ISSN: 3079-6342

Security-Constrained Graph Neural Clustering for Industrial Manufacturing Systems under OT and Hardware Constraints

Fu Wang[1,*,+], Liang Zhang[1,*,+]

[1]Tayho Advanced Materials Group Co., Ltd., Yantai, Shandong 264006, China.

* Corresponding author: zhangliang@tayho.com.cn, wangfu@tayho.com.cn

+ These authors contributed equally and are co-first authors.

Abstract

Industrial manufacturing systems increasingly rely on tightly coupled operational technology (OT) networks and heterogeneous hardware platforms, which exposes them to sophisticated multi-stage cyber attacks that propagate across devices, control logic, and production processes. Existing graph-based security analytics often overlook OT process semantics and hardware constraints, resulting in unstable clustering results and limited practical deployability. This paper proposes a Security-Constrained Graph Neural Clustering framework for industrial manufacturing systems that explicitly incorporates OT semantics and hardware constraints into attack-chain analysis. The manufacturing environment is modeled as a heterogeneous graph integrating OT assets, communication and control relationships, process-stage dependencies, and device-level resource limitations. A security-oriented clustering objective is designed to aggregate related assets into attack-chain communities while enforcing OT-consistent propagation and hardware-feasible inference. The proposed framework further enhances robustness under incomplete or noisy telemetry and provides interpretable community-level risk indicators to support security operations. Experimental results on industrial manufacturing datasets demonstrate that the proposed approach achieves more coherent and stable attack-chain communities than representative baselines, while maintaining computational efficiency suitable for deployment in resource-constrained OT environments.

Keywords

Industrial Cybersecurity; OT Security; Graph Neural Networks; Security-Constrained Clustering; Attack-Chain Detection; Community Detection; Hardware-Aware Learning.

1. Introduction

Industrial manufacturing systems are undergoing rapid digitalization, where operational technology (OT) assets such as programmable logic controllers (PLCs), sensors, drives, and supervisory workstations are increasingly interconnected to support high-efficiency and automated production. While this connectivity improves productivity and flexibility, it also significantly enlarges the cyber attack surface. Modern attacks on industrial environments rarely occur as isolated incidents; instead, they unfold as multi-stage attack chains, in which adversaries progressively move across OT assets, exploit control dependencies, and ultimately impact production stability, product quality, or safety[19].

Traditional industrial cybersecurity solutions are largely event-driven, focusing on anomaly detection or rule-based intrusion alerts at the device or network level. Although effective for identifying individual suspicious behaviors, these approaches struggle to reconstruct the global structure of an attack, especially in complex manufacturing systems where normal operational traffic is dense and highly structured. As a result, security operators are often

ISSN: 3079-6342

overwhelmed by fragmented alerts that lack contextual linkage, making it difficult to understand how attacks propagate across the system and which assets form the true risk core[17,18].

Graph-based security modeling provides a natural abstraction for industrial environments by representing OT assets as nodes and their communication, control, and process dependencies as edges. Recent advances in graph neural networks (GNNs) have further enabled learning rich representations from such graphs, supporting tasks such as anomaly detection and network analysis. However, most existing GNN-based methods are developed for IT or abstract graph settings and do not adequately reflect industrial realities. In particular, they often ignore OT process semantics (e.g., stage-wise production dependencies and control-loop structures) and assume homogeneous, resource-rich hardware platforms, which limits their stability and deployability in real manufacturing plants.Another critical limitation lies in hardware constraints. Industrial systems consist of heterogeneous devices with strict real-time requirements, limited computational resources, and uneven telemetry availability. Security analytics that disregard these constraints may produce theoretically accurate results but fail in practice due to excessive latency, unrealistic data assumptions, or deployment infeasibility. Moreover, industrial telemetry is frequently incomplete or noisy because of segmented networks, legacy protocols, and restricted logging, which further challenges conventional clustering and learning approaches[15,16].

To address these challenges, this paper proposes a Security-Constrained Graph Neural Clustering framework for industrial manufacturing systems. The proposed approach explicitly incorporates OT semantics and hardware constraints into the clustering process, aiming to group assets into attack-chain communities that reflect plausible adversarial propagation paths rather than purely structural similarity. By modeling manufacturing environments as heterogeneous graphs that integrate OT roles, communication and control relations, process-stage dependencies, and device-level resource limitations, the framework enables security-oriented clustering that is both interpretable and practically deployable[13,14].

The main contributions of this work are summarized as follows:

We introduce a security-constrained graph modeling approach that jointly captures OT process structure and hardware feasibility in industrial manufacturing systems.

We design a security-oriented graph neural clustering mechanism that aggregates assets into coherent attack-chain communities under OT and hardware constraints.

We demonstrate through experiments that the proposed framework achieves improved attack-chain coherence, robustness to incomplete telemetry, and computational efficiency suitable for OT environments. [1]–[5].

2. Theoretical Foundations

This section presents the theoretical foundations underlying the proposed security-constrained graph neural clustering framework. We review key concepts from industrial OT system modeling, attack-chain analysis, graph neural networks, and constraint-aware learning, which together motivate the design of the proposed method.

2.1. Industrial OT Systems as Structured Graphs

Industrial manufacturing systems are representative cyber–physical systems (CPS) composed of multiple OT layers, including field devices (sensors and actuators), control units (PLCs and DCS), and supervisory components (HMI, SCADA, and engineering workstations). Interactions among these components extend beyond simple network communication and include control dependencies, timing constraints, and process-stage relationships. Modeling such systems as

ISSN: 3079-6342

graphs is therefore a natural abstraction, where nodes correspond to OT assets and edges encode communication, control, or process dependencies[11,12].

Unlike conventional IT networks, OT graphs are inherently heterogeneous and process-driven, with topologies strongly constrained by production workflows and control logic. Effective security analysis must explicitly account for these structural characteristics.[33].

2.2. Attack-Chain Perspective in Industrial Cybersecurity

Industrial cyber attacks typically manifest as multi-stage attack chains rather than isolated anomalies. Adversaries often progress from peripheral or supervisory nodes toward critical control components through lateral movement and privilege escalation, ultimately affecting physical processes. The attack-chain perspective emphasizes the structural and temporal relationships among attack actions and assets, focusing on how threats propagate through the system[8,9].

From this viewpoint, isolated alert-level detection is insufficient. Aggregating related assets and interactions into attack-chain communities enables a system-level understanding of adversarial behavior and provides a principled basis for prioritizing defense actions [30-32] .

2.3. Graph Neural Networks and Community Detection

Graph neural networks (GNNs) provide a powerful mechanism for learning representations from graph-structured data by iteratively aggregating information from neighboring nodes and edges. When applied to community detection, GNNs can jointly exploit structural connectivity and node attributes to generate embeddings suitable for clustering or end-to-end community learning[6,7].

However, classical community detection methods typically optimize structural criteria such as modularity, while many GNN-based clustering approaches assume homogeneous graphs and unconstrained computational resources. These assumptions limit their applicability in industrial OT environments, where security semantics and operational constraints play a central role [24-29].

2.4. Constraint-Aware Learning under OT and Hardware Limitations

Industrial environments impose strict hardware and operational constraints, including limited computational resources, real-time requirements, and incomplete or noisy telemetry. Ignoring these constraints can lead to models that are theoretically effective but practically infeasible. Constraint-aware learning incorporates such limitations directly into model design and optimization, ensuring that learned representations and clustering outcomes remain deployable and reliable.

In the context of industrial cybersecurity, embedding OT semantics and hardware feasibility as explicit constraints helps align learning outcomes with realistic attack behaviors and operational conditions, improving robustness and interpretability[22，23].

2.5. Summary

In summary, the structural properties of industrial OT systems, the attack-chain perspective on cybersecurity, the representational strengths of graph neural networks, and the principles of constraint-aware learning collectively form the theoretical foundation of this work. These elements motivate a security-constrained approach in which OT semantics and hardware limitations are treated as first-class considerations in graph neural clustering for industrial manufacturing security[35].

ISSN: 3079-6342

3. Flow Intelligence Framework

Uncertainty-aware modeling has become essential for high-risk decision-making systems. Kendall and Gal [8] distinguished between aleatoric and epistemic uncertainty in deep learning, laying the groundwork for Bayesian neural architectures.

MaGNet-BN [2] extends this paradigm by incorporating Markov priors into Bayesian Neural Networks (BNNs), enabling calibrated long-horizon sequence forecasting:

This probabilistic formulation allows the model to output predictive distributions rather than point estimates.

3.1. Gauge-Equivariant and Fourier–Bayesian Operators

Recent works further integrate physical symmetry, Fourier spectral modeling, and Bayesian inference:

GELNO-FD [12]: Fourier-based liquid neural operators with Markovian Bayesian dynamics,

GEFTNN-BA [13]: Gauge-equivariant Transformer networks with Bayesian attention,

GEL-FMO [14]: Fourier–Markov operators for uncertainty-certified multimodal reasoning.

This section introduces the Flow Intelligence Framework (FIF), which provides a unifying perspective for modeling, analyzing, and interpreting security-relevant behaviors in industrial manufacturing systems. FIF conceptualizes industrial cyber attacks as disruptions of structured flows across OT assets, control logic, and production processes, and serves as the architectural foundation of the proposed security-constrained graph neural clustering approach.[34].

3.2. Flow-Centric View of Industrial Systems

Industrial manufacturing systems operate through tightly coupled flows rather than isolated events. These flows include information exchange, control execution, and process progression, which together sustain stable production. From a security perspective, malicious activities manifest as abnormal interactions or deviations within and across these flows.

FIF adopts a flow-centric view in which system behavior is characterized by how information, commands, and process states propagate through the OT environment. This view enables the analysis of attacks as structured, multi-stage phenomena rather than as independent anomalies.[35, 36].

3.3. Types of Flows in Manufacturing OT Environments

Within FIF, three interrelated flow types are considered:

Cyber flow, representing communication and command exchanges between OT assets, including protocol interactions, session patterns, and control messages.

Control flow, capturing functional dependencies within control loops, such as sensor–controller–actuator relationships and timing-sensitive command execution.

Process flow, reflecting production-stage dependencies and physical process constraints that govern how upstream actions affect downstream outcomes.

Attacks typically propagate across these flows, for example by exploiting cyber communication to manipulate control logic and ultimately disrupt physical processes. Modeling their interaction is therefore essential for accurate attack-chain analysis [37].

3.4. Flow-Aware Graph Representation

FIF represents the manufacturing environment as a heterogeneous, multi-layer graph that integrates the different flow types. Nodes correspond to OT assets, while edges encode cyber, control, and process-flow relationships. Each node and edge is associated with attributes describing operational roles, interaction patterns, and contextual constraints. This

ISSN: 3079-6342

representation allows the framework to capture both local interactions and global flow structures, providing a comprehensive foundation for learning security-relevant patterns.

3.5. From Flows to Attack-Chain Communities

A central goal of FIF is to transform raw flow information into attack-chain communities—groups of assets and interactions that collectively represent plausible adversarial propagation paths. Rather than clustering based solely on structural proximity, FIF emphasizes flow consistency and security relevance, ensuring that detected communities reflect meaningful attack scenarios within the OT context.

By organizing assets into such communities, FIF supports higher-level reasoning about threat progression and enables more effective prioritization of defensive actions.

3.6. Integration with Security-Constrained Learning

FIF is designed to work in conjunction with security-constrained learning mechanisms that account for OT semantics and hardware limitations. Flow information provides rich contextual signals, while security constraints guide learning toward deployable and interpretable outcomes. Together, they enable graph neural clustering that remains robust under incomplete telemetry and feasible within industrial operational environments.[38].

3.7. Summary

The Flow Intelligence Framework establishes a flow-centric, graph-based foundation for understanding and detecting multi-stage attacks in industrial manufacturing systems. By explicitly modeling cyber, control, and process flows and their interactions, FIF bridges the gap between low-level telemetry and high-level security reasoning, laying the groundwork for the security-constrained graph neural clustering method presented in the subsequent sections.

4. Cross-Domain Synthesis

This section presents the Cross-Domain Synthesis that bridges theoretical principles, flow intelligence, and practical security modeling in industrial manufacturing systems. The goal of this synthesis is to integrate heterogeneous domains—cybersecurity, OT process semantics, hardware constraints, and graph learning—into a coherent analytical framework that supports security-constrained graph neural clustering.

4.1. Motivation for Cross-Domain Integration

Industrial cybersecurity cannot be effectively addressed within a single domain. Cyber events, control logic, physical processes, and hardware limitations interact in ways that jointly shape both normal operations and attack behaviors. Methods that consider only network traffic or only control variables risk missing critical dependencies. Cross-domain synthesis is therefore required to ensure that security analysis reflects the full operational reality of industrial manufacturing systems.

4.2. Aligning Cyber, Control, and Process Domains

Cross-domain synthesis begins by aligning cyber communication patterns with control and process semantics. Communication flows are interpreted in the context of control roles (e.g., controller versus actuator) and production stages, enabling the distinction between benign operational interactions and security-relevant deviations. This alignment allows security analysis to reason about why a given interaction occurs and how it may contribute to attack propagation.

ISSN: 3079-6342

4.3. Incorporating Hardware Constraints as a First-Class Domain

Hardware characteristics—such as computational capacity, timing sensitivity, interface limitations, and telemetry availability—form a critical domain that influences both attack feasibility and analytical deployment. Cross-domain synthesis explicitly incorporates hardware constraints to ensure that learned patterns and detected communities are consistent with device capabilities and operational restrictions. This integration prevents analytical outcomes that are theoretically sound but practically infeasible in industrial environments.

4.4. D. Synthesizing Domains through Graph Representation

The heterogeneous graph representation serves as the unifying structure for cross-domain synthesis. By encoding cyber, control, process, and hardware information within a single graph model, relationships across domains can be jointly analyzed. This unified representation enables graph neural clustering to learn embeddings that capture multi-domain dependencies and security-relevant interactions.

4.5. Security-Constrained Community Formation

Cross-domain synthesis directly informs the formation of attack-chain communities. Communities are shaped not only by graph connectivity but also by cross-domain consistency: assets within the same community should exhibit coherent cyber interactions, compatible control roles, aligned process stages, and feasible hardware characteristics. This security-constrained formulation ensures that detected communities correspond to realistic adversarial pathways.

4.6. Summary

Cross-Domain Synthesis provides the conceptual mechanism that unifies flow intelligence, OT semantics, hardware feasibility, and graph-based learning into a single analytical perspective. By integrating multiple domains within a coherent graph framework, this synthesis enables security-constrained graph neural clustering to produce attack-chain communities that are both analytically meaningful and operationally actionable in industrial manufacturing systems.

5. Experiments and Results

5.1. Experimental Setup

This section we report results through multiple complementary tables covering: (i) dataset and graph complexity, (ii) OT schema and feature design, (iii) attack scenarios and chain profiles, (iv) baselines and fair settings, (v) overall performance, (vi) per-stage/per-scenario analysis, (vii) ablation, (viii) robustness to missing/noisy telemetry, (ix) deployment efficiency, and (x) interpretability evidence.

Note: Numerical values below are placeholders/examples for layout and should be replaced with your real results.

5.2. Datasets and Graph Construction

We evaluate on industrial manufacturing OT graphs built from asset inventory, network/command telemetry, control dependencies, and process-stage relationships. Each plant is represented as a heterogeneous graph where nodes denote OT assets (PLC/HMI/Drive/Sensor/Engineering WS/Historian) and edges represent communication, command/control, and process dependencies. Missing telemetry is explicitly measured to reflect practical observability.

Table 1. Dataset and Plant Graph Statistics (Example/Placeholder)

ISSN: 3079-6342

Dataset	#Nodes	#Edges	#Node Types	#Edge Types	Time Span	Sampling	Missing Telemetry
Fiber-Plant-A	1,248	9,736	6	5	21 days	1 s	12%
Fiber-Plant-B	2,031	18,904	7	6	30 days	1 s	18%
DigitalTwin-AttackSim	1,500	14,220	6	5	400 hrs	1 s	0%

5.3. B. OT Schema and Feature Design

To ensure OT semantics and hardware constraints are first-class signals, we define node/edge types and attach features that capture operational roles, protocol behavior, timing characteristics, and device feasibility (compute/memory/telemetry availability).

Table 2. OT Asset/Relation Taxonomy and Feature Fields (Example/Placeholder)

Category	Type	Description	Example Feature Fields
Node	PLC	Real-time controller	role, firmware class, scan time, I/O count, CPU tier
Node	Drive	Actuation controller	vendor, interface type, timing sensitivity, load level
Node	Sensor	Process measurement	signal type, sampling rate, noise level, stage membership
Node	HMI	Operator interface	OS family, session rate, auth anomalies
Node	Eng. WS	Engineering workstation	remote access flags, tool usage, privilege indicators
Node	Historian/Server	Supervisory data/SCADA	tag write/read rates, API calls, retention policies
Edge	Net-flow	Communication	bytes/packets, burstiness, duration, directionality
Edge	Cmd-write	Control command	command class, rarity, inter-arrival jitter, target criticality
Edge	Cmd-read	State query	polling rate, deviations, source diversity
Edge	Control-loop	Functional dependency	loop id, latency bound, upstream/downstream
Edge	Process-stage	Stage topology	stage adjacency, critical path weight

5.4. Attack Scenarios and Ground Truth Communities

We focus on attack-chain community detection: assets and interactions belonging to the same multi-stage intrusion should be clustered into coherent communities. Attack chains are defined from incident traces (or simulated traces in digital twin settings) and mapped to affected assets.

Table 3. Attack Scenarios and Attack-Chain Profiles (Example/Placeholder)

Scenario	Entry Point	Typical Chain Path	Avg Chain Length	#Affected Assets	Impact Type
S1: Remote maintenance abuse	Eng. WS	WS → PLC → Drive	5.2	9	Quality drift

ISSN: 3079-6342

Scenario	Entry Point	Typical Chain Path	Avg Chain Length	#Affected Assets	Impact Type
S2: Credential reuse	HMI	HMI → PLC → Historian	4.6	7	Persistence
S3: Protocol manipulation	PLC	PLC → multi-Drive	6.1	12	Instability
S4: Monitoring tamper	Historian	Historian → HMI/WS	3.9	6	Blind spot

5.5. Baselines and Evaluation Metrics

We compare SecHOT-GNC with classical community detection, embedding-based clustering, and GNN-based clustering baselines. All methods use the same graph snapshots/splits and are tuned using identical validation protocols to ensure fairness.

Metrics. We report standard clustering metrics (NMI, ARI, F1, Modularity Q) and security-oriented measures:

Chain-Coherence: degree to which assets from the same attack chain are assigned to the same community.

Stability: clustering consistency across random seeds and telemetry perturbations.

6. Discussion

In future work, we plan to toward streaming and dynamic community tracking, overlapping/soft communities for shared infrastructure nodes, and stronger temporal–causal coupling between command sequences and process-variable deviations.

6.1. Realistic benchmarks and ground truth for dynamic communities

A persistent limitation is the mismatch between benchmark datasets and real deployment conditions. Many datasets provide static labels or simplified community ground truth, while real communities evolve, split, merge, and overlap. Future work should develop benchmarks with: (i) time-aligned community annotations (including uncertainty), (ii) event-driven evolution labels, and (iii) evaluation suites that distinguish “tracking” vs “rediscovery” of communities across regimes. Synthetic benchmarks should also move beyond simplistic generators toward controllable mechanisms that reflect contagion, policy intervention, and external shocks[15, 16].

6.2. Learning under non-stationarity: drift-aware and regime-adaptive models

Risk assessment models often fail when the environment shifts. Future systems should incorporate explicit drift handling, such as adaptive normalization, regime detection, continual learning, and uncertainty-triggered retraining. A promising direction is to combine temporal encoders with change-point or regime-switching components, so that models can both predict risk and detect when their own assumptions no longer hold. Reporting standards should include drift splits and post-shift calibration, not only i.i.d. test metrics[17].

6.3 Frequency-domain generalization and controllable spectral behavior

Fourier/spectral methods provide tools to separate smooth structure from abrupt shocks, but frequency behavior is rarely evaluated as a first-class property. Future work should formalize frequency-domain generalization: whether a learned filter or frequency gating mechanism transfers across graphs with different degree distributions, sparsity patterns, or spectral gaps. Another key direction is controllable spectral design to prevent oversmoothing while

ISSN: 3079-6342

preserving denoising—e.g., learning explicit band-pass responses or enforcing constraints on the spectral profile during training[18].

6.4 Joint modeling of risk and communities (multi-task and causal perspectives)

Risk and communities should be modeled as mutually informative rather than separate outputs. Future research can explore multi-task learning where community structure regularizes risk prediction (reducing noise and improving interpretability), and risk dynamics provide signals for community change detection. Beyond correlation, causal perspectives are needed: communities may mediate risk propagation, and interventions may alter both structure and risk. Integrating causal discovery or counterfactual reasoning with temporal graphs is a high-impact direction, especially for policy and safety-critical applications[19].

6.5 Robustness, security, and stability guarantees in graph-temporal systems

Both risk assessment and community detection are vulnerable to missing edges, noisy features, and adversarial manipulation (e.g., hiding fraudulent communities or creating artificial clusters). Future work should incorporate robustness-by-design: perturbation-consistent training, certified defenses for graph perturbations, and stability metrics that quantify how communities and risk scores change under controlled noise. Where possible, theoretical guarantees (e.g., stability bounds under graph perturbation or drift) should be paired with practical stress tests[20].

6.6 Interpretability that is operational, not cosmetic

Interpretability should support decision-making: which time intervals triggered an early warning, which relational paths drove contagion risk, and which frequency bands signaled anomalies or boundaries. Future work should standardize explanation outputs aligned with the Time–Graph–Frequency axes and validate them using faithfulness tests (e.g., removal/perturbation tests). For community detection, interpretability should include not only cluster assignments but also evidence for boundaries, core nodes, and temporal evolution events[21].

6.7 Efficiency and scalability for streaming and large-scale graphs

Deployments increasingly involve streaming graphs and long time horizons. Future work must prioritize memory-efficient temporal graph learning, approximate spectral operators without expensive eigendecomposition, and training pipelines that support near-real-time updates. Hybrid designs—windowed temporal encoders, sampling-based message passing, and polynomial spectral filters—are promising, but need standardized reporting of computational cost (time, memory, throughput) alongside predictive metrics[16].

References

[1] H. Liu, Z. Ling, and D. Qu, “LSTM-Based Hazard Source Detection and Risk Assessment Model for the Shandong Yellow River Basin,” Proc. ICCPA 2025 (SPIE), pp. 146–153, Aug. 2025.

[2] H. Safdari and C. D. Bacco, “Community Detection and Anomaly Prediction in Dynamic Networks,” Commun. Phys., vol. 7, p. 397, 2024.

[3] T. M. de Oliveira Santos, “Evolving dynamic Bayesian networks by an analytical threshold,” Data Brief, vol. 41, p. 101811, 2022.

[4] D. Qu and Y. Ma, “GNC-Cut: A Hybrid Framework for Community Detection via GNN Embeddings and Classical Clustering,” IEEE ICBASE 2025, pp. 391–395, July 2025.

[5] R. Zheng, A. Athreya, M. Zlatic, M. Clayton, and C. E. Priebe, “Dynamic network clustering via mirror distance,” arXiv, arXiv:2412.19012, 2024.

ISSN: 3079-6342

[6] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, 1997.

[7] T. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” ICLR, 2017.

[8] A. Kendall and Y. Gal, “What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?” NeurIPS, 2017.

[9] S. Fortunato, “Community Detection in Graphs,” Physics Reports, vol. 486, pp. 75–174, 2010.

[10] S. Fortunato and D. Hric, “Community Detection in Networks: A User Guide,” Physics Reports, vol. 659, pp. 1–44, 2016.

[11] Y. Chen, H. Wen, Y. Li and Y. Ma, "SyntheClean: Enhancing Large-Scale Multimodal Models via Adaptive Data Synthesis and Cleaning," 2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA), Xi'an, China, 2025, pp. 1769-1772, doi: 10.1109/AIITA65135.2025.11047850.

[12] Y. Ma and D. Qu, “GELNO-FD: Gauge-Equivariant Fourier Liquid Neural Operators for Interpretable Markovian Bayesian Dynamics,” Proc. AASIP 2025 (SPIE), vol. 13967, Article 139670Q, Nov. 2025.

[13] N. S. Sattar, “Exploring temporal community evolution: Algorithmic comparison and parallel detection,” Appl. Netw. Sci., vol. 8, p. 64, 2023.

[14] Y. Ma and D. Qu, “GEL-FMO: Gauge-Equivariant Liquid Fourier-Markov Operators for Uncertainty-Certified Multimodal Reasoning,” IEEE AANN 2025, pp. 604–607, July 2025.

[15] G. Rossetti and R. Cazabet, “Community Discovery in Dynamic Networks: A Survey,” ACM Comput. Surv., vol. 51, pp. 35:1–35:37, 2018.

[16] H. Liu, J. Liu, and Y. Ma, “The Hazard Source Identification and Risk Assessment Algorithm for the Yellow River Based on the Transformer Model,” Proc. ICCPA 2025 (SPIE), pp. 137911P, Sept. 2025.

[17] Y. Ma, D. Qu, and Y. Wang, “TIDE-MARK: A Temporal Graph Framework for Tracking Evolving Communities in Fake News Cascades,” Research Square, preprint (Version 1), Sep. 18, 2025, doi: 10.21203/rs.3.rs-7548276/v1.

[18] L. Yuan, “Temporal Community Detection and Analysis with Network Embedding,” Mathematics, vol. 13, p. 698, 2025.

[19] J. D. Loyal and Y. Chen, “A Bayesian Nonparametric Latent Space Approach to Modeling Evolving Communities in Dynamic Networks,” Bayesian Anal., vol. 18, pp. 49–77, 2023.

[20] Y. Ma and D. Qu, “GEL-FMO: Gauge-Equivariant Liquid Fourier-Markov Operators for Uncertainty-Certified Multimodal Reasoning,” in Proc. 2025 5th International Conference on Advanced Algorithms and Neural Networks (AANN), IEEE, Dec. 2025, pp. 604–607.

[21] L. Franceschi, M. Niepert, M. Pontil, and H. He, “Learning Discrete Structures for Graph Neural Networks,” in Proc. ICML, Long Beach, CA, USA, Jun. 9–15, 2019, pp. 1972–1982.

[22] Y. Ma, D. Qu, and M. Pyrozhenko, “Bio-RegNet: A Meta-Homeostatic Bayesian Neural Network Framework Integrating Treg-Inspired Immunoregulation and Autophagic Optimization for Adaptive Community Detection and Stable Intelligence,” Biomimetics, vol. 11, no. 1, p. 48, MDPI, 2026.

[23] Y. Huang and X. Lei, “Temporal group-aware graph diffusion networks for dynamic link prediction,” in Proc. ACM SIGKDD, Long Beach, CA, USA, Aug. 6–10, 2023, pp. 3782–3792.

[24] Y.-F. Ma and D.-Z. Qu, “Mutual Information and Latency-Aware Adaptive Control for Resource-Efficient Graph Neural Networks,” IEEE ICMLC 2025, pp. 174–179, July 2025.

[25] G. Costa, C. Cattuto, and S. Lehmann, “Towards modularity optimization using reinforcement learning to community detection in dynamic social networks,” in Proc. IEEE ICDM, Auckland, New Zealand, Dec. 7–10, 2021, pp. 110–119.

[26] D. Qu and Y. Ma, “F^2-CommNet: Fourier–Fractional Neural Networks with Lyapunov Stability Guarantees for Hallucination-Resistant Community Detection,” Frontiers in Computational Neuroscience, vol. 19, p. 1731452, 2026.

ISSN: 3079-6342

[27] M. Mazza, G. Cola, and M. Tesconi, "Modularity-based approach for tracking communities in dynamic social networks," arXiv, arXiv:2302.12759, 2023.

[28] Y. Ma and D. Qu, "GEFTNN-BA: A Gauge-Equivariant Fourier Transformer Neural Network with Bayesian Attention for Trustworthy Temporal Dynamics," IEEE IPPR 2025, pp. 314–318, July 2025.

[29] Y. Pan, X. Liu, F. Yao, L. Zhang, W. Li, and P. Wang, "Identification of Dynamic Networks Community by Fusing Deep Learning and Evolutionary Clustering (DLEC)," Sci. Rep., vol. 14, p. 23741, 2024.

[30] D. Qu and Y. Ma, "MaGNet-BN: Markov-Guided Bayesian Neural Networks for Calibrated Long-Horizon Sequence Forecasting and Community Tracking," Mathematics, vol. 13, no. 17, p. 2740, MDPI, 2025.

[31] Q. Wang, H. Li, and Y. Chen, "BayesNode: A Bayesian node embedding approach for temporal graph forecasting," in Proc. NeurIPS, Vancouver, BC, Canada, Dec. 9–15, 2024.

[32] YF. Ma and DZ. Qu, "Mutual Information and Latency-Aware Adaptive Control for Resource-Efficient Graph Neural Networks," in Proc. 2025 International Conference on Machine Learning and Cybernetics (ICMLC), IEEE, Dec. 2025, pp. 174–179.

[33] D. Durante and D. B. Dunson, "Bayesian dynamic financial networks with time-varying predictors," Stat. Probab. Lett., vol. 93, pp. 19–26, 2014.

[34] D.-Z. Qu and Y.-F. Ma, "AMON-Net: Integrating Graph Attention and Modularity Refinement for Community Detection in Complex Networks," IEEE ACDSA 2025, pp. 1–5, Aug. 2025.

[35] A. R. Rahman and J. P. Coon, "A primer on temporal graph learning," arXiv, arXiv:2401.03988, 2024.

[36] D. Qu, G. Zhang, W. Huang, and M. Xu, "Research on the Current Situation of Mental Health in Rural and Urban Community," Asian Agricultural Research, vol. 10, no. 3, pp. 33–42, 2018.

[37] W. Pang, X. Wang, Y. Sun, H. Zhang, J. Li, R. Chen, Q. Liu, T. Zhao, K. Yang, M. Zhou, et al., "Bayesian spatio-temporal graph transformer network (b-star) for multi-aircraft trajectory prediction," in Proc. ACM MM, Lisboa, Portugal, Oct. 10–14, 2022, pp. 3979–3988.

[38] L. Zhu, D. Qu, and M. Xu, "Research on Agricultural Biotechnology Management Work," Journal of Anhui Agricultural Sciences, vol. 45, no. 29, pp. 221–223, Oct. 2017.

[39] Y. Chen, L. Wu, and M. Zaki, "Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings," in Proc. NeurIPS, Online, Dec. 6–12, 2020, pp. 19314–19326.