

Universidade do Minho
Escola de Engenharia

Shashank Mishra

Predictive Incident Detection Architectural Proposal for Local AI in SOC System in Continuous Time

Junho de 2026

Predictive Incident Detection Architectural Proposal for Local AI in SOC
Shashank Mishra
UMinho | 2026

Shashank Mishra

Predictive Incident Detection Architectural Proposal for Local AI in SOC

Master Dissertation

Master's on International Cybersecurity and Cyberintelligence

Work developed under the guidance of:

Professor Henrique Santos

June 2026

COPYRIGHT AND TERMS OF USE OF THE WORK BY THIRD PARTIES

This is an academic work that third parties may use, provided that internationally accepted rules and best practices regarding copyright and related rights are respected.

Therefore, this work may be used under the terms of the license indicated below.

If the user needs permission to use the work under conditions not covered by the indicated license, they should contact the author through the RepositoriUM of the University of Minho.

License granted to users of this work.



Predictive Incident Detection Architectural Proposal for Local AI in SOC © 2026 by Shashank Mishra is licensed under CC BY-NC-SA 4.0. To view a copy of this license, visit https://creativecommons.org/licenses/by-nc-sa/4.0/

Braga, 25th June 2026

Shashank Mishra



Acknowledgments

First and foremost, I would like to express my sincere gratitude to my academic advisor, Professor Henrique Santos, and my mentor at Eurotux, Rodrigo Araujo. Thank you, Professor Santos, for your steadfast support, expertise, and encouragement throughout the development of this thesis. To Rodrigo Araujo, I am deeply grateful for introducing me to such a compelling thesis topic and for providing exceptional guidance, insight, and mentorship along the way. I also extend my appreciation to Eurotux for providing the opportunity to undertake this project and to learn within a professional environment that values collaboration and innovation. Your combined support has been instrumental in making this work possible.

On a personal note, I am deeply indebted to my friend, Shubham. Thank you for your unwavering belief in me, your endless patience, and your invaluable help throughout this entire process. Navigating the demands of this journey was daunting, but having you in my corner as a constant source of strength and support made all the difference.

Finally, my deepest and most heartfelt thanks go to my family. To my parents, thank you for your unconditional love, your countless sacrifices, and for always believing in my potential even before my dreams fully took shape. To my brother, thank you for being my anchor and for standing by my side in every possible way. I could not have reached the finish line without you all, and this milestone belongs to you just as much as it belongs to me.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the pro cess leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Braga, 30th June 2026

Shashank Mishra

(Student name)

DECLARATION ON THE USE OF ARTIFICIAL INTELLIGENCE (AI) TOOLS

I hereby declare having acted this academic work with integrity and confirm that I have not resorted to any form of undue use of AI tools, nor to the falsification or omission of information, sources, data, results or processes in any of the stages leading to its elaboration.

I further declare that any use of AI tools was done in a transparent and responsi ble manner, being duly noted in the work, and that I critically verified the accuracy of the content produced with their support, assuming full responsibility for what I submit.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho, as well as the applicable standards of academic integrity and data protection.

Braga, 30th June 2026

Shashank Mishra

(Student name)

“If I have seen further, it is by standing on the shoulders of Giants”

Sir Issac Newton

Abstract

The dissolution of the traditional network perimeter and the adoption of zero-trust architectures have inundated modern Security Operations Centers (SOCs) with high-velocity, heterogeneous telemetry, creating highly stochastic environments where Advanced Persistent Threats (APTs) thrive by mimicking benign behavior over extended temporal horizons. Traditional machine learning Intrusion Detection Systems (IDS) rely on monolithic, static models that suffer from severe performance degradation due to continuous concept and feature drift, label scarcity, and structural graph bottlenecks. To resolve these operational gaps, this thesis proposes AEGIS-CT, a highly scalable, self-supervised, and predictive IDS pipeline that mathematically bifurcates the continuous physics of network traffic from the discrete semantic reasoning of attacker intent. The architecture is structured across three specialized phases: Phase 1 standardizes incomplete, mixed-type streaming data into a unified latent space using a Gaussian Copula module and autonomously mitigates structural and semantic drift via an unsupervised detector (ADWIN-U) coupled with Representation Enhancement for Concept Drift Adaptation (ReCDA). Phase 2 contextualizes anomalies by utilizing an Ontology-Grounded Retrieval-Augmented Generation (OG-RAG) framework, mapping telemetry to a strict STIX and AttacKG+ cybersecurity knowledge graph to extract unambiguous, semantic macro-states without relying on numerical data evaluation by Large Language Models (LLMs). Phase 3 integrates these bifurcated payloads into an Episodic Continuous-Time Temporal Graph Neural Network (CT-TGNN). By leveraging Time-Aware Batch-Normalized Ordinary Differential Equations (TA-BN-ODE) and decoupling continuous computation from historical memory via a Global State Registry, the system evaluates temporal network evolution without adjacency matrix memory explosion. Rigorous theoretical evaluation establishes strict mathematical bounds on computational complexity, demonstrating the reduction of point process scaling from $\mathcal{O}(n^2)$ to $\mathcal{O}(n^{3/2})$. A logical dry-run against the SolarWinds supply-chain compromise validates the pipeline's theoretical capacity to prevent adversarial drift poisoning, bridge massive temporal dormancy gaps, and translate complex topological deviations into highly explainable, sample-level alerts designed for seamless Human-AI teaming.

Index

1 Introduction **1**

1.1 The Context of Modern Security Operations . . . 1

1.2 The Proposed Solution: A Predictive, Self-Supervised Pipeline . . . 1

1.3 Problem Formulation & Design Requirements . . . 2

1.4 Structure of the Thesis . . . 3

2 Literature Review **4**

2.1 Literature Retrieval and Selection Strategy . . . 4

2.2 Background & Related Work . . . 4

2.2.1 SSL IDS Paradigms . . . 5

2.2.2 Ingestion & Unsupervised Drift Adaptation . . . 8

2.2.3 Semantic Enrichment & Ontology Mapping . . . 10

2.2.4 Mitigation of Graph Structural Bottlenecks & Continuous Inference . . . 12

3 Proposed Architecture **15**

3.1 Phase 1: Ingestion, Normalization & Unsupervised Drift Adaptation . . . 15

3.1.1 Gaussian Copula (GC) Module: Data Normalization and Imputation . . . 15

3.1.2 ADWIN-U: Unsupervised Concept Drift Detection . . . 16

3.1.3 Representation Enhancement for Concept Drift Adaptation (ReCDA) . . . 18

3.2 Phase 2: Ontology-Grounded Correlation & State Extraction . . . 19

3.2.1 Temporal Bounding and Alert Contextualization . . . 20

3.2.2 Semantic Enrichment and Preprocessing . . . 21

3.2.3 Factual Deduction and Semantic-Preserving Pruning . . . 22

3.2.4 The Encoder: Mathematical Transformation for CT-TGNN . . . 23

3.3 Phase 3: Episodic Continuous-Time Node Classification . . . 24

3.3.1 Episodic Initialization via Global State Registry . . . 25

3.3.2 Multi-Scale Temporal Encoding and Continuous ODE Dynamics . . . 25

3.3.3 Volatility Calculation and Discrete State Jump . . . 26

3.3.4 Explainable Classification and Unconditional Write-Back . . . 27

4 Mathematical and Algorithmic Framework **28**

4.1 Phase 1: Ingestion, Normalization, and Drift Management . . . 28

4.1.1 Structural Normalization: The Gaussian Copula Framework . . . 28

4.1.2 Concept Drift Detection (ADWIN-U) . . . 29

4.1.3 Representation Enhancement for Drift Adaptation (ReCDA) . . . 29
4.2 Phase 2: Ontology-Grounded Correlation and State Extraction . . . 30
4.2.1 Graph Ontology and State Summary Formalization . . . 30
4.2.2 Continuous Alert Bifurcation and the Edge Feature Calculator . . . 31
4.3 Phase 3: Episodic Continuous-Time Node Classification . . . 33
4.3.1 Continuous Micro-Edge Processing and State Evolution . . . 33
4.3.2 Point Process Events and Discrete State Jumps . . . 34
4.3.3 Constraint and Loss Formulation . . . 35

5 Theoretical Evaluation & Architectural Analysis 36
5.1 Feasibility Analysis and Architectural Dependencies . . . 36
5.1.1 Summary of Architectural Dependencies . . . 36
5.1.2 Component Dependency Breakdown . . . 37
5.1.3 Operational Feasibility within an Enterprise SOC . . . 39
5.1.4 Rigorous Operational Feasibility and Cost-Benefit Analysis . . . 39
5.2 Computational Complexity and Memory Bounds . . . 40
5.2.1 Phase 1: Ingestion and Drift Management Constraints . . . 41
5.2.2 Phase 2: Token Boundaries and Clustering Limits . . . 41
5.2.3 Phase 3: Continuous Integration and Scalability Proofs . . . 42
5.3 Theoretical Evaluation: Dry-Run of an APT Attack . . . 42
5.3.1 The SolarWinds Campaign Timeline and SIEM Telemetry . . . 43
5.3.2 Logical Pipeline Execution and Dry-Run Analysis . . . 43

6 Conclusion 49
6.1 Synthesis of the Research Problem and Solution . . . 49
6.2 Alignment with Design Requirements . . . 49
6.3 Significance and Impact . . . 50
6.4 Future Work and Next Steps . . . 50

Bibliography 52

Figures Index

Phase 1 Architecture for continuous drift detection & mitigation 16
Phase 2 Architecture mapping the transformation of raw SIEM alerts 19
The Phase 3 Architecture routing continuous and discrete payloads through the CT-TGNN to Alert 24
Dry-Run of the SolarWinds Attack represented in modified AttacKG+ Structure 44

Acronyms

AD FS *Active Directory Federation Services*.

ADWIN *Adaptive Windowing*.

API *Application Programming Interface*.

APT *Advanced Persistent Threat*.

BAR *Balanced Accuracy by the Amount of Requested Labeled Data*.

C2 *Command and Control*.

CKG *Cybersecurity Knowledge Graph*.

CoT *Chain-of-Thought*.

CT-TGNN *Continuous-Time Temporal Graph Neural Network*.

CTI *Cyber Threat Intelligence*.

CVE *Common Vulnerabilities and Exposures*.

DDM *Drift Detection Method*.

DDoS *Distributed Denial of Service*.

DGA *Domain Generation Algorithm*.

DR *Design Requirement*.

DTDG *Discrete-Time Dynamic Graphs*.

EDDM *Early Drift Detection Method*.

EM *Expectation-Maximization*.

GAT *Graph Attention Network*.

GC *Gaussian Copula*.

GCN *Graph Convolutional Network*.

GNN *Graph Neural Network*.

HAC *Hierarchical Agglomerative Clustering*.

I/O *Input/Output*.

IDS *Intrusion Detection System*.

LLM *Large Language Model*.

MFA *Multi-Factor Authentication*.

MLSecOps *Machine Learning Security Operations*.

ODE *Ordinary Differential Equation*.

OG-RAG *Ontology-Grounded Retrieval-Augmented Generation*.

PC *Pairwise Constrained*.

RAG *Retrieval-Augmented Generation*.

ReCDA *Representation Enhancement for Concept Drift Adaptation*.

SIEM *Security Information and Event Management*.

SOC *Security Operations Center*.

SPC *Statistical Process Control*.

SSAD *Semi-Supervised Anomaly Detector*.

SSL *Semi-Supervised Learning*.

TA-BN *Temporal Adaptive Batch Normalization*.

TA-BN-ODE *Time-Aware Batch-Normalized Ordinary Differential Equations*.

TGCN *Temporal Graph Convolutional Network*.

TTP *Tactics, Techniques, and Procedures*.

U2R *User-to-Root*.

1 Introduction

1.1 The Context of Modern Security Operations

The paradigm of enterprise cybersecurity has fundamentally shifted. With the dissolution of the traditional network perimeter and the widespread adoption of zero-trust architectures, internal network telemetry has become hyper-critical. Modern Security Operations Centers (SOCs) are consequently overwhelmed by heterogeneous, high-velocity data streams generated by raw Security Information and Event Management (SIEM) logs (Zhuo et al., 2025). Within these highly stochastic environments, adversaries deploy Advanced Persistent Threats (APTs)—sophisticated, low-and-slow campaigns designed to evade signature-based detection by mimicking benign user behaviors over extended temporal horizons.

To combat this, recent efforts to automate incident discovery have increasingly relied on machine learning (ML) paradigms (Jalalvand, Baruwal Chhetri, Nepal, & Paris, 2024). However, these current approaches often treat the intrusion detection pipeline as a series of isolated sub-problems. Existing Intrusion Detection System (IDS) research largely operates on the fundamentally erroneous assumption that a singular, monolithic algorithm—often trained in offline, static environments—can manage the evolving complexities of enterprise networks. Consequently, models deployed in live SOCs suffer from severe performance degradation due to continuous feature and concept drift (Pan et al., 2026), an inability to process multi-modal data organically, and a reliance on vast repositories of labeled data that simply do not exist in real-world scenarios.

1.2 The Proposed Solution: A Predictive, Self-Supervised Pipeline

To bridge this critical operational gap, this thesis proposes AEGIS-CT, a highly scalable, self-supervised, and predictive IDS pipeline explicitly designed for enterprise zero-trust networks. Moving beyond siloed ML applications, this framework synthesizes advanced data normalization, LLM-driven ontology mapping, and continuous-time graph neural networks into a cohesive, modular architecture.

Crucially, this architecture serves not just as an anomaly detector, but as intelligent scaffolding for emerging Agentic AI workflows—specifically, by utilizing the Phase 2 Local LLM to autonomously query, reason over semantic ontologies, and deduce tactical state summaries without human intervention—establishing a practical, mathematically rigorous foundation for autonomous, end-to-end threat remediation. The proposed architecture handles data gracefully from ingestion to prediction across three highly specialized, sequential phases:

- Phase 1: Ingestion, Normalization & Unsupervised Drift Management: Acting as an unsupervised data stabilizer, this phase tackles the highly stochastic nature of raw SIEM logs. It standardizes incomplete, mixed-type streaming data into a unified latent space using a Gaussian Copula module. To ensure long-term viability, it continuously adapts to both structural and semantic concept drifts using an unsupervised

detector coupled with Representation Enhancement (ReCDA), eliminating the bottleneck of manual data labeling.

- Phase 2: Ontology-Grounded Correlation & Semantic Extraction: This phase bridges the gap between raw alerts and human-readable threat intelligence. Utilizing a Temporal Bounding Mechanism, it fetches historical contexts and maps discrete semantic features to a rigid Cybersecurity Knowledge Graph (CKG) utilizing the STIX framework over an AttacKG+ structure. By leveraging an Ontology-Grounded RAG (OG-RAG) framework, it resolves entity polysemy and extracts actionable MITRE TTP labels and semantic summaries.
- Phase 3: Episodic Continuous-Time Node Classification: Addressing the traditional memory explosions of graph-based deep learning, this phase evaluates the enterprise topology using an episodic Continuous-Time Temporal Graph Neural Network (CT-TGNN). By calculating temporal volatility and fusing discrete semantic "shocks" with continuous state representations, a self-supervised Deviation Network Decoder (TFLAG) dynamically decodes these mathematical projections into highly explainable, sample-level alerts.

1.3 Problem Formulation & Design Requirements

To establish a verifiable blueprint for this synthesized architecture, the framework is engineered to satisfy five core operational mandates. These Design Requirements (DR) directly address the distinct, compounding challenges of live SOC environments and serve as structural validation parameters throughout this thesis:

- DR-1: Robust Ingestion of Incomplete and Heterogeneous Data Streams: Enterprise logs do not adhere to pristine, academic formats. The architecture must dynamically process unpredictable, mixed-type streaming SIEM alerts encompassing Boolean, ordinal, and continuous features. It must mathematically project these into a cohesive latent space to robustly impute missing values, thereby preventing oscillating gradients and biased model initializations during downstream processing (Zhuo et al., 2025; Arazzi et al., 2025).
- DR-2: Unsupervised Dual-Drift Adaptation under Label Scarcity: Zero-trust networks are not static; they evolve as new services are deployed and user behaviors shift. An effective architecture must autonomously detect and adapt to complex, dual-drift scenarios (both Feature Drift and Concept Drift). It must do so in an unsupervised or self-supervised manner to re-tune decision boundaries dynamically, preventing catastrophic model degradation without relying on the operationally unviable standard of continuous manual labeling (Shyaa et al., 2024; Arazzi et al., 2025).
- DR-3: Contextualized Alert Enrichment and Unambiguous CKG Construction: An isolated alert lacks the tactical narrative required to detect an APT. The architecture must contextualize incoming alerts by dynamically fetching related historical states and integrating external Cyber Threat Intelligence (CTI). By parsing this enriched data to resolve semantic ambiguities (such as synonymy and polysemy) and mapping

it to a strict CKG ontology, the system provides a comprehensively contextualized, factual baseline for subsequent prediction (B. Li, Yang, Deng, & Pan, 2025; Y. Zhang et al., 2025; Z. Jiang et al., 2024).

- DR-4: Mitigation of Structural Bottlenecks in Dynamic Graphs: Traditional static Graph Neural Networks (GNNs) are fundamentally misaligned with streaming network data, struggling with adjacency matrix memory explosion, catastrophic forgetting, and poor temporal tracking. To scale securely within an enterprise, the architecture must maintain continuous-time integrity without maintaining a perpetual global graph in active memory, utilizing episodic updates to securely track network evolution (Anaedevha, Trofimov, & Borodachev, n.d.; Z. Wang et al., 2025; Zola, Segurola-Gil, Bruse, Galar, & Orduna-Urrutia, 2022; King & Huang, 2023).
- DR-5: Explainable Alert Prioritization for Human-AI Teaming: Black-box detection algorithms generate friction in live SOCs. The architecture must move beyond binary, uninterpretable classification. It must provide prioritized, sample-level alerts backed by transparent, tactically grounded explanations, mathematically linking topological anomalies back to explicit MITRE ATT&CK tactics to facilitate seamless SOC Human-AI Teaming (Jalalvand et al., 2024).

1.4 Structure of the Thesis

The thesis is organized into six chapters. Following the introduction, Chapter 2 reviews the foundational literature concerning Self-Supervised Learning (SSL) Intrusion Detection Systems (IDS), unsupervised drift adaptation, and continuous graph inference. Chapter 3 introduces the core architectural contribution: a decoupled, tri-phase pipeline that sequentially executes structural normalization and drift adaptation (Phase 1), ontology-grounded semantic state extraction (Phase 2), and predictive continuous-time node classification (Phase 3). Chapter 4 formalizes the rigorous mathematical and algorithmic frameworks underpinning these mechanisms, including Gaussian Copula mappings and continuous ordinary differential equations (ODEs). Chapter 5 evaluates the architecture's real-world viability by establishing strict computational complexity bounds, conducting an operational feasibility analysis for enterprise Security Operations Center (SOC) integration, and theoretically validating the pipeline through a logical dry-run of the SolarWinds Advanced Persistent Threat (APT). Finally, Chapter 6 concludes the research by synthesizing the findings, assessing alignment with the initial design requirements, and outlining future directions for empirical validation.

2 Literature Review

2.1 Literature Retrieval and Selection Strategy

To establish the theoretical foundation for this research, peer-reviewed literature was sourced from prominent academic databases, primarily *IEEE Xplore*, *ACM Digital Library*, *Springer Link*, *ScienceDirect*, and *Google Scholar*. To ensure a comprehensive and up-to-date collection, traditional keyword searches were augmented with AI-assisted discovery platforms—including *Connected Papers*, *Consensus*, and *Elicit*. These tools streamlined the rapid identification of citation networks, automated the extraction of key empirical findings, and helped identify conceptually related research beyond exact keyword matches.

The search strategy targeted the intersection of the primary study domains using Boolean combinations of keywords such as: "Graph Neural Networks," "Concept/Feature Drift," "Cybersecurity Ontology" (e.g., STIX), and "Self-Supervised Intrusion Detection." Priority was given to high-impact journal articles, flagship conference proceedings (e.g., IEEE S&P, ACM CCS, NDSS), and authoritative industry white papers detailing advanced persistent threat (APT) telemetry.

To ensure relevance and methodological rigor, a two-stage screening process was applied to the retrieved literature:

1. **Initial Phase (Abstract & Keyword Analysis):** Documents were first evaluated against the framework's Design Requirements (DR-1 through DR-5). Papers failing to address streaming telemetry, graph-structured data, or drift mechanics were excluded.

2. **Secondary Phase (Full-Text Appraisal):** The remaining documents underwent a full-text review scrutinizing their technical architectures, mathematical proofs, and dataset constraints (e.g., synthetic vs. real-world SIEM logs).

This focused screening ensured that the selected literature accurately maps the state-of-the-art in unsupervised dual-drift management, LLM-driven ontology correlation, and continuous-time temporal graph architectures, effectively highlighting the specific research gaps this thesis aims to resolve.

2.2 Background & Related Work

Based on the sources retrieved, the following section evaluates the state-of-the-art literature concerning the three primary operational pillars of modern Security Operations Centers (SOC): data ingestion, semantic enrichment, and temporal graph inference. For each domain, the operational challenges are defined, the limitations of current academic paradigms are analyzed, and the specific theoretical components selected for the proposed architecture are motivated.

2.2.1 SSL IDS Paradigms

a) The Operational Challenge

When designing an Intrusion Detection System (IDS) for a live Security Operations Center (SOC), researchers are immediately confronted with the paradox of label availability. Constraining the architectural search space to strictly Supervised Learning paradigms is practically unviable; SOCs process billions of highly heterogeneous events daily, rendering the continuous acquisition of perfectly labeled ground-truth data impossible. Conversely, relying entirely on Unsupervised Learning (pure anomaly detection) introduces unmanageable false-positive rates, as these algorithms inherently struggle to differentiate between benign network evolution and malicious actions without any contextual ground truth. Consequently, modern IDS architectures are strictly constrained to Semi-Supervised Learning (SSL) paradigms, which attempt to leverage a massive volume of unlabeled telemetry guided by a minuscule fraction of known threat indicators (Mvula, Branco, Jourdan, & Viktor, 2024).

b) Comprehensive Review of Existing Approaches

The SSL landscape for intrusion detection is broadly categorized into Inductive wrapper methods, unsupervised preprocessing, and transductive paradigms (Mvula et al., 2024; Zhu, 2005). Inductive wrapper approaches capitalize on the self-training paradigm to iteratively update training sets with high-confidence pseudo-labels. These include Single Classifier Self-Training, which frequently utilizes base learners such as Decision Trees (Andresini et al., 2021; Jaiswal, Manjunatha, Madhu, & Chidananda, 2016; Pachghare, Khatavkar, & Kulkarni, 2012; Wagh & Kolhe, 2014; Y. Wang, Yun, & Zhang, 2015). However, single-classifier approaches exhibit significant variance and frequently struggle to detect specific threat vectors, such as User-to-Root (U2R) attacks. To mitigate this, ensemble-based self-training frameworks have integrated diverse classifiers (e.g., Random Forest, Neural Networks, and k-NN) (Khonde & Ulagamuthalvi, 2019), enabling collective decision-making that enhances accuracy but significantly increases computational complexity due to the need for continuous feature selection via Variable Importance Indices. Other advanced self-training variants rely on Semi-Supervised SVMs (S3VM) (Haweliya & Nigam, 2014) to determine prediction confidence via hyperplane margins, or Semi-Supervised Deep Belief Networks (SS-DBN) (N. Shi, Yuan, Hernandez, Roy, & Esterline, 2018), which utilize Restricted Boltzmann Machines (RBM) to estimate confidence through node weights.

Beyond self-training, the introduction of Co-Training and active learning (Mao, Lee, Parikh, Chen, & Huang, 2009) has advanced IDS capabilities by allowing models to interact with an oracle to select the most informative data points for manual labeling. These frameworks leverage multiple views of the data, utilizing techniques such as d-transform-based confidence measurements or adapting the strategy to handle false alarms via the Two-Teachers-One-Student (2T1S) framework (Chiu, Lee, Chang, Luo, & Huang, 2010).

Tri-Training (W. Li, Meng, & Au, 2020; Lin, 2021; Viegas, Santin, Cogo, & Abreu, 2020; Y. Wang et al., 2015; H. Zhang & Li, 2020; L. Zhang, Du, & Zhang, 2019) extends the co-training idea to three learners. Crucially, these learners are trained on a single view rather than distinct views, mathematically eliminating the complex requirement of generating multiple views for network traffic. Whether used in conjunction with active learning, unsupervised feature extraction, or simply three distinct supervised classifiers, tri-training has emerged as one of the most promising wrapper approaches, consistently yielding precision rates between 90% and 99% on benchmark datasets like KDD99.

In addition to wrapper methods, Generative Models utilize probabilistic frameworks and advanced Deep Learning architectures to detect attacks. Approaches based on Mixture Models (MMs) (M. Zhang & Mei, 2012) address false positives by using a binary representation comprised of True Positive (TP) and False Positive (FP) components. Similarly, Gaussian Mixture Models (GMMs) (Gulati, Greenstadt, Dandekar, & Walsh, 2013) estimate probability density to compute the posterior probability of unlabeled samples, effectively distinguishing between benign and malicious distributions based on attack frequency. Advanced architectures such as Autoencoders (AEs), Variational Autoencoders (VAEs)—including variants like the Forked-VAE— (Lee, Kim, & Choi, 2021; Min, Yoo, Kim, Shin, & Shin, 2021; Mohammadi & Sabokrou, 2019; Zavrak & Iskefiyeli, 2020) and Adversarial AEs (AAEs) (Hara & Shiomoto, 2020) leverage reconstruction errors and latent variable classifications to detect network anomalies, even with minimal labeled data. Furthermore, Conditional GANs (CGANs) (Ezeme, Mahmoud, & Azim, 2020) have been employed to balance datasets by generating data distributions that align with predetermined profiles of normal and anomalous samples.

Semi-Supervised Clustering algorithms map intricate patterns suggestive of anomalies by guiding the clustering process with partial supervision. Constraints-based approaches, such as Pairwise Constrained (PC) K-Means and Fuzzy C-Means (FCM), inject supervised information via balance parameters to establish must-link and cannot-link constraints. Density-based clustering, like DBSCAN and Shared Nearest Neighbors (Zimba, Chen, Wang, & Chishimba, 2020), isolates hosts participating in Advanced Persistent Threats (APTs). Furthermore, hierarchical and metric-based techniques leverage Mahalanobis distance (Casas, D'Alconzo, Settanni, Fiadino, & Skopik, 2016) or external threat intelligence like VirusTotal ratings (Watkins et al., 2017) to extract pure clusters and concentrate on unknown patterns. Partial-label strategies (P. Yang et al., 2020) also guide these clustering mechanisms using a minimal amount of labeled data to minimize labeling effort.

Unsupervised Preprocessing remains a critical pipeline step to structure data prior to classification. Cluster-then-label paradigms, such as the Semi-Supervised Anomaly Detector (SSAD) (Yu, Guo, Lan, & Ban, 2009), divide network streams into time-based chunks for clustering, utilizing history information and attenuation to adapt to changing traffic patterns. Pre-training approaches deploy deep architectures like Deep Discriminative Belief Networks (DDBN) (Xian, 2020) with manifold learning regularization, or Cloud Grey Wolf Optimization (CGWO) (H. Yang & Zhou, 2018) for penalty parameter tuning. Feature extraction pipelines utilize n-gram representations for payload identification

(Angiulli, Argento, & Furfaro, 2015), multiscale residual temporal convolutional modules (Abdel-Basset, Hawash, Chakrabortty, & Ryan, 2021), and Genetic Algorithms (GA) (Varal & Wagh, 2018) to optimize feature selection and reduce dimensionality before classification.

Finally, Transductive Methods exploit the underlying structural patterns and relationships within network data. Graph-based approaches, such as the Spectral Graph Transducer (SGT) and Gaussian Fields Approach (GFA) (C. Chen, Yunchao Gong, & Yingjie Tian, 2008), construct graphs to uncover hidden patterns that distinguish normal behavior from anomalies. Specialized transductive structures include bipartite graphs connecting DNS server IPs to domains to identify DNS hijacking (Jia, Dong, Li, & Stokes, 2021), and Scalable Reputation Propagation (SRP) algorithms (L. Shi, Lin, Fang, & Zhai, 2015) that identify suspicious sources by propagating reputation scores through the network.

c) Limitations of Current Paradigms

While traditional Inductive SSL methods demonstrate high theoretical accuracy, they suffer from critical algorithmic brittleness when deployed against volatile, multi-stage Advanced Persistent Threats (APTs). Self-training models suffer from severe confirmation bias, frequently failing to detect minority classes such as User-to-Root (U2R) attacks. Co-training models require the assumption of independent and sufficient multi-view features, a condition rarely satisfied by raw network packets. While Tri-Training resolves some independence issues and achieves high precision on static datasets (e.g., KDD99), maintaining and updating three separate ensemble classifiers on high-speed streaming data introduces prohibitive computational overhead.

Furthermore, traditional preprocessing and clustering methods (like PC K-Means or SSAD) operate on flat, tabular representations of network traffic. By discarding the inherently relational nature of network communications (e.g., an infected host communicating with a lateral target), these models must rely entirely on localized feature distances, blinding them to the distributed, multi-hop operational chains characteristic of modern ransomware and APTs.

d) Motivating the Proposed Architecture

To overcome the confirmation bias of self-training and the structural blindness of flat clustering, the proposed architecture abandons traditional Inductive SSL in favor of an intrinsically semi-supervised, Transductive Graph Neural Network (GNN) paradigm.

Network telemetry is not tabular; it is fundamentally a graph, where IP addresses are nodes and communication flows are edges. GNNs mathematically exploit the principle of *label smoothness* across this topology—the transductive assumption that heavily connected nodes likely share the same state (benign or compromised). By utilizing continuous-time graph architectures, the system naturally propagates the semantic

weight of a single known threat (a sparse label) across the entire active network topology via message passing. This inherently non-parametric and discriminative approach allows the framework to accurately trace multi-stage lateral movement under extreme label scarcity, achieving the robustness of ensemble Tri-training without the crippling computational overhead of maintaining multiple discrete classifiers.

2.2.2 Ingestion & Unsupervised Drift Adaptation

a) The Operational Challenge

Deploying static machine learning models within live Security Operations Centers (SOCs) presents insurmountable operational challenges. At the ingestion layer, enterprise SIEMs process billions of security events daily in a single-pass streaming environment. This data is fundamentally hostile: it is high-dimensional, highly sparse, and characterized by an unbounded feature space containing simultaneously mix-typed features (Boolean flags, ordinal severities, and continuous byte counts) (Zhuo et al., 2025). Furthermore, SOC analysts are routinely overwhelmed by thousands of daily alerts with massive false-positive rates, inducing severe alert fatigue and making the continuous acquisition of perfectly labeled ground-truth data operationally impossible (Jalalvand et al., 2024). Compounding this label scarcity is the evolutionary nature of cyber threats. Adversaries continuously mutate their patterns, triggering complex concept drifts (changes in statistical properties of the target variable) and feature drifts (changes in feature relevance) (Shyaa et al., 2024). Consequently, architectures must be capable of tracking multiple drift typologies without relying on constant manual triage or perfectly labeled streams.

b) Comprehensive Review of Existing Approaches

To address the challenges of evolving data streams, the literature extensively explores concept drift detection and adaptation algorithms. Drift detection mechanisms are broadly categorized into Statistical Process Control (SPC), windowing techniques, and ensemble-based monitoring (Shyaa et al., 2024).

SPC methods assume that drift has occurred when a base learner's online error rate exceeds a statistical significance threshold. Prominent implementations include the Drift Detection Method (DDM) (Gama, Medas, Castillo, & Rodrigues, 2004), which tracks errors as a Bernoulli random variable, and the Early Drift Detection Method (EDDM) (Baena-Garcıa et al., 2006), which measures the distance between classification errors to better capture slow, gradual drifts. Other variants, such as the Reactive Drift Detection Method (RDDM) (Barros, Cabral, Gonçalves Jr, & Santos, 2017) and the Fast Hoeffding Drift Detection Method (FHDDM) (Pesaranghader & Viktor, 2016), improve global accuracy and timely detection but frequently require rigid parameter tuning (Shyaa et al., 2024).

Windowing techniques identify drift by analyzing data distributions across sliding timeframes. Adaptive Windowing (ADWIN) (Bifet & Gavalda, 2007) dynamically adjusts window sizes based on observed statistical

changes in real-time, while the Statistical Test of Equal Proportions (STEPD) (Nishida & Yamauchi, 2007) and the Wilcoxon Rank Sum Test Drift Detector (WSTD) (de Barros, Hidalgo, & de Lima Cabral, 2018) apply rigorous statistical tests to rapidly identify sudden changes. Additionally, margin-based detectors like MD3 (Sethi & Kantardzic, 2015) attempt to track density distribution changes in unlabeled samples to mitigate the reliance on immediate feedback.

Once drift is detected, systems must adapt. Ensemble learning frameworks combine diverse base learners to maintain accuracy across evolving concepts. Algorithms such as the Streaming Ensemble Algorithm (SEA) (Street & Kim, 2001), Accuracy Updated Ensemble (AUE) (Brzeziński & Stefanowski, 2011), and the Learn++ family (Polikar, Upda, Upda, & Honavar, 2001) dynamically adjust classifier weights or incrementally add new learners to accommodate incoming data. In specific IDS applications, researchers have deployed Adaptive Random Forests (ARF) (Seth, Chahal, & Singh, 2024) and modified Online Sequential Extreme Learning Machines (OS-ELM) (Shyaa et al., 2023) to handle nonstationary traffic. More recent continual learning frameworks integrate Continuous Kernel Learning (Y. Chen & Dai, 2024; J. Chen et al., 2024) or utilize streaming auto-encoders to isolate drift before triggering updates.

c) Limitations of Current Paradigms

Despite the breadth of these mechanisms, they fundamentally fail when subjected to real-SOC constraints. First, attempting to process the mix-typed features of raw network traffic through standard ensembles or rigid imputation models mathematically corrupts the data, inducing garbled gradients and rendering the models susceptible to adversarial self-poisoning.

More critically, traditional drift detection paradigms—spanning nearly all SPC methods (e.g., DDM, EDDM), ensemble triggers, and standard windowing (ADWIN)—are fundamentally flawed due to their strict reliance on classifier error rates (Shyaa et al., 2024). In a label-scarce SOC lacking ground-truth information about drift positions, calculating error rates to trigger an adaptation window is an operational paradox. Even when margin-based or unsupervised detectors identify a shift, state-of-the-art adaptation frameworks suffer from catastrophic forgetting or prohibitive computational overhead. Massive ensemble updates (e.g., AUE, SEA) scale poorly with high-dimensional data and require vast batches of newly labeled data to retrain incoming classifiers. These systems routinely fail to distinguish between benign network evolution (virtual drift) and malicious anomaly-driven concept drift, ultimately exacerbating false-positive rates and alarm fatigue.

d) Motivating the Proposed Architecture

To bypass these computational, dimensional, and label-dependent bottlenecks, the proposed architecture synthesizes three highly specialized components:

- **Dynamic Preprocessing**: It utilizes a Gaussian Copula module for dynamic online preprocessing. By isolating the marginal distributions of individual features from their complex multivariate dependencies, the copula mathematically standardizes the unbounded, mix-typed streams into a unified joint-Gaussian latent space. This eliminates garbled gradients and provides robust imputation without distorting the underlying network reality.

- **Label-Independent Detection**: To resolve the operational paradox of error-rate dependency and label scarcity, the architecture adopts ADWIN-U. This eliminates label dependency entirely by strictly monitoring unsupervised statistical distributions (e.g., skewness, kurtosis, and median absolute deviation) to detect structural data shifts prior to classification failure, strictly optimizing the Balanced Accuracy by the Amount of Requested Labeled Data (BAR) metric.

- **Efficient Adaptation**: Finally, to bypass catastrophic forgetting and the massive overhead of ensemble retraining, the architecture integrates Representation Enhancement for Concept Drift Adaptation (ReCDA). Rather than retraining the entire model or suffering from self-poisoning, ReCDA utilizes contrastive loss to force the model to learn a strictly drift-invariant representation. By mapping historical and unlabeled drift samples into an adjacent latent sub-space, it requires only a minuscule fraction of true labels to dynamically re-tune the decision boundary, preserving inference integrity against both concept and feature drift under severe label scarcity.

2.2.3 Semantic Enrichment & Ontology Mapping

a) The Operational Challenge

Elevating raw topological anomalies into actionable intelligence requires sophisticated semantic parsing of both localized SIEM logs and unstructured external Cyber Threat Intelligence (CTI). However, processing this data presents severe operational hurdles. The overwhelming volume of intelligence makes it difficult to distinguish genuine threats from noise. Furthermore, unstructured CTI text is plagued by complex semantics, non-unified formats, and overlapping entity relationships within multi-stage attack chains (Arazzi et al., 2025). Because cyber threats are highly dynamic, manual annotation to structure this data into usable threat intelligence is far too time-consuming and resource-intensive, demanding continuous efforts that cannot scale. Delaying this parsing is inherently dangerous due to the time-sensitive nature of threat intelligence, which rapidly renders outdated information obsolete.

b) Comprehensive Review of Existing Approaches

To automate parsing and construct Cybersecurity Knowledge Graphs (CKGs), the field has aggressively shifted towards Natural Language Processing (NLP) and Large Language Models (LLMs) (Arazzi et al., 2025). Foundational text processing has largely been superseded by domain-adapted language models (e.g., SecureBERT, CySecBERT) designed to capture semantic cybersecurity nuances.

For log parsing specifically, traditional syntax-based parsers are increasingly being replaced by LLM-driven frameworks. For example, *LogPrécis* leverages pre-trained language models to analyze Unix shell attack logs, mapping raw commands into intermediate representations that reflect attackers' objectives guided by the MITRE ATT&CK framework (Boffa et al., 2024). Similarly, models like *AECR* fine-tune lightweight LLMs (e.g., ChatGLM3) to automatically extract and categorize Attack Technique (AT) intelligence from unstructured reports (M. Chen et al., 2025).

To extract and structure this text into knowledge graphs, researchers deploy both pipeline and joint-extraction architectures alongside comprehensive cybersecurity ontologies. Frameworks like *Open-CyKG* utilize neural attention-based models to extract relation triples from APT reports, integrating them into a unified knowledge graph and using agglomerative clustering for disambiguation (Sarhan & Spruit, 2021). Modern ontology mapping frequently leverages highly structured, prompt-based LLM frameworks like *AttacKG+*, which employs LLMs to rewrite, parse, identify, and summarize raw text directly into MITRE TTPs (Y. Zhang et al., 2025), or standardizes relationships using the STIX 2.1 framework.

c) Limitations of Current Paradigms

Despite algorithmic advancements, existing pipelines suffer from critical generalization and integration failures. Human-crafted log parsing rules degrade rapidly on complex data. When deploying general LLMs to solve this, systems encounter massive network latency, huge computational overhead, and inconsistent outputs. As demonstrated by the *SecLLMHolmes* evaluation framework, even leading LLMs (e.g., GPT-4, PaLM2) struggle with high false-positive rates and flawed reasoning when evaluating complex security contexts (Ullah et al., 2024).

For CTI extraction, traditional pipeline approaches suffer from severe cascading error propagation, where downstream relation extraction fails due to initial entity recognition errors (Arazzi et al., 2025). To combat this, recent joint-entity relation models (e.g., *CyberRel* (Guo et al., 2021) and *CTI-JE* (Liu, Han, Zuo, Lv, & Guo, 2024)) attempt to learn entities and relations simultaneously. However, applying pre-trained models to the cybersecurity domain still introduces a semantic gap; standard 512-token input windows are fundamentally insufficient to model complete, multi-stage attack chains, leading to semantic fragmentation (Arazzi et al., 2025).

When querying these extracted graphs, standard Retrieval-Augmented Generation (RAG) struggles to adapt to specialized knowledge without expensive fine-tuning, often generating suboptimal contexts that lead to poor,

hallucinated LLM responses. Finally, existing attack graph constructions require extensive expertise to tune and predominantly generate isolated, ad-hoc provenance graphs lacking standardized semantic schemas. This semantic failure explicitly prevents the bridging of abstract global threat intelligence with localized temporal node behavior, rendering the graphs incompatible with external CTI sharing platforms.

d) Motivating the Proposed Architecture

To bypass LLM hallucination and the severe computational overhead of monolithic models, the proposed architecture bifurcates the incoming data stream, routing continuous numerical tensors entirely around the LLM.

To resolve semantic fragmentation and the cascading errors of traditional extraction pipelines, the framework abandons isolated clustering and standard RAG. Instead, it integrates Hierarchical Agglomerative Clustering (HAC) with an Ontology-Grounded RAG (OG-RAG) approach (Sharma, Kumar, & Li, 2025). By utilizing embedding-based semantic fusion prior to retrieval, the system dynamically resolves unclear terminology and overlapping entity relationships without the need for expensive fine-tuning. Consequently, the localized LLM retrieves a minimal, optimal set of hyperedges, forcing it to deduce Tactics, Techniques, and Procedures (TTPs) from a pre-disambiguated, factual hypergraph rather than uncontextualized noise.

Finally, to bridge the interoperability gap and securely synthesize multi-source data, the architecture queries localized MISP instances to inject STIX-compliant CTI directly into the structured *AttacKG+* schema (Y. Zhang et al., 2025). This strict separation of continuous behavioral micro-edges from discrete semantic macro-states grounds the LLM's deductions in a complete, unambiguous ontology, directly solving the challenge of integrating global CTI into localized temporal prediction.

2.2.4 Mitigation of Graph Structural Bottlenecks & Continuous Inference

a) The Operational Challenge

While Graph Neural Networks (GNNs) are theoretically the ideal paradigm for tracking the lateral movement of Advanced Persistent Threats (APTs), deploying them in live enterprise environments presents massive computational and structural challenges. Telemetry is defined by heterogeneous transient dynamics, varying time intervals, and highly evolving node features. Generality and scalability are of paramount importance; however, tracking long-term historical dependencies across billions of nodes frequently leads to critical computational bottlenecks. Furthermore, live network behavior is inherently stochastic. Relying on single, deterministic graph representations is insufficient for capturing the complex, diverse relationships inherent in multivariate time series. Models must dynamically impute missing structural information and quantify uncertainty without relying on excessive domain knowledge to define the rules of interaction.

b) Comprehensive Review of Existing Approaches

To overcome these structural and spatial challenges, foundational research relies heavily on Core GNN architectures, which are broadly categorized into spectral and spatial approaches designed to capture complex relational structures through message passing (Khemani, Patil, Kotecha, & Tanwar, 2024; Scarselli, Gori, Tsoi, Hagenbuchner, & Monfardini, 2008; J. Zhou et al., 2020). Graph Convolutional Networks (GCNs) are frequently utilized as backbones to capture spatial dependencies through neighborhood filters (Kipf & Welling, 2016a). To address missing information, architectures like FS-GNN ((Lei et al., 2025)) utilize GCN and Graph Attention Network (GAT) (Veličković et al., 2017) encoders alongside trainable multi-channel feature completion modules. Other frameworks, such as Euler (King & Huang, 2023) and GraphSAGE (Hamilton, Ying, & Leskovec, 2017), focus on link prediction through scalable max-pooling aggregation and distributed model-agnostic frameworks to mitigate the neighborhood explosion problem inherent in large graphs (Hamilton et al., 2017). Furthermore, to address the insufficiency of single-graph representations, application-specific models like PMGC (J. Chen, Feng, & Wirjanto, 2025) and MTAD-GAT (Zhao et al., 2020) integrate long-term static graphs with a series of short-term dynamic graphs, while frameworks like GDN (Deng & Hooi, 2021) create global static graphs utilizing one-hot embeddings and sparsity constraints.

To extend these spatial models into the temporal domain, Spatial-Temporal GNNs (STGNNs) have emerged to capture graphs where both the topological structure and node inputs evolve over time (Cao et al., 2020). The literature traditionally augments spatial layers with sequence-to-sequence learning or recurrent units. This has led to the development of Temporal GCNs (TGCN), parameter-evolving networks like EvolveGCN (Pareja et al., 2020) (where the GCN parameters are passed through a recurrent unit), and temporal self-attention layers seen in DySAT (Sankar, Wu, Gou, Zhang, & Yang, 2020). Finally, to address the stochastic nature of network traffic and quantify uncertainty, several frameworks abandon deterministic embeddings in favor of probabilistic extensions, utilizing Variational Graph Autoencoders (VGAE) (Kipf & Welling, 2016b) or Graph2Gauss (G2G) (Bojchevski & Günnemann, 2017) to represent each node as a multivariate Gaussian distribution.

c) Limitations of Current Paradigms

Despite this vast array of architectural innovations, scaling these models to live, zero-trust SOCs exposes critical computational and temporal flaws. As traditional spatial GNNs deepen to capture extended multi-hop APT lateral movement, they encounter the fatal challenge of "over-smoothing"—a phenomenon where repeated message-passing causes the embeddings of diverse nodes to converge into indistinguishable representations, effectively erasing the discriminative features of malicious nodes (K. Zhou et al., 2020; Rong, Huang, Xu, & Huang, 2019). Complex attention mechanisms, while theoretically robust, frequently suffer from parameter bloat and latency. For example, empirical studies show that GAT models, despite containing significantly more parameters

than standard GCNs, routinely underperform in anomaly classification quality (Khemani et al., 2024). Similarly, temporal attention models like DySAT experience severe quadratic computational decay on large-scale graphs, and parameter-evolving networks like EvolveGCN struggle to achieve adequate anomaly detection at an enterprise scale.

More critically, foundational TGCNs augmented with sequence-to-sequence learning successfully encode temporal dynamics but fundamentally fail to capture underlying causal processes. Even when models utilize causal discovery (e.g., CARGO's (Math & Lienhart, 2025) one-shot causal graphs or CTGCN's divide-and-conquer techniques), they rely on Discrete-Time Dynamic Graphs (DTDG). DTDGs slice continuous network telemetry into arbitrary temporal snapshots (Δt). This discrete splicing inherently creates dangerous inter-event blind spots; attackers executing rapid reconnaissance probes or slow, irregularly timed exfiltrations easily evade detection by operating across or between these arbitrary boundaries. Conversely, attempting to maintain global graphs in active memory triggers adjacency matrix explosion, while aggressively windowing them causes catastrophic forgetting and false-positive "cold-start" anomalies.

d) Motivating the Proposed Architecture

To scale securely without sacrificing temporal integrity, losing causal dependencies, or suffering from the depth-induced over-smoothing prevalent in traditional multi-layer GNNs, the proposed architecture abandons discrete snapshotting entirely. It implements an episodic Continuous-Time TGNN (CT-TGNN) grounded in Graph Neural Ordinary Differential Equations (Graph Neural ODEs). Unlike traditional architectures that assume a fixed graph topology or rely on discrete message-passing layers—which are highly inappropriate for volatile network security and prone to information degradation—Graph Neural ODEs extend continuous dynamics to graph domains through coupled differential equations over node embeddings.

By utilizing localized Time-Aware Batch-Normalized Ordinary Differential Equations (TA-BN-ODE) augmented with multi-scale temporal decomposition—featuring parallel ODE branches with learned time constants spanning eight orders of magnitude—the system achieves unified modeling of continuous system dynamics and discrete event occurrences. This continuous-depth paradigm inherently bypasses the over-smoothing limitation of standard GNNs. Furthermore, the ODE mechanism smoothly rolls the localized graph dynamics forward to the exact timestamp of the incoming alert ($h_i(t_0^-)$), eliminating temporal blind spots while achieving a massive 60-90% parameter reduction.

To completely bypass adjacency matrix explosion, this continuous inference is executed episodically. The active ODE computation is decoupled from historical memory via a localized Global State Registry. Upon alert ingestion, the system fetches a node's historically preserved continuous state, processes the topological update, and unconditionally writes the finalized embedding ($h_i(t_0^+)$) back to the registry. This guarantees unbroken long-term memory, naturally neutralizes cold-start anomalies, and provides a continuous mathematical trajectory that the downstream decoder translates into highly accurate incident predictions at sub-50 millisecond latencies.

3 Proposed Architecture

This chapter introduces the core contribution of this thesis: a highly scalable, self-supervised, and predictive Intrusion Detection System (IDS) pipeline designed for dynamic zero-trust enterprise networks. To overcome the structural bottlenecks and high false-positive rates of legacy monolithic models, the proposed framework is structured as a decoupled, three-phase architecture that explicitly bifurcates the continuous physics of network traffic from the discrete semantic reasoning of attacker intent.

The chapter is organized sequentially to track the lifecycle of a security event through the pipeline. Section 3.1 details Phase 1, which manages the robust ingestion and structural normalization of chaotic SIEM logs while autonomously mitigating concept drift via unsupervised statistical trackers and representation enhancement. Section 3.2 outlines Phase 2, demonstrating how raw alerts are contextualized, disambiguated via an ontology-grounded factual hypergraph, and mathematically pruned to decouple continuous edge features from discrete Large Language Model (LLM) state summaries. Finally, Section 3.3 presents Phase 3, the predictive engine of the framework. This section formalizes how an Episodic Continuous-Time Temporal Graph Neural Network (CT-TGNN) integrates these bifurcated payloads to evaluate temporal volatility, yielding highly explainable, sample-level alerts designed for seamless SOC human-AI teaming.

3.1 Phase 1: Ingestion, Normalization & Unsupervised Drift Adaptation

3.1.1 Gaussian Copula (GC) Module: Data Normalization and Imputation

To address the structural bottlenecks of raw SIEM logs detailed in Chapter 2.2.2, the pipeline routes all incoming alerts through a Gaussian Copula (GC) module (Zhuo et al., 2025), denoted as *Gaussian Copula* in Figure 1. This component is explicitly designed to satisfy Design Requirement 1 (**DR-1**) by enabling the robust ingestion of incomplete, heterogeneous streams, while simultaneously addressing the feature drift mandate of **DR-2**.

a) Standardizing Mix-Typed Data

To resolve the mixed-type data instability discussed in Chapter 2.2.2, the GC module acts as a continuous mathematical transformer. Directly addressing **DR-1**, it projects the heterogeneous features ingested (spanning continuous, ordinal, and Boolean variables) into a cohesive latent space. By mathematically smoothing discrete features into continuous intervals and standardizing continuous features to a baseline distribution, the module proactively prevents the oscillating gradients and biased model initializations that disrupt downstream predictive engines.

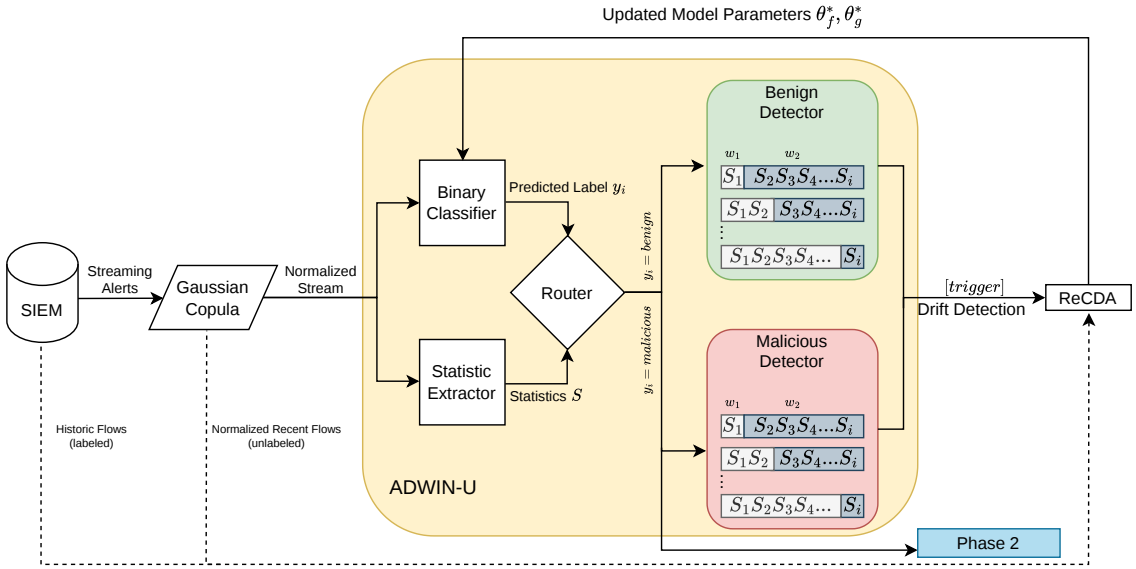


Figure 1: Phase 1 Architecture for continuous drift detection & mitigation

b) Reconstructing Missing Log Features

Rather than relying on naive zero-padding, which introduces artificial bias, the GC module fulfills the robust imputation mandate of **DR-1**. By mapping all telemetry into a shared latent space, the module captures the underlying generative correlations between log fields. When an incomplete log is ingested, it analyzes the observed features and statistically infers the unobserved attributes. This guarantees the seamless generation of fully populated feature vectors without discarding valuable, fragmented telemetry.

c) Dynamic Adaptation to Evolving Feature Spaces

To satisfy the autonomous adaptation requirements of **DR-2**, the pipeline must handle the continuous evolution of feature spaces (Feature Drift) without relying on continuous manual labeling or offline retraining. The GC module achieves this via an online learning mechanism operating over a sliding buffer of recent logs. As data flows through the system, it monitors sequence structures; upon detecting new variables or localized distributional shifts, it dynamically updates its internal correlation baselines. This on-the-fly adjustment allows the pipeline to gracefully assimilate new structural variables, preventing feature drift from triggering downstream false anomalies.

3.1.2 ADWIN-U: Unsupervised Concept Drift Detection

To directly address the semantic Concept Drift mandate of **DR-2**, the pipeline evaluates the normalized stream without relying on the impossible standard of continuous ground-truth labels. As illustrated by the yellow box in Figure 1, the normalized, continuous stream from the GC module is routed into the ADWIN-U framework (Assis &

Souza, 2025). This component functions autonomously under severe label scarcity, requiring labeled data only *after* a drift is confirmed to execute a model update.

a) Statistical Extraction and Baseline Prediction

As depicted in Figure 1, the normalized stream entering ADWIN-U immediately bifurcates. It is simultaneously processed by a Binary Classifier, which yields a provisional predicted label (y_i), and a Statistic Extractor, which isolates higher-order statistical moments (denoted as Statistics S) from the data sequence. Rather than tracking classifier error rates, ADWIN-U detects drift by monitoring the geometric shape of the data. Based on empirical performance in multi-detector configurations, the Statistic Extractor specifically isolates Skewness and Kurtosis, as these measures are highly sensitive to underlying distribution shifts.

b) The Two-Detector Routing Mechanism

Central to the ADWIN-U architecture is the Router (Figure 1), which enables a two-detector configuration that significantly outperforms single-detector approaches in both accuracy and efficiency. The Router evaluates the predicted label (y_i) and directs the corresponding statistics (S) into one of two independent tracking pipelines: the Benign Detector or the Malicious Detector. As shown in the diagram, if y_i is flagged as malicious, the system immediately forwards the alert to Phase 2 for semantic extraction, while concurrently logging its statistical footprint in the Malicious Detector. This segregation ensures that a sudden spike in novel benign traffic is not misinterpreted as a malicious attack.

c) Compressed Windowing and Unsupervised Detection

To remain computationally lightweight, the detectors do not maintain exhaustive historical arrays. Instead, they utilize an exponential histogram technique to maintain a compressed "bucket" representation of the data over time. As illustrated inside the Benign and Malicious Detectors in Figure 1, the system continuously attempts to partition its historical statistic arrays ($S_1, S_2, \ldots, S_i$) into two adjacent sub-windows (w_1 and w_2). If the absolute difference between the mean statistical values of w_1 and w_2 exceeds a dynamic, statistically defined boundary, the system confirms that the underlying concept of the data has fundamentally changed.

d) Triggering the Retraining Protocol

When a semantic shift is confirmed by either detector, a $[trigger]$ edge is executed (Figure 1). This signal initiates the Representation Enhancement (ReCDA) module and subsequently resets all ADWIN-U tracking windows.

3.1.3 Representation Enhancement for Concept Drift Adaptation (ReCDA)

As illustrated in Figure 1, when a semantic shift is confirmed by either the Benign or Malicious detector within ADWIN-U, the $[trigger]$ edge is executed, initiating the Representation Enhancement (ReCDA) module (S. Yang et al., 2024). Directly addressing the label scarcity mandate of **DR-2**, ReCDA bypasses the need for massive manual labeling during a live concept drift event. As depicted by the dashed data flows in Figure 1, the module operates by fetching a constrained pool of *Historic Flows (labeled)* directly from the SIEM and pairing them with the *Normalized Recent Flows (unlabeled)* provided by the Gaussian Copula.

a) Drift-Aware Perturbation: Synthesizing the Data Streams

ReCDA must mathematically bridge the distribution gap between the historical baselines and the newly detected traffic behaviors. It achieves this through a probabilistic perturbation mechanism. The module systematically merges the labeled historic flows with the unlabeled recent drift flows by swapping specific structural features between the two sets based on a defined perturbation rate. This creates synthesized, 'drift-aware' samples - a hybrid traffic view that introduces realistic variations, forcing the model to perceive new feature changes without destroying the underlying semantic meaning of the original network traffic.

b) Representation Alignment via Contrastive Learning

Once these hybrid samples are synthesized, the module must ensure that the underlying feature extractor does not break when exposed to these new network variations. Both the original historic samples and their newly synthesized perturbed counterparts are fed into a shared encoder network.

Using contrastive learning techniques, the model forces these paired representations to map into closely adjacent latent sub-spaces, while pushing uncorrelated traffic flows away. By successfully narrowing the distribution gap in the latent space, the feature extractor avoids model collapse. It learns to recognize the drifted, unlabeled recent flows as variations of known behaviors, enabling it to process unseen attack categories without manual intervention, strictly satisfying the unsupervised adaptation requirement of **DR-2**.

c) Instructive Sampling at the Decision Boundary

While the feature extractor is now robust against drift, the actual classifier's decision boundary must be re-tuned to output accurate predictions. To avoid overfitting to old data or requiring thousands of new labels, ReCDA utilizes a highly targeted Instructive Sampling tactic on the fetched historic data pool.

The module evaluates the historical dataset and selects a tiny fraction of instructive samples. It specifically targets data points that sit exactly at the fuzzy decision boundary, i.e. samples that appear highly dissimilar to the

recent drift in the raw feature space, but highly similar in the newly aligned latent space. By extracting only these critical anchors, the module gathers the richest category-discriminative knowledge possible, drastically reducing the volume of historical labels required for retraining and adhering to the lightweight operational constraints of **DR-2**.

d) Constrained Classifier Tuning and Autonomous Adaptation Loop

Finally, the Phase 1 baseline classifier is re-tuned using these instructive anchor samples. To ensure that the feature extractor does not forget the drift-invariant properties it learned during representation alignment, the update is governed by a Consistency Regularization penalty. This acts as a mathematical tether, allowing the decision boundary to shift to accommodate the new attack behaviors while firmly anchoring the underlying feature representations.

As depicted in the feedback loop of Figure 1, this process generates the updated model parameters (θ_f^*, θ_g^*), which are dynamically fed back to update the Binary Classifier. This continuous, self-supervised loop directly solves the Concept Drift challenge, ensuring that only highly reliable, normalized data and high-confidence malicious triggers are forwarded to Phase 2.

A core architectural priority of this pipeline is minimizing the cost of human intervention. Standard detectors often achieve high accuracy simply by spamming the system with constant retraining requests. ReCDA optimizes the Balanced Accuracy by the Amount of Requested Labeled Data (BAR) metric (proposed in (Assis & Souza, 2025)). By relying purely on unsupervised statistical shifts to initiate the $[trigger]$, it only requests the absolute minimum number of expensive human-verified labels required to retrain the model, making it highly viable for enterprise deployment despite the challenges of high-dimensional log data.

3.2 Phase 2: Ontology-Grounded Correlation & State Extraction

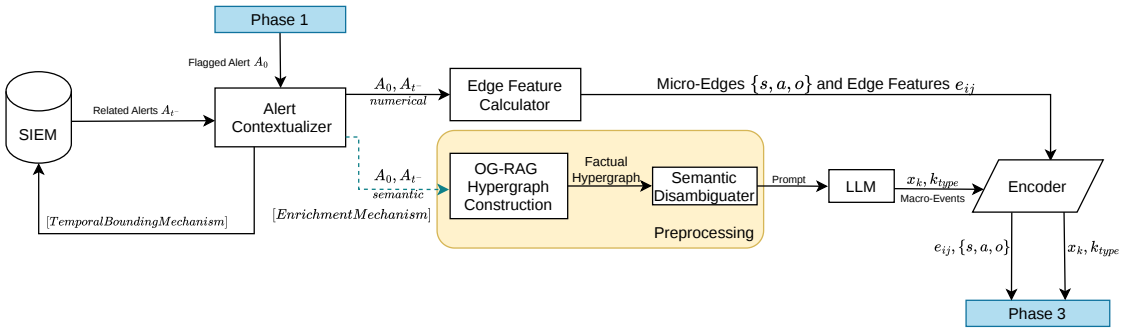


Figure 2: Phase 2 Architecture mapping the transformation of raw SIEM alerts

An isolated alert, even when accurately classified as malicious by Phase 1, lacks the tactical narrative required to detect a broader Advanced Persistent Threat (APT). To directly address the contextualization and enrichment mandates of Design Requirement 3 (**DR-3**), Phase 2 transforms raw anomalous alerts into a structured, fully

contextualized baseline. As illustrated in Figure 2, this phase operates through a bifurcated architecture that rigorously separates deterministic topological extraction from semantic LLM processing.

3.2.1 Temporal Bounding and Alert Contextualization

When a flagged alert (A_0) is received from the Phase 1 Malicious Detector, it is immediately routed to the Alert Contextualizer (Figure 2). Because an APT executes over prolonged temporal horizons, analyzing A_0 in isolation is insufficient. To fulfill **DR-3**, the Contextualizer invokes a Temporal Bounding Mechanism to query the SIEM and dynamically fetch related historical alerts (A_{t-}). This securely grounds the threat actor's current actions within the network's recent historical context.

To manage this historical fetching efficiently and avoid computational bottlenecks, the pipeline relies on an `Episodic_ID` metadata field appended to the SIEM logs. This identifier tracks exactly which analytical episodes a specific historical alert has already contributed to. This mechanism efficiently prevents redundant double-calculations within a single temporal window, while still permitting the same critical historical alert to be deliberately re-fetched and reintegrated with new context later, allowing it to mathematically participate in multiple evolving stages of the attack chain.

a) Data Bifurcation and Edge Feature Calculation

Once the enriched contextual pool (A_0, A_{t-}) is assembled, the Alert Contextualizer executes a strict data bifurcation, dividing the payload into numerical and semantic streams (represented by the diverging solid and dashed lines in Figure 2).

The numerical features in A_0, A_{t-} are routed directly to the Edge Feature Calculator. This deterministic component serves two critical functions:

1. **Topological Extraction:** It deterministically parses the raw numerical data to extract the topological micro-edges $\{s, a, o\}$ (Subject, Action, Object) alongside continuous edge features e_{ij} (e.g., packet inter-arrival times, byte distributions).

2. **Temporal Normalization:** To prevent point process intensity errors in the downstream Continuous-Time Neural ODEs (Phase 3), the calculator encodes relative time differences ($\Delta t = t_0 - t_{historic}$) rather than feeding raw, absolute timestamps.

These extracted continuous micro-edges ($\{s, a, o\}$ and e_{ij}) completely bypass the Preprocessing and LLM modules, flowing directly to the final Encoder. This architectural decision is vital: by forcing numerical data to bypass the Large Language Model entirely, the pipeline optimizes token limits and mathematically neutralizes the LLM's primary vector for statistical hallucination, ensuring that downstream predictive models receive pristine topological data.

3.2.2 Semantic Enrichment and Preprocessing

To fulfill the external intelligence integration and ambiguity resolution mandates of **DR-3**, the pipeline must ground volatile local telemetry within a globally recognized attack narrative. As depicted by the dashed data flow in Figure 2, the semantic part of A_0, A_{t-} is passed through an Enrichment Mechanism (e.g., querying a localized MISP instance). The Mechanism is not explicitly defined to allow flexibility for users to integrate their preexisting enrichment mechanism here. This mechanism should seamlessly integrate external Cyber Threat Intelligence (CTI), creating a comprehensive textual payload that is subsequently routed into the Preprocessing module.

Within the Preprocessing module (highlighted in yellow in Figure 2), the unstructured semantic data is mathematically organized and disambiguated through two highly specialized sequential processes.

a) OG-RAG Hypergraph Construction

Traditional pairwise graphs often fail to capture the multi-dimensional relationships inherent in complex cyber-attacks. To overcome this, the module utilizes an Ontology-Grounded Retrieval-Augmented Generation (OG-RAG) methodology to construct a Factual Hypergraph (Sharma et al., 2025). This process bridges unstructured text and structured CKG ontologies:

1. **Ontology Mapping and Flattening:** An initial local LLM call parses the enriched text chunks, extracting domain-specific values and mapping them to a predefined cybersecurity ontology (e.g., STIX/AttacKG+). To ensure computational efficiency, the resulting nested factual blocks are flattened via a recursive algorithm, transforming them into self-contained direct relationships without losing their semantic integrity.
2. **Multi-Dimensional Hypergraph Creation:** The flattened blocks are used to construct the hypergraph. *Hypernodes* represent key-value pairs (where the key is a concatenated entity and attribute), while *Hyperedges* represent complete factual blocks. Unlike traditional graph edges that only connect two nodes, a single hyperedge can encapsulate multiple nodes simultaneously, accurately modeling complex, multi-entity threat behaviors in a structured format.

b) Semantic Disambiguation via SecureBERT and HAC

While the hypergraph structures the data, raw threat intelligence is notoriously plagued by semantic ambiguities, such as synonymous threat actor names or polysemous attack techniques. To generate the unambiguous CKG baseline demanded by **DR-3**, the hypergraph is evaluated by the Semantic Disambiguater (Figure 2).

This component addresses ambiguity mathematically by pairing a domain-specific encoder with a clustering algorithm:

- **Contextual Embedding:** A cybersecurity-fine-tuned model, like SecureBERT_Plus, processes the hypernodes to generate deep, context-aware vector embeddings of the extracted entities and attributes.
- **Mathematical Resolution:** These embeddings are subsequently processed using Hierarchical Agglomerative Clustering (HAC). HAC calculates the cosine distances between the contextual embeddings and dynamically merges highly similar nodes.

By clustering synonymous nodes and resolving entity redundancies entirely in the vector space, the Semantic Disambiguater mathematically guarantees the structural purity of the Factual Hypergraph. This pristine, optimized hypergraph is then converted into a compact, formatted dictionary prompt, providing the ultimate LLM with a strictly factual, minimally noisy context window.

3.2.3 Factual Deduction and Semantic-Preserving Pruning

The mathematically disambiguated output from the Preprocessing module is formatted into a strict, highly compact prompt and passed to the local Large Language Model (LLM). Driven by the contextualization and structural mandates of **DR-3**, the LLM evaluates this pristine Factual Hypergraph to construct a fully unambiguous Cyber Knowledge Graph (CKG) baseline. Specifically, the LLM deduces the precise MITRE TTP label (k_{type}) and generates an expanded Macro-Event state summary (x_k), which captures critical contextual shifts such as permission changes and toolset deployments. This final semantic payload strictly adheres to the AttacKG+ ontology (Y. Zhang et al., 2025), guaranteeing that the output accurately reflects complex, multi-stage threat behaviors.

It is critical to note that traditional generative AI workflows often require deterministic fallback mechanisms to account for stochastic failure rates, such as hallucinating non-existent TTPs or outputting malformed data structures. However, this architecture explicitly circumvents such failure modes through its utilization of the OG-RAG methodology. Because the LLM does not generate responses unconditionally but is rigidly constrained by the mathematically disambiguated Factual Hypergraph and the strict parameters of the STIX/AttacKG+ ontology, the probability of out-of-schema hallucinations is structurally neutralized. The retrieval-augmented nature of the prompt forces the LLM to deduce from a verified, factually restricted context window, inherently acting as its own programmatic safeguard and rendering a secondary deterministic fallback mechanism mathematically redundant.

While this semantic extraction perfectly satisfies **DR-3**, routing the entirety of the historical graph to Phase 3 can introduce downstream computational bottlenecks. Therefore, the pipeline can accommodate an optional, Semantic-Preserving Pruning mechanism prior to Phase 3 transmission.

Standard graph pruning is often blind, strictly preferring lower-order hops and indiscriminately discarding high-hop pathways to reduce redundancy. However in cybersecurity, erasing a multi-hop path destroys the AttacKG+ narrative (e.g., erasing the middle step between *Initial Access* and *Lateral Movement*). Because the pipeline utilizes the LLM to deduce MITRE TTPs *before* any mathematical compression occurs, the system can leverage those

semantic labels to prune the graph intelligently.

If applied, this pruning mechanism operates under three deterministic rules to protect the structural integrity of the attack chain:

- A_0 **Immunity:** The current flagged alert (A_0) and its immediate 1-hop connections are absolutely never pruned. They serve as the unalterable structural anchor for the current state evaluation.
- **Intra-Phase Pruning (Noise Reduction):** If the historical batch (A_{t-}) contains massive amounts of repetitive multi-hop loops that all fall under the exact same MITRE tactical phase (e.g., an attacker executing fifty redundant internal reconnaissance scans), the algorithm collapses these redundant loops into optimized first-order links. This vastly reduces the number of edges sent to Phase 3 without losing the semantic fact that reconnaissance occurred.
- **Cross-Phase Preservation (Narrative Protection):** If a multi-hop sequence crosses different tactical phases (e.g., a node involved in *Privilege Escalation* connecting to a node involved in *Exfiltration*), the pruning algorithm is bypassed. The script strictly preserves these specific bridging edges, ensuring that the full multi-hop narrative of the overarching APT campaign successfully reaches the Continuous-Time Graph Neural Network (CT-TGNN) in Phase 3.

3.2.4 The Encoder: Mathematical Transformation for CT-TGNN

While the semantic outputs generated by the LLM and the Semantic-Preserving Pruning algorithms perfectly satisfy the contextual narrative mandates of **DR-3**, they remain in string format (e.g., "T1110" or "admin"). As illustrated by the final *Encoder* block in Figure 2, this textual representation cannot be ingested directly by Phase 3. Continuous-Time Graph Neural Networks (CT-TGNN) and Neural Ordinary Differential Equations (Neural ODEs) strictly require dense, continuous numerical vectors and mathematical arrays to function; passing raw text will inherently cause the mathematical integration to fail.

To bridge this gap and provide the mathematically precise baseline required for subsequent prediction (**DR-3**), the Encoder performs a rigorous deterministic transformation, converting the entire bifurcated payload into compatible tensors.

Transforming the Topological Micro-Edges ($\{s, a, o\}$ and Timestamps) The raw structural topology is mapped into a continuous latent space:

- **Nodes (Subject and Object):** Textual entities are mapped to unique integer IDs. When a novel entity is discovered, it is initialized in the graph using a learned node embedding, establishing its continuous state vector $h_i(t) \in R^d$.

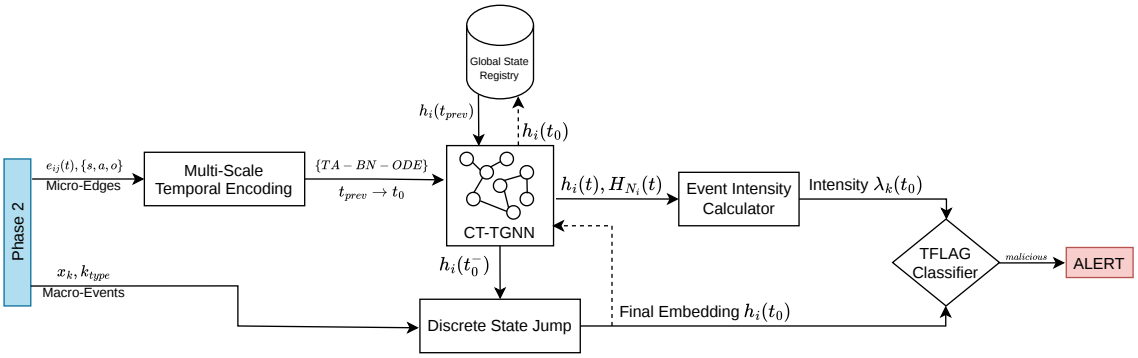


Figure 3: The Phase 3 Architecture routing continuous and discrete payloads through the CT-TGNN to Alert

- **Edges (Actions):** The relational action strings (e.g., "authenticates") are transformed into dense vectors via a Categorical Encoding matrix.
- **Temporal Normalization:** Timestamps cannot be processed as standard date-time strings. They are strictly normalized into numerical floats representing time differentials (Δt). This is critical because the downstream Neural ODE explicitly relies on the numerical time variable to project continuous node dynamics forward.

Transforming the Discrete Macro-Events (x_k and k_{type}) The semantic payload is mathematically compressed to act as a discrete state jump, defined as $u_\psi(x_k, k_{type})$, which will shock the CT-TGNN in Phase 3:

- **The TTP Label (k_{type}):** The textual MITRE tactical stage deduced by the LLM is mapped to a predefined integer class $k_{type} \in \{1, \ldots, K\}$, distinctly categorizing the event type.
- **The State Summary (x_k):** The expanded text-based state summary is compressed into a dense event feature vector $x_k \in R^{d_x}$. Categorical metadata (such as utilized tools or specific user permissions) is passed through a categorical encoder to become dense continuous vectors. Additionally, any associated timing and flow statistics ($\mu_{\Delta t}, \sigma_{\Delta t}, CV_{\Delta t}$) are strictly standardized via z-score normalization (zero mean, unit variance) before being concatenated into the final x_k array.

Through this final encoding step, Phase 2 completely satisfies **DR-3**, outputting a highly contextualized, fully disambiguated, and rigorously mathematical baseline ready for continuous-time predictive modeling.

3.3 Phase 3: Episodic Continuous-Time Node Classification

The final phase of the architecture serves as the predictive engine. Its primary objective is to evaluate the contextualized payloads from Phase 2 to predict and isolate Advanced Persistent Threats (APTs). In designing this phase, the architecture must explicitly solve two critical enterprise challenges: mitigating the memory explosion

inherent in dynamic graph tracking (**DR-4**) and providing highly explainable, non-binary alerts for Human-AI teaming (**DR-5**). As illustrated in Figure 3, this is achieved through an episodic, continuous-time framework coupled with a self-supervised deviation classifier.

3.3.1 Episodic Initialization via Global State Registry

As depicted on the left side of Figure 3, Phase 3 receives the bifurcated data streams from Phase 2: the structural Micro-Edges ($e_{ij}(t), \{s, a, o\}$) and the semantic Macro-Events (x_k, k_{type}).

To directly address the structural bottlenecks identified in **DR-4**, this architecture completely abandons the perpetual maintenance of a massive, global adjacency matrix in active memory. Traditional static Graph Neural Networks (GNNs) suffer from catastrophic forgetting and memory explosion when attempting to track streaming network data. Instead, this pipeline utilizes an episodic update mechanism.

When a new processing episode begins, the system queries the *Global State Registry* (top of Figure 3) to retrieve the historically preserved continuous node embeddings, denoted as the state vector $h_i(t_{prev})$. By fetching only the isolated sub-graph states required for the current temporal window, the pipeline mathematically sidesteps memory explosion while maintaining continuous-time integrity.

Furthermore, in a dynamic zero-trust network, the sudden appearance of novel entities (e.g., newly spun-up microservices or anomalous external IPs) routinely triggers "cold-start" mathematical instabilities in standard models. Rather than initializing these unseen nodes with random, meaningless vectors, the architecture securely initializes them using the dense contextual semantics dynamically extracted in Phase 2. This guarantees immediate mathematical stability and perfectly grounds the node's starting state in verifiable intelligence before continuous integration begins.

3.3.2 Multi-Scale Temporal Encoding and Continuous ODE Dynamics

As illustrated in Figure 3, the continuous micro-edge payload ($e_{ij}(t), \{s, a, o\}$) generated in Phase 2 bypasses the discrete semantic logic and flows directly into the *Multi-Scale Temporal Encoding* module.

This component addresses a fundamental structural reality of cybersecurity: attack patterns execute at vastly different velocities. A microsecond timing exploit and a slow, hours-long reconnaissance campaign cannot be accurately captured by a model evaluating a single, fixed temporal window. To overcome this, the encoding module decomposes the continuous temporal analysis into parallel branches operating simultaneously across four distinct temporal lenses, defined by fixed time constants (τ_s):

- $\tau_1 = 10^{-6}$ seconds: Captures microsecond-level timing attacks.
- $\tau_2 = 10^{-3}$ seconds: Captures millisecond-level burst exploits.

- $\tau_3 = 1$ second: Captures typical request-response patterns.
- $\tau_4 = 3600$ seconds: Captures slow, hour-scale reconnaissance campaigns.

Rather than treating all time scales equally, the module dynamically weights these branches using a learned attention mechanism ($\alpha_s(t) \in [0, 1]$). By dynamically shifting these attention weights based on the velocity of the incoming traffic, the framework intrinsically satisfies **DR-5**. It provides a transparent, tactically grounded explanation of the attack's temporal velocity, allowing SOC analysts to differentiate between a rapid automated exploit and a methodical human-driven campaign.

Once temporally encoded, this continuous stream is fed into the central *CT-TGNN* component (Figure 3) (Anaedevha et al., n.d.). To strictly satisfy **DR-4**'s mandate against maintaining a massive, perpetual global graph in active memory, the CT-TGNN avoids static adjacency matrices. Instead, it utilizes an adaptive, localized Neural Ordinary Differential Equation solver (the $\{TA - BN - ODE\}$ engine).

The solver initializes its mathematical clock at the historically fetched state (t_{prev}) and chronologically integrates the network's spatial topology step-by-step from $t_{prev} \rightarrow t_0$. The continuous evolution of the node's hidden state is governed by the underlying ODE:

$$\frac{dh_i(t)}{dt} = f_\theta(h_i(t), H_{N_i}(t), A(t))$$

Because Neural ODEs strictly integrate forward in time to preserve chronological integrity, historical alerts that logically precede the current graph state cannot be processed sequentially. Instead, Phase 2 inherently forces them to be injected as discrete memory updates. Furthermore, to prevent the $\{TA - BN - ODE\}$ from losing its baseline understanding of historical attack patterns as it integrates forward, catastrophic forgetting is intrinsically mitigated through Elastic Weight Consolidation (EWC) embedded directly within the ODE solver.

3.3.3 Volatility Calculation and Discrete State Jump

As the localized ODE solver within the CT-TGNN rolls the spatial topology forward in continuous time, it constantly monitors the evolving dynamics of the sub-graph. As depicted in Figure 3, the CT-TGNN continuously outputs the evolving node states ($h_i(t)$) and their surrounding neighborhood contexts ($H_{N_i}(t)$) to the *Event Intensity Calculator*.

This module utilizes a Deep Spatio-Temporal Point Process to compute a pre-emptive volatility metric, denoted as *Intensity* $\lambda_k(t_0)$. Directly satisfying the mandate of **DR-5** to move beyond uninterpretable binary classifications, this intensity metric provides SOC analysts with a mathematically sound, non-binary prioritization score, quantifying the exact escalating volatility of the network environment immediately prior to the alert.

Exactly at the time of the alert (t_0), the continuous integration pauses. As shown in the lower pathways of Figure 3, the CT-TGNN passes this strictly pre-alert topological state, denoted as $h_i(t_0^-)$, downward into the *Discrete State Jump* module.

Simultaneously, this module receives the discrete *Macro-Events* payload from Phase 2, which comprises the categorically encoded AttacKG+ State Summary (x_k) and the deduced MITRE TTP label (k_{type}). The injection of these semantic variables triggers an instantaneous mathematical shock to the ODE. This mechanism fundamentally fuses purely semantic Cyber Threat Intelligence (CTI) into the continuous mathematical state, generating the *Final Embedding* $h_i(t_0)$.

By strictly linking the topological anomalies back to explicit MITRE ATT&CK tactics (k_{type}) during this discrete jump, the architecture structurally guarantees the tactically grounded, explainable baseline required for seamless Human-AI Teaming (**DR-5**).

3.3.4 Explainable Classification and Unconditional Write-Back

To finalize the threat evaluation, the dual outputs of the preceding modules—the pre-emptive *Intensity* $\lambda_k(t_0)$ and the semantically infused *Final Embedding* $h_i(t_0)$—are routed into the *TFLAG Classifier* (Figure 3).

Rather than relying on uninterpretable, black-box binary thresholds, the TFLAG Classifier (W. Jiang, Chai, Liu, Wang, & Zhang, 2025) evaluates anomalies based on extreme topological and semantic deviations from the historical baseline. If the classifier detects a severe deviation and triggers the *malicious* edge (resulting in an *ALERT*), the SOC analyst is not handed a blind true/false flag.

Instead, to fully bridge the gap between high-dimensional mathematical anomaly detection and practical SOC operations (**DR-5**), a final semantic translation step is proposed. A local LLM, employing Chain-of-Thought (CoT) reasoning, ingests the anomalous outputs—specifically the pre-alert intensity ($\lambda_k(t_0)$) and the enriched AttacKG+ context (x_k, k_{type}) injected during the discrete jump. The CoT framework logically synthesizes these topological and semantic variables to generate a human-readable, tactically grounded narrative of the incident. This ensures the SOC analyst receives an organically derived, actionable alert that explicitly justifies the statistical magnitude of the anomaly using verifiable MITRE ATT&CK context, facilitating seamless Human-AI teaming.

Finally, to resolve the memory explosion bottleneck mandated by **DR-4**, the architecture employs an unconditional write-back mechanism. Regardless of whether the classification outcome is benign or malicious, the updated mathematical state is sent back to the *Global State Registry*, as indicated by the upward dashed arrow $h_i(t_0)$ in Figure 3.

This critically preserves the evolutionary mathematical memory of the entity within the lightweight registry. Consequently, the localized, episodic graph maintained by the CT-TGNN can be safely deleted from active memory at the end of the temporal window, guaranteeing highly scalable, continuous-time network tracking without computational bloat.

4 Mathematical and Algorithmic Framework

This chapter formally defines the mathematical and theoretical underpinnings of the Intrusion Detection System (IDS) pipeline introduced in Chapter 3. The chapter is divided into three primary sections, corresponding directly to the three operational phases of the architecture.

4.1 Phase 1: Ingestion, Normalization, and Drift Management

As outlined in Chapter 3, Phase 1 is responsible for the ingestion and standardization of raw, chaotic SIEM logs. Because enterprise networks are highly dynamic, this phase must mathematically mitigate two distinct types of data degradation: structural feature drift (handled via a Gaussian Copula and Expectation-Maximization) and semantic concept drift (managed via ADWIN-U detection and ReCDA). This section formally defines the mathematical framework enabling these transformations.

4.1.1 Structural Normalization: The Gaussian Copula Framework

To standardize mixed-type features and handle structural drift, a Gaussian Copula (GC) maps heterogeneous data $x \in R^d$ into a standard multivariate normal distribution. This mapping is defined by an element-wise monotone function g, such that:

$$x = g(z) \tag{4.1}$$

where z is the latent variable following $z \sim \mathcal{N}(0, \Sigma)$, and Σ represents the correlation matrix. The mixed-type logs are partitioned into continuous and discrete variables $x := (x^c, x^d)$, allowing the unified latent vector to be defined as:

$$z := f^{-1}(x) = (f^{-1}(x^c), \text{cutoff}^{-1}(x^d)) \tag{4.2}$$

where f^{-1} and cutoff^{-1} represent the respective inverse marginal transformation functions for continuous and discrete features.

Missing fields are dynamically imputed using conditional expectations within this latent space. Given observed features x^o and missing features x^m, the expected missing latent data is calculated via the correlation sub-matrices:

$$\hat{z}^m = \Sigma_{mo}\Sigma_{oo}^{-1}\hat{z}^o \tag{4.3}$$

where $\hat{z}^m$ is the imputed latent vector for missing fields, $\hat{z}^o$ represents the corresponding observed latent values, and Σ_{mo} and Σ_{oo} are the correlation sub-matrices mapping the missing to observed variables.

To continuously adapt to structural drift, the correlation matrix Σ is optimized without full retraining using an online Expectation-Maximization (EM) algorithm. The update relies on a harmonic mean rule to ensure valid covariance bounds and prevent catastrophic forgetting:

$$\Sigma^{(t)} = E((1-\alpha)\Sigma^{(t-1)} + \alpha \cdot \hat{\Sigma}) \tag{4.4}$$

where $\alpha \in (0, 1]$ is a decaying step size, $\hat{\Sigma}$ is the newly estimated correlation matrix from the current batch, and $E(\cdot)$ is a mathematical normalization operator ensuring the resulting matrix remains a valid correlation matrix.

4.1.2 Concept Drift Detection (ADWIN-U)

While the GC normalizes structural feature drift, the system also accounts for semantic concept drift. ADWIN-U statistically monitors the normalized data stream for severe distributional shifts. The two detectors maintained by ADWIN-U actively attempt to partition their arrays of S into two sub-windows, w_1 and w_2. Drift is confirmed when the absolute difference between the means of these two sub-windows exceeds a dynamic statistical bound ϵ (derived from the Hoeffding inequality):

$$|\mu_{w_1} - \mu_{w_2}| \geq \epsilon$$

where μ_{w_1} and μ_{w_2} are the mean values of the respective sub-windows, and ϵ dictates the required confidence threshold. When this threshold is breached, the $[trigger]$ edge is executed, initiating the retraining module and subsequently resetting the ADWIN-U windows.

4.1.3 Representation Enhancement for Drift Adaptation (ReCDA)

Triggered by ADWIN-U, the Representation Contrastive Drift Adaptation (ReCDA) framework re-tunes the model using a self-supervised process, requiring only a fraction of labeled historical data. This process relies on two core mathematical steps:

a) Drift-Aware Perturbation and Alignment

Rather than adding arbitrary noise, ReCDA explicitly merges a historical sample x_i^h with a randomly sampled recent drift sample x_i^r. A mask vector $m \in \{0, 1\}^d$ is generated from a Bernoulli distribution with a perturbation rate σ. The perturbed sample $\tilde{x}_i$ is calculated using the Hadamard product ($\odot$):

$$\tilde{x}_i = m \odot x_i^r + (1 - m) \odot x_i^h \tag{4.5}$$

The original sample x_i^h and perturbed sample $\tilde{x}_i$ are then mapped to latent vectors z_i and $\tilde{z}_i$ via a shared encoder. ReCDA forces these representations into an adjacent latent sub-space by minimizing a contrastive loss function over the batch size $|\mathcal{B}|$:

$$\mathcal{L}_{ra}(x_i) = -\log \frac{\exp(z_i \cdot \tilde{z}_i/\tau)}{\sum_{j=1}^{|\mathcal{B}|} \exp(z_i \cdot \tilde{z}_j/\tau)} \tag{4.6}$$

where τ is a tunable temperature hyperparameter that controls the penalty strength for hard negative samples. This optimization guarantees that the historical sample and its drift-aware equivalent remain proximal in the latent space, creating a drift-invariant feature extractor.

b) Weakly-Supervised Classifier Tuning

Once the latent space is unified, the classifier's decision boundary must be mathematically retuned using an instructive matrix G. The model measures the cosine similarity (sim) between the historical data and the clustering centroids $\bar{x}_j^r$ of the recent dataset to identify boundary samples:

$$G_{i,j} = \frac{sim(x_i^h, \bar{x}_j^r \mid \bar{y}_j)}{sim(f(x_i^h), f(\bar{x}_j^r) \mid \bar{y}_j) + \epsilon} \tag{4.7}$$

where $sim(\cdot, \cdot)$ denotes the cosine similarity function, $f(\cdot)$ represents the feature extractor, $\bar{y}_j$ represents the pseudo-label of the centroid, and ϵ is a conservatively small constant (e.g., $\epsilon = 10^{-8}$) added to the denominator strictly to prevent division by zero. The top-δ historical samples with the highest scores in G are selected as anchors, providing rich category-discriminative knowledge at the decision boundary.

Finally, the model retunes the classifier using a standard Cross-Entropy loss ($\mathcal{L}_{ce}$). To ensure the feature extractor retains its newly learned drift-invariant properties, a Consistency Regularization penalty ($\mathcal{L}_{cr}$) is applied using the original and updated encoder parameters (θ_f and θ_f^*):

$$\mathcal{L}_{cr}(x_i) = ||f(x_i \mid \theta_f) - f(x_i \mid \theta_f^*)||^2 \tag{4.8}$$

The final mathematical objective for retuning the decision boundary is thus defined as:

$$\min_{\theta_g, \theta_f} [\mathcal{L}_{ce}(x_i, y_i \mid \theta_g, \theta_f) + \lambda \mathcal{L}_{cr}(x_i \mid \theta_f)] \tag{4.9}$$

where θ_g represents the learnable parameters of the classifier, y_i represents the true labels of the historical anchor samples, and λ is a scalar hyperparameter governing the weight of the consistency regularization term.

4.2 Phase 2: Ontology-Grounded Correlation and State Extraction

Following the structural normalization and drift mitigation of Phase 1, Phase 2 is responsible for translating the standardized log stream into a bifurcated architecture. To prevent computational bottlenecks and token waste in the Local LLM, this phase completely decouples the numerical network statistics from the semantic reasoning. This section mathematically grounds the ontology used to map semantic features into a Discrete Macro-Event Payload, and defines the deterministic Edge Feature Calculator responsible for the Continuous Payload.

4.2.1 Graph Ontology and State Summary Formalization

To establish a rigorous relational structure, the pipeline imposes the STIX 2.1 framework onto an AttacKG+ structure, blending micro-observations from the SIEM with macro-intelligence from MISP.

The ontology is categorized into four primary domains:

- **Category A: Threat Agents (The "Who"):** Captures macro-level ThreatActors (e.g., APT29) and micro-level SystemActors (e.g., SystemProcess, UserCredential).
- **Category B: Attack Vectors (The "How"):** Defines the methodologies, encompassing Malware, Tools, and specific Vulnerabilities (CVEs).
- **Category C: Infrastructure and Targets (The "Where"):** Maps macro-level Indicators (e.g., IP reputation) and VictimOrgs against micro-level NetworkHosts, Sockets, and Files.
- **Category D: Relational Attributes (The Edges):** Defines the topological links $\{s, a, o\}$. Crucially, these edges are embedded with the continuous feature vector e_{ij}. A formal edge representation is structured as:

$$[\text{SystemActor}] \xrightarrow{\text{communicatesWith } \{e_{ij}\}} [\text{NetworkHost}] \tag{4.10}$$

a) Enriched State Summary (x_k)

The original AttacKG+ State Summary lacks the multidimensional context required for the Phase 3 Continuous-Time Temporal Graph Neural Network (CT-TGNN). Therefore, the LLM-derived State Summary x_k is enriched into a highly dense semantic vector encompassing four distinct contexts:

1. **Behavioral Semantics:** Captures the attack narrative, including acquired permission states, targeted file collections, identified toolsets, and exploited CVEs.
2. **Defensive Context:** Details the security posture by identifying internal mechanisms that have been bypassed or disabled (e.g., firewall evasion).
3. **Topological Routing and Reputation:** While routing metadata (IPs/Hashes) directs the discrete state jump in Phase 3, this segment also injects numeric indicator trust scores derived from MISP, geographic origins, and severity confidence levels.
4. **Threat Attribution:** Contextualizes the event by linking it to broader ongoing campaigns or specific Intrusion Sets (e.g., Lazarus Group).

4.2.2 Continuous Alert Bifurcation and the Edge Feature Calculator

While the LLM handles the semantic formulation of x_k, the physical and numerical attributes of the alert within the time window $(t_{\text{prev}}, t_0]$ are deterministically extracted to form the continuous micro-edge feature vector $e_{ij}(t) \in R^{d_e}$. The Edge Feature Calculator computes rigorous mathematical proofs of network behavior to drive the continuous Neural ODE (TA-BN-ODE) in Phase 3.

The feature vector $e_{ij}(t)$ comprises the following specific calculations:

a) Temporal and Timing Features

To differentiate between automated exploits (e.g., C2 beaconing) and human-driven reconnaissance, the calculator evaluates inter-arrival time (Δt) statistics. Along with the mean ($\mu_{\Delta t}$), standard deviation ($\sigma_{\Delta t}$), minimum, and maximum, the system calculates the Coefficient of Variation:

$$CV_{\Delta t} = \frac{\sigma_{\Delta t}}{\mu_{\Delta t}} \tag{4.11}$$

where $\sigma_{\Delta t}$ is the standard deviation and $\mu_{\Delta t}$ is the mean of inter-arrival times. A low $CV_{\Delta t}$ mathematically penalizes the edge by indicating highly rigid, automated periodic behavior.

b) Packet Size Distributions and Directionality

For encrypted traffic where deep packet inspection is impossible, size metadata is leveraged. Beyond standard aggregations (mean, standard deviation, histograms), a directionality ratio is computed to detect anomalies such as data exfiltration:

$$r_{\text{bytes}} = \frac{\sum_{\text{up}} s_i}{\sum_{\text{down}} s_i} \tag{4.12}$$

where s_i represents the size of the i-th packet, and the ratio divides the total upstream payload by the downstream payload. Sudden spikes in r_{bytes} signal large outbound data transfers.

c) Normalization and Privacy Transformation

Before the continuous payload $e_{ij}(t)$ (typically $d_e \in [50, 100]$ dimensions) is routed to Phase 3, it undergoes a final transformation to ensure mathematical stability within the Neural ODE. Every feature is subjected to Z-Score normalization, enforcing a zero mean and unit variance.

Furthermore, if the intelligence crosses zero-trust service boundaries, the calculator can apply an optional (ϵ, δ)-differential privacy mechanism. Calibrated Gaussian noise is mathematically injected into sensitive features, satisfying the condition:

$$P[\mathcal{M}(D) \in S] \leq e^{\epsilon} P[\mathcal{M}(D') \in S] + \delta \tag{4.13}$$

where $\mathcal{M}$ is the randomized mechanism, D and D' are adjacent datasets differing by only one record, S represents any subset of potential outputs, ϵ bounds the maximum privacy loss (distinguishability), and δ allows a small probability of catastrophic privacy failure. This preserves topological integrity while anonymizing specific host vulnerabilities.

4.3 Phase 3: Episodic Continuous-Time Node Classification

The final phase of the pipeline evaluates the bifurcated threat payload using an Episodic Continuous-Time Temporal Graph Neural Network (CT-TGNN). In this graph ontology, nodes represent the microservices or entities deployed within the environment, and edges represent the communication channels between them.

This architecture relies on the spatial-temporal graph topology of the CT-TGNN framework to map deterministic interactions between subjects and objects. However, to guarantee gradient stability and sub-100ms processing speed under enterprise workloads, the temporal ODE integration and point-process mechanisms are rigidly governed by the theorems established in the CT-TGNN authors 2026 peer reviewed framework in (Anaedevha, Trofimov, & Borodachev, 2026).

The architecture formally models intrusion detection as a "continuous-discrete hybrid dynamical system." The mathematical treatment of this hybrid system is explicitly split into continuous micro-edge processing and discrete point process events.

4.3.1 Continuous Micro-Edge Processing and State Evolution

For streaming edges representing encrypted communications, the processing follows continuous mathematical dynamics. The node states $h_i(t)$ encode vital intelligence—such as recent communication patterns, authentication statuses, and historical threat indicators.

When the deterministic Phase 2 Edge Feature Calculator streams a new continuous payload $e_{ij}(t)$, the node states evolve continuously according to a coupled ordinary differential equation (ODE):

$$\frac{dh_i(t)}{dt} = f_\theta(h_i(t), H_{N_i}(t), A(t), t) \tag{4.14}$$

where N_i is the set of incoming neighbors, $H_{N_i}(t)$ represents the aggregate state of those neighbors, $A(t)$ is the temporal adjacency matrix encoding edge existence, and t is the continuous time variable.

The learnable vector field f_θ executes spatial aggregation over the graph:

$$f_\theta(h_i(t), H_{N_i}(t), A(t), t) = \sigma\left(W_{\text{self}}(t)h_i + \sum_{j \in N_i} \frac{\alpha_{ij}(t)}{|N_i|} W_{\text{neigh}}(t)h_j\right) \tag{4.15}$$

where $W_{\text{self}}(t)$ and $W_{\text{neigh}}(t)$ are time-dependent weight matrices, $\alpha_{ij}(t)$ denotes attention coefficients weighing neighbor contributions based on the edge features $e_{ij}(t)$, $|N_i|$ is the neighborhood size for normalization, and $\sigma(\cdot)$ is an Exponential Linear Unit (ELU) activation ensuring continuous differentiability for ODE integration.

a) Multi-Scale Temporal Decomposition

To capture patterns across vastly different frequencies—from rapid, microsecond burst exploits to slow, month-long advanced persistent threats (APTs)—the continuous dynamics are split into parallel branches using multi-scale

temporal decomposition:

$$f_\theta(h,t) = \sum_{s=1}^{S} \alpha_s(t) f_{\theta_s}\left(h, \frac{t}{\tau_s}\right) \tag{4.16}$$

where S is the total number of distinct time scales. Validated by (Anaedevha et al., 2026), the system utilizes $S = 4$ scales defined as $\tau_s \in \{10^{-6}, 10^{-3}, 1, 3600\}$ seconds. The time t is scaled by τ_s, and the learned attention weights $\alpha_s(t)$ determine which frequency branch dominates the current prediction based on the node's temporal context.

The continuous state is then integrated to predict future topology using adaptive Runge-Kutta solvers (e.g., Dormand-Prince) with dynamic step-size control:

$$h_i(t_1) = h_i(t_0) + \int_{t_0}^{t_1} f_\theta(h_i(\tau), H_{N_i}(\tau), A(\tau), \tau) d\tau \tag{4.17}$$

b) Gradient Stability via TA-BN (Theorem 1)

A known vulnerability of continuous-depth Neural ODEs in real-time traffic is gradient explosion during continuous backpropagation (adjoint computation). To resolve this, the architecture applies Temporal Adaptive Batch Normalization (TA-BN), where normalization parameters are strictly tied to the integration time t:

$$\text{TA-BN}(z,t) = \gamma(t) \odot \frac{z - \mu(t)}{\sqrt{\sigma^2(t) + \epsilon}} + \beta(t) \tag{4.18}$$

where z is the pre-normalized activation, $\mu(t)$ and $\sigma^2(t)$ are the continuous-time batch mean and variance, $\gamma(t)$ and $\beta(t)$ are time-dependent affine scaling and shifting parameters, and ϵ is a conservatively small constant (e.g., $\epsilon = 10^{-5}$) added for numerical stability. Relying on **Theorem 1 (Adjoint Gradient Stability for TA-BN-ODE)** from (Anaedevha et al., 2026), this formulation mathematically guarantees that as long as the time-dependent parameters are bounded, the adjoint state gradients will not explode, ensuring the continuous graph node embeddings evolve smoothly.

4.3.2 Point Process Events and Discrete State Jumps

While the micro-edges continuously roll the physical state forward, the occurrence of specific security events (semantic payloads from Phase 2) triggers instantaneous mathematical updates.

a) Event Intensity and Scalability (Lemma 2)

The system continuously computes the event intensity (occurrence rate) to predict temporally when an attack will occur:

$$\lambda_k(t) = g_\phi(h_i(t), H_{N_i}(t)) \tag{4.19}$$

where $\lambda_k(t)$ is the event intensity of type k, and g_ϕ is a learnable softmax activation function depending on the current node state $h_i(t)$ and the aggregated neighborhood state $H_{N_i}(t)$.

Evaluating the survival integral of these point processes typically scales quadratically $\mathcal{O}(n^2)$, causing severe computational bottlenecks. To achieve enterprise-grade scalability, this thesis leverages **Lemma 2 (Quadrature + Log-Barrier Survival Approximation)** from (Anaedevha et al., 2026). By applying a logarithmic barrier penalty at equispaced nodes rather than computing the exact integral, the computational complexity is mathematically reduced to $\mathcal{O}(n^{3/2})$.

b) Bidirectional Coupling and Discrete Update

When an alert triggered by the Phase 2 Local LLM dictates a definitive malicious or benign semantic payload, a discrete state jump occurs at time t_k:

$$h_i(t_k^+) = h_i(t_k^-) + u_\psi(x_k, k_{\text{type}}) \tag{4.20}$$

where $h_i(t_k^-)$ is the state solved by the ODE just before the event, $h_i(t_k^+)$ is the updated state immediately following the event, x_k contains the Phase 2 semantic features (State Summary), k_{type} represents the specific MITRE event type, and u_ψ is a learned discrete update function.

4.3.3 Constraint and Loss Formulation

To alert SOC analysts, the framework evaluates policies against the predicted states. A constraint function flags policy violations based on communication at time t:

$$p(v_i, v_j, t) = I\left[c_{\text{policy}}^T[h_i(t)||h_j(t)] \geq \tau_{\text{policy}}\right] \tag{4.21}$$

where $p(v_i, v_j, t)$ evaluates the communication edge between nodes v_i and v_j, c_{policy} is a learnable policy vector applied to the concatenated embeddings $[h_i(t)||h_j(t)]$, τ_{policy} is the detection threshold, and $I[\cdot]$ is an indicator function outputting a boolean $\{0, 1\}$.

Finally, all temporal parameters, continuous weight matrices, and point process parameters are jointly optimized through a composite loss function:

$$L(\Theta) = L_{\text{node}} + \lambda_{\text{event}} L_{\text{event}} + \lambda_{\text{reg}} L_{\text{reg}} \tag{4.22}$$

where Θ represents the complete set of learnable model parameters, L_{node} represents the classification accuracy of predicting the MITRE $\{s, a, o\}$ triplets, L_{event} contains the log-likelihood training the model to predict attack occurrence rates, L_{reg} applies structural regularization penalties to prevent overfitting, and λ_{event}, λ_{reg} act as scalar tuning weights governing the relative influence of each respective loss component.

5 Theoretical Evaluation & Architectural Analysis

Having established the mathematical and structural foundations in the preceding chapters, it is necessary to rigorously evaluate the architecture's viability prior to empirical deployment. This chapter critically assesses the framework across three primary dimensions: operational feasibility, computational scalability, and theoretical efficacy against real-world threat vectors. This chapter conducts a comprehensive feasibility analysis in Section 5.1, then Section 5.2 establishes strict mathematical bounds on computational complexity and memory usage. Finally, Section 5.3 synthesizes these structural proofs through a logical dry-run of the SolarWinds supply chain compromise. By mapping this highly sophisticated Advanced Persistent Threat (APT) to the pipeline, the section demonstrates exactly how the architecture processes, predicts, and explains complex cyberattacks across their entire lifecycle.

5.1 Feasibility Analysis and Architectural Dependencies

To validate the feasibility of deploying the proposed architecture within an enterprise environment, we must evaluate the computational nature, dependencies, and training requirements of each component. This analysis confirms that the architecture balances rigorous statistical mathematics and deep learning with low-overhead deterministic scripting, ensuring it can operate in high-speed streaming environments without catastrophic hardware failure.

5.1.1 Summary of Architectural Dependencies

The proposed pipeline is fundamentally hybrid, relying on distinct software paradigms rather than a monolithic machine learning framework. The overarching dependencies required to build and integrate this architecture are:

- **Deep Learning Frameworks:** Core neural network components (ReCDA, SecureBERT_Plus, CT-TGNN, TFLAG) require PyTorch or TensorFlow.
- **Scientific & Mathematical Libraries:** Extensive use of `NumPy`, `SciPy`, and `Pandas` is necessary for the Gaussian Copula matrix operations, ADWIN-U incremental tracking, and the Edge Feature Calculator's flow aggregations.
- **Graph Processing Ecosystem:** Libraries such as `NetworkX` or `PyTorch Geometric` are strictly required to instantiate the chronological behavior graphs and execute spatial aggregations.
- **Differential Equation Solvers:** The continuous temporal integration in Phase 3 requires specialized libraries like `torchdiffeq`, specifically utilizing the Dormand-Prince (`dopri5`) adaptive solver.
- **High-Speed Data Infrastructure:** An in-memory datastore (e.g., Redis or Memcached) is mandatory for the Global State Registry to handle sub-millisecond querying without I/O bottlenecks.

- **External APIs and Parsers:** The system requires SIEM API connectors, Threat Intelligence (MISP) APIs, and network packet parsers (e.g., `pyshark` or Zeek logs) to extract raw metadata.
- **Large Language Models (LLMs):** High-capacity instruction-tuned LLMs (e.g., GPT-4o for Phase 2, Meta-Llama-3.1-8B-Instruct for Phase 3) operate the zero-shot reasoning frameworks.

5.1.2 Component Dependency Breakdown

a) Phase 1: Ingestion and Drift Management

- **Gaussian Copula (GC) & Online EM:** This is a deterministic mathematical script. It requires linear algebra libraries for matrix inversion and calculating conditional expectations, alongside statistical functions for Cumulative Distribution Functions (CDFs). It uses a sliding window buffer and requires no traditional neural network training, relying instead on an iterative Online Expectation-Maximization loop to update the correlation matrix.
- **ADWIN-U:** An unsupervised algorithmic detector. To prevent memory explosion, it must implement an exponential histogram data structure, bounding space complexity to $\mathcal{O}(\log W)$. It requires incremental trackers to calculate skewness and kurtosis on the fly and a classification model wrapper to route statistics based on predicted class labels. It requires zero active training.
- **ReCDA:** This is a deep learning model requiring active neural network training. It depends on PyTorch/TensorFlow, a perturbation module to generate random Bernoulli mask vectors, and a lightweight clustering algorithm to find drift data centroids. It requires batches of both unlabeled drift traffic and historically labeled data. Training involves self-supervised contrastive loss and weakly-supervised fine-tuning via cross-entropy and consistency regularization.

b) Phase 2: Ontology-Grounded Correlation

The majority of Phase 2 is highly feasible as it requires zero model fine-tuning or active training loops.

- **Alert Contextualizer:** A deterministic script requiring read access to the Global State Registry, a SIEM API connector for the Time-Bounded Fetch, and a CTI Enrichment API connector for retrieving reputation scores.
- **Edge Feature Calculator:** Requires data processing libraries to compute temporal metrics, a TLS parser for unencrypted handshake metadata, and a Z-score normalization function.
- **OG-RAG Construction:** Operates zero-shot. It requires a predefined domain-specific ontology (AttacKG+), a structured LLM bounded to 8192 tokens, the OG-RAG Flattening Algorithm to prevent nested data loss, and an Adaptive Parsing Cache to save API calls.

- **Semantic Disambiguation (SecureBERT_Plus + HAC):** Requires a pre-trained SecureBERT_Plus domain transformer enforced with a strict sequence limit of $N \leq 512$ tokens. It relies on clustering libraries to execute Hierarchical Agglomerative Clustering using cosine distances. Neither requires active training.
- **LLM Deductive Reasoning:** Requires a deductive prompting framework, a strictly enforced JSON schema, and the hypergraph output from HAC. It operates zero-shot via in-context learning.
- **Pruning & Final Encoder:** A deterministic script requiring a graph library, scripted pruning logic to collapse redundant intra-phase loops, and a payload router to bifurcate the data into discrete and continuous streams.

c) Phase 3: Continuous Integration and Classification

Phase 3 isolates its heavy deep learning operations from its data storage and explainability layers.

- **Global State Registry:** Requires a low-latency, in-memory datastore (e.g., Redis or Memcached) to handle millions of concurrent sub-millisecond read/write queries without I/O bottlenecks. It requires zero training.
- **Episodic CT-TGNN:** A heavily training-dependent deep learning model. It requires PyTorch/TensorFlow, `torchdiffeq` for the Dormand-Prince ODE solver, and `PyTorch Geometric` for multi-scale temporal convolutions. Training strictly requires the Adjoint Method to prevent memory explosion during backpropagation, and the Deep Point Process relies on Log-Barrier Optimization to mathematically reduce complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^{3/2})$.
- **TFLAG (Deviation Network):** Developed in Python using PyTorch, this decoder strictly requires upstream inputs: a Temporal Graph Encoder and dense continuous node embeddings (z_{src} and z_{dst}). Internally, it requires an MLP decoder, an anomaly scoring module, and an adjustable dynamic memory repository to log normal event distributions. Training requires a dual loss function architecture (Cross-Entropy Loss and Deviation Loss) and necessitates benign baseline data to establish reference scores via self-supervised labeling.
- **Chain-of-Thought (CoT) LLM:** Operates zero-shot. It depends on a high-capacity instruction-tuned LLM constrained by deterministic settings (Temperature = 0.2, Top-p = 0.9, max 256 tokens). It requires a fixed CoT prompting template enforcing a structured reasoning sequence and necessitates the anonymization of raw payloads before inference to prevent data leakage.

5.1.3 Operational Feasibility within an Enterprise SOC

Implementing this tri-phase architecture represents a bold, paradigm-shifting proposal for modern Security Operations Centers (SOCs). Traditional SOC environments are heavily indexed on reactive, static rule-based SIEMs (e.g., legacy Splunk or QRadar deployments) that require relatively low computational overhead but demand massive human capital to manually triage false positives. Conversely, the proposed architecture shifts the burden from human analysts to a highly complex, predictive algorithmic pipeline.

Evaluating this from an operational SOC perspective reveals several critical implementation hurdles, offset by transformative strategic advantages.

a) Implementation Challenges and Overhead

- **Infrastructure and Compute Costs:** The architecture cannot run on standard commodity hardware. Phase 3 (CT-TGNN) and Phase 2 (LLM Reasoning) require dedicated GPU clusters to process tensor operations and transformer attention mechanisms in real-time. Furthermore, the Global State Registry necessitates an enterprise-grade, high-capacity in-memory datastore cluster to prevent I/O bottlenecks during the continuous state updates.
- **MLSecOps and Continuous Training:** Unlike plug-and-play legacy systems, this pipeline requires a mature Machine Learning Security Operations (MLSecOps) framework. While Phase 2 relies heavily on zero-shot inference, components like ReCDA and the Episodic CT-TGNN demand active neural network training. The SOC must maintain data pipelines capable of feeding benign baselines for self-supervised learning (e.g., TFLAG's reference scores) and executing continuous learning loops without suffering from catastrophic forgetting.

5.1.4 Rigorous Operational Feasibility and Cost-Benefit Analysis

While the proposed architecture is mathematically rigorous and theoretically capable of predictive incident detection, deploying it within an enterprise Security Operations Center (SOC) introduces immense computational, financial, and MLSecOps overhead. A rigorous, impartial feasibility analysis requires evaluating the pipeline not just on its detection efficacy, but on its throughput limits, hardware dependencies, and maintenance costs under real-world enterprise data volumes.

a) Computational Bottlenecks and MLSecOps Overhead

Enterprise networks routinely generate upwards of 10,000 Events Per Second (EPS). While Phase 1 (Gaussian Copula and ADWIN-U) can handle this velocity using lightweight statistical trackers, the downstream deep learning

components pose severe operational risks:

- **LLM Inference Latency (Phase 2):** The most prohibitive bottleneck in this pipeline is the reliance on a Local LLM (e.g., Llama-3-8B or GPT-4 class models) for zero-shot deductive reasoning and state extraction. Evaluating a factual hypergraph with an 8192-token context window is computationally massive. Even with the Temporal Bounding mechanism and pruning, executing LLM inference on high-frequency anomalous clusters introduces latency. If relying on commercial APIs, this translates to huge financial costs; if hosted locally, it requires a dedicated, highly available multi-GPU cluster strictly for Phase 2.
- **Continuous Neural Network Tuning (Phase 1 - ReCDA):** Implementing ReCDA requires active, continuous neural network training (contrastive and cross-entropy loss) on streaming data. In a live SOC, online training can be unstable. Without oversight, the model risks poisoning if the unlabeled drift batch contains stealthy adversarial noise.
- **ODE Solver Throughput Constraints (Phase 3):** The Dormand-Prince ODE solver requires multiple function evaluations per temporal interval. Under a Distributed Denial of Service (DDoS) attack or an aggressive network scan, the sheer volume of discrete point-process interruptions (the t_0 state jumps) will force the ODE solver to constantly halt and recalculate, severely degrading the $\mathcal{O}(V)$ spatial aggregation throughput.

b) Conclusion on Operational Viability

Ultimately, the tri-phase continuous-time architecture represents the theoretical ceiling of predictive intrusion detection. However, its unadulterated deployment is currently only feasible for highly mature, well-resourced organizations possessing dedicated GPU infrastructure and specialized MLSecOps teams. For standard enterprise SOCs, implementing the core structural flow while utilizing the aforementioned substitutions provides a pragmatic balance between revolutionary predictive capabilities and operational stability.

5.2 Computational Complexity and Memory Bounds

To guarantee that the decoupled architecture can withstand the high-throughput requirements of an enterprise zero-trust environment, it is necessary to establish strict theoretical bounds on both time and space complexity across all three phases. By defining explicit architectural constraints, we demonstrate that the pipeline prevents the memory explosions and processing bottlenecks common in traditional deep learning intrusion detection systems.

5.2.1 Phase 1: Ingestion and Drift Management Constraints

Phase 1 must continuously absorb high-velocity log streams without catastrophic memory expansion. While the Gaussian Copula's online Expectation-Maximization operates with a defined time complexity of $\mathcal{O}(K \times M \times |\mathcal{B}| + T)$, the semantic drift mechanisms enforce their own rigorous limits:

- **ADWIN-U Logarithmic Space:** Instead of a hard-capped window size, ADWIN-U utilizes a variable-length window W that only shrinks upon detecting structural drift ($|\hat{\mu}_{W_1} - \hat{\mu}_{W_2}| \geq \epsilon$). To avoid linear memory explosion, it employs an exponential histogram technique. Consequently, the space complexity is strictly logarithmic, bounded at $\mathcal{O}(\log(W))$ for a single detector, and $\mathcal{O}(2\log(W))$ when monitoring both independent classes concurrently.
- **ReCDA Encoder Efficiency:** The Representation Enhancement module utilizes a highly compressed feature extractor, with the hidden layer dimensionality explicitly constrained to $d_h = 16$. During the weakly-supervised tuning phase, the instructive anchor selection rate (δ) evaluates up to 100% of the historical dataset $|D_h|$ to populate the instructive matrix G, ensuring a bounded tuning duration proportional strictly to the size of the historical buffer.

5.2.2 Phase 2: Token Boundaries and Clustering Limits

The most significant vulnerability in Phase 2 involves quadratic memory scaling within transformer attention mechanisms and the cubic scaling of hierarchical clustering. The architecture preempts these bottlenecks via strict deterministic truncation.

- **Transformer Token Limits (N):** The text processing is bifurcated to prevent $\mathcal{O}(N^2 \cdot d)$ complexity collapse. For the Local LLM constructing the factual hypergraph (OG-RAG), documents are strictly chunked to a maximum token limit of $N = 8192$. This perfectly balances broad attack-chain context with the attention mechanism's memory ceiling. Subsequently, the SecureBERT_Plus engine—responsible for creating the contextual vectors—enforces a rigid sequence limit of $N = 512$ tokens, guaranteeing near-instantaneous embedding generation for localized log entities.
- **HAC Complexity and Node Saturation (V):** Hierarchical Agglomerative Clustering calculates cosine distances across all entity vectors, scaling at $\mathcal{O}(V^2 \log V)$ or $\mathcal{O}(V^3)$. Because Phase 2 utilizes a Time-Bounded Fetch within $(t_{\text{prev}}, t_0]$, the expected episodic volume of semantic entities is mathematically restricted to $V \approx 250 - 300$. To guarantee pipeline stability during anomalies like massive network scans, a hard architectural cutoff is enforced at $V_{\text{max}} = 7500$. If V exceeds this threshold, the deterministic script aggressively pre-filters the array before executing the clustering algorithm.

5.2.3 Phase 3: Continuous Integration and Scalability Proofs

Phase 3 processes the hybrid continuous-discrete payload. Its viability relies on proving that the ODE solver and graph message passing will not stall under high event loads.

- **ODE Solver Bounds (L_{steps}):** The pipeline employs an adaptive Runge-Kutta 4-5 (Dormand-Prince) solver, meaning integration steps scale dynamically. Based on the Lipschitz continuity of the continuous vector field (bounded by a constant L_f) and stabilized via Temporal Adaptive Batch Normalization (TA-BN), the solver is mathematically guaranteed to cross the interval without requiring infinite steps. Practically, $L_{\text{steps}} \propto \mathcal{O}(N_{\text{events}})$. With the maximum sequence length capped at 1024, the Number of Function Evaluations (NFE) is strictly bounded per episode.

- **Graph Sparsity and Spatial Aggregation:** Spatial aggregation time complexity depends on whether the graph is dense $\mathcal{O}(|V|^2)$ or sparse $\mathcal{O}(|E|)$. By adopting Assumption 2 (Bounded Degree) from (Anaedevha et al., 2026), the maximum in-degree is capped ($\max_i |N_i| \leq d_{\text{max}}$), independent of graph size. Because Phase 2 prunes redundant noise, $|E| \leq d_{\text{max}} \times |V|$. Thus, message passing complexity is linear $\mathcal{O}(|V|)$. Furthermore, the dimensions are highly controlled: node features $d_n = 256$, edge features $d_e = 87$, and the multi-scale ODE evaluates exactly $S = 4$ fixed time constants, applying a manageable computational multiplier.

- **Deep Spatio-Temporal Log-Barrier Optimization:** A standard transformer-based point process for evaluating Event Intensity requires $\mathcal{O}(N^2)$ calculations. However, by approximating the survival integral using equispaced nodes $m = \lceil c\sqrt{n} \rceil$, the temporal adaptive framework employs a log-barrier optimization. This mathematically reduces the point process time complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^{3/2})$, ensuring the predictive temporal mechanism remains exceptionally lightweight despite the complex spatial topology.

5.3 Theoretical Evaluation: Dry-Run of an APT Attack

To validate the theoretical efficacy of the proposed three-phase architecture, this section performs a logical dry-run of a highly sophisticated Advanced Persistent Threat (APT): the SolarWinds supply chain compromise. This specific campaign was selected because it perfectly encapsulates the systemic bottlenecks that cause legacy SIEMs to fail, notably exploiting massive temporal gaps (dormancy), cross-domain ambiguity, and the hijacking of trusted entities to mask feature drift.

5.3.1 The SolarWinds Campaign Timeline and SIEM Telemetry

The attack unfolded over several months, intentionally blending into legitimate network traffic to evade static detection rules. The campaign can be abstracted into five critical stages:

1. **Initial Access and Implantation (Supply Chain Compromise - T1195.002):** Attackers compromised the SolarWinds build pipeline, injecting a backdoor into a legitimate DLL update.

   Telemetry: Sysmon Event 11 (`FileCreate`) records `SolarWinds.Orion.Core.BusinessLayer.dll` written to disk. Sysmon Event 7 (`Image Loaded`) records `SolarWinds.BusinessLayerHost.exe` loading the tampered DLL. Because this mimics a trusted software update, the logs appear completely benign.

2. **The Dormancy Period:** The malware actively checks for sandbox environments and pauses all malicious activity for a period of 12 to 14 days, generating absolute telemetry silence.

3. **Execution and C2 Beaconing (T1568.002):** Following dormancy, the implant beacons to a Command and Control (C2) server using a Domain Generation Algorithm (DGA).

   - *Telemetry:* Zeek `dns.log` or Sysmon Event 22 records massive query spikes to `*.avsvmcloud[.]com`. The subdomains exhibit extreme lengths (> 30 characters) and moderate Shannon entropy (≈ 4.0).

4. **Lateral Movement and Privilege Escalation (T1606.002):** Attackers steal the Active Directory Federation Services (AD FS) SAML signing certificate to mint forged "Golden SAML" tokens, bypassing MFA and impersonating any user.

   - *Telemetry:* Windows Security Event 4624 (Logon Type 3) shows anomalous host-to-host authentication. Sysmon Event 1 records the execution of `AdFind.exe`. Azure AD Audit Logs record successful logins to critical cloud applications from anomalous IPs.

5. **Data Exfiltration (T1114.002):** Programmatic retrieval of executive emails via the Microsoft Graph API using forged tokens.

   - *Telemetry:* O365 Exchange Audit Logs (`MailboxLogin`, `FileDownloaded`) show massive data retrieval.

5.3.2 Logical Pipeline Execution and Dry-Run Analysis

To empirically validate the theoretical architecture, we perform a logical dry-run of the pipeline against the SolarWinds supply-chain compromise. Figure 4 illustrates the logically structured, bifurcated output of Phase 2

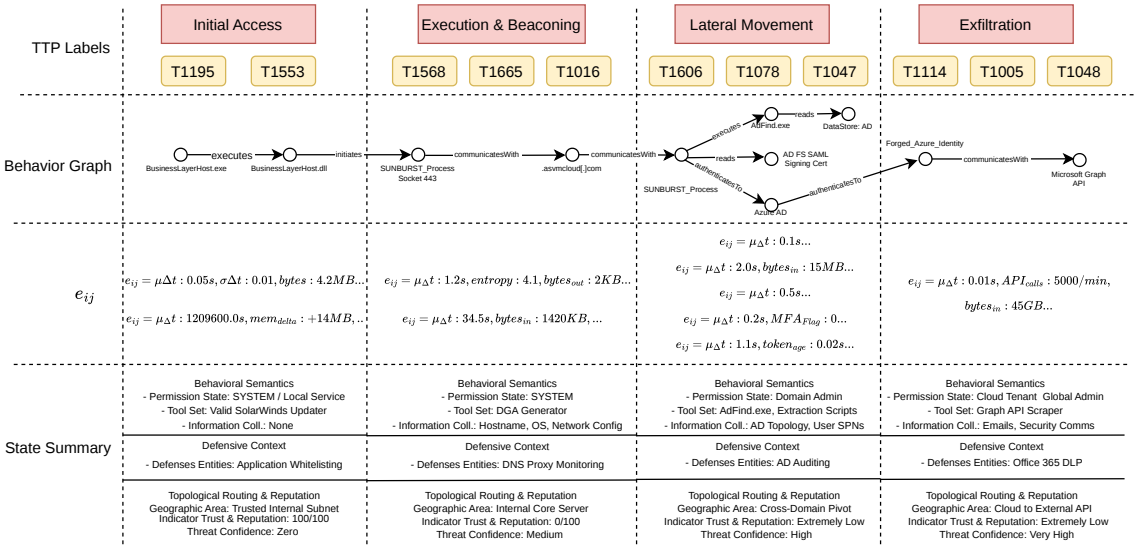


Figure 4: Dry-Run of the SolarWinds Attack represented in modified AttacKG+ Structure

across four chronological stages of the Advanced Persistent Threat (APT): Initial Access, Execution & Beaconing, Lateral Movement, and Exfiltration.

For visual brevity, Figure 4 omits the dense auxiliary noise of the underlying Factual Hypergraph, isolating instead the core causal attack chain. The diagram is divided into four horizontal strata that directly map to the architecture's data streams: the top two strata represent the semantic discrete payload (MITRE TTP Labels and the Behavior Graph), the third stratum represents the continuous deterministic payload (e_{ij}), and the bottom stratum details the highly enriched State Summary vectors deduced by the Local LLM.

The following traces exactly how the proposed architecture processes this timeline, mitigating the bottlenecks that blind legacy systems.

a) Phase 1 Processing and Mitigation of Adversarial Poisoning

When the tampered SolarWinds update occurs (*Initial Access* stage in Figure 4), the system ingests the Sysmon Event 7 logs. Because the software is highly trusted and structurally standard, the Gaussian Copula (GC) module normalizes these mixed-type features into the continuous latent space without triggering any baseline alerts.

A critical consideration in any retrospective analysis is hindsight bias—specifically, the assumption that the pipeline flawlessly isolates the anomaly without being overwhelmed by false positives or actively poisoned by the adversary. During the subsequent 14-day dormancy period, the malware generates absolute telemetry silence. In a legacy SIEM, the surrounding benign network evolution during this two-week gap would typically generate thousands of false-positive alerts due to strict static baselines. However, the ADWIN-U module continuously monitors this benign evolution. When normal structural shifts occur, the unsupervised ReCDA module seamlessly updates the latent sub-space, dynamically suppressing false positives and maintaining a mathematically correct baseline without

manual SOC intervention.

Fourteen days later, as the attack transitions to the *Execution & Beaconing* stage, the `SolarWinds.BusinessLayerHost.exe` process begins emitting massive amounts of DNS queries. This introduces a sophisticated adversarial threat: if the attacker masked this DGA beaconing within massive volumes of synthetic benign traffic, could they force a semantic Concept Drift, tricking ReCDA into accepting the malicious C2 beaconing as the "new normal"?

To prevent this adversarial drift poisoning, the pipeline relies on the strict mathematical gating of the Phase 1 Malicious Detector. ReCDA is mathematically restricted to updating the decision boundary using only unlabeled samples that fall within an ambiguous confidence threshold. The DGA beaconing, however, introduces high Shannon entropy and mathematically rigid inter-arrival times that severely breach the established bounds of the Gaussian Copula latent space. The classifier explicitly flags this as a high-confidence anomaly (A_0), immediately routing it to Phase 2 and permanently quarantining it from the ReCDA update pool. Consequently, the attacker's attempt to force structural drift fails, the latent space remains anchored to true benign behavior, and the C2 anomaly is successfully isolated.

b) Phase 2 Processing

The anomalous DNS alert (A_0) is routed to the Alert Contextualizer, which systematically constructs the bifurcated narrative depicted in Figure 4.

It is important to note that Figure 4 intentionally omits a significant amount of data for visual brevity. A true production hypergraph contains thousands of interconnected noisy nodes; the diagram isolates only the core causal attack chain. Similarly, the depicted Edge Features (e_{ij}) and State Summaries represent only a curated fraction of the highly dimensional vectors mathematically evaluated by the pipeline at each stage.

The following details how Phase 2 bifurcates and processes this information across the attack lifecycle:

Continuous Edge Feature (e_{ij}) Evolution Rather than wasting LLM context tokens on numeric evaluation, the deterministic Edge Feature Calculator extracts features representative of behavior, represented in the third stratum of Figure 4. The evolution of these continuous variables provides the ODE solver with vital physical context:

- **Initial Access:** The features reflect standard software update behavior: stable inter-arrival times ($\mu_{\Delta t} = 0.05s$, very low variance $\sigma_{\Delta t}$) and standard payload sizes ($bytes = 4.2MB$).
- **Execution & Beaconing:** The Temporal Bounding Mechanism bridges the gap to the initial DLL modification. The calculator extracts the massive temporal dormancy gap ($\mu_{\Delta t} = 1,209,600s$, or exactly 14 days). Concurrently, it captures the DGA beaconing physics: high payload entropy ($entropy : 4.1$) paired with tiny outbound queries ($bytes_{\text{out}} : 2KB$).

- **Lateral Movement:** As the attacker pivots to the AD FS servers, the edge features capture sudden internal data spikes ($bytes_{\text{in}} : 15MB$). Crucially, as the forged SAML token is used against Azure AD, the calculator records instantaneous token exchange rates ($token_{\text{exp}} : 0.02s$) and mathematically flags the absence of secondary authentication challenges ($MFA_{\text{flag}} : 0$).

- **Exfiltration:** During the final stage, the physical edge features capture undeniable corporate espionage: massive volumetric spikes in Microsoft Graph API interactions ($API_{\text{calls}} : 5000/\text{min}$) and massive outbound data transfers ($bytes_{\text{in}} : 45GB$).

Semantic State Summary and Threat Confidence Evolution While the continuous micro-edges track the physics of the attack, the Local LLM deduces the semantic narrative, outputting the State Summary. The bottom stratum of Figure 4 demonstrates how this summary dynamically evolves, particularly highlighting the pipeline's *Threat Confidence* field:

- **Stage 1 (Baseline):** The Threat Confidence is **Zero**. The Permission State is SYSTEM (normal for a core monitoring service), the Tool Set is the valid SolarWinds installer, and the MISP Indicator Trust is 100/100. It is entirely indistinguishable from routine enterprise maintenance.

- **Stage 2 (Anomaly Detection):** The Threat Confidence escalates to **Medium**. The Defensive Context flags tripped DNS proxy monitors due to volume spikes. The Indicator Trust plummets to 0/100 because the system is beaconing to `avsvmcloud[.]com`—an unmapped, newly registered public infrastructure. The threat is categorized as an isolated, novel anomaly.

- **Stage 3 (Critical Pivot):** The Threat Confidence spikes to **High**. The LLM recognizes a massive escalation in Permission State (Local SYSTEM to Domain/Global Administrator). The Tool Set reflects malicious query utilities (`AdFind.exe`) and cryptographic extraction scripts. Because the activity originates from a non-standard IP framework targeting the federated cloud tenant, the narrative explicitly recognizes Advanced Persistent Threat (APT) behavior.

- **Stage 4 (Terminal Breach):** The Threat Confidence reaches **Very High (99%)**. The Permission State is now Cloud Tenant Global Admin, and Office 365 Data Loss Prevention (DLP) filters are actively tripping. The LLM categorically attributes the event cluster as a confirmed, highly targeted corporate espionage operation.

Ontology-Grounded TTP Deduction Guided by the Factual Hypergraph constructed via OG-RAG, the Local LLM mathematically resolves entity synonymy and deduces the exact MITRE ATT&CK sequence (Top Stratum of Figure 4). The diagram displays a subset of these TTPs, deduced through the following logic:

- **Initial Access:** Because the anomaly originates from a trojanized but legitimately signed update binary, the

LLM maps the hypergraph to **T1195.002** (Supply Chain Compromise) and **T1553.002** (Subvert Trust Controls via Code Signing).

- **Execution:** The presence of high-entropy subdomain generation triggers **T1568** (Dynamic Resolution/DGA). Because the C2 hostnames mimic legitimate victim infrastructure, the LLM deduces **T1665** (Hide Infrastructure), alongside **T1016** (Internet Connection Discovery) as the malware probes for outbound HTTP proxies.
- **Lateral Movement:** The execution of `AdFind.exe` and WMI queries yields **T1047** (Windows Management Instrumentation) and various Discovery tactics (omitted from diagram, e.g., T1018, T1087). The extraction of the AD FS SAML signing certificate and its subsequent use to bypass MFA mathematically forces the deduction of **T1606.002** (Forge Web Credentials: SAML Tokens) and **T1078** (Valid Accounts).
- **Exfiltration:** The automated scripting against the Microsoft Graph API to scrape executive emails results in **T1114.002** (Remote Email Collection) and **T1005** (Data from Local System), ultimately culminating in **T1048** (Exfiltration Over Alternative Protocol) as the password-protected archives are siphoned out of the network.

By strictly separating the numeric physics (e_{ij}) from these semantic TTP deductions, Phase 2 guarantees that the downstream ODE solver in Phase 3 receives mathematically correct topological coordinates, while the SOC analysts receive highly explainable attack narrative.

c) Phase 3 Processing

Phase 2 passes the strictly formatted continuous micro-edges and discrete macro-events to the Episodic CT-TGNN in Phase 3.

- **Continuous Dynamics:** The CT-TGNN initializes the SolarWinds node embedding from the Global State Registry ($h_i(t_{\text{prev}})$). The localized TA-BN-ODE solver smoothly rolls the physical topology forward across the 14-day dormancy gap, sequentially integrating the e_{ij} continuous payload (e.g., 1420 KB data spikes and extreme Δt gaps) to arrive at the precise pre-alert state, $h_i(t_0^-)$.
- **Discrete State Jump:** Exactly at t_0, the LLM-deduced State Summary acts as a discrete mathematical shock. The realization of Golden SAML privilege escalation, MFA bypass, and raw cryptographic extraction scripts is fused into the continuous state, updating the node to the finalized embedding $h_i(t_0^+)$.
- **Explainable Classification:** The self-supervised TFLAG Deviation Network evaluates the concatenated tensor $[h_i(t_0^+), \lambda_k(t_0)]$. The combination of a 14-day temporal gap, a structural shift from internal to external topology, and a massive semantic escalation to SAML token forgery causes an extreme mathematical

deviation from the registry baseline. TFLAG classifies the behavior as a critical APT. The LLM Chain-of-Thought (CoT) step then synthesizes this into a highly explainable, sample-level alert, tracking exactly how a trusted DLL escalated into cloud data exfiltration.

- **Unconditional Write-Back:** Finally, the compromised state $h_i(t_0^+)$ is unconditionally written back to the Global State Registry. This perpetually preserves the mathematical memory of the network. If the attackers attempt to deploy a secondary payload through this vector months later, the architecture inherently recognizes the SolarWinds node as fundamentally poisoned.

6 Conclusion and Future Directions

6.1 Synthesis of the Research Problem and Solution

Modern enterprise zero-trust networks face an asymmetric threat landscape. Legacy rule-based Security Information and Event Management (SIEM) systems and static machine learning classifiers are fundamentally misaligned with the realities of highly dynamic network environments. As demonstrated by the SolarWinds compromise, Advanced Persistent Threats (APTs) exploit temporal dormancy, cross-domain ambiguity, and trusted supply chains to blind reactive forensic systems.

To solve this, this thesis proposes a highly scalable, AEGIS-CT. The core original contribution of this research is the formulation of a "continuous-discrete hybrid dynamical system." Rather than relying on monolithic black-box models, this architecture mathematically bifurcates the physics of network traffic from the semantic reasoning of attacker intent. By marrying the continuous temporal integration of Graph Neural Networks (CT-TGNN) with the zero-shot deductive reasoning of Large Language Models (LLMs), the proposed pipeline significantly mitigates the operational bottlenecks that have historically hindered real-time, predictive incident detection, establishing a rigorous foundation for scalable inference despite ongoing challenges in extreme high-volume scenarios.

6.2 Alignment with Design Requirements

The proposed architecture was intentionally decoupled to satisfy five foundational Design Requirements (DRs) necessary for enterprise deployment:

- **DR-1: Robust Ingestion of Incomplete and Heterogeneous Data Streams:** Solved in Phase 1 via the Gaussian Copula (GC) and Online Expectation-Maximization (EM) loop. This deterministic module seamlessly ingests mixed-type SIEM alerts and reconstructs missing values within a unified latent space without relying on unstable neural network backpropagation.
- **DR-2: Unsupervised Dual-Drift Adaptation under Label Scarcity:** Addressed by pairing the ADWIN-U statistical detector with the Representation Enhancement (ReCDA) module. This allows the system to autonomously detect concept drift and retune its decision boundaries dynamically using self-supervised contrastive loss, completely circumventing the need for continuous manual labeling.
- **DR-3: Contextualized Alert Enrichment and Unambiguous CKG Construction:** Satisfied in Phase 2. By leveraging an Alert Contextualizer and the OG-RAG framework, the system bridges temporal dormancy gaps and clusters entities (via SecureBERT_Plus and HAC) to mathematically resolve cross-domain ambiguities. This produces a pristine, factual baseline for the LLM to deduce MITRE ATT&CK tactics.
- **DR-4: Mitigation of Structural Bottlenecks in Dynamic Graphs:** Resolved in Phase 3 through the implementation of an Episodic CT-TGNN paired with a Global State Registry. The use of the Adjoint Method

and Log-Barrier Optimization mathematically guarantees that the system can continuously integrate temporal dynamics without triggering the adjacency matrix memory explosions fatal to traditional static GNNs.

- **DR-5: Explainable Alert Prioritization for Human-AI Teaming:** Achieved by combining the TFLAG Deviation Network with a Chain-of-Thought (CoT) LLM. Instead of outputting binary anomaly scores, the architecture translates topological graph deviations into highly explainable, sample-level threat narratives, fundamentally shifting the SOC analyst's role from manual data triage to strategic response.

6.3 Significance and Impact

The implications of this research extend far beyond theoretical mathematics; it offers a paradigm shift for Security Operations Centers (SOCs). Currently, SOCs are plagued by alert fatigue and routinely detect APTs months after the initial perimeter breach.

This architecture matters because it transitions enterprise security from reactive forensics to **predictive incident detection**. By perpetually preserving the mathematical memory of the network and tracking micro-behaviors across multi-scale differential equations, the system can anticipate a lateral movement or privilege escalation event *before* the attacker successfully completes the exploit chain. Security analysts, MLSecOps engineers, and enterprise stakeholders directly benefit from an architecture that collapses hours of manual, cross-platform investigation into instantaneous, highly explainable, and tactically grounded alerts.

6.4 Future Work and Next Steps

While the theoretical evaluation proves the architecture's predictive capabilities, the rigorous feasibility analysis identified critical operational hurdles that must guide future empirical research:

1. **Empirical Baseline Validation:** The immediate next step is to deploy the pipeline against standard, live enterprise benchmark datasets to measure true inference latency. Specifically, testing against datasets like CSE-CIC-IDS2018 or UNSW-NB15 will be critical for evaluating the Dormand-Prince ODE solver under the stress of simulated volumetric Distributed Denial of Service (DDoS) telemetry, while datasets like Splunk's BOTSv3 (Boss of the SOC) can be utilized to assess the semantic tracking of long-term APT progression.

2. **Mitigating LLM Inference Costs:** Phase 2 currently relies heavily on evaluating massive 8192-token context windows, which presents severe financial and hardware bottlenecks for standard SOCs under heavy alert loads. Future work should explore substituting the LLM extraction step with heuristic, deterministic mapping to a static AttacKG+ schema, reserving the LLM solely for the final CoT alert generation.

3. **Offline Ensemble Strategies:** To mitigate the MLSecOps risks of continuous online neural network tuning (ReCDA), future iterations should explore offline ensemble swapping, allowing the architecture to maintain its predictive edge while ensuring training stability in live, adversarial environments.

Bibliography

Abdel-Basset, M., Hawash, H., Chakrabortty, R. K., & Ryan, M. J. (2021, August). Semi-Supervised Spatiotemporal Deep Learning for Intrusions Detection in IoT Networks. *IEEE Internet of Things Journal*, *8*(15), 12251–12265. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9359667/ doi: 10.1109/JIOT.2021.3060878

Anaedevha, R. N., Trofimov, A. G., & Borodachev, Y. V. (n.d.). CT-TGNN: Continuous-Time Temporal Graph Neural Networks for Real-Time Threat Propagation Analysis in Encrypted Zero-Trust Microservices. *TechRxiv*, *2026*(0108). Retrieved 2026-05-05, from https://www.techrxiv.org/doi/full/10.36227/techrxiv.176784247.72825467/v1 doi: 10.36227/techrxiv.176784247.72825467/v1

Anaedevha, R. N., Trofimov, A. G., & Borodachev, Y. V. (2026). Temporal adaptive neural ordinary differential equations with deep spatio-temporal point processes for real-time network intrusion detection. *Complex & Intelligent Systems*, *12*(6), 165.

Andresini, G., Pendlebury, F., Pierazzi, F., Loglisci, C., Appice, A., & Cavallaro, L. (2021, November). INSOMNIA: Towards Concept-Drift Robustness in Network Intrusion Detection. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security* (pp. 111–122). New York, NY, USA: Association for Computing Machinery. Retrieved 2026-06-17, from https://dl.acm.org/doi/10.1145/3474369.3486864 doi: 10.1145/3474369.3486864

Angiulli, F., Argento, L., & Furfaro, A. (2015, November). Exploiting N-Gram Location for Intrusion Detection. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)* (pp. 1093–1098). Retrieved 2026-06-18, from https://ieeexplore.ieee.org/abstract/document/7372252 (ISSN: 1082-3409) doi: 10.1109/ICTAI.2015.155

Arazzi, M., R. Arikkat, D., Nicolazzo, S., Nocera, A., Rehiman K.A., R., P., V., & Conti, M. (2025, November). NLP-based techniques for Cyber Threat Intelligence. *Computer Science Review*, *58*, 100765. Retrieved 2026-04-30, from https://www.sciencedirect.com/science/article/pii/S1574013725000413 doi: 10.1016/j.cosrev.2025.100765

Assis, D. N., & Souza, V. M. A. (2025, November). ADWIN-U: adaptive windowing for unsupervised drift detection on data streams. *Knowledge and Information Systems*, *67*(11), 10005–10034. Retrieved 2026-04-23, from https://doi.org/10.1007/s10115-025-02523-1 doi: 10.1007/s10115-025-02523-1

Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., & Morales-Bueno, R. (2006). Early drift detection method. In *Fourth international workshop on knowledge discovery from data streams* (Vol. 6, pp. 77–86).

Barros, R. S., Cabral, D. R., Gonçalves Jr, P. M., & Santos, S. G. (2017). Rddm: Reactive drift detection method. *Expert Systems with Applications*, *90*, 344–355.

Bifet, A., & Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. In *Proceedings of the*

2007 siam international conference on data mining (pp. 443–448).

Boffa, M., Drago, I., Mellia, M., Vassio, L., Giordano, D., Valentim, R., & Houidi, Z. B. (2024). Logprécis: Unleashing language models for automated malicious log analysis: Précis: A concise summary of essential points, statements, or facts. *Computers & Security*, *141*, 103805.

Bojchevski, A., & Günnemann, S. (2017). Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*.

Brzeziński, D., & Stefanowski, J. (2011). Accuracy updated ensemble for data streams with concept drift. In *International conference on hybrid artificial intelligence systems* (pp. 155–163).

Cao, D., Wang, Y., Duan, J., Zhang, C., Zhu, X., Huang, C., ... others (2020). Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, *33*, 17766–17778.

Casas, P., D'Alconzo, A., Settanni, G., Fiadino, P., & Skopik, F. (2016, October). POSTER: (Semi)-Supervised Machine Learning Approaches for Network Security in High-Dimensional Network Data. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1805–1807). New York, NY, USA: Association for Computing Machinery. Retrieved 2026-06-18, from `https://dl.acm.org/doi/10.1145/2976749.2989069` doi: 10.1145/2976749.2989069

Chen, C., Yunchao Gong, & Yingjie Tian. (2008, October). Semi-supervised learning methods for network intrusion detection. In *2008 IEEE International Conference on Systems, Man and Cybernetics* (pp. 2603–2608). Singapore, Singapore: IEEE. Retrieved 2026-06-18, from `http://ieeexplore.ieee.org/document/4811688/` doi: 10.1109/ICSMC.2008.4811688

Chen, J., Feng, M., & Wirjanto, T. S. (2025, March). Prospective Multi-Graph Cohesion for Multivariate Time Series Anomaly Detection. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining* (pp. 98–106). New York, NY, USA: Association for Computing Machinery. Retrieved 2026-04-23, from `https://dl.acm.org/doi/10.1145/3701551.3703494` doi: 10.1145/3701551.3703494

Chen, J., Yang, S., Gao, T., Ying, Y., Li, T., & Li, P. (2024). Multi-type concept drift detection under a dual-layer variable sliding window in frequent pattern mining with cloud computing. *Journal of Cloud Computing*, *13*(1), 40.

Chen, M., Zhu, K., Lu, B., Li, D., Yuan, Q., & Zhu, Y. (2025). Aecr: Automatic attack technique intelligence extraction based on fine-tuned large language model. *Computers & Security*, *150*, 104213.

Chen, Y., & Dai, H.-L. (2024). Concept drift adaptation with continuous kernel learning. *Information Sciences*, *670*, 120649.

Chiu, C.-Y., Lee, Y.-J., Chang, C.-C., Luo, W.-Y., & Huang, H.-C. (2010). Semi-supervised Learning for False Alarm Reduction. In P. Perner (Ed.), *Advances in Data Mining. Applications and Theoretical Aspects* (pp. 595–605). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-642-14400-4$_4$6

de Barros, R. S. M., Hidalgo, J. I. G., & de Lima Cabral, D. R. (2018). Wilcoxon rank sum test drift detector. *Neurocomputing*, *275*, 1954–1963.

Deng, A., & Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 35, pp. 4027–4035).

Ezeme, O. M., Mahmoud, Q. H., & Azim, A. (2020). Design and Development of AD-CGAN: Conditional Generative Adversarial Networks for Anomaly Detection. *IEEE Access*, *8*, 177667–177681. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/abstract/document/9201442 doi: 10.1109/ACCESS.2020.3025530

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Brazilian symposium on artificial intelligence* (pp. 286–295).

Gulati, N., Greenstadt, R., Dandekar, K. R., & Walsh, J. M. (2013, September). GMM Based Semi-Supervised Learning for Channel-Based Authentication Scheme. In *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)* (pp. 1–6). Las Vegas, NV, USA: IEEE. Retrieved 2026-06-18, from http://ieeexplore.ieee.org/document/6692216/ doi: 10.1109/VTCFall.2013.6692216

Guo, Y., Liu, Z., Huang, C., Liu, J., Jing, W., Wang, Z., & Wang, Y. (2021). Cyberrel: Joint entity and relation extraction for cybersecurity concepts. In *International conference on information and communications security* (pp. 447–463).

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems*, *30*.

Hara, K., & Shiomoto, K. (2020, April). Intrusion Detection System using Semi-Supervised Learning with Adversarial Auto-encoder. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium* (pp. 1–8). Budapest, Hungary: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9110343/ doi: 10.1109/NOMS47738.2020.9110343

Haweliya, J., & Nigam, B. (2014, January). Network Intrusion Detection using Semi Supervised Support Vector Machine. *International Journal of Computer Applications*, *85*(9), 27–31. Retrieved 2026-06-17, from http://research.ijcaonline.org/volume85/number9/pxc3893245.pdf doi: 10.5120/14870-3245

Jaiswal, A., Manjunatha, A. S., Madhu, B., & Chidananda, M. P. (2016, December). Predicting unlabeled traffic for intrusion detection using semi-supervised machine learning. In *2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)* (pp. 218–222). Retrieved 2026-06-17, from https://ieeexplore.ieee.org/abstract/document/7955218 doi: 10.1109/ICEECCOT.2016.7955218

Jalalvand, F., Baruwal Chhetri, M., Nepal, S., & Paris, C. (2024, November). Alert Prioritisation in Security Operations

Centres: A Systematic Survey on Criteria and Methods. *ACM Comput. Surv.*, *57*(2), 42:1–42:36. Retrieved 2026-04-22, from https://dl.acm.org/doi/10.1145/3695462 doi: 10.1145/3695462

Jia, J., Dong, Z., Li, J., & Stokes, J. W. (2021, June). Detection Of Malicious DNS and Web Servers using Graph-Based Approaches. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 2625–2629). Toronto, ON, Canada: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9415092/ doi: 10.1109/ICASSP39728.2021.9415092

Jiang, W., Chai, T., Liu, H., Wang, K., & Zhang, H. (2025, September). *TFLAG:Towards Practical APT Detection via Deviation-Aware Learning on Temporal Provenance Graph.* arXiv. Retrieved 2026-05-25, from http://arxiv.org/abs/2501.06997 (arXiv:2501.06997 [cs.CR]) doi: 10.48550/arXiv.2501.06997

Jiang, Z., Liu, J., Chen, Z., Li, Y., Huang, J., Huo, Y., ... Lyu, M. R. (2024, July). LILAC: Log Parsing using LLMs with Adaptive Parsing Cache. *Proc. ACM Softw. Eng.*, *1*(FSE), 7:137–7:160. Retrieved 2026-04-30, from https://dl.acm.org/doi/10.1145/3643733 doi: 10.1145/3643733

Khemani, B., Patil, S., Kotecha, K., & Tanwar, S. (2024). A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, *11*(1), 18.

Khonde, S. R., & Ulagamuthalvi, V. (2019, July). Ensemble-based semi-supervised learning approach for a distributed intrusion detection system. *Journal of Cyber Security Technology*, *3*(3), 163–188. Retrieved 2026-06-18, from https://doi.org/10.1080/23742917.2019.1623475 (_eprint: https://doi.org/10.1080/23742917.2019.1623475) doi: 10.1080/23742917.2019.1623475

King, I. J., & Huang, H. H. (2023, June). Euler: Detecting Network Lateral Movement via Scalable Temporal Link Prediction. *ACM Trans. Priv. Secur.*, *26*(3), 35:1–35:36. Retrieved 2026-05-15, from https://dl.acm.org/doi/10.1145/3588771 doi: 10.1145/3588771

Kipf, T. N., & Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N., & Welling, M. (2016b). Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Lee, J.-H., Kim, J.-W., & Choi, M.-J. (2021, September). SSAE - DeepCNN Model for Network Intrusion Detection. In *2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS)* (pp. 78–83). Tainan, Taiwan: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9562594/ doi: 10.23919/APNOMS52696.2021.9562594

Lei, S., Chang, X., Yu, Z., He, D., Huo, C., Wang, J., & Jin, D. (2025). Feature-structure adaptive completion graph neural network for cold-start recommendation. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 39, pp. 12022–12030).

Li, B., Yang, Q., Deng, C., & Pan, H. (2025, September). CyberKG: Constructing a Cybersecurity Knowledge

Graph Based on SecureBERT_plus for CTI Reports. *Informatics*, *12*(3), 100. Retrieved 2026-05-05, from https://www.mdpi.com/2227-9709/12/3/100 doi: 10.3390/informatics12030100

Li, W., Meng, W., & Au, M. H. (2020, July). Enhancing collaborative intrusion detection via disagreement-based semi-supervised learning in IoT environments. *Journal of Network and Computer Applications*, *161*, 102631. Retrieved 2026-06-17, from https://linkinghub.elsevier.com/retrieve/pii/S1084804520301053 doi: 10.1016/j.jnca.2020.102631

Lin, Z. (2021, October). Network Intrusion Detection based of Semi-Supervised Ensemble Learning Algorithm for Imbalanced Data. In *2021 International Conference on Networking and Network Applications (NaNA)* (pp. 338–344). Lijiang City, China: IEEE. Retrieved 2026-06-17, from https://ieeexplore.ieee.org/document/9634927/ doi: 10.1109/NaNA53684.2021.00065

Liu, Y., Han, X., Zuo, W., Lv, H., & Guo, J. (2024). Cti-je: a joint extraction framework of entities and relations in unstructured cyber threat intelligence. In *2024 27th international conference on computer supported cooperative work in design (cscwd)* (pp. 2728–2733).

Mao, C.-H., Lee, H.-M., Parikh, D., Chen, T., & Huang, S.-Y. (2009, March). Semi-supervised co-training and active learning based approach for multi-view intrusion detection. In *Proceedings of the 2009 ACM symposium on Applied Computing* (pp. 2042–2048). New York, NY, USA: Association for Computing Machinery. Retrieved 2026-06-17, from https://dl.acm.org/doi/10.1145/1529282.1529735 doi: 10.1145/1529282.1529735

Math, H., & Lienhart, R. (2025, December). *Towards Practical Multi-label Causal Discovery in High-Dimensional Event Sequences via One-Shot Graph Aggregation.* arXiv. Retrieved 2026-05-15, from http://arxiv.org/abs/2509.19112 (arXiv:2509.19112 [cs]) doi: 10.48550/arXiv.2509.19112

Min, B., Yoo, J., Kim, S., Shin, D., & Shin, D. (2021). Network Anomaly Detection Using Memory-Augmented Deep Autoencoder. *IEEE Access*, *9*, 104695–104706. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/abstract/document/9496635 doi: 10.1109/ACCESS.2021.3100087

Mohammadi, B., & Sabokrou, M. (2019, October). End-to-End Adversarial Learning for Intrusion Detection in Computer Networks. In *2019 IEEE 44th Conference on Local Computer Networks (LCN)* (pp. 270–273). Osnabrueck, Germany: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/8990759/ doi: 10.1109/LCN44214.2019.8990759

Mvula, P. K., Branco, P., Jourdan, G.-V., & Viktor, H. L. (2024, June). A Survey on the Applications of Semi-supervised Learning to Cyber-security. *ACM Comput. Surv.*, *56*(10), 253:1–253:41. Retrieved 2026-04-22,

from https://dl.acm.org/doi/10.1145/3657647 doi: 10.1145/3657647

Nishida, K., & Yamauchi, K. (2007). Detecting concept drift using statistical testing. In *International conference on discovery science* (pp. 264–269).

Pachghare, V. K., Khatavkar, V. K., & Kulkarni, P. A. (2012, July). Pattern Based Network Security Using Semi-supervised Learning. *International Journal of Information and Network Security (IJINS)*, *1*(3), 228–234. Retrieved 2026-06-17, from http://www.iaesjournal.com/online/index.php/IJINS/article/view/704 doi: 10.11591/ijins.v1i3.704

Pan, Y., Zhao, L., Leng, T., Luo, Z., Cai, L., Yu, A., & Meng, D. (2026). DriftTrace: Combating Concept Drift in Security Applications Through Detection and Explanation. *IEEE Transactions on Information Forensics and Security*, *21*, 1957–1972. Retrieved 2026-04-23, from https://ieeexplore.ieee.org/abstract/document/11367729 doi: 10.1109/TIFS.2026.3659398

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., ... Leiserson, C. (2020). Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 34, pp. 5363–5370).

Pesaranghader, A., & Viktor, H. L. (2016). Fast hoeffding drift detection method for evolving data streams. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 96–111).

Polikar, R., Upda, L., Upda, S. S., & Honavar, V. (2001). Learn++: An incremental learning algorithm for supervised neural networks. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, *31*(4), 497–508.

Rong, Y., Huang, W., Xu, T., & Huang, J. (2019). Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*.

Sankar, A., Wu, Y., Gou, L., Zhang, W., & Yang, H. (2020). Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th international conference on web search and data mining* (pp. 519–527).

Sarhan, I., & Spruit, M. (2021). Open-cykg: An open cyber threat intelligence knowledge graph. *Knowledge-based systems*, *233*, 107524.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The graph neural network model. *IEEE transactions on neural networks*, *20*(1), 61–80.

Seth, S., Chahal, K. K., & Singh, G. (2024). Concept drift–based intrusion detection for evolving data stream classification in ids: approaches and comparative study. *The Computer Journal*, *67*(7), 2529–2547.

Sethi, T. S., & Kantardzic, M. (2015). Don't pay for validation: Detecting drifts from unlabeled data using margin density. *Procedia Computer Science*, *53*, 103–112.

Sharma, K., Kumar, P., & Li, Y. (2025, November). OG-RAG: Ontology-grounded retrieval-augmented generation for large language models. In C. Christodoulopoulos, T. Chakraborty, C. Rose, & V. Peng (Eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 32962–32981). Suzhou, China: Association for Computational Linguistics. Retrieved 2026-06-01, from https://aclanthology.org/2025.emnlp-main.1674/ doi: 10.18653/v1/2025.emnlp-main.1674

Shi, L., Lin, D., Fang, C. V., & Zhai, Y. (2015, November). A Hybrid Learning from Multi-behavior for Malicious Domain Detection on Enterprise Network. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)* (pp. 987–996). Atlantic City, NJ, USA: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/7395774/ doi: 10.1109/ICDMW.2015.38

Shi, N., Yuan, X., Hernandez, J., Roy, K., & Esterline, A. (2018, December). Self-Learning Semi-Supervised Machine Learning for Network Intrusion Detection. In *2018 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 59–64). Las Vegas, NV, USA: IEEE. Retrieved 2026-06-17, from https://ieeexplore.ieee.org/document/8947898/ doi: 10.1109/CSCI46756.2018.00019

Shyaa, M. A., Ibrahim, N. F., Zainol, Z., Abdullah, R., Anbar, M., & Alzubaidi, L. (2024, November). Evolving cybersecurity frontiers: A comprehensive survey on concept drift and feature dynamics aware machine and deep learning in intrusion detection systems. *Engineering Applications of Artificial Intelligence*, *137*, 109143. Retrieved 2026-04-23, from https://www.sciencedirect.com/science/article/pii/S0952197624013010 doi: 10.1016/j.engappai.2024.109143

Shyaa, M. A., Zainol, Z., Abdullah, R., Anbar, M., Alzubaidi, L., & Santamaría, J. (2023). Enhanced intrusion detection with data stream classification and concept drift guided by the incremental learning genetic programming combiner. *Sensors*, *23*(7), 3736.

Street, W. N., & Kim, Y. (2001). A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the seventh acm sigkdd international conference on knowledge discovery and data mining* (pp. 377–382).

Ullah, S., Han, M., Pujar, S., Pearce, H., Coskun, A., & Stringhini, G. (2024). Llms cannot reliably identify and reason about security vulnerabilities (yet?): A comprehensive evaluation, framework, and benchmarks. In *2024 ieee symposium on security and privacy (sp)* (pp. 862–880).

Varal, A. S., & Wagh, S. K. (2018, July). Misuse and Anomaly Intrusion Detection System using Ensemble Learning Model. In *2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE)* (pp. 1722–1727). Bhubaneswar, India: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9009147/ doi: 10.1109/ICRIEECE44171.2018.9009147

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. *arXiv*

preprint arXiv:1710.10903.

Viegas, E. K., Santin, A. O., Cogo, V. V., & Abreu, V. (2020, June). A Reliable Semi-Supervised Intrusion Detection Model: One Year of Network Traffic Anomalies. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)* (pp. 1–6). Dublin, Ireland: IEEE. Retrieved 2026-06-17, from https://ieeexplore.ieee.org/document/9148916/ doi: 10.1109/ICC40277.2020.9148916

Wagh, S. K., & Kolhe, S. R. (2014, September). Effective intrusion detection system using semi-supervised learning. In *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)* (pp. 1–5). Delhi, India: IEEE. Retrieved 2026-06-17, from http://ieeexplore.ieee.org/document/6954236/ doi: 10.1109/ICDMIC.2014.6954236

Wang, Y., Yun, X., & Zhang, Y. (2015, November). Rethinking Robust and Accurate Application Protocol Identification: A Nonparametric Approach. In *2015 IEEE 23rd International Conference on Network Protocols (ICNP)* (pp. 134–144). San Francisco, CA, USA: IEEE. Retrieved 2026-06-17, from http://ieeexplore.ieee.org/document/7437123/ doi: 10.1109/ICNP.2015.43

Wang, Z., Sun, Y., Zhang, X., Xu, B., Yang, Z., & Lin, H. (2025, March). Continual learning with high-order experience replay for dynamic network embedding. *Pattern Recognition*, *159*, 111093. Retrieved 2026-04-23, from https://www.sciencedirect.com/science/article/pii/S0031320324008446 doi: 10.1016/j.patcog.2024.111093

Watkins, L., Beck, S., Zook, J., Buczak, A., Chavis, J., Robinson, W. H., ... Mishra, S. (2017, January). Using semi-supervised machine learning to address the Big Data problem in DNS networks. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)* (pp. 1–6). Retrieved 2026-06-18, from https://ieeexplore.ieee.org/abstract/document/7868376 doi: 10.1109/CCWC.2017.7868376

Xian, G. (2020). Cyber Intrusion Prevention for Large-Scale Semi-Supervised Deep Learning Based on Local and Non-Local Regularization. *IEEE Access*, *8*, 55526–55539. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9037285/ doi: 10.1109/ACCESS.2020.2981162

Yang, H., & Zhou, Z. (2018, July). A Novel Intrusion Detection Scheme Using Cloud Grey Wolf Optimizer. In *2018 37th Chinese Control Conference (CCC)* (pp. 8297–8302). Wuhan: IEEE. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/8483324/ doi: 10.23919/ChiCC.2018.8483324

Yang, P., Shan, C., Wang, D., Su, L., Li, J., & Wan, X. (2020, September). Mechanism of Parked Domains Recognition Based on Authoritative DNS Servers. In *Proceedings of the 2020 The 2nd World Symposium on Software Engineering* (pp. 128–134). Chengdu China: ACM. Retrieved 2026-06-18, from https://dl.acm.org/doi/10.1145/3425329.3425335 doi: 10.1145/3425329.3425335

Yang, S., Zheng, X., Li, J., Xu, J., Wang, X., & Ngai, E. C. H. (2024, August). ReCDA:

Concept Drift Adaptation with Representation Enhancement for Network Intrusion Detection. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 3818–3828). New York, NY, USA: Association for Computing Machinery. Retrieved 2026-04-23, from https://dl.acm.org/doi/10.1145/3637528.3672007 doi: 10.1145/3637528.3672007

Yu, Y., Guo, S., Lan, S., & Ban, T. (2009). Anomaly Intrusion Detection for Evolving Data Stream Based on Semi-supervised Learning. In M. Köppen, N. Kasabov, & G. Coghill (Eds.), *Advances in Neuro-Information Processing* (pp. 571–578). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-642-02490-0$_70$

Zavrak, S., & Iskefiyeli, M. (2020). Anomaly-Based Intrusion Detection From Network Flow Features Using Variational Autoencoder. *IEEE Access*, *8*, 108346–108358. Retrieved 2026-06-18, from https://ieeexplore.ieee.org/document/9113298/ doi: 10.1109/ACCESS.2020.3001350

Zhang, H., & Li, J. (2020, December). A New Network Intrusion Detection based on Semi-supervised Dimensionality Reduction and Tri-LightGBM. In *2020 International Conference on Pervasive Artificial Intelligence (ICPAI)* (pp. 35–40). Taipei, Taiwan: IEEE. Retrieved 2026-06-17, from https://ieeexplore.ieee.org/document/9302690/ doi: 10.1109/ICPAI51961.2020.00014

Zhang, L., Du, H., & Zhang, Y. (2019, October). Network Anomaly Detection Based on Cooperative Semi-Supervised Support Vector Machine. In *2019 International Conference on Networking and Network Applications (NaNA)* (pp. 171–180). Daegu, Korea (South): IEEE. Retrieved 2026-06-17, from https://ieeexplore.ieee.org/document/9027541/ doi: 10.1109/NaNA.2019.00039

Zhang, M., & Mei, H. (2012). A New Method for Filtering IDS False Positives with Semi-supervised Classification. In D.-S. Huang, C. Jiang, V. Bevilacqua, & J. C. Figueroa (Eds.), *Intelligent Computing Technology* (pp. 513–519). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-642-31588-6$_66$

Zhang, Y., Du, T., Ma, Y., Wang, X., Xie, Y., Yang, G., ... Chang, E.-C. (2025, March). AttacKG+: Boosting attack graph construction with Large Language Models. *Computers & Security*, *150*, 104220. Retrieved 2026-05-15, from https://www.sciencedirect.com/science/article/pii/S0167404824005261 doi: 10.1016/j.cose.2024.104220

Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., ... Zhang, Q. (2020). Multivariate time-series anomaly detection via graph attention network. In *2020 ieee international conference on data mining (icdm)* (pp. 841–850).

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., ... Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI open*, *1*, 57–81.

Zhou, K., Huang, X., Li, Y., Zha, D., Chen, R., & Hu, X. (2020). Towards deeper graph neural networks with differentiable group normalization. *Advances in neural information processing systems*, *33*, 4917–4928.

Zhu, X. J. (2005). *Semi-Supervised Learning Literature Survey* (Technical Report). University of Wisconsin-Madison Department of Computer Sciences. Retrieved 2026-06-18, from

https://minds.wisconsin.edu/handle/1793/60444 (Accepted: 2012-03-15T17:19:12Z)

Zhuo, S., https://orcid.org/0000-0001-5610-005X, View Profile, Wu, D., https://orcid.org/0000-0002-7788-9202, View Profile, ... View Profile (2025, September). Online Learning from Mix-typed, Drifted, and Incomplete Streaming Features. *ACM Transactions on Knowledge Discovery from Data*, *19*(8), 1–28. Retrieved 2026-04-23, from https://dl.acm.org/doi/10.1145/3744712 doi: 10.1145/3744712

Zimba, A., Chen, H., Wang, Z., & Chishimba, M. (2020, May). Modeling and detection of the multi-stages of Advanced Persistent Threats attacks based on semi-supervised learning and complex networks characteristics. *Future Generation Computer Systems*, *106*, 501–517. Retrieved 2026-06-18, from https://www.sciencedirect.com/science/article/pii/S0167739X19316267 doi: 10.1016/j.future.2020.01.032

Zola, F., Segurola-Gil, L., Bruse, J. L., Galar, M., & Orduna-Urrutia, R. (2022, April). Network traffic analysis through node behaviour classification: a graph-based approach with temporal dissection and data-level preprocessing. *Computers & Security*, *115*, 102632. Retrieved 2026-04-23, from https://www.sciencedirect.com/science/article/pii/S0167404822000311 doi: 10.1016/j.cose.2022.102632