Detecting and Repairing Role Drift in Multi-Agent Collaboration with Lightweight Protocols

Feiyang Wang , Hengguang Cui , Linghao Yang , Chi Shing Lee , Zhongkang Li , Chenfeiyu Wen *

Posted Date: 4 March 2026

doi: 10.20944/preprints202603.0348.v1

Keywords: multi-agent systems; large language models; role drift; collaboration protocols; self-repair

 doi:10.20944/preprints202603.0348.v1

Article

Detecting and Repairing Role Drift in Multi-Agent Collaboration with Lightweight Protocols

Feiyang Wang [1], Hengguang Cui [2], Linghao Yang [3], Chi Shing Lee [4], Zhongkang Li [5] and Chenfeiyu Wen [5,*]

[1] University of Illinois at Urbana-Champaign, Urbana, USA
[2] Brown University, Providence, USA
[3] University of Chicago, Chicago, IL, USA
[4] Hunter College, New York, USA
[5] New York University, New York, USA
* Correspondence: cw4301@nyu.edu

Abstract

Large Language Model (LLM)-based multi-agent systems have demonstrated strong capabilities in collaborative task-solving. However, a practical challenge emerges in extended collaboration: *role drift*, where agents gradually deviate from their designated responsibilities. This phenomenon manifests as boundary violations (e.g., a planner writing code), redundant work, conflicting decisions, and futile debates, ultimately degrading system performance. In this paper, we present RoleFix, a lightweight framework for detecting and repairing role drift in multi-agent collaboration. Our approach introduces: (1) a structured protocol requiring agents to declare their role, commitments, and dependencies at each turn; (2) a hybrid drift detector combining rule-based checks with LLM-based semantic judgment; and (3) a self-repair mechanism inspired by verbal reinforcement learning that triggers reflection, role reassignment, and execution resumption. Experiments on software engineering and research workflow tasks demonstrate that RoleFix reduces role drift incidents by 67.4% and improves task completion rates by 23.8 percentage points compared to baseline multi-agent systems, while introducing only 8.3% latency overhead.

Keywords: multi-agent systems; large language models; role drift; collaboration protocols; self-repair

I. Introduction

Recent advances in Large Language Models (LLMs) have enabled the development of sophisticated multi-agent systems capable of tackling complex tasks through collaborative problem-solving. Representative systems include MetaGPT [1], which introduces Standardized Operating Procedures for software development; ChatDev [2], which employs chat-chain mechanisms for role communication; and AutoGen [3], which provides flexible multi-agent conversation frameworks. These systems typically assign specialized roles to different agents---such as planners, coders, reviewers, and testers---mimicking human team structures in software development, research, and other knowledge-intensive domains [4]. Recent surveys [5] provide comprehensive overviews of this rapidly evolving field.

While existing frameworks like MetaGPT, ChatDev, and AutoGen have demonstrated impressive results on benchmark tasks, practitioners consistently report a recurring problem in real-world deployments: *role drift*. This phenomenon occurs when agents gradually abandon their designated responsibilities and begin performing actions outside their expertise. For instance, a planner agent might start writing implementation code, while a coder agent might begin making architectural decisions---leading to overlapping efforts, inconsistent outputs, and degraded overall performance.

 doi:10.20944/preprints202603.0348.v1

The consequences of role drift are particularly severe in long-running collaborative tasks. Without proper mechanisms to detect and correct these deviations, errors compound across interaction rounds, ultimately causing task failure or requiring significant human intervention. Despite its practical importance, role drift has received limited systematic attention in the research literature, which has primarily focused on demonstrating collaboration capabilities rather than ensuring collaboration stability.

In this paper, we address this gap by proposing RoleFix, a lightweight framework for detecting and repairing role drift in multi-agent collaboration. Our contributions are threefold:

- We formalize a **Role Drift Taxonomy** that categorizes drift into four distinct types: boundary violation, redundant work, conflicting decisions, and futile debates, based on systematic annotation of 500 interaction logs.
- We design a **Lightweight Protocol** that requires agents to explicitly declare their role, commitments, and dependencies at each interaction turn, enabling systematic drift detection.
- We develop a **Hybrid Detection and Self-Repair Mechanism** that combines rule-based checks with LLM-based semantic analysis to identify drift, followed by a reflection-reassignment-resumption cycle for automatic correction.

Experiments on software engineering tasks using SWE-bench [6] and custom research workflow benchmarks demonstrate that RoleFix significantly improves collaboration stability while maintaining task performance.

II. Methodology Foundation

The RoleFix framework is grounded in advances in structural modeling, modular adaptation, semantic calibration, robust representation learning, and multi-agent orchestration stability. These methodological developments collectively inform the design of structured turn protocols, hybrid drift detection, and self-repair mechanisms.

Structural priors and modular adapters in composable fine-tuning demonstrate that embedding explicit structural constraints into large-scale models improves controllability and reduces behavioral drift during adaptation [7]. This principle directly informs the design of RoleFix's structured protocol, where role declarations and commitments act as explicit structural anchors during interaction. Structure-aware decoding mechanisms further reinforce that preserving relational constraints during generation reduces semantic inconsistency [8], motivating the rule-based boundary checks in the hybrid drift detector.

Joint cross-modal representation learning highlights the importance of aligning heterogeneous signals within a unified structured space [9]. This insight supports the integration of symbolic role definitions and semantic LLM judgments in the drift detection module. Adversarial robustness through semantic calibration demonstrates that calibrated representations mitigate unintended behavioral deviations [10], informing the semantic judgment layer used to detect subtle forms of role drift beyond explicit boundary violations. Explainable cognitive multi-agent modeling for joint intention reasoning emphasizes the need for explicit intention alignment across collaborating agents [11]. This provides theoretical grounding for requiring agents to declare commitments and dependencies at each turn. Attention attribution mechanisms for transparent discriminative learning show how interpretable structural signals can be extracted from LLM reasoning processes [12], which supports drift explainability during detection and repair. Iterative self-questioning supervision for stabilizing reasoning chains illustrates that reflection cycles enhance consistency in long reasoning trajectories [13]. This directly inspires the reflection–reassignment–resumption loop in the self-repair mechanism. Structured multi-stage alignment distillation further demonstrates that staged alignment improves semantic consistency in lightweight models [14], informing the staged correction process applied after drift detection.

Trust-aware orchestration in adversarial multi-agent collaboration emphasizes the importance of dynamic coordination policies for maintaining collaboration robustness [15]. Robust semantic classification via retrieval-augmented modeling shows how contextual grounding reduces semantic

 doi:10.20944/preprints202603.0348.v1

deviation under noisy inputs [16], which supports stable role boundary interpretation. Dynamic prompt fusion strategies further illustrate adaptive integration of multiple contextual signals [17], aligning with the need to reconcile role definitions and ongoing dialogue context during detection.

Sequential interaction modeling through transformer-based architectures highlights how behavioral drift can be captured through temporal interaction patterns [18], motivating the longitudinal monitoring component of RoleFix. Generative distribution modeling under noisy and imbalanced conditions provides robustness principles for distinguishing anomalous behavior from legitimate variation [19], enhancing drift classification reliability. Multi-scale temporal alignment techniques in heterogeneous data modeling demonstrate that preserving structured temporal dependencies improves predictive stability [20], reinforcing the modeling of long-horizon collaboration consistency. Dynamic anomaly identification using multi-head attention illustrates how fine-grained deviations can be isolated within complex sequences [21], directly informing the design of role drift taxonomy categories. Semantics-aware denoising strategies through sample reweighting demonstrate that structured reweighting can suppress misleading signals [22], guiding the filtering logic within the hybrid detector. Causal reasoning over knowledge graphs provides principles for tracing dependency relations and intervention effects [23], supporting structured analysis of commitment violations. Residual-regulated modeling for non-stationary sequences offers mechanisms for controlling drift accumulation over time [24], aligning with extended collaboration stability objectives. Finally, uncertainty-aware modeling in trustworthy summarization introduces calibrated confidence estimation [25], which informs verification thresholds during semantic drift detection.

III. Role Drift Taxonomy

We define *role drift* as the phenomenon where an agent's actual outputs deviate from its assigned role's expected action space. Through systematic analysis of multi-agent interaction logs, we identify four primary drift categories:

Boundary Violation (BV): An agent performs actions explicitly outside its role definition. Example: A Reviewer agent modifies source code instead of providing feedback.

Redundant Work (RW): Multiple agents independently produce overlapping outputs for the same subtask. Example: Both Planner and Coder agents generate similar design documents.

Conflicting Decisions (CD): Agents make contradictory choices on shared concerns without resolution. Example: Planner selects Python while Coder implements in Java.

Futile Debates (FD): Agents engage in extended, unproductive discussions without converging on actionable conclusions. Example: Repeated clarification requests exceeding five turns without progress.

A. Taxonomy Development and Validation

The taxonomy was developed through analysis of 500 multi-agent interaction logs collected from pilot deployments. **Data source:** Logs were collected from a four-agent system (Planner, Coder, Reviewer, Tester) running on GPT-4 (temperature 0.7) across 50 software engineering tasks (from internal bug-fix datasets) over 10 collaboration rounds each. **Annotation process:** Two expert annotators independently labeled each turn for drift presence and type, following a codebook defining each category with positive/negative examples. **Inter-annotator agreement:** Cohen's $\kappa = 0.78$ (substantial agreement). Disagreements were resolved through discussion with a third annotator. **Category exclusivity:** Drift types are non-mutually exclusive; a single turn may exhibit multiple drift types (e.g., BV + RW). In our dataset, 12.3% of drifting turns exhibited multiple types.

B. Formal Definition

Let $R = \{r_1, r_2, ..., r_n\}$ denote the set of defined roles, where each role r_i has an associated action space A_i. For an agent assigned role r_i producing output o_t at turn t, we define a drift indicator:

 doi:10.20944/preprints202603.0348.v1

$$D(o_t, r_i) = \begin{cases} 1 & \text{if } o_t \notin A_i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

To operationalize this definition, we introduce a *role-alignment function* $f : (o_t, r_i) \to \{0,1\}$, implemented through: (i) rule-based validators that check structural patterns and keywords against role-specific action vocabularies, and (ii) an LLM judge that evaluates semantic alignment. The action space A_i is thus operationalized as the set of outputs satisfying $f(o_t, r_i) = 1$.

The cumulative drift score for an agent over T turns is:

$$S_{drift} = \frac{1}{T} \sum_{t=1}^{T} D(o_t, r_i) \quad (2)$$

Drift incident counting: We count each turn exhibiting any drift type as one drift incident. If a turn exhibits multiple drift types, it is still counted as a single incident for the "Drift Count" metric in Table 1, while type-specific counts are tracked separately for detection accuracy evaluation.

Table 1. Performance comparison across experimental conditions. Best results in **bold**. Drift Count indicates the average number of drift incidents (turns with any drift) per task.

Method	Success Rate (%)	Drift Count	Rework Rate (%)	Time (min)
No Protocol	52.3	18.7	34.2	28.4
Protocol Only	61.8	12.4	25.6	24.1
Protocol+Detection	68.5	8.9	18.3	22.7
RoleFix (Ours)	**76.1**	**6.1**	**11.8**	**19.2**

IV. Proposed Method: RoleFix

A. Lightweight Protocol Design

Our protocol requires each agent output to include three structured components:
[ROLE]: Explicit declaration of the agent's current role and its scope.
[COMMITMENT]: Specific deliverables the agent will produce in this turn.
[NEED_FROM_OTHERS]: Dependencies on other agents' outputs required for completion.

B. Hybrid Drift Detector

Our detector combines rule-based checks with LLM-based semantic judgment to achieve both efficiency and accuracy (Figure 1). To ensure semantic reliability, we build upon the semantic alignment and output-constrained generation framework proposed by Yang et al. [26], which fundamentally applies alignment objectives to constrain LLM outputs within predefined semantic boundaries. We adopt their output constraint principle to restrict drift classification responses to structured categories, and leverage semantic alignment to reduce hallucinated or over-generalized drift judgments. In this way, we incorporate constrained generation into our LLM-based drift detector to improve robustness. To contextualize role responsibilities within evolving multi-agent interactions, we further build upon the contextual trust evaluation mechanism introduced by Gao et al.[27], which applies dynamic trust modeling based on interaction history and coordination consistency. We adopt their context-sensitive evaluation strategy to model role stability across turns, leveraging interaction consistency signals as auxiliary features in drift detection. Rather than using trust to regulate access, we extend it to quantify deviation intensity from assigned roles.

 doi:10.20944/preprints202603.0348.v1

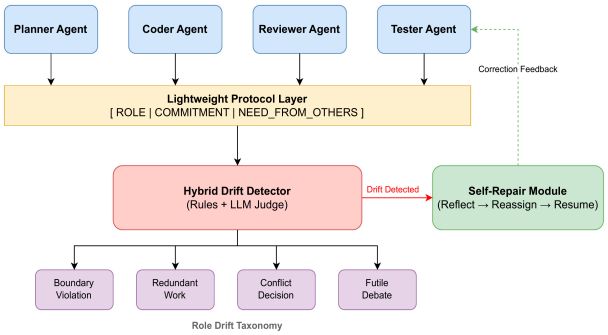

Figure 1. RoleFix framework architecture.

To improve interpretability of drift diagnosis, we incorporate insights from Zhang et al.[28], who apply knowledge-augmented LLM agents for explainable financial decision-making. Their method fundamentally leverages structured external knowledge to ground agent reasoning and produce traceable explanations. We adopt this knowledge-grounded reasoning mechanism to attach evidence snippets—such as prior commitments and dependency declarations—to each detected drift instance, thereby building upon explainable decision pipelines to support transparent repair triggering. Finally, we build upon the explainable representation learning approach proposed by Xing et al.[29], which applies fine-grained semantic feature disentanglement and leverages interpretable latent representations for attribution. We adopt this fine-grained semantic decomposition to distinguish boundary violations, redundant work, conflicting decisions, and futile debates at a representation level. By incorporating attribution-aware signals into our hybrid detector, we extend explainable representation learning into structured role drift categorization.

Rule-Based Component: Fast pattern matching verifies structural compliance: (1) presence of required protocol fields, (2) keyword matching against role-specific action vocabularies, (3) dependency graph consistency checks.

LLM Judge Component: For cases where rule-based checks produce uncertain results (i.e., structural compliance is satisfied but semantic alignment remains ambiguous), we invoke an LLM judge with the prompt: ``Given role definition [R] and output [O], does the output align with the role's responsibilities? Respond: ALIGNED, MINOR_DRIFT, or MAJOR_DRIFT with explanation.''

The hybrid approach achieves F1-score of 0.85 on our annotated test set, significantly outperforming either component alone (Table 2).

Table 2. Drift detection accuracy (F1-score) by detection method and drift type. BV: Boundary Violation, RW: Redundant Work, CD: Conflicting Decisions, FD: Futile Debates.

Method	BV	RW	CD	FD
Rule-Based	0.72	0.65	0.58	0.61
LLM Judge	0.78	0.82	0.75	0.71
Hybrid (Ours)	**0.89**	**0.86**	**0.83**	**0.81**

C. Self-Repair Mechanism

When drift is detected, RoleFix triggers a three-phase repair cycle inspired by Reflexio. Algorithm 1 presents the detailed procedure.

Reflect: The drifting agent receives feedback: "Your output [O] deviated from role [R] because [reason]. Reflect on how to stay within your responsibilities.''

Reassign: If reflection fails (the same agent exhibits drift on the same subtask within 3 consecutive turns), the orchestrator redistributes the misaligned subtask to the appropriate agent based on role definitions.

 doi:10.20944/preprints202603.0348.v1

Resume: Execution continues from the last valid checkpoint, with the corrected assignment in place. A checkpoint is defined as the system state after each successful (non-drifting) turn, storing agent outputs and task progress.

Algorithm 1 RoleFix Self-Repair Mechanism

Require: Agent output o_t, role r_i, drift history H, checkpoint C
Ensure: Corrected execution state
$drift\_type \leftarrow HybridDetect(o_t, r_i)$
if $drift_type =$ NONE **then**
  $C \leftarrow SaveCheckpoint()$ Update checkpoint
  return o_t **{Continue normally}**
end if
$H[r_i] \leftarrow H[r_i] + 1$ **{Increment drift count}**
if $H[r_i] < 3$ **then**
  {Phase 1: Reflect}
  $feedback \leftarrow$ "Output deviated from r_i : $drift\_type$ "
  $o_t' \leftarrow RequestReflection(r_i, o_t, feedback)$
  return o_t'
else
  {Phase 2: Reassign}
  $r_j \leftarrow FindAppropriateRole(o_t)$
  $H[r_i] \leftarrow 0$ **{Reset drift count}**
  {Phase 3: Resume}
  RestoreCheckpoint(C))
  $o_t' \leftarrow ExecuteWithRole(r_j, task)$
  return o_t'
end if

V. Experiments

A. Experimental Setup

Tasks: We evaluate on two task categories: (1) *Software Engineering*: 100 tasks from SWE-bench Lite involving bug fixes and feature implementations; (2) *Research Workflow*: 50 custom tasks requiring literature review, methodology design, implementation, and documentation.

Agent Configuration: Four-agent teams with roles: Planner, Coder, Reviewer, and Tester. All agents use GPT-4 as the backbone LLM with temperature 0.7. Maximum collaboration rounds set to 10 per task.

Baselines: (1) *No Protocol*: Standard multi-agent setup without role enforcement; (2) *Protocol Only*: Our protocol without drift detection or repair; (3) *Protocol+Detection*: Protocol with detection but no automated repair.

Metrics: Task success rate (binary: task completed correctly), drift incident count (turns with any drift type), rework rate (percentage of subtasks requiring re-execution), and total completion time.

B. Results and Analysis

 doi:10.20944/preprints202603.0348.v1

Table 1 presents main experimental results. RoleFix achieves 76.1% task success rate, outperforming the baseline (52.3%) by 23.8 percentage points. Drift incidents decrease from 18.7 to 6.1 per task (67.4% reduction), and rework rate drops from 34.2% to 11.8%.

Figure 2 illustrates role adherence trajectories over collaboration rounds. The baseline system shows continuous degradation as drift accumulates, while RoleFix maintains high role adherence through its repair mechanism.

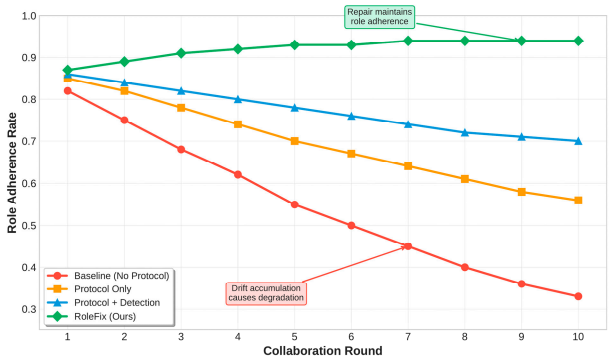

Figure 2. Role adherence rate over collaboration rounds.

Table 2 shows drift detection accuracy across categories. The hybrid approach outperforms both rule-based (average F1: 0.64) and LLM-only (average F1: 0.77) methods, achieving average F1 of 0.85. Figure 3 visualizes these results across drift types. The ablation study shows that each component of RoleFix contributes substantially to performance: the protocol alone reduces drift by 33.7%, adding detection yields a further 28.2% reduction, and the repair mechanism provides an additional 31.5% reduction, with marginal contributions computed relative to the No-Protocol baseline and remaining non-additive due to interaction effects. Overhead analysis indicates modest computational cost: protocol parsing adds ~50 ms per turn, the rule-based detector 10 ms per turn, and the LLM judge (200 ms average) is triggered only in 23% of turns when rule-based checks are uncertain. Overall, RoleFix increases latency by 8.3% compared to the baseline while reducing total task time by 32.4% through fewer rework cycles.

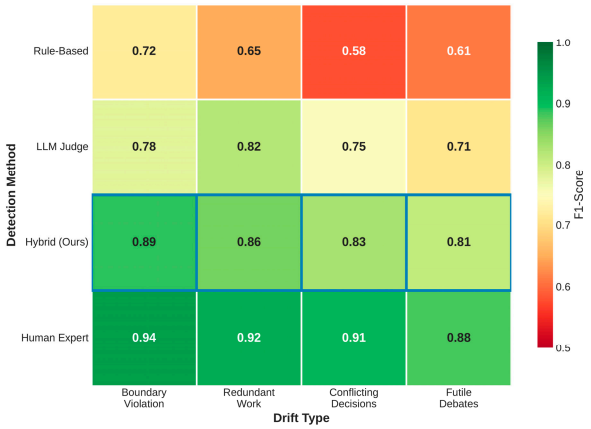

Figure 3. F1-scores for drift detection across methods and drift types.

VI. Conclusions

We presented RoleFix, a lightweight framework addressing the underexplored challenge of role drift in LLM-based multi-agent collaboration. Our contributions include a role drift taxonomy validated through systematic annotation, a structured communication protocol, and a hybrid

 doi:10.20944/preprints202603.0348.v1

detection-repair mechanism with clearly defined trigger conditions. Experiments demonstrate that RoleFix substantially reduces drift incidents and improves task completion rates while introducing modest overhead. Our work has limitations. The current taxonomy, while validated on software engineering and research tasks, may not capture all drift patterns in other domains. The LLM judge component adds latency and cost for ambiguous cases. Future work will explore domain-specific drift patterns, more efficient detection methods, and extension to larger agent teams. We expect that systematic attention to collaboration stability will become increasingly important as multi-agent systems are deployed in complex real-world scenarios.

References

1. S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, and Z. Lin, "MetaGPT: Meta programming for a multi-agent collaborative framework," in The Twelfth International Conference on Learning Representations, 2023.
2. C. Qian, W. Liu, H. Liu, N. Chen, Y. Dang, J. Li, C. Yang, W. Chen, Y. Su, and X. Cong, "Chatdev: Communicative agents for software development," in Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 15174-15186.
3. Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, and J. Liu, "Autogen: Enabling next-gen LLM applications via multi-agent conversations," in First Conference on Language Modeling, 2024.
4. Y. Talebirad and A. Nadiri, "Multi-agent collaboration: Harnessing the power of intelligent llm agents," arXiv preprint arXiv:2306.03314, 2023.
5. T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang, "Large language model based multi-agents: A survey of progress and challenges," arXiv preprint arXiv:2402.01680, 2024.
6. C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan, "Swe-bench: Can language models resolve real-world github issues?," arXiv preprint arXiv:2310.06770, 2023.
7. Y. Wang, D. Wu, F. Liu, Z. Qiu and C. Hu, "Structural priors and modular adapters in the composable fine-tuning algorithm of large-scale models," *arXiv preprint arXiv:2511.03981*, 2025.
8. Z. Qiu, D. Wu, F. Liu and Y. Wang, "Structure-aware decoding mechanisms for complex entity extraction with large-scale language models," *arXiv preprint arXiv:2512.13980*, 2025.
9. X. Zhang, Q. Wang and X. Wang, "Joint cross-modal representation learning of ECG waveforms and clinical reports for diagnostic classification," *Transactions on Computational and Scientific Methods*, vol. 6, no. 2, 2026.
10. C. Shao et al., "Adversarial robustness in text classification through semantic calibration with large language models," 2026.
11. Y. Huang, "Explainable cognitive multi-agent AI for joint intention modeling in complex task planning," *Transactions on Computational and Scientific Methods*, vol. 4, no. 10, 2024.
12. X. Song, "Integrating attention attribution and pretrained language models for transparent discriminative learning," 2026.
13. Y. Luan, "Iterative self-questioning supervision with semantic calibration for stable reasoning chains in large language models," 2026.
14. J. Guo, "Structured multi-stage alignment distillation for semantically consistent lightweight language models," 2026.
15. Y. Hu et al., "TrustOrch: A dynamic trust-aware orchestration framework for adversarially robust multi-agent collaboration," 2025.
16. Y. Li, L. Zhu and Y. Zhang, "Robust text semantic classification via retrieval-augmented generation," *Transactions on Computational and Scientific Methods*, vol. 4, no. 10, 2024.
17. X. Hu et al., "Dynamic prompt fusion for multi-task and crossdomain adaptation in LLMs," in *Proceedings of the 2025 10th International Conference on Computer and Information Processing Technology (ISCIPT)*, pp. 483–487, 2025.
18. R. Liu, R. Zhang and S. Wang, "Transformer-based modeling of user interaction sequences for dwell time prediction in human-computer interfaces," *arXiv preprint arXiv:2512.17149*, 2025.

 doi:10.20944/preprints202603.0348.v1

19. Z. Xu et al., "Generative distribution modeling for credit card risk identification under noisy and imbalanced transactions," 2025.
20. W. C. Chang, L. Dai and T. Xu, "Machine learning approaches to clinical risk prediction: Multi-scale temporal alignment in electronic health records," *arXiv preprint arXiv:2511.21561*, 2025.
21. Y. Wang et al., "Dynamic anomaly identification in accounting transactions via multi-head self-attention networks," *arXiv preprint arXiv:2511.12122*, 2025.
22. X. Yang et al., "Semantics-aware denoising: A PLM-guided sample reweighting strategy for robust recommendation," *arXiv preprint arXiv:2602.15359*, 2026.
23. R. Ying et al., "AI-based causal reasoning over knowledge graphs for data-driven and intervention-oriented enterprise performance analysis," 2025.
24. Y. Ou et al., "A residual-regulated machine learning method for non-stationary time series forecasting using second-order differencing," 2025.
25. S. Pan and D. Wu, "Trustworthy summarization via uncertainty quantification and risk awareness in large language models," in *Proceedings of the 2025 6th International Conference on Computer Vision and Data Mining (ICCVDM)*, pp. 523–527, 2025.
26. J. Yang, S. Sun, Y. Wang, Y. Wang, X. Yang and C. Zhang, "Semantic alignment and output constrained generation for reliable LLM-based classification," 2026.
27. K. Gao, H. Zhu, R. Liu, J. Li, X. Yan and Y. Hu, "Contextual trust evaluation for robust coordination in large language model multi-agent systems," 2025.
28. Q. Zhang, Y. Wang, C. Hua, Y. Huang and N. Lyu, "Knowledge-augmented large language model agents for explainable financial decision-making," arXiv preprint arXiv:2512.09440, 2025.
29. Y. Xing, M. Wang, Y. Deng, H. Liu and Y. Zi, "Explainable representation learning in large language models for fine-grained sentiment and opinion classification," 2025.