# MFGAF: Multi-faceted Granular Analysis Framework for LLM-Generated Text Detection

Jia Chen[1] and Haizhou Wang[1,*]

***Abstract*— Large Language Models (LLMs) have become increasingly proficient in generating human-like text, yet their widespread deployment raises significant concerns, including disseminating fake information, privacy violations, and academic dishonesty. Detecting LLM-generated text is vital for mitigating these risks and is often framed as a binary classification task. Although zero-shot textual analysis methods have gained popularity due to their generalizability, they face key challenges: (1) limited ability to capture holistic and granular consistency and (2) insufficiently comprehensive textual analysis, particularly regarding syntax and lexical patterns. To address these problems, we propose a novel Multi-faceted Granular Analysis Framework (MFGAF), which leverages *Rewriting Concordance* and *Completive Concordance* to detect LLM-generated text through multi-granular textual dissection. Specifically, MFGAF is designed with two perspectives, Rewriting and Completion, to comprehensively capture global and local LLM-generated features. Additionally, for each perspective, a Multi-Granular Textual Dissection mechanism is constructed to thoroughly analyze LLM-generated text. Finally, MFGAF leverages LLMs to achieve adaptive integration of text analysis results and reflect on the correctness of these results. Our method achieves an average F1 improvement of 3.06% compared to the best baselines, demonstrating its robustness and effectiveness through extensive evaluations.**

## I. INTRODUCTION

Large Language Models (LLMs) have shown remarkable capabilities in text generation tasks, from question answering to code generation. However, their widespread deployment and accessibility introduce significant risks. For instance, LLMs can facilitate cybersecurity threats such as phishing, propaganda, and social engineering, and in educational settings, they may contribute to academic dishonesty [1], [2]. Additionally, LLM-generated code can introduce software vulnerabilities [3], and the generated content may pollute training data for future models [4]. Therefore, detecting and auditing LLM-generated text is critical for mitigating these adverse effects.

As in prior work, the problem of detecting LLM-generated text is commonly framed as a binary classification task: determining whether a text segment is generated by an LLM or authored by a human [5]. Existing detection methods are typically categorized into three types. First, feature-based methods rely on model output logits or losses, but such information is often inaccessible in commercial LLM services, leading to surrogate-based approximations and performance limitations [6], [7]. Second, while supervised fine-tuning trains classifiers on labeled data, it is prone to overfitting and limited generalization [5]. Third, zero-shot textual analysis methods examine linguistic features, such as grammar and style, to detect discrepancies between LLM- and human-written text [8]. These zero-shot methods have become prevalent due to their convenience and generalizability.

Despite the progress of zero-shot methods, several challenges remain:

1) **Limited Capture of Holistic and Granular Consistency:** Some methods detect LLM-generated text via rewriting, assuming such text exhibits consistent patterns before and after rewriting [9]. However, they often overlook cases where only part of the content is modified or refined, resulting in incomplete detection.
2) **Lack of Comprehensive Textual Analysis:** Although LLM-generated text often maintains semantic consistency across rewrites [10], finer-grained aspects beyond semantics—such as syntax and vocabulary preferences—are frequently ignored, limiting the depth of analysis.

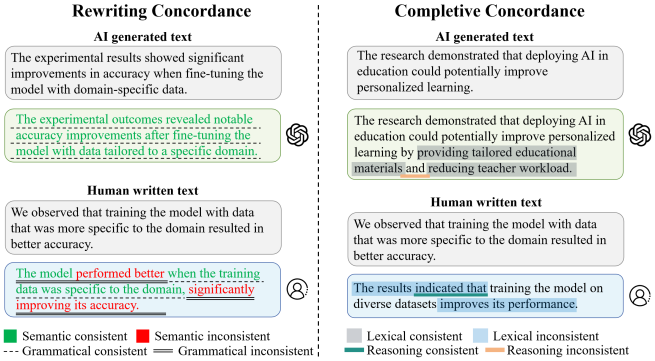


Fig. 1. Consistency Analysis of LLM-generated Text: Rewriting vs. Completive Concordance Perspectives.

To address these challenges, we propose a novel **M**ulti-**F**aceted **G**ranular **A**nalysis **F**ramework (MFGAF)[1] for robust and comprehensive LLM-generated text detection. Our method is grounded in a key observation: although LLMs excel at producing human-like content, their outputs exhibit consistent patterns across semantics, syntax, lexical choice, and reasoning—often distinct from those of human authors. These consistencies persist even when the text is rewritten, paraphrased, or completed. MFGAF introduces two complementary analytical perspectives—*Rewriting Concordance* and *Completive Concordance*—as illustrated in Fig. 1. We

[1]Jia Chen and Haizhou Wang are with School of Cyber Science and Engineering, Sichuan University, Chengdu, China.(*Corresponding author, email: whzh.nc@scu.edu.cn)

[1]https://github.com/yiyepianzhounc/MFGAF

further apply a *multi-granular textual dissection* across linguistic levels, and use weighted concordance scores from both perspectives to inform the final detection decision.

Our contributions can be summarized as follows:

- We propose the Multi-faceted Granular Analysis Framework (MFGAF), which integrates dual perspectives (Rewriting and Completive Concordance) with multi-granular analysis (semantics, syntax, lexical, reasoning) for LLM-generated text detection.
- We design key modules: *Multi-faceted Concordance* analysis and *Multi-granular Textual Dissection*, which effectively capture holistic and fine-grained textual consistencies to enhance detection performance.
- We develop an innovative *LLM-based Resonance Score Calculation* module that adaptively integrates multi-dimensional analysis results and reflects on their correctness, improving robustness and generalizability in zero-shot scenarios.

## II. Related Work

### A. Large Language Models

Large Language Models have revolutionized numerous natural language processing tasks, including machine translation, question answering, and text generation [11]. Recent advancements have enabled LLMs to follow specific instructions through instruction-tuning [12]. LLMs are widely deployed in applications such as chatbots, content creation tools, and automated systems [11], [13]. However, concerns regarding their misuse have emerged, particularly in contexts like academic dishonesty, misinformation generation, and malicious automation [2], [14], [15]. As these models become increasingly accessible, ensuring ethical use and reliable detection mechanisms is critical.

### B. LLM-Generated Text Detection

Nowadays, LLM-generated text detection approaches are divided into active (e.g., watermarking [16]) and passive methods; the former suffers from text degradation and rephrasing vulnerabilities, while the latter analyzes inherent text properties without model modification.

Passive detection employs either: 1) White-box techniques using model internals (log probability [17], entropy [18]), limited by requiring model access; 2) Black-box methods analyzing semantic/stylistic patterns, including n-gram analysis (DNA-GPT [19]), rewriting differences (Raidar [20]), and probability curvature (DetectGPT [8]). Recent advances focus on enhancing black-box robustness through zero-shot adaptations [21].

## III. METHODOLOGY

To overcome the limitations of existing methods that focus on single transformations or coarse-grained features, MFGAF adopts a novel framework built upon the core principles of complementary perspectives (Rewriting Resonance Analysis and Completion Resonance Analysis) and deep multi-granular analysis. This design enables capturing subtle inconsistencies often missed by prior work [9]. MFGAF detects LLM-generated text by analyzing consistent patterns across multiple linguistic dimensions, including semantics, syntax, lexical choices, and reasoning, observed in text transformations. Uniquely, MFGAF leverages the analytical power of LLMs themselves to dynamically integrate evidence from these dimensions, ensuring adaptability across various domains. A schematic overview of the MFGAF framework is presented in Fig. 2.

### A. Problem Definition

MFGAF is designed to tackle the problem of detecting LLM-generated text by capturing and analyzing inherent consistencies across multiple dimensions during text transformation. Specifically, it examines two scenarios: text rewriting and text completion. In both scenarios, LLM-generated text tends to exhibit predictable patterns compared to human-authored text. The objective is to identify and quantify these predictable patterns across four key linguistic dimensions: semantics, syntax, lexical choices, and reasoning, leveraging deviations from human norms for detection.

### B. Dual-view Text Processing

MFGAF operates through two primary, complementary perspectives: **R**ewriting **R**esonance **A**nalysis (RRA) and **C**ompletion **R**esonance **A**nalysis (CRA). Both perspectives employ specialized text processing procedures designed to elicit LLM-specific consistency patterns.

*1) Text Rewriting Procedure:* The RRA perspective probes the LLM's tendency towards stylistic or structural normalization while preserving core meaning. It tests consistency under meaning-preserving paraphrase. We begin by taking the original input text $T_{\text{raw}}$ and using a robust large language model, such as GPT-4, to generate its rewritten counterpart, $T_{\text{modified}}$. Controlled rewriting prompts ensure the process retains original semantic content while encouraging stylistic variations. This step isolates the the linguistic consistencies between the original and modified text.

*2) Text Masking and Completion Procedure:* The CRA perspective challenges the LLM's ability to maintain coherence and contextual appropriateness when generating text from partial input. A portion (e.g., a sentence or major clause) of the original text $T_{\text{raw}}$ is masked, resulting in $T_{\text{mask}}$. The LLM then fills the masked regions, producing $T_{\text{completed}}$. This evaluates consistency in constrained generative settings.

### C. Multi-granular Textual Dissection

Once the rewriting or completion is performed, we proceed with a **Multi-granular Textual Dissection** across four linguistic dimensions to capture detailed consistencies from both perspectives. The analysis involves the following:

*1) Semantic Resonance Analysis (SRA):* To assess semantic consistency, we use Sentence-BERT [22] to generate dense embeddings for both $T_{\text{raw}}$ and $T_{\text{transformed}}$. The semantic resonance score is calculated as the cosine similarity:

$$\text{Score}_{\text{sem}} = \cos(\text{Embed}(T_{\text{raw}}), \text{Embed}(T_{\text{transformed}})), \quad (1)$$

where $\text{Embed}(T)$ denotes the embedding of text $T$.

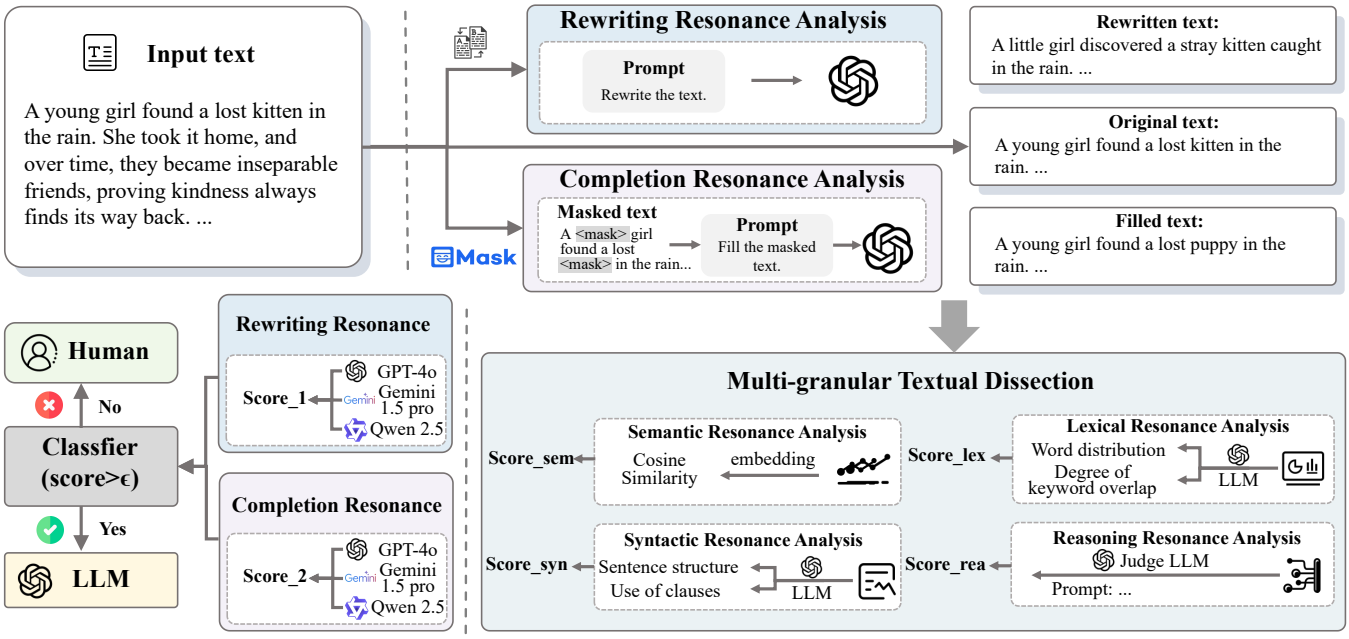


Fig. 2. Schematic diagram of the proposed MFGAF framework

*2) Syntactic Resonance Analysis (SyRA):* This dimension captures structural consistencies. We extract syntactic features using spaCy [23], including dependency parse trees, average clause length, syntactic tree depth, and **P**art-**o**f-**S**peech (POS) tag distributions. The score is computed by $f_{\text{SyntacticAnalysis}}$, which incorporates the normalized tree edit distance and the L2 differences in the feature vectors:

$$\text{Score}_{\text{syn}} = f_{\text{SyntacticAnalysis}}(T_{\text{raw}}, T_{\text{transformed}}). \tag{2}$$

*3) Lexical Resonance Analysis (LRA):* This assesses vocabulary overlap. After filtering stop words, we extract content words and compute the Jaccard index:

$$\text{Score}_{\text{lex}} = \frac{|V(T_{\text{raw}}) \cap V(T_{\text{transformed}})|}{|V(T_{\text{raw}}) \cup V(T_{\text{transformed}})|}, \tag{3}$$

where $V(T)$ is the set of content words in text $T$.

*4) Reasoning Resonance Analysis (ReRA):* To evaluate logical alignment, we use Bart-large-mnli [24] to predict the probability of entailment from $T_{\text{raw}}$ to $T_{\text{transformed}}$. This forms the reasoning score:

$$\text{Score}_{\text{rea}} = f_{\text{ReasoningAnalysis}}(T_{\text{raw}}, T_{\text{transformed}}) := p_{\text{entail}}. \tag{4}$$

### D. LLM-based Resonance Score Calculation

MFGAF uses LLMs to integrate the four resonance scores per perspective. Prior to evaluation, all scores are min-max normalized to [0, 1].

*1) LLM-Based Consistency Evaluation:* For each perspective, the LLM receives $T_{\text{raw}}$, $T_{\text{transformed}}$, and the normalized scores. It outputs a score (0-100) and rationale via a structured prompt.

*2) Multi-LLM Assessment for Enhanced Robustness:* Three different LLMs conduct evaluations independently. Their average is used as the final score.

*3) Reflection Process for Inconsistent Evaluations:* If $\text{Var}(\text{Scores}) < \tau_{\text{variance}}$, compute:

$$\text{Score}_{\text{final}} = \frac{1}{3}\sum_{i=1}^{3} \text{Score}_i. \tag{5}$$

Otherwise, apply reweighting:

$$\text{Score}_{\text{final}} = \sum_{i=1}^{3} w_i \cdot \text{Score}_i, \quad \sum w_i = 1. \tag{6}$$

The weights $w_i$ may reflect the historical accuracy or the confidence of the model.

### E. Final Decision Mechanism

Once final consistency scores from RRA and CRA are obtained, the overall detection score is computed:

$$\text{Score}_{\text{final}} = \alpha \cdot \text{Score}_{\text{rewrite}} + (1-\alpha) \cdot \text{Score}_{\text{completion}}. \tag{7}$$

If $\text{Score}_{\text{final}} \geq \tau$, the text is classified as LLM-generated; otherwise, as human-authored. Parameters $\alpha$, $\tau$, and $\tau_{\text{variance}}$ are optimized using validation data.

This fusion strategy combines the diversity of linguistic signals with LLM-guided integration, ensuring a robust and generalizable detection process.

## IV. Experiments

In this section, we evaluate the performance of the proposed **MFGAF** for detecting LLM-generated text. The experiments aim to address the following research questions:

- **RQ1:** How does the proposed method compare with state-of-the-art techniques in detecting LLM-generated text?

- **RQ2:** To what extent do the individual components of the proposed framework contribute to its overall performance?
- **RQ3:** What is the robustness of the proposed method when faced with common text manipulations like rephrasing and length variation?
- **RQ4:** Can the method maintain good performance levels even in a few-shot setting with limited labeled data?
- **RQ5:** How does the choice of large language models (LLMs) affect detection performance? Specifically, does model scale (e.g., GPT-3.5 vs. smaller models) significantly influence detection accuracy?
- **RQ6:** How does MFGAF balance detection accuracy and inference latency, particularly when employing lightweight LLMs?

### A. Experimental Setup

*1) Datasets:* We conduct experiments on datasets provided in *Raidar: GeneRative AI Detection viA Rewriting* [20], covering a wide range of text types, including creative writing, student essays, code, reviews, and academic abstracts. These datasets include human-written and LLM-generated texts, where the LLM-generated texts are created using GPT models (GPT-3.5-turbo and GPT-4). Notably, this dataset has also been used in other studies, further validating its robustness and applicability for AI detection tasks [25].

*2) Benchmark Methods and Evaluation:* We evaluate MFGAF against several state-of-the-art detection approaches, categorized as follows:

- **Probability-Based Methods**: These rely on internal model probabilities or scores, including DetectGPT [8], Fast-DetectGPT [21], and Ghostbuster [26].
- **Text Pattern Analysis Methods**: These analyze linguistic features and textual patterns without requiring direct model probabilities. This category includes methods like GPTZero [27], DNA-GPT [19] (which analyzes divergent n-grams), SeqXGPT [17] (focusing on sentence-level features), and Raidar [20] (which uses rewriting).

Evaluation is based on F1-score across six text domains (News, Creative Writing, Student Essay, Code, Reviews, Abstracts), sourced from the dataset used in [20]. Additional robustness assessments are performed under text manipulations like rephrasing and truncation.

*3) Implementation Details:* The key hyperparameters of MFGAF include the word mask ratio for perturbation, the threshold $\tau_{\text{variance}}$ in the Reflection Process for Inconsistent Evaluations, and the parameter $\alpha$ in the *Final decision mechanism* module. Our experimental setup utilizes `gpt-3.5-turbo` as the base LLM for all methods, with both temperature and `top_p` set to 0.9. The mask ratio is set to 0.4, aiming to mask short sentences as a whole. Furthermore, $\tau_{\text{variance}}$ and $\alpha$ are tuned from the set $\{0.1, 0.2, 0.4, 0.5, 0.6, 0.8\}$, and their optimal values of 0.5 and 0.6 are selected.

TABLE I
F1-SCORE COMPARISON ACROSS DATASETS

| Methods | News | Creative | Essays | Code | Reviews | Abstracts |
|---|---|---|---|---|---|---|
| *Probability-Based Methods* | | | | | | |
| DetectGPT [8] | 58.25 | 60.74 | 46.03 | 67.89 | 70.16 | 68.07 |
| Fast-DetectGPT [21] | 59.32 | 62.19 | 60.56 | 75.34 | 74.23 | 75.97 |
| Ghostbuster [26] | 53.01 | 41.93 | 43.54 | 67.07 | 72.67 | 77.72 |
| *Text Pattern Analysis Methods* | | | | | | |
| GPTZero [27] | 55.89 | 50.73 | 53.09 | 63.13 | 66.95 | 66.04 |
| DNA-GPT [19] | 57.32 | 63.13 | 59.76 | 87.32 | 74.67 | 69.87 |
| SeqXGPT [17] | 56.98 | 59.65 | 66.43 | 79.28 | 78.64 | 72.67 |
| Raidar [20] | 61.49 | 63.76 | 65.91 | 92.88 | 85.05 | 80.97 |
| **Ours (MFGAF)** | **66.47** | **65.21** | **68.42** | **95.01** | **89.79** | **83.53** |

### B. Overall Performance (RQ1)

To answer RQ1, we compare MFGAF's performance against the selected baseline methods across all six domains. Table I summarizes the F1 scores.

The results clearly demonstrate MFGAF's superior performance. It achieves the highest F1-score across all evaluated domains, surpassing even the strongest baseline (Raidar) by a significant margin. This consistent top-tier performance underscores the effectiveness of MFGAF's multi-faceted granular analysis approach compared to existing zero-shot techniques.

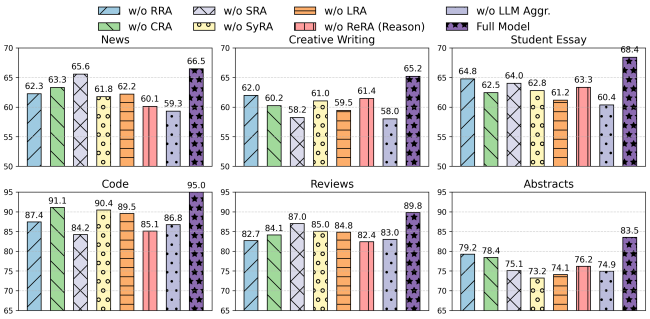


Fig. 3. Ablation Study Results

### C. Component Analysis (RQ2)

To address RQ2, we perform an ablation study by systematically removing or altering key components of MFGAF:

- **View Removal**: Evaluating performance using only RRA or only CRA.
- **Dimension Exclusion**: Evaluating performance after removing each of the four granular analysis dimensions (Semantic, Syntactic, Lexical, Reasoning) individually.
- **Score Aggregation**: Replacing the LLM-based aggregation with a simple weighted sum (weights tuned on validation data).

The results of these ablation experiments are illustrated in Fig. 3, clearly demonstrating the performance impact of each component. Removing any component degrades performance, indicating their collective contribution. The most significant performance drops occur when the LLM-based

aggregation mechanism is removed (an average decrease in F1-score of 7. 7%) and the Reasoning Resonance Analysis (an average decrease in F1-score of 6. 6%). These findings validate the necessity of analyzing multiple dimensions and highlight the benefit of the adaptive LLM-based score integration strategy.

### *D. Robustness against Text Manipulations (RQ3)*

To evaluate the robustness of MFGAF, we test its performance under three types of text manipulations: rephrasing, truncation, and expansion. These operations simulate real-world variations and assess the generalizability of detection methods. Rephrasing changes wording while keeping meaning intact, potentially affecting semantic consistency [28]. Truncation shortens the text, which may compromise coherence and completeness [19], while expansion adds content that can introduce redundancy or unnecessary elaboration [29].

As shown in Table II, the MFGAF framework performs well even under significant text modifications, maintaining high detection accuracy across different datasets. These results suggest that the proposed framework is resilient to common text manipulations, making it a reliable tool for detecting LLM-generated content in diverse scenarios.

TABLE II
PERFORMANCE UNDER TEXT MANIPULATION

| Manipulation Type | News | Creative | Essays | Code | Reviews | Abstracts |
|---|---|---|---|---|---|---|
| Rephrasing | 58.99 | 60.44 | 61.22 | 82.39 | 76.22 | 79.45 |
| Truncation | 59.02 | 60.20 | 60.94 | 78.11 | 74.83 | 77.01 |
| Expansion | 59.56 | 61.01 | 62.04 | 85.32 | 80.14 | 81.60 |

### *E. Few-shot Performance (RQ4)*

We also evaluate the framework in a few-shot learning scenario, where we train the model with limited labeled data (ranging from 10 to 100 samples per text type). The results in Fig. 4 demonstrate that the proposed method remains effective even with small amounts of labeled data, making it suitable for real-world applications where labeled examples are scarce.

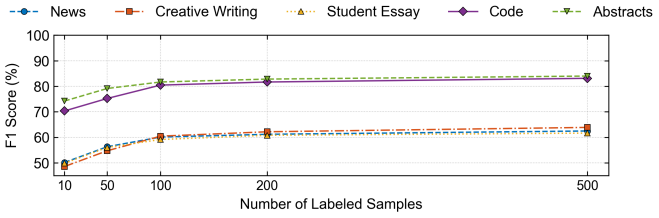


Fig. 4. Few-shot Detection Performance

### *F. Impact of Detection Model (RQ5)*

In our experiments, we explore the impact of different large language models (LLMs) on detection performance. Specifically, we assess the performance of various LLMs, such as GPT-3.5, Ada, Text-Davinci-002, GPT-4, and Llama-7B, under the same experimental setting. Our findings indicate that larger models, such as GPT-3.5 and GPT-4, consistently outperform smaller models, yielding better detection accuracy across different datasets. This suggests that the model size plays a significant role in the effectiveness of LLM-generated text detection. Table III shows the detection performance comparison across different models.

TABLE III
IMPACT OF DIFFERENT DETECTION MODELS ON PERFORMANCE

| Methods | News | Creative | Essays | Code | Reviews | Abstracts |
|---|---|---|---|---|---|---|
| GPT-3.5 | 66.47 | 65.21 | 68.42 | 95.01 | 89.79 | 83.53 |
| Ada | 60.32 | 59.14 | 61.28 | 84.12 | 80.56 | 75.20 |
| Text-Davinci-002 | 62.45 | 60.28 | 63.72 | 87.80 | 82.90 | 78.11 |
| GPT-4 | **69.15** | **68.23** | **72.34** | **96.72** | **91.80** | **85.63** |
| Llama-7B | 61.22 | 59.98 | 62.17 | 81.34 | 79.12 | 74.55 |

### *G. Accuracy vs. Latency Trade-off Analysis (RQ6)*

To address RQ6, we evaluate the trade-off between detection performance (average F1-score) and inference latency for MFGAF using different LLM configurations, including large cloud-based models and smaller local alternatives. The results, along with the Raidar baseline, are summarized in Table IV.

TABLE IV
ACCURACY (F1 SCORE) VS. INFERENCE LATENCY (TESTED ON NVIDIA A100, 80GB RAM, INTEL XEON GOLD 6338 CPU)

| Model Configuration | F1 Score (Avg %) | Latency (s) |
|---|---|---|
| MFGAF (using GPT-4) | 80.65 | 4.48 |
| MFGAF (using GPT-3.5) | 78.07 | 2.75 |
| MFGAF (using Local 70B) | 79.20 | 3.01 |
| MFGAF (using Local 7B) | 74.81 | 2.21 |
| Raidar (Baseline) [20] | 75.01 | 4.25 |

As shown in Table IV, there is a clear trade-off between accuracy and latency. GPT-4 achieves the highest accuracy (80.65%) but incurs the longest latency (4.48 seconds). GPT-3.5 provides a more balanced option with an F1 score of 78.07% and lower latency (2.75 seconds), outperforming the Raidar baseline (4.25 seconds) in both aspects.

For scenarios requiring faster inference or offline deployment, lightweight local models offer practical alternatives. The 7B local LLM delivers the lowest latency (2.21 seconds), albeit with a slightly reduced F1 score (74.81%). This makes it suitable for real-time or edge deployments where speed is prioritized over accuracy. The 70B local model improves accuracy (79.20%) at the cost of increased latency (3.01 seconds) and higher hardware resource demands.

In summary, MFGAF offers flexibility in balancing accuracy and latency. GPT-3.5 serves as a strong default for general-purpose use, while GPT-4 delivers maximum accuracy when latency is less critical. For latency-sensitive

applications, the 7B local model enables the fastest inference with acceptable performance trade-offs.

## V. Conclusion

In conclusion, we proposed MFGAF to address the challenges of detecting LLM-generated text. By incorporating the *Multi-faceted Concordance* and *Multi-granular Textual Dissection* modules, MFGAF enables a more comprehensive understanding of both holistic and granular textual consistencies. This design supports robust detection across diverse LLM-generated content. Furthermore, the *LLM-based Resonance Score Calculation* module adaptively integrates analysis results to further enhance detection accuracy.

Experiments demonstrate that MFGAF outperforms existing approaches, establishing a new benchmark in zero-shot detection scenarios. Nonetheless, its higher computational cost may limit efficiency in large-scale deployments.

Future work may explore the modular and extensible design of MFGAF for cross-modal detection, including LLM-generated images, audio, and video, which is becoming increasingly important as generative media continues to evolve.

## ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China under grant No. 2022YFC3303101.

## References

[1] E. Ferrara, "The rise of generative ai and implications for cybersecurity," *arXiv preprint arXiv:2303.04248*, 2023.

[2] R. Perrin and S. Downes, "Academic integrity in the digital age: A survey of undergraduate engineering students," *Canadian Journal of Higher Education*, vol. 45, no. 2, pp. 265–284, 2015.

[3] S. Wang, M. Geng, B. Lin, Z. Sun, M. Wen, Y. Liu, L. Li, T. F. Bissyandé, and X. Mao, "Natural language to code: How far are we?," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 375–387, ACM, 2023.

[4] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. Anderson, "The curse of recursion: Training on generated data makes models forget," *arXiv preprint arxiv:2305.17493*, 2023.

[5] X. He, X. Shen, Z. Chen, M. Backes, and Y. Zhang, "Mgtbench: Benchmarking machine-generated text detection," in *Proceedings of the 31st ACM SIGSAC Conference on Computer and Communications Security*, pp. 2251–2265, ACM, 2024.

[6] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck, "Automatic detection of generated text is easiest when humans are fooled," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1808–1822, ACL, 2020.

[7] C. Guo, T. Wang, Y. Qin, C. Zhou, S. Han, and X. a. S. Li, "The closer, the better: Rethinking model-based text detectors," *arXiv preprint arXiv:2305.03163*, 2023.

[8] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn, "Detectgpt: zero-shot machine-generated text detection using probability curvature," in *Proceedings of the 40th International Conference on Machine Learning*, JMLR.org, 2023.

[9] Y. Huang, J. Cao, H. Luo, X. Guan, and B. Liu, "MAGRET: Machine-generated text detection with rewritten texts," in *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 8336–8346, ACL, 2025.

[10] D. Zhou and S. Sterman, "Ai.llude: Investigating rewriting ai-generated text to support creative expression," in *Proceedings of the 16th Conference on Creativity & Cognition*, pp. 241–254, ACM, 2024.

[11] Y. Annepaka and P. Pakray, "Large language models: a survey of their development, capabilities, and applications," *Knowl.Inf.Syst.*, vol. 67, no. 3, pp. 2967–3022, 2025.

[12] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, *et al.*, "Instruction tuning for large language models: A survey," *arXiv preprint arXiv:2308.10792*, 2023.

[13] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie, "A survey on evaluation of large language models," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 3, pp. 1–45, 2024.

[14] K.-J. Tokayev, "Ethical implications of large language models a multidimensional exploration of societal, economic, and technical concerns," *International Journal of Social Analytics*, vol. 8, no. 9, pp. 17–33, 2023.

[15] X. Wang, J. Peng, K. Xu, H. Yao, and T. Chen, "Reinforcement learning-driven LLM agent for automated attacks on LLMs," in *Proceedings of the 5th Workshop on Privacy in Natural Language Processing*, pp. 170–177, ACL, 2024.

[16] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," in *Proceedings of the 40th International Conference on Machine Learning*, vol. 202, pp. 17061–17084, PMLR, 2023.

[17] P. Wang, L. Li, K. Ren, B. Jiang, D. Zhang, and X. Qiu, "SeqXGPT: Sentence-level AI-generated text detection," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1144–1156, ACL, 2023.

[18] T. Lavergne, T. Urvoy, and F. Yvon, "Detecting fake content with relative entropy scoring," in *Proceedings of the 2008 International Conference on Uncovering Plagiarism, Authorship and Social Software Misuse*, vol. 377, pp. 27–31, CEUR-WS.org, 2008.

[19] X. Yang, W. Cheng, Y. Wu, L. Petzold, W. Y. Wang, and H. Chen, "Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text," *arXiv preprint arXiv:2305.17359*, 2023.

[20] C. Mao, C. Vondrick, H. Wang, and J. Yang, "Raidar: generative AI detection via rewriting," in *Proceedings of the 12th International Conference on Learning Representations*, vol. 7, pp. 1–18, http://OpenReview.net, 2024.

[21] G. Bao, Y. Zhao, Z. Teng, L. Yang, and Y. Zhang, "Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature," *arXiv preprint arXiv:2310.05130*, 2023.

[22] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-pack: Packed resources for general chinese embeddings," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 641–649, ACM, 2024.

[23] Y. Vasiliev, *Natural Language Processing with Python and spaCy: A Practical Introduction*, pp. 45–52. San Francisco, CA, USA: No Starch Press, 1st ed., 2020.

[24] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, ACL, 2020.

[25] H.-Q. Nguyen-Son, M.-S. Dao, and K. Zettsu, "SimLLM: Detecting sentences generated by large language models using similarity between the generation and its re-generation," in *Proceedings of the 21th Conference on Empirical Methods in Natural Language Processing*, pp. 22340–22352, ACL, 2024.

[26] V. Verma, E. Fleisig, N. Tomlin, and D. Klein, "Ghostbuster: Detecting text ghostwritten by large language models," in *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 1702–1717, ACL, 2024.

[27] E. Tian, 2023. Available at: `https://gptzero.me`.

[28] B. Zhu, L. Yuan, G. Cui, Y. Chen, C. Fu, B. He, Y. Deng, Z. Liu, M. Sun, and M. Gu, "Beat LLMs at their own game: Zero-shot LLM-generated text detection via querying ChatGPT," in *Proceedings of the 20th Conference on Empirical Methods in Natural Language Processing*, pp. 7470–7483, ACL, 2023.

[29] H. Dang, K. Benharrak, F. Lehmann, and D. Buschek, "Beyond text generation: Supporting writers with continuous automatic text summaries," in *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–13, ACM, 2022.