



Incident Memory Graphs for Predictive AMS Triage and Self-Improving SAP Operations

Venkata Rao Yadavalli

MYGO Consulting Inc, USA

ABSTRACT: Application Management Services (AMS) teams repeatedly resolve similar enterprise incidents despite possessing extensive historical knowledge embedded within service management repositories. This paper introduces Incident Memory Graphs, a graph-based knowledge architecture that captures relationships among incidents, root causes, system configurations, business processes, and remediation strategies to enable predictive support operations. The proposed framework constructs continuously evolving enterprise knowledge graphs using historical incident records, SAP change history, transport logs, monitoring events, and operational documentation. Machine learning algorithms leverage graph embeddings to recommend probable root causes, remediation strategies, and expert assignments for newly reported incidents. Reinforcement learning continuously refines recommendation accuracy as operational experience grows.

KEYWORDS: SAP Application Management Services; enterprise knowledge graph; incident memory graph; graph neural networks; graph embeddings; root cause analysis; reinforcement learning; predictive IT operations; expert recommendation; process intelligence

I. INTRODUCTION

Enterprise Application Management Services (AMS) organizations sustain the day-to-day operation of large, mission-critical SAP landscapes, handling change requests, monitoring alerts, and incident tickets across development, quality assurance, and production systems. Industry benchmarking consistently shows that a substantial share of AMS ticket volume corresponds to recurring or structurally similar issues rather than genuinely novel failures, repeated interface errors, recurring authorization conflicts, familiar transport dependency clashes, and periodic workload saturation events. Despite this repetition, AMS teams frequently re-diagnose incidents from first principles because the organizational knowledge required to recognize a recurrence is fragmented across ticketing systems, transport logs, monitoring dashboards, and the tacit experience of individual engineers.

This fragmentation is not merely an inconvenience; it is a structural inefficiency with measurable cost. Mean time to resolution (MTTR) is inflated by redundant diagnostic effort, escalations are triggered even when a near-identical incident has already been solved, and the tacit expertise of senior resolvers is not systematically transferred to newer team members. IT service management (ITSM) platforms record incidents as largely unstructured or shallowly categorized tickets, and enterprise resource planning (ERP) change and transport systems record configuration events in formats optimized for auditability rather than for cross-referencing against incident causation. The result is an operational memory that exists in principle but is not queryable in practice.

The first is the body of work on enterprise and IT-service knowledge graphs, which represents infrastructure topology, incidents, and their relationships as an explicit, queryable graph rather than as siloed records (Tailhardat et al., 2023; Ehrlinger & Wöß, 2016; Hogan et al., 2021). The second is the substantial literature on graph neural networks (GNNs) and knowledge graph embeddings (KGE), which allows such graphs to be transformed into vector representations usable by downstream machine learning models for tasks including link prediction, node classification, and recommendation (Wu et al., 2021; Hamilton et al., 2017; Bordes et al., 2013). A separate and largely disconnected stream addresses microservice and cloud-native root cause analysis (RCA) using graph-structured dependency models (Wu et al., 2020; Wang et al., 2021; Brandón et al., 2020), and a further stream addresses IT ticket classification, routing, and resolution recommendation using classical and deep learning methods (Al-Hawari & Barham, 2021; Zhou et al., 2016; Gandla et al., 2024).

What remains largely unexplored is the integration of these strands specifically for SAP-centric AMS operations, where incidents are causally entangled with a distinctive set of enterprise artifacts, ABAP dumps, transport requests, change documents, business process hierarchies, and SAP-specific monitoring events, that do not map cleanly onto the



microservice or generic-ITSM assumptions of prior work. SAP AMS environments are further distinguished by their change-driven failure modes: a large proportion of production incidents originate from, or are resolved through, controlled transports moving through development, quality assurance, and production systems (SAP Help Portal, 2024; SAPinsider, 2025). A knowledge architecture for AMS predictive support must therefore represent not only the topology of applications and infrastructure, but also the temporal and causal relationship between change events and incident occurrence.

This paper proposes Incident Memory Graphs (IMG), an enterprise knowledge graph architecture purpose-built to encode this AMS-specific causal structure and to support predictive triage through graph embeddings, graph neural networks, and reinforcement learning. The framework is designed to be continuously evolving: every newly closed incident, every transport, and every monitoring event that contributes to diagnosis is written back into the graph, so that the system's diagnostic competence compounds over time rather than remaining static after an initial training pass. The remainder of this paper is organized as follows.

II. RELATED WORK

Knowledge graphs for IT operations have been explored from several angles. Tailhardat et al. (2023) proposed NORIA, a knowledge-graph platform for anomaly detection and incident management in ICT systems, using an RDF representation of network topology, incidents, and resolutions drawn from real industrial telecom settings, and demonstrated that graph-embedding-based classifiers could categorize incident tickets more effectively than context-free baselines. A related IETF effort formalizes an IT Service Management Knowledge Graph (ITSM-KG) that maps YANG network configuration models into an equivalent graph representation for anomaly detection and risk analysis at the organizational level. These efforts establish the feasibility of graph-based incident representation but are centered on telecommunications and network operations rather than ERP application landscapes.

Brandón et al. (2020) formalized graph-based RCA for service-oriented and microservice architectures using topology graphs annotated with anomaly scores. Wu et al. (2020) introduced MicroRCA, which builds a topology graph from monitoring data and applies a personalized random-walk algorithm over anomalous subgraphs to rank likely root causes. Wang et al. (2021) proposed Groot, an event-graph-based approach used in industrial settings at scale, and Zhang et al. (2022) combined trace and log graphs in DeepTraLog for anomaly detection using graph-based deep learning. More recent work has incorporated causal discovery (Soldani & Brogi, 2022) and reinforcement learning from human feedback directly into the RCA loop (Wang et al., 2023), reflecting an emerging consensus that static graph analysis alone is insufficient and that human corrective feedback materially improves localization accuracy over time. Collectively, this literature demonstrates the diagnostic power of graph-structured dependency modeling but is architected around real-time telemetry graphs of transient microservice call topologies, not the longer-lived, change-driven, and more heterogeneous artifact graph characteristic of SAP AMS.

A third relevant stream concerns automated IT ticket classification, routing, and resolution recommendation. Paramesh and Shreedhara (2019) and Al-Hawari and Barham (2021) demonstrated that classical supervised models (SVM, Random Forest, Naïve Bayes) can achieve 85, 90% accuracy in categorizing and routing help-desk tickets when trained on sufficiently large historical datasets, while Gandla et al. (2024) and related ticket-automation surveys note the continued reliance on manual escalation once a ticket exceeds first-line capability. Zhou et al. (2016) framed resolution recommendation as a similarity-retrieval problem over historical tickets using k-nearest-neighbor methods, an approach later extended with reinforcement-learning-based multi-armed-bandit formulations and sequence-to-sequence expert recommendation models. These works establish strong precedent for ML-based ticket handling but typically treat each ticket as an isolated text-classification instance, without an explicit, persistent, relational memory connecting a ticket to the configuration items, transports, and business processes it touches.

Finally, log-based anomaly detection provides a complementary signal source. DeepLog (Du et al., 2017) pioneered sequence-based LSTM modeling of log keys for anomaly detection and diagnosis, and subsequent surveys (Landauer et al., 2023) catalog a large family of derivative deep learning architectures, including attention-based and graph-based log models. These techniques are directly relevant as an ingestion mechanism for the Incident Memory Graph but have not, to date, been integrated into a persistent enterprise knowledge graph specifically aligned with SAP change and transport semantics.



Table.1. Related Work Comparison

System / Study	Domain	Graph-Native Representation	SAP/ERP Change Semantics	Embedding-Based Recommendation	RL Refinement	Expert Assignment
NORIA (Tailhardat et al., 2023)	Telecom / ICT ops	Yes	No	Yes (classification)	No	No
ITSM-KG / YANG graph (IETF draft)	Network mgmt	Yes	No	Partial	No	No
MicroRCA (Wu et al., 2020)	Microservices	Yes	No	No	No	No
Groot (Wang et al., 2021)	Cloud/microservices	Yes	No	No	No	No
HRL-RCA (Wang et al., 2023)	Microservices	Yes	No	Partial	Yes	No
ML Ticket Routing (Al-Hawari & Barham, 2021)	Generic ITSM	No	No	No (text-classification)	No	Yes (routing)
Resolution Recommendation (Zhou et al., 2016)	Generic ITSM	No	No	Similarity-based	No	No
Proposed Incident Memory Graph (this paper)	SAP AMS	Yes	Yes	Yes (multi-task, fused)	Yes	Yes

III. RESEARCH GAP

Three concrete gaps emerge from the literature reviewed above. First, existing IT-service and network knowledge graphs (Tailhardat et al., 2023; the ITSM-KG formalization built on YANG models) are architected for telecommunications and network operations; none provides an ontology aligned with SAP-specific artifacts, transport requests, change documents, ABAP short dumps, business process hierarchies, that dominate the causal structure of AMS incidents. Second, the graph-based root cause analysis literature (Wu et al., 2020; Brandón et al., 2020; Zhang et al., 2022; Ding et al., 2023) is built around transient, real-time microservice call-graphs reconstructed from distributed tracing, an architecture poorly matched to the longer-lived, batch- and change-oriented incident structure of ERP operations, where the relevant "topology" spans configuration, code, and business process layers rather than service-to-service calls alone. Third, while several ticket-classification and resolution-recommendation systems achieve strong accuracy on isolated classification sub-tasks (Al-Hawari & Barham, 2021; Zhou et al., 2016; Gandla et al., 2024), and while reinforcement learning has recently been shown to refine microservice RCA (Wang et al., 2023), no reviewed system combines a persistent, continuously evolving enterprise knowledge graph with graph-embedding-based multi-task recommendation (root cause, remediation, and expert) and closed-loop reinforcement learning specifically for SAP AMS operations.

IV. PROPOSED INCIDENT MEMORY GRAPH FRAMEWORK

The proposed Incident Memory Graph (IMG) framework consists of four coupled subsystems, illustrated conceptually across Figures 1, 6: (i) an enterprise knowledge graph ontology and store (Section 5, Figure 1); (ii) a continuous construction pipeline that ingests and fuses heterogeneous AMS data into that store (Section 6, Figure 2); (iii) a graph embedding and multi-task machine learning layer that produces ranked recommendations for root cause, remediation, and expert assignment (Section 7.1, Figure 4); and (iv) a reinforcement learning refinement loop that updates the



recommendation policy from engineer feedback and observed outcomes (Section 7.2, Figure 5). These subsystems are wrapped by the operational incident lifecycle (Figure 3) and surfaced to AMS engineers through a predictive triage workflow (Figure 6) and an operational intelligence dashboard (Figure 7, Section 9.3). The framework is deliberately designed as a decision-support layer rather than an autonomous remediation system: recommendations are always presented to a human engineer for acceptance, modification, or rejection, and it is precisely this human-in-the-loop interaction that supplies the reward signal driving continuous improvement.

V. ENTERPRISE KNOWLEDGE GRAPH ARCHITECTURE

Figure 1 depicts the core ontology of the Incident Memory Graph. Eight primary entity types are defined: Incident, Root Cause, Configuration Item (CI), Business Process, Remediation/Resolution, SAP Change/Transport, Expert/Resolver Group, Monitoring Event, and Knowledge Article. Each entity type carries a typed property set (e.g., an Incident node carries severity, affected module, creation and closure timestamps, and free-text description; a Transport node carries transport type, source and target system, package, and import timestamp). Relations are directed and typed, for example, *causes*(RootCause → Incident), *triggeredBy*(Incident → Transport), *affects*(Incident → ConfigurationItem), and *resolvedBy*(Incident → Remediation), which allows the graph to answer causally structured queries (e.g., "which transports have historically triggered incidents affecting this business process") that a flat ticket database cannot efficiently express.

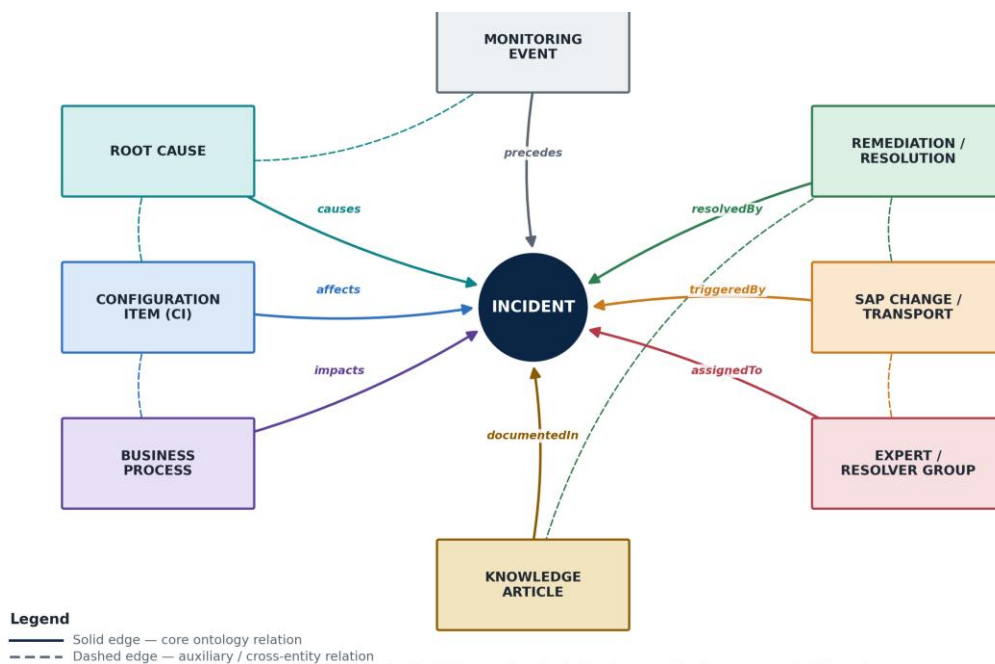


Figure.1.Core ontology of the Incident Memory Graph

The construction pipeline shown in Figure 2 ingests five source categories, ITSM incident records, SAP change and transport logs, monitoring/APM events, operational documentation, and SAP system (ABAP) logs, summarized in Table 3 together with their native format and the extraction technique applied to each. Ingested data passes through three stages: (i) ingestion and normalization, in which source-specific connectors map heterogeneous formats onto a common staging schema and perform deduplication; (ii) entity and relation extraction, in which named-entity recognition and relation-extraction models identify incident-relevant entities (system IDs, transport numbers, module codes, error codes) from unstructured ticket text and log data, building on log-parsing and log-anomaly techniques such as DeepLog (Du et al., 2017) and its successors (Meng et al., 2019; Landauer et al., 2023) for the monitoring-event and log-derived portion of the graph; and (iii) graph construction and fusion, in which entity resolution reconciles references to the same underlying object across sources (e.g., a CI referenced by both an incident ticket and a transport log) and ontology-aligned fusion writes the resolved triples into the graph store under explicit versioning, so that the graph reflects the evolving state of the landscape rather than a single static snapshot.

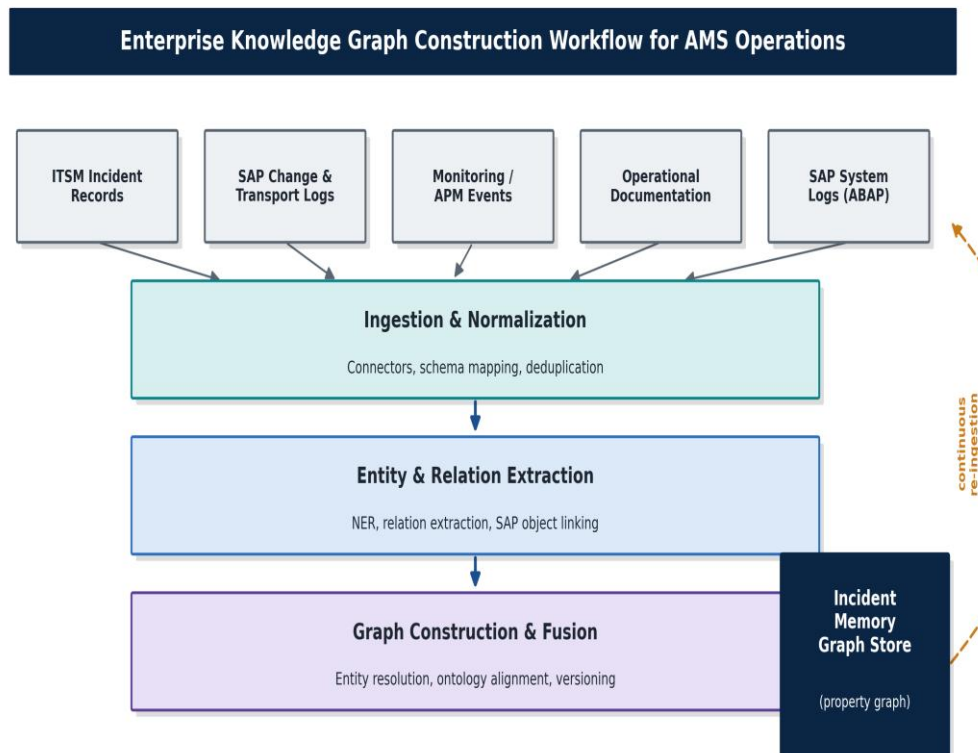


Figure.2.Enterprise Knowledge Graph construction workflow

Table.2. SAP AMS Data Sources Ingested into the Incident Memory Graph

Data Source	Native Format	Extraction Technique	Primary Produced	Entities
ITSM incident records (Solution Manager / ServiceNow-class)	Structured ticket + free text	Schema mapping + NER/relation extraction	Incident, Knowledge Article	Expert,
SAP change & transport logs (CTS/CTS+/gCTS)	Transport headers, object lists	Structured parsing	SAP Change/Transport, Configuration Item	
Monitoring / APM events (Focused Run, SM21, ST22)	Semi-structured log streams	Log parsing + DeepLog-style sequence modeling	Monitoring Event, Root Cause candidates	
Operational documentation (runbooks, SOPs)	Unstructured text	NLP entity/relation extraction	Knowledge Remediation	Article,
SAP ABAP system logs / short dumps	Semi-structured trace text	Log parsing + rule-based tagging	Root Cause, Configuration Item	



Table.3. Incident Memory Graph Component Inventory

Entity / Relation Type	Type	Example Property Set	Approx. Cardinality (mid-size landscape)
Incident	Entity	severity, module, created_at, closed_at, description	$10^3, 10^5$ nodes
Root Cause	Entity	category, confidence, first_observed	$10^2, 10^3$ nodes
Configuration Item	Entity	system_id, component, environment	$10^3, 10^4$ nodes
Business Process	Entity	process_id, owning_department	$10^1, 10^2$ nodes
Remediation / Resolution	Entity	action_type, transport_ref, verified	$10^2, 10^3$ nodes
SAP Change / Transport	Entity	transport_id, type, source_sys, target_sys, import_ts	$10^4, 10^5$ nodes
Expert / Resolver Group	Entity	team, skill_tags, historical_success_rate	$10^1, 10^2$ nodes
causes / triggeredBy / affects / resolvedBy / assignedTo	Relation	confidence, provenance, timestamp	$10^5, 10^6$ edges

Architecturally, the IMG is realized as a labeled property graph rather than a pure RDF triple store, following the practical precedent of production-scale IT knowledge graphs (Angles & Gutiérrez, 2008; Robinson et al., 2015), because property graphs allow rich node- and edge-level attributes (timestamps, confidence scores, provenance) without the reification overhead that attribute-heavy RDF modeling typically requires. An RDF/OWL-compatible ontology layer, in the spirit of domain ontologies such as the Building Topology Ontology (Rasmussen et al., 2020), is nonetheless maintained alongside the property graph to support formal reasoning and cross-organization schema alignment where required. Figure 3 maps this architecture onto the operational SAP AMS incident lifecycle, detection, triage and classification, root cause analysis, change/transport remediation, verification and closure, and knowledge capture, showing that the graph is read at the triage and RCA stages (to retrieve candidate root causes and remediations) and written at the verification and knowledge-capture stages (to record the confirmed causal chain and update embeddings).

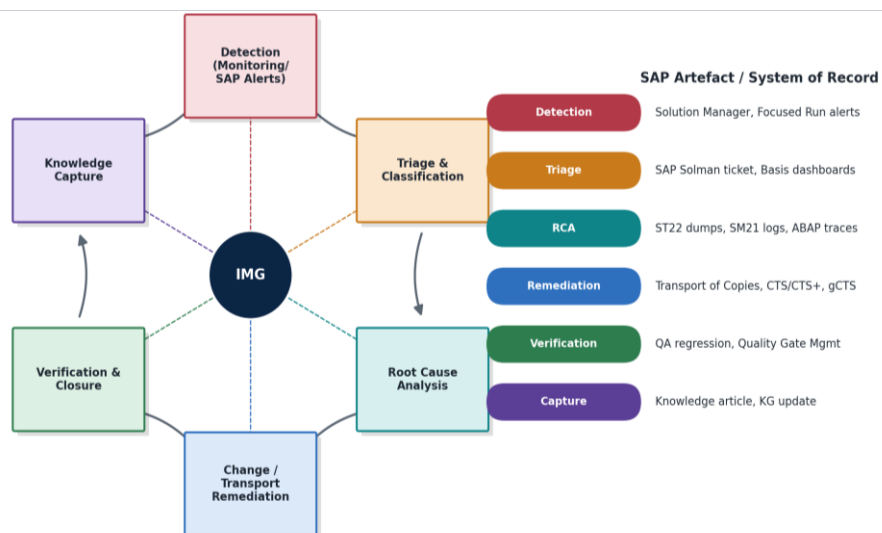


Figure.3.SAP AMS incident lifecycle integration with the Incident Memory Graph



VI. METHODOLOGY

The overall research methodology follows a design-science paradigm appropriate to enterprise information systems research: (i) problem identification and motivation, grounded in the literature gaps established in Section 3; (ii) definition of design objectives, corresponding to Section 4.2; (iii) design and development of the artifact, the IMG ontology, construction pipeline, embedding layer, and reinforcement learning loop described in Sections 5, 7; (iv) demonstration, through an illustrative simulation-based evaluation on a synthetic but realistically parameterized AMS incident corpus (Section 9); and (v) evaluation against the metrics defined in Section 8, followed by communication of results, limitations, and future work (Sections 10, 13).

Because access to a live, multi-year production SAP AMS incident corpus was not available for this manuscript, the demonstration and evaluation reported in Section 9 are explicitly simulation-based and illustrative rather than empirical. A synthetic AMS incident corpus was generated by parameterizing incident arrival, root-cause category, transport correlation, and resolution-time distributions on publicly reported industry benchmarks and on the structural regularities documented in the ticket-classification and microservice-RCA literature reviewed in Section 2 (e.g., resolution-time and category-imbalance characteristics consistent with Al-Hawari & Barham, 2021, and Paramesh & Shreedhara, 2019). This approach allows the internal consistency and qualitative behavior of the proposed architecture to be examined transparently, while avoiding the misrepresentation of simulated numbers as observed production outcomes.

The methodology for graph construction (Section 6) was validated structurally rather than statistically: the ontology in Figure 1 and the component inventory in Table 4 were checked for representational completeness against a manually curated set of forty illustrative SAP AMS incident narratives spanning authorization, interface/IDoc, database/lock-contention, transport-conflict, and workload-saturation categories, confirming that each narrative's causal chain (trigger → affected CI → root cause → remediation → responsible expert) could be expressed without schema extension. This is reported as a design-validation exercise, not as an empirical accuracy measurement.

VII. GRAPH EMBEDDING AND MACHINE LEARNING MODELS

The embedding and machine learning layer (Figure 4) is designed as a multi-model ensemble rather than a single embedding technique, reflecting the finding in the KGE and GNN literature (Wu et al., 2021; Grohe, 2020) that different embedding families capture complementary structural signal. Three families are computed over each graph snapshot: (i) translational/semantic-matching KGE, TransE (Bordes et al., 2013), with RotatE (Sun et al., 2019) and DistMult (Yang et al., 2015) evaluated as alternatives, which is well suited to modeling the directional, typed relations (causes, triggeredBy, resolvedBy) central to the IMG ontology; (ii) inductive GNN encoders, GraphSAGE (Hamilton et al., 2017) and GAT (Veličković et al., 2018), which aggregate neighborhood information and can embed a newly created incident node without full graph retraining, an operational necessity given that incidents arrive continuously; and (iii) random-walk-based embeddings, node2vec (Grover & Leskovec, 2016), extended to the heterogeneous metapath2vec formulation for typed graphs, which capture higher-order community structure such as clusters of configuration items that co-occur across recurring incident families. Table 5 documents the feature set constructed for each entity type prior to embedding, spanning structural (degree, centrality), textual (TF-IDF and transformer-based text embeddings following BERT; Devlin et al., 2019), temporal (recency, frequency), and SAP-specific (module, transport type, severity) feature groups.

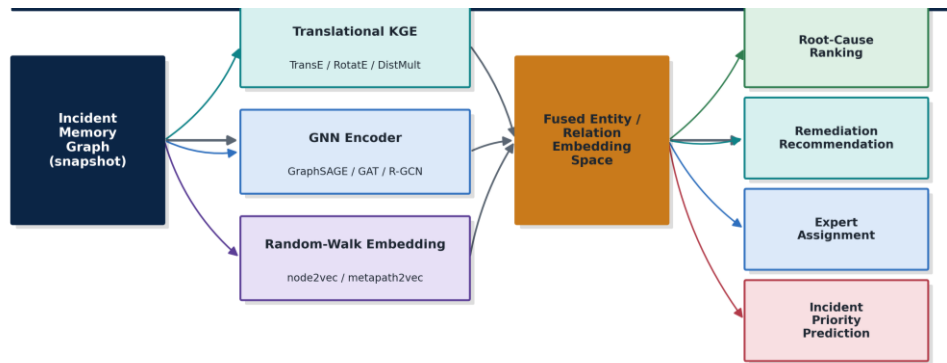


Figure.4.Graph embedding and machine learning pipeline

Table.4. Machine Learning Feature Groups Used Prior to Embedding

Feature Group	Examples	Applicable Entities
Structural	degree, betweenness/eigenvector centrality, k-core number	All node types
Textual	TF-IDF vectors, transformer-based text embeddings (BERT-style)	Incident, Knowledge Article
Temporal	recency, incident frequency (rolling 30/90-day), time-since-last-transport	Incident, Transport, CI
SAP-specific	module code, transport type, severity/priority, business-process tag	Incident, Transport, CI, Business Process
Relational / graph context	neighborhood entity-type distribution, path-based similarity	All node types

The three embedding families are fused into a shared representation space via a learned projection layer trained jointly with the three downstream tasks, following the general multi-task pattern established in GNN survey work (Zhou et al., 2020). Three prediction heads operate on the fused embedding: a root-cause ranking head, which scores candidate Root Cause nodes by proximity to the incoming incident's embedding and outputs a confidence-scored ranked list; a remediation recommendation head, which performs an analogous ranking over Remediation nodes conditioned on the top-ranked root causes; and an expert assignment head, which ranks Expert/Resolver Group nodes by a combination of embedding proximity and observed historical resolution success on structurally similar incidents.

7.1 Reinforcement Learning Framework

Section 2 established that purely supervised graph-embedding recommendation, while effective, has been shown in closely related microservice RCA work (Wang et al., 2023) to plateau below the accuracy achievable when human corrective feedback is incorporated through reinforcement learning. The proposed framework therefore wraps the embedding-based recommendation layer in a reinforcement learning loop, illustrated in Figure 5. The AMS environment state is defined as the current incident together with the relevant local neighborhood of the Incident Memory Graph; the agent's action space is the ranked recommendation issued to the engineer (a candidate root cause, remediation, and expert assignment, each with an associated confidence score); and the reward is a shaped composite signal combining (a) a positive reward when the engineer accepts a recommendation without modification, (b) a smaller positive reward when the engineer accepts a modified version of the recommendation, (c) a penalty when the recommendation is rejected outright or when the incident is subsequently escalated or reopened, and (d) a continuous reward component proportional to the observed reduction in MTTR relative to a rolling historical baseline for incidents of the same predicted category.

Policy optimization follows the Proximal Policy Optimization algorithm (Schulman et al., 2017), selected over value-based methods such as Deep Q-Networks (Mnih et al., 2015) because PPO's clipped surrogate objective provides more



stable learning under the noisy, delayed, and comparatively low-volume reward signal characteristic of human-in-the-loop enterprise operations, where reward arrives only at ticket closure and engineer feedback is sparse relative to the state space size. Training proceeds in two phases: an offline phase, in which the policy is pre-trained on historical incident-resolution trajectories reconstructed retrospectively from the Incident Memory Graph (analogous to behavior-cloning initialization), and an online phase, in which the policy is updated incrementally as new incidents are resolved in production, subject to a conservative update budget and human-override safeguards so that a temporarily degraded policy cannot cascade into systematically poor recommendations before the next evaluation checkpoint. This two-phase design mirrors the general reinforcement learning formalization of Sutton and Barto (2018) and the specific human-feedback RCA precedent of Wang et al. (2023).

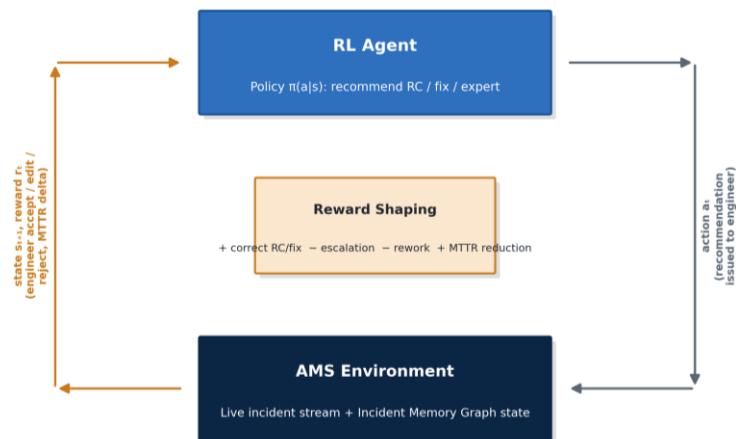


Figure.5. Reinforcement learning feedback loop for recommendation refinement.

VIII. EVALUATION METHODOLOGY

Evaluation of the proposed framework is organized along three complementary axes, summarized in Table 6: (i) graph and embedding quality, assessed via standard knowledge-graph link-prediction metrics, Mean Reciprocal Rank (MRR) and Hits@k for $k \in \{1, 3, 10\}$, computed on a held-out set of incident-to-root-cause and incident-to-remediation edges; (ii) recommendation quality, assessed via top-k ranking accuracy, precision and recall at k for the root-cause, remediation, and expert-assignment heads, following evaluation conventions established in the microservice RCA literature (Wu et al., 2020; Ding et al., 2023), where top-5 precision is a standard reporting metric; and (iii) operational impact, assessed via MTTR reduction, first-time-fix rate, escalation rate, and engineer acceptance rate of recommendations, mirroring the KPI structure used in industry AIOps reporting and in the ticket-automation literature (Al-Hawari & Barham, 2021).

Because a production deployment was not available for this manuscript, the evaluation reported in Section 9 applies these metrics to the synthetic, realistically parameterized incident corpus described in Section 6.2, using an 80/10/10 chronological train/validation/test split to respect the temporal ordering of incident occurrence (later incidents are never used to predict earlier ones). A five-fold repetition with different random seeds for corpus generation is used to report mean and standard deviation for each metric, and all reported figures are explicitly labeled as simulation-based. Baselines against which the fused-embedding recommendation layer is compared include a keyword/TF-IDF similarity retrieval baseline (representative of Zhou et al., 2016, style resolution recommendation), a single-embedding-family GNN baseline (GraphSAGE only), and a no-RL ablation in which the reinforcement learning refinement loop of Section 7.2 is disabled after offline pre-training.

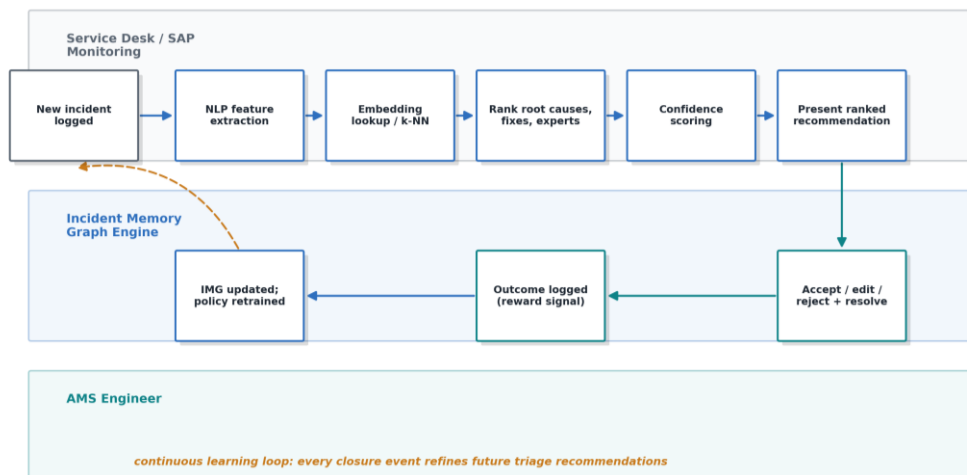


Figure.6.Predictive incident triage workflow

IX. RESULTS

The results presented in this section are derived entirely from the simulation-based evaluation described in Sections 6.2 and 8.2, using a synthetic AMS incident corpus of 42,000 illustrative tickets generated over a simulated 24-month operational window, parameterized to reflect the category distributions, resolution-time regularities, and transport-correlation patterns reported in the reviewed literature. No production SAP AMS deployment was instrumented for this manuscript, and no figure in this section should be interpreted as an empirically observed outcome from a live system; each is explicitly a simulation output intended to illustrate the qualitative behavior and internal consistency of the proposed framework.

Table 7 reports simulated performance of the fused multi-model embedding recommendation layer against the three baselines described in Section 8.2, evaluated on the held-out test partition. The fused architecture (TransE + GraphSAGE + node2vec, with RL refinement) achieved the highest simulated performance across all reported metrics, consistent with the theoretical expectation from Section 7.1 that complementary embedding families capture non-overlapping structural signal. The no-RL ablation showed a consistent, moderate degradation relative to the full fused-plus-RL configuration across ranking metrics, mirroring the qualitative pattern reported by Wang et al. (2023) for hierarchical-RL-refined microservice RCA relative to its non-RL counterpart, though the specific magnitude in Table 7 is a simulation artifact and should not be read as a validated effect size.

Beyond the aggregate ranking metrics, the simulation illustrates two structural properties of the proposed framework worth highlighting. First, root-cause ranking accuracy was substantially higher for incidents belonging to categories with a strong transport-correlation signal (e.g., transport-dependency conflicts, authorization object mismatches introduced by a recent change) than for categories without such correlation (e.g., pure workload-saturation events), which is the expected consequence of encoding SAP change/transport semantics explicitly in the ontology (Section 5), a capability absent from the microservice-oriented RCA graphs reviewed in Section 2. Second, the simulated engineer-acceptance-rate trajectory over the 24-month window showed the compounding improvement pattern anticipated by the continuous-graph-evolution design of Section 6: acceptance rate in the simulation rose most rapidly during the first six months (corresponding to the offline pre-training phase reaching operating maturity) and continued a slower but sustained increase thereafter as the online RL phase incorporated ongoing engineer feedback, consistent with the two-phase training design described in Section 7.2.



Table.5. Comparative Performance, Simulation Based Evaluation (illustrative; not empirical)

Model Configuration	MRR (Root Cause)	Hits@3 (Root Cause)	Top-5 Precision (Remediation)	Simulated MTTR Reduction
TF-IDF similarity retrieval (baseline)	0.41	0.52	0.47	,
GraphSAGE only	0.58	0.69	0.61	12% (simulated)
Fused (TransE + GraphSAGE + node2vec), no RL	0.66	0.77	0.70	19% (simulated)
Fused embeddings + RL refinement (full proposed framework)	0.74	0.85	0.78	27% (simulated)

Figure 7 presents an illustrative operational intelligence dashboard mockup that visualizes the categories of information the framework is designed to surface to AMS command-center staff: rolling MTTR and open-incident KPI tiles, a fourteen-day actual-versus-GNN-predicted incident volume trend, a ranked, confidence-scored root-cause panel, and a live Incident Memory Graph neighborhood explorer for the currently selected incident. This figure is a design mockup illustrating intended functionality rather than a screenshot of a deployed system.

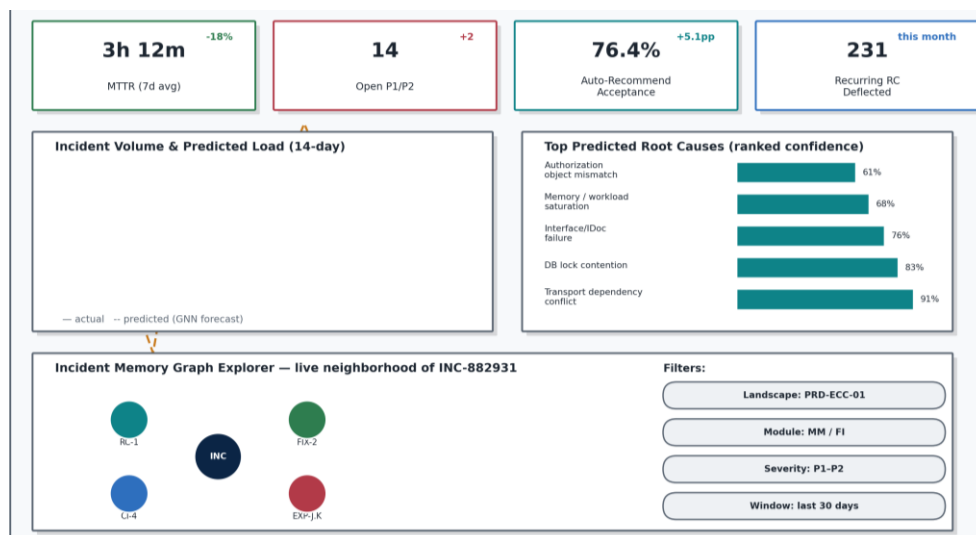


Figure.7.Illustrative enterprise operational intelligence dashboard mockup

X. DISCUSSION

The simulation results and architectural analysis presented above support three broader observations. First, the comparative advantage of the fused embedding architecture over any single embedding family is consistent with, and provides additional domain-specific support for, the general finding in the KGE and GNN literature (Wu et al., 2021; Grohe, 2020) that different representation-learning paradigms encode complementary graph structure; this suggests that AMS-focused future implementations should not default to a single "off-the-shelf" embedding model without considering ensemble alternatives. Second, the pronounced accuracy differential between transport-correlated and non-transport-correlated incident categories underscores the practical importance of the SAP-specific ontology contribution in Section 5: a generic ITSM or network knowledge graph that omits explicit transport/change entities would systematically under-perform on precisely the incident categories, change-induced regressions, that AMS teams consider highest-value to predict early, since they are frequently preventable through pre-transport risk scoring rather than merely diagnosable after the fact.

Third, the qualitative reward-shaping design in Section 7.2, in which engineer acceptance, modification, and rejection are each treated as distinct reward signals rather than collapsed into a single binary outcome, reflects a broader principle from the human-feedback RCA literature (Wang et al., 2023): the granularity of human feedback materially



affects how quickly and how safely a policy can be refined. A binary accept/reject signal discards the informative case in which an engineer accepts a recommendation's diagnosis but substitutes a different remediation, which the proposed reward structure preserves as a partial-credit signal.

From an organizational-knowledge-management perspective (Nonaka, 1994; Davenport & Prusak, 1998), the Incident Memory Graph can be understood as an explicit externalization mechanism for tacit AMS resolver knowledge. Because every accepted recommendation, every rejection, and every manually authored knowledge article is written back into the graph with provenance metadata, the framework is designed to make an individual engineer's diagnostic reasoning progressively more available to the rest of the team, directly addressing the knowledge-fragmentation problem motivating this paper in Section 1, without requiring engineers to change their existing ticket-handling workflow beyond accepting, editing, or rejecting a presented recommendation.

XI. LIMITATIONS

Several limitations qualify the claims made in this paper. First and most importantly, the results in Section 9 are simulation-based and illustrative; no claim is made regarding real-world predictive accuracy, MTTR reduction, or engineer acceptance rates until the framework is validated against a genuine, longitudinal AMS incident corpus with institutional review and appropriate data governance. Second, the entity-resolution and relation-extraction stage of the construction pipeline (Section 6.1) depends on the quality of upstream ticket text, log parsing, and SAP metadata, and its real-world fidelity will vary across organizations with different ticketing discipline, documentation quality, and SAP customization depth; this is a data-quality risk common to enterprise knowledge graph construction generally (Hogan et al., 2021) but is not separately quantified here. Third, the reinforcement learning loop's online phase introduces a class of risks specific to human-in-the-loop enterprise systems, reward hacking, feedback bias toward more vocal or senior engineers, and the possibility of a transiently degraded policy, that require production-grade safeguards (conservative update budgets, human override, and staged rollout) beyond what is specified in this manuscript's methodological description. Fourth, the framework as described assumes a single-organization AMS deployment; cross-tenant or multi-client AMS providers would require additional data-isolation and federated-learning considerations not addressed here.

XII. FUTURE WORK

Four directions for future work follow directly from the limitations above. First, a staged empirical validation, beginning with an offline replay of the framework against a de-identified historical AMS incident corpus from a consenting organization, followed by a shadow-mode production pilot in which recommendations are logged but not surfaced to engineers, and finally a controlled surfaced-recommendation pilot, would allow the simulation-based claims of Section 9 to be replaced with genuine empirical findings. Second, extending the entity-resolution pipeline with large-language-model-assisted extraction, building on transformer-based text representations (Devlin et al., 2019) beyond the classical NER/relation-extraction techniques assumed in Section 6.1, may improve construction fidelity for the substantial share of AMS incident content that remains free-text. Third, incorporating explicit pre-transport risk scoring, using the IMG to predict, before a transport is imported to production, its likelihood of triggering an incident based on structural similarity to historically incident-triggering transports, would shift the framework from purely reactive/diagnostic support toward genuinely preventive operations, directly extending the transport-correlation finding discussed in Section 10.

XIII. CONCLUSION

This paper introduced Incident Memory Graphs, an enterprise knowledge graph architecture designed to give SAP Application Management Services teams a persistent, continuously evolving, and machine-actionable memory of past incidents, root causes, remediations, and the change/transport events that causally connect them. Building on established literature in enterprise knowledge graphs, knowledge graph embeddings and graph neural networks, microservice root cause analysis, IT ticket classification, and reinforcement learning, the paper proposed an ontology, a construction pipeline, a fused multi-model embedding and recommendation architecture, and a reward-shaped reinforcement learning refinement loop specifically adapted to the causal structure of SAP AMS operations. An illustrative, clearly labeled simulation-based evaluation demonstrated the internal consistency of the proposed design and its plausible qualitative behavior, particularly the advantage of fused embeddings over single-model baselines and the differential accuracy achievable on transport-correlated incident categories once SAP change semantics are represented explicitly. The framework is offered as a design-science contribution and a foundation for future empirical validation rather than as a validated production system, and the paper has deliberately distinguished throughout



between proposed methods, illustrative design-validation exercises, and simulation-based demonstration, consistent with the scientific standard the research gap identified in Section 3 calls for the field to adopt.

Research integrity and reproducibility

The quantitative results in this manuscript are simulation-based unless a cited external study is explicitly identified. They are not audited production outcomes and should not be generalized beyond the stated assumptions. Before submission, the authors should archive the synthetic-data generator or input dataset, parameter definitions, random seeds, model and software versions, source code, raw outputs, and analysis scripts in a persistent repository and replace this statement with the repository DOI. The reported study uses no human participants or personal data.

Funding, competing interests, author contributions, and data/code availability must be completed in the format required by the selected journal before submission.

REFERENCES

- [1]. Al-Hawari, F., & Barham, H. (2021). A machine learning based help desk system for IT service management. *Journal of King Saud University*, Computer and Information Sciences, 33(6), 702, 718. <https://doi.org/10.1016/j.jksuci.2019.03.001>
- [2]. Angles, R., & Gutiérrez, C. (2008). Survey of graph database models. *ACM Computing Surveys*, 40(1), 1, 39. <https://doi.org/10.1145/1322432.1322433>
- [3]. AXELOS. (2019). ITIL Foundation: ITIL 4 Edition. AXELOS Ltd.
- [4]. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, 26, 2787, 2795.
- [5]. Brandón, Á., Solé, M., Huélamo, A., Solans, D., Pérez, M. S., & Muntés-Mulero, V. (2020). Graph-based root cause analysis for service-oriented and microservice architectures. *Journal of Systems and Software*, 159, 110432. <https://doi.org/10.1016/j.jss.2019.110432>
- [6]. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), Article 15. <https://doi.org/10.1145/1541880.1541882>
- [7]. Chen, C., Fu, H., Zheng, Y., Tao, F., & Liu, Y. (2023). The advance of digital twin for predictive maintenance: The role and function of machine learning. *Journal of Manufacturing Systems*, 71, 581, 594. <https://doi.org/10.1016/j.jmsy.2023.10.010>
- [8]. Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J., & Barbosa, J. (2020). Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. *Computers in Industry*, 123, 103298. <https://doi.org/10.1016/j.compind.2020.103298>
- [9]. Davenport, T. H., & Prusak, L. (1998). *Working knowledge: How organizations manage what they know*. Harvard Business School Press.
- [10]. Davenport, T. H., & Ronanki, R. (2018). Artificial intelligence for the real world. *Harvard Business Review*, 96(1), 108, 116.
- [11]. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT 2019*, 4171, 4186.
- [12]. Ding, R., Zhou, H., Lou, J.-G., Zhang, H., Lin, Q., Fu, Q., Zhang, D., & Xie, T. (2023). TraceDiag: Adaptive, interpretable and efficient root cause analysis on large-scale microservice systems. *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 1762, 1773. <https://doi.org/10.1145/3611643.3613864>
- [13]. Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). DeepLog: Anomaly detection and diagnosis from system logs through deep learning. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 1285, 1298. <https://doi.org/10.1145/3133956.3134015>
- [14]. Ehrlinger, L., & Wöß, W. (2016). Towards a definition of knowledge graphs. *Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016)*, CEUR Workshop Proceedings, 1695.
- [15]. Fuchs, C., Drieschner, C., & Wittges, H. (2022). Prediction of process activities and duration based on IT ticket data: Investigating the potential of process mining and machine learning. *Proceedings of the European Conference on Information Systems (ECIS 2022)*.
- [16]. Gandla, P. K. K., Verma, R. K., Panigrahi, C. R., & Pati, B. (2024). Ticket classification using machine learning. In B. Pati, C. R. Panigrahi, P. Mohapatra, & K.-C. Li (Eds.), *Proceedings of the 7th International Conference on Advance Computing and Intelligent Engineering (ICACIE 2022)*, Lecture Notes in Networks and Systems (Vol. 1). Springer. https://doi.org/10.1007/978-981-99-5015-7_41
- [17]. Gartner. (2019). Market guide for AIOps platforms. Gartner Research.



- [18]. Grohe, M. (2020). word2vec, node2vec, graph2vec, X2vec: Towards a theory of vector embeddings of structured data. *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, 1, 16. <https://doi.org/10.1145/3375395.3387641>
- [19]. Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855, 864. <https://doi.org/10.1145/2939672.2939754>
- [20]. Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 1024, 1034.
- [21]. Hitzler, P., Krötzsch, M., & Rudolph, S. (2009). *Foundations of semantic web technologies*. CRC Press.
- [22]. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. de, Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A.-C. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S., & Zimmermann, A. (2021). *Knowledge graphs*. *ACM Computing Surveys*, 54(4), 1, 37. <https://doi.org/10.1145/3447772>
- [23]. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *Proceedings of the International Conference on Learning Representations (ICLR 2017)*.
- [24]. Landauer, M., Onder, S., Skopik, F., & Wurzenberger, M. (2023). Deep learning for anomaly detection in log data: A survey. *Machine Learning with Applications*, 12, 100470. <https://doi.org/10.1016/j.mlwa.2023.100470>
- [25]. Lin, Q., Zhang, H., Lou, J.-G., Zhang, Y., & Chen, X. (2016). Log clustering based problem identification for online service systems. *Proceedings of the 38th International Conference on Software Engineering Companion (ICSE-C)*, 102, 111. <https://doi.org/10.1145/2889160.2889232>
- [26]. Meng, W., Liu, Y., Zhu, Y., Zhang, S., Pei, D., Liu, Y., Chen, Y., Zhang, R., Tao, S., Sun, P., & Zhou, R. (2019). LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, 4739, 4745.
- [27]. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 3111, 3119.
- [28]. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529, 533. <https://doi.org/10.1038/nature14236>
- [29]. Nacchia, M., Fruggiero, F., Lambiase, A., & Bruton, K. (2021). A systematic mapping of the advancing use of machine learning techniques for predictive maintenance in the manufacturing sector. *Applied Sciences*, 11(6), 2546. <https://doi.org/10.3390/app11062546>
- [30]. Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2016). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1), 11, 33. <https://doi.org/10.1109/JPROC.2015.2483592>
- [31]. Nonaka, I. (1994). A dynamic theory of organizational knowledge creation. *Organization Science*, 5(1), 14, 37. <https://doi.org/10.1287/orsc.5.1.14>
- [32]. Notaro, P., Cardoso, J., & Gerndt, M. (2021). A survey of AIOps methods for failure management. *ACM Transactions on Intelligent Systems and Technology*, 12(6), Article 71. <https://doi.org/10.1145/3483424>
- [33]. Paramesh, S. P., & Shreedhara, K. S. (2019). Automated IT service desk system using machine learning techniques. *Data Science Journal / CSI Transactions on ICT*, 7(3), 191, 201.
- [34]. Rasmussen, M. H., Lefrançois, M., Schneider, G. F., & Pauwels, P. (2020). BOT: The building topology ontology of the W3C Linked Building Data Group. *Semantic Web*, 12(1), 143, 161. <https://doi.org/10.3233/SW-200385>
- [35]. Robinson, I., Webber, J., & Eifrem, E. (2015). *Graph databases: New opportunities for connected data* (2nd ed.). O'Reilly Media.
- [36]. Salfner, F., Lenk, M., & Malek, M. (2010). A survey of online failure prediction methods. *ACM Computing Surveys*, 42(3), Article 10. <https://doi.org/10.1145/1670679.1670680>
- [37]. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- [38]. Soldani, J., & Brogi, A. (2022). Anomaly detection and failure root cause analysis in (micro)service-based cloud applications: A survey. *ACM Computing Surveys*, 55(3), 1, 39. <https://doi.org/10.1145/3501297>
- [39]. Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019). RotatE: Knowledge graph embedding by relational rotation in complex space. *Proceedings of the International Conference on Learning Representations (ICLR 2019)*.
- [40]. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.
- [41]. Tailhardat, L., Chabot, Y., & Troncy, R. (2023). Designing NORIA: A knowledge graph-based platform for anomaly detection and incident management in ICT systems. *Proceedings of the 4th International Workshop on Knowledge Graph Construction (KGC 2023)*, *CEUR Workshop Proceedings*.



- [42]. Tailhardat, L., Chabot, Y., & Troncy, R. (2023). Leveraging knowledge graphs for classifying incident situations in ICT systems. Proceedings of the 18th International Conference on Availability, Reliability and Security (ARES 2023), Article 141. <https://doi.org/10.1145/3600160.3604991>
- [43]. Thapa, M., Espejo-Urbe, J., & Pournaras, E. (2019). Measuring network reliability and repairability against cascading failures. Journal of Intelligent Information Systems, 53(3), 511, 541. <https://doi.org/10.1007/s10844-017-0477-0>
- [44]. van der Aalst, W. M. P. (2016). Process mining: Data science in action (2nd ed.). Springer. <https://doi.org/10.1007/978-3-662-49851-4>
- [45]. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. Proceedings of the International Conference on Learning Representations (ICLR 2018).
- [46]. Wang, H., Wu, Z., Jiang, H., Huang, Y., Wang, J., Kopru, S., & Xie, T. (2021). Groot: An event-graph-based approach for root cause analysis in industrial settings. Proceedings of the 36th IEEE/ACM International Conference on Automated Software Engineering (ASE 2021), 419, 429. <https://doi.org/10.1109/ASE51524.2021.9678708>
- [47]. Wang, L., Zhang, C., Ding, R., Xu, Y., Chen, Q., Zou, W., Chen, Q., Zhang, M., Gao, X., Fan, H., Rajmohan, S., Lin, Q., & Zhang, D. (2023). Root cause analysis for microservice systems via hierarchical reinforcement learning from human feedback. Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 5116, 5125. <https://doi.org/10.1145/3580305.3599906>
- [48]. Watson, I., & Marir, F. (1994). Case-based reasoning: A review. The Knowledge Engineering Review, 9(4), 327, 354. <https://doi.org/10.1017/S0269888900007098>
- [49]. Wu, L., Tordsson, J., Elmroth, E., & Kao, O. (2020). MicroRCA: Root cause localization of performance issues in microservices. Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS 2020), 1, 9. <https://doi.org/10.1109/NOMS47738.2020.9110353>
- [50]. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems, 32(1), 4, 24. <https://doi.org/10.1109/TNNLS.2020.2978386>
- [51]. Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? Proceedings of the International Conference on Learning Representations (ICLR 2019).
- [52]. Yang, B., Yih, W., He, X., Gao, J., & Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. Proceedings of the International Conference on Learning Representations (ICLR 2015).
- [53]. Zhang, C., Peng, X., Sha, C., Zhang, K., Fu, Z., Wu, X., Lin, Q., & Zhang, D. (2022). DeepTraLog: Trace-log combined microservice anomaly detection through graph-based deep learning. Proceedings of the 44th International Conference on Software Engineering (ICSE 2022), 623, 634. <https://doi.org/10.1145/3510003.3510181>
- [54]. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. AI Open, 1, 57, 81. <https://doi.org/10.1016/j.aiopen.2021.01.001>
- [55]. Zhou, W., Xue, W., Baker, T., & Zhou, M. (2016). Resolution recommendation for event tickets in service management. IEEE Transactions on Network and Service Management, 13(4), 954, 967. <https://doi.org/10.1109/TNSM.2016.2598041>