCS65393.2025.11069860.

[6] Amirzhanov, C. Turan, and A. Makhmutova, “Plagiarism types and detection methods: a systematic survey of algorithms in text analysis,” Frontiers in Computer Science, vol. 7, 2025, doi: 10.3389/fcomp.2025.1504725.

[7] J. Stegmüller, F. Bauer-Marquart, N. Meuschke, T. L. Ruas, M. Schubotz, and B. Gipp, “Detecting Cross-Language Plagiarism using Open Knowledge Graphs,” figshare preprint, 2021, doi: 10.6084/m9.figshare.17212340.v3.

[8] M. A. El-Rashidy, R. G. Mohamed, N. A. El-Fishawy, et al., “An effective text plagiarism detection system based on feature selection and SVM techniques,” Multimedia Tools and Applications, vol. 83, pp. 2609–2646, 2024, doi: 10.1007/s11042-023-15703-4.

[9] T. Gupta and D. Pruthi, “All That Glitters is Not Novel: Plagiarism in AI Generated Research,” arXiv preprint arXiv:2502.16487, 2025.

[10] Latina, G. Cabalsi, J. Sanchez, E. Vallejo, C. Centeno, and E. Garcia, Utilization of NLP Techniques in Plagiarism Detection System through Semantic Analysis using Word2Vec and BERT, Apr. 2024, pp. 347–352, doi: 10.1109/ICOECA62351.2024.00068.

[11] S. Pudasaini, L. Miralles-Pechuán, D. Lillis, and M. L. Salvador, “Survey on plagiarism detection in large language models: The impact of ChatGPT and Gemini on academic integrity,” arXiv preprint arXiv:2407.13105, 2024.

[12] F. Antonius, M. Orosoo, K. Aanandha, I. Patra, and S. Prema, “Enhanced Plagiarism Detection Through Advanced Natural Language Processing and E-BERT Framework of the Smith-Waterman Algorithm,” International Journal of Advanced Computer Science and Applications, vol. 14, Oct. 2023, doi: 10.14569/IJACSA.2023.0140944.

[13] M. A. El-Rashidy, R. G. Mohamed, N. A. El-Fishawy, et al., “Reliable plagiarism detection system based on deep learning approaches,” Neural Computing and Applications, vol. 34, pp. 18837–18858, 2022, doi: 10.1007/s00521-022-07486-w.

[14] Y. Shen and Z. Lin, “PatentGrapher: A PLM-GNNs Hybrid Model for Comprehensive Patent Plagiarism Detection Across Full Claim Texts,” IEEE Access, vol. 12, pp. 182717–182725, 2024, doi: 10.1109/ACCESS.2024.3508762.

[15] X. Gao, Z. Zhang, M. Xie, T. Liu, and Y. Fu, “Graph of AI Ideas: Leveraging Knowledge Graphs and LLMs for AI Research Idea Generation,” arXiv preprint arXiv:2503.08549, 2025.