Leveraging Knowledge Graphs For Classifying Incident Situations in ICT Systems

Lionel Tailhardat
Orange, EURECOM
France
lionel.tailhardat@orange.com

Raphael Troncy
EURECOM
France
raphael.troncy@eurecom.fr

Yoan Chabot
Orange
France
yoan.chabot@orange.com

ABSTRACT

The complexity of Information and Communications Technology (ICT) systems, such as enterprise or Internet access provider networks, entails uncertainty in causal reasoning for efficient incident management. In this work, we propose to use knowledge graphs and explicit representation of incident context to enable support teams to provide a quick and effective response to complex incident situations. Formal analysis and expert opinions are used to analyze challenges in providing knowledge about relationships between events and incidents in network operations. We make use of an RDF knowledge graph generated from a real industrial settings and representing the network topology in terms of equipments and applications, past incidents and their resolutions. We then demonstrate the effectiveness of using a graph embeddings-based classifier to categorize incident tickets based on context and link anomaly models with their logical representation.

CCS CONCEPTS

• **Networks** → *Network performance evaluation*; • **Information systems** → *Decision support systems*; • **Computing methodologies** → **Artificial intelligence**; **Logical and relational learning**.

KEYWORDS

Incident Management, Anomaly Detection, Knowledge Graph, Graph Query, Graph Embeddings

ACM Reference Format:
Lionel Tailhardat, Raphael Troncy, and Yoan Chabot. 2023. Leveraging Knowledge Graphs For Classifying Incident Situations in ICT Systems. In *The 18th International Conference on Availability, Reliability and Security (ARES 2023), August 29–September 01, 2023, Benevento, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3600160.3604991

1 INTRODUCTION

The complexity of Information and Communications Technology (ICT) systems, such as enterprise or Internet access provider networks, entails uncertainty in causal reasoning for incident management. Indeed, the complexity of these systems extends beyond the sole set of equipments and links to include behavioral rules defined at both the network and application levels for resilient routing of the data and feature-rich services. Quick and efficient response to incident situations (e.g. network traffic disruption, cyber security attack) on top of this complexity involves specific expertise for accurate understanding of system behavior. Tools such as Network Monitoring Systems (NMSs) [23, 36] and Security Information and Event Monitoring systems (SIEMs) [24] are used to assist operators with incident management. However, ICT systems have interactions with other external systems and are characterized by multiple stacks of configuration, yielding incoherences between the local understanding of a subsystem's state and observable artifacts of a situation at a higher level of analysis. The consequences for incident management include long recovery times, difficulties in capitalizing on new incidents, and difficulties in generalizing when problems occur.

Mastering this complexity has been a long-standing effort by the industry and academic communities, including standardization for smooth knowledge sharing between practitioners, and logical and statistical modeling of the network dynamics [15, 18, 26, 30, 44, 46]. Yet, there is still a gap to fill in order to achieve the learning or use of a manipulable representation of anomalies for decision support: rule-based models require constant update efforts and do not scale because of computational complexity (e.g. a single national backbone router typically has 10,000 configuration lines), and black-box models hardly suits explainability requirements for decision-making systems on critical networks.

In order to enhance decision support tools, we propose to better capture the context of labeled anomalies through a multi-faceted knowledge graph and to use it to classify incident types. More precisely, we make use of an RDF knowledge graph [3] structured by the NORIA-O [27] ontology and we explore how graph embeddings provide a suitable representation for categorizing incident tickets. Our main contributions are the following. Firstly, we present a detailed study including expert opinions on the challenges and nature of inference techniques required for capturing explicit anomaly models in relation to incident management processes. Secondly, we introduce a method for capturing the context of anomalies and examine how it can partially correspond to logical formulations. This analysis includes exploring whether logical formulations could suffice and how statistical approaches can compensate for their limitations. These contributions lead to the development of a graph embeddings-based classification method for categorizing incident tickets, and the identification of SPARQL query patterns for anomaly detection based on a qualitative analysis of incident tickets. Finally, we provide an illustration of the synergy between knowledge graph modeling with NORIA-O and anomaly detection approaches.

ARES 2023, August 29–September 01, 2023, Benevento, Italy

ACM ISBN 979-8-4007-0772-8/23/08…$15.00
https://doi.org/10.1145/3600160.3604991

The remainder of this paper is organized as follows. Section 2 analyzes the challenges inherent in the incident management process for a detailed understanding of situations. Section 3 details our approach that makes use of the knowledge graph. Section 4 describes our experiments and evaluations. Section 5 presents some related work. Finally, we conclude the paper and outline some future work in Section 6.

2 CHALLENGES AND MOTIVATIONS

This Section, through the analysis of challenges inherent to incident management processes and tools (Section 2.1), the definition of use cases by experts (Section 2.2), and the analysis of anomaly detection techniques based on explicit representation (Section 2.3), elaborates on the principles for situation categorization with reasoning services (Section 2.4). These principles serve as the basis for the approach developed in Section 3.

2.1 Incident Management Process

Within the fields of IT networks and telecommunication services, the study of tasks and actions related to incident management is an active research area for many years and has led to a consensus on the forms of organization to be adopted in order to effectively respond to various situations. For example, the ITIL Incident Management process [45] and NIST SP 800-61 [37] recommendations apply respectively to IT management and cybersecurity domains.

In both cases, the process is described as a sequence of iterative steps including the diagnosis of the situation and leading to the remediation and correction of an undesirable situation. As such, it is akin to an "action-observation-reward-goal" process model with the following scenario: *1)* a failure (issue) on an asset induces events and alarms on the asset's neighborhood; *2)* responding to a trouble ticket (an alert), a network or security administrator analyzes events and alarms to distinguish primary events (causes) from secondary events (effects); *3)* contextualizing events and alarms with respect to "in policy" or "out of policy" activity models enables the administrator to select a remediation action; *4)* based on the remediation action results, the administrator closes the trouble ticket (the issue) or loops back for further analysis and corrective actions.

To support network and security administrators in their task, numerous tools and procedures are available for diagnosing the state of the systems (decision support tools such as NMSs [23, 36] or SIEMs [24], remote access to devices, on-site measurements and indicators), monitoring the life cycle of incidents, and capitalizing on knowledge of the causes and solutions to incidents (help desk ticketing systems, knowledge bases). This variety of tools and solutions is a wealth in itself that corresponds to the variety of technical or functional scopes to be managed. However, this is at the same time a challenge for a unified approach to the diagnostic stage: in practice, decision-making on the remediation action to be taken for a given situation must be based on a multiplicity of viewpoints stemming from various specialized tools.

2.2 Scoping the Diagnostic Phase with Experts

Assuming that the above-mentioned unified approach to the diagnostic stage is an achievable goal, decision-making depends heavily on how the operating parameters of the systems are obtained and represented. We argue that observables (i.e. artifacts of the network assets' events and states) result from a generative process in a semi-open world: as assets' states dynamically vary with respect to other agents (e.g. neighboring assets, servicing technicians, end users, randomness) based on behavioral rules (e.g. failover mechanisms, remediation procedure), sets of states are interpreted through higher level (composite) concepts. Although these perspectives provide indications on the entities to represent and how to do it, the nature of the processing carried out on these concepts for the diagnostic phase remains a broad subject.

To get more specific on the nature of the analysis and responses that are performed, we conducted interviews with a panel of support experts from Orange, an international telecommunications infrastructure and service provider. This panel consists of 16 experts who collectively represent 150 operations team members. We used the following methodology: *1)* ask experts for "pitfalls and wishes"; *2)* analyze responses with clarifying questions following ideas from the Agile framework [43] and ISO/IEC/IEEE 29148:2011 guide [1]; *3)* write exemplified anomaly detection use cases following the Cockburn-style template [4]. As a result, six use cases were defined to serve as a framework for the implementation of an automated reasoning system. Table 1 provides a short version of these use cases.

Table 1: List of use cases from expert panel interviews

#	Description
1	Circumscribe assets and causes search space for multi-applications incident situations
2	Alert on impaired service situations occurring on (distributed) fail-over architectures
3	Assess legitimacy of a given network flow
4	Track single identity from a set of various activity traces
5	Analyze false-positive and recurrent cyber security alerts
6	Analyze compliance of web navigation traces from institutional website

We notice that the use case #1 is the most challenging and encompasses the other use cases in that it generalizes the heuristic established in the incident diagnostic phase. During the interviews, the experts notably regularly raised the need for a confidence indicator alongside the inference results, which supports the need to search for a general mechanism. Therefore, we further study the use case #1 in the remainder of this paper. We observe that the effectiveness of the situation understanding relates to the composition of successive interpretation functions T_P, T_S, T_H applied to operating parameters of the managed systems (Eq. 1):

$$E \sim T_H(T_S(T_P(op))) \tag{1}$$

where op is the data representative of a system state, measured by a probe P, encoded into the decision support system S and then understood by a human operator H for a potential decision-making. Hence, the inference process (e.g. alerting on undesirable user/system trajectory, predicting next user/system action for corrective maintenance action) is akin to a sequential decision-making problem under uncertainty, where states and transitions are two different ways of representing the system's dynamics.

The experts also emphasized that the confidence indicator, seen as uncertainty about the interpretation to be given to a situation, should not be systematically used to reject inferences because it can itself contain information about complex situations (common cause failures, multi-application failures) for relating trouble tickets. To illustrate this, let us consider that each trouble ticket holds for a single independent incident or problem. However, it may occur that several tickets are linked, thus covering related incidents. Network assets being linked, mutual information arise about causes and consequences from the knowledge of the faults at hand. Hence, it is possible to infer and exploit a parent/child relationship reflecting the hierarchy of incidents from this confidence indicator: the child tickets describe incidents which are considered as consequences of the incident described in the parent ticket. Eq. (2) expresses this with modal logic in order to give substance to this idea:

$$\begin{array}{rc} \exists x, \exists y : & (F.x \rightarrow T.x) \land (F.y \rightarrow T.y) \\ \forall x, \forall y : & \diamond(F.x \rightarrow F.y) \\ \models \exists x, \exists y : & T.x \rightarrow T.y \end{array} \tag{2}$$

where (x, y) are network assets (e.g. router, server, application), F a fault indicator, T a trouble ticket, and $\diamond$ a possibility operator.

In fact, the notion of probability in the association of tickets for common cause failures directly relates to the need to capture incident contexts broadly, which would make it possible to generate a single incident ticket for a complex situation instead of soliciting various teams simultaneously and without coordination through multiple tickets. We remark that to further develop on this approach with tractable computational complexity, we may hypothesize that alarm spreading and cascading failures are bounded with respect to time and space.

2.3 Anomaly Modeling Techniques

For representing ICT systems and associated events, we consider RDF knowledge graphs [3], which are directed acyclic graphs where every nodes and edges are universally identified (URIs) and equipped with formal semantics. RDF knowledge graphs have proven to be flexible for data integration and logical reasoning over heterogeneous data, notably thanks to shared semantics provided by ontologies. We also use the SKOS standard [5] to represent vocabularies: terms defined in these vocabularies get a structured definition and an identity enabling re-use across applications.

SPARQL is the standard language to query RDF knowledge graphs. We can rely on SPARQL to query for anomalies that would have been already detected and either materialized in the graph or could be deduced using the semantics of the model. This approach constraints anomaly detection to logic-related use cases, which fits some expectations from Table 1. Going beyond logical reasoning is nevertheless required, as shown with the above uncertainty example (Eq. 2). We also remark that the use case #1 (Table 1) holds an implicit sub-graph computation requirement not available from the SPARQL 1.1 standard (e.g. A^* search algorithm).

To overcome these apparent limitations, we define three families of anomaly modeling techniques (Table 2) including the notion of time and explainability capabilities thanks to the use of explicit representation:

- *Model-Based Design* assumes that the knowledge graph holds the necessary and sufficient data to infer unwanted situations with information retrieval (e.g. SPARQL queries);
- *Process Mining*, including conformance checking tools and Petri nets (P/T nets) representation, is effective for situations tied to a decision-model and bounded in time and space;
- *Statistical Learning* with graph embeddings [20] assumes that anomaly models (i.e. the generalizing context of a set of situations) derive from the structure of the knowledge graph.

Table 2: Anomaly modeling technique families

Principles	Strengths	Weaknesses
Model-Based Design		
Query the graph to retrieve anomalies and their context.	Detecting anomalies “recorded” somehow in the graph thanks to the alarm system; straightforward translation of simple anomaly detection rules; multiple abstraction levels (subsumption).	Relies on expert knowledge; lack of probabilistic reasoning; hard to represent sequential decisions; may require to infer more prior information about the anomaly, e.g. its type using classification.
Process Mining		
Align a sequence of entities to activity models, then use this relatedness to guide the repair.	Detecting anomalies with multiple alerting signals and sequential decisions; replayable models.	Relies on expert knowledge; may require denoising models; probabilistic relatedness.
Statistical Learning		
Relate entities based on context similarities, then use this relatedness to alert and guide the repair.	Detecting anomalies with multiple alerting signals.	Requires fine tuning of the context definition depending on use case and temporality requirements; probabilistic relatedness.

2.4 Towards Reasoning Services

We posit that we are working with explicit representation techniques that allow, in particular, to provide assistance for the automatic filling of trouble tickets. Such incident situation categorization can be done at several stages: before the ticket creation (early detection), at the ticket opening (cause/solution similarity based on ticket descriptors and context), during the resolution (cause/solution refinement and proposal of next action based on the actions taken). According to the use cases list defined in Table 2, we summarize the above analysis by proposing the following set of reasoning and inference services:

(1) Predicting the category of a trouble ticket (i.e. the initial nature or technical impact, such as “isolated customer site”, “traffic disruption”, “integrity violation”). This is a classification problem, with classes defined in a user-provided controlled vocabulary represented as SKOS concepts.
(2) Predicting the probable cause of a trouble ticket. We can imagine that this would also be a classification problem with references to a controlled vocabulary.
(3) Detecting anomalies before a trouble ticket is even created. This service could be implemented with Link Prediction techniques [22]. The link being predicted is not necessarily related to incident remediation initially, but can rather be a link that would lead to the creation of an alarm or a trouble ticket.

(4) Adding comments to a given trouble ticket, namely, a comment proposing the next best action to undertake based on the observations of a given situation.
(5) Calculate the *n* closest anomalies given an observed anomaly (with the ambition of then transposing/adapting the remediation plans that have worked in the past).

We notice that these five services have a common point concerning the capture of the context related to the nature of the incident. Therefore, we choose to start by addressing the first case in the next section to provide guidance on the incremental construction of these services in the future.

3 APPROACH

In this section, we present two approaches to explicitly represent anomaly models. Firstly, we approach decision support in Section 3.1 as a classification problem and develop a model to predict the category of a trouble ticket using graph embeddings. Secondly, we assume that the anomaly models learned by the classifier have a correspondence, possibly partial, with a logical representation. We analyze trouble tickets qualitatively in Section 3.2 and highlight corresponding SPARQL queries for comparison with the classifier. We intentionally set aside the process mining approach discussed in Section 2.3 because it only captures local processes and therefore misses out on the need for learning from a larger context that is enabled by graph embeddings. We present the related experiments and results in Section 4.

3.1 Multiclass Classifier with Graph Embeddings

Understanding a network-impacting incident based only on network monitoring functions is an ill-posed problem (as explained in Section 2). We propose that using graph embeddings could help solve the inverse problem. Indeed, we assume that a trouble ticket represents an approximate dual of the anomaly structure in the network's parameter space. Thus, we can create an anomaly model by aggregating the graph representations of each incident in the network's parameter space that have the same characteristics (e.g. problem category, probable cause). In what follows, we focus on the task of predicting the category of a trouble ticket by building a multiclass classifier upon the context of trouble ticket entities.

For knowledge representation of ICT systems, we leverage on the NORIA-O conceptual model [27], an OWL-2 ontology published at https://w3id.org/noria that re-uses and extends well-known ontologies such as SEAS [32, 33], FOLIO [14], UCO [52], ORG [17], BOT [29] and BBO [6]. This model allows to describe a network infrastructure, its events (user login, network route priority reconfiguration), diagnosis and repair actions (connectivity check, firmware upgrade) that are performed during incident management. Therefore, it is used as the main data model for the experiments described in this paper as it can model complex ICT system situations and serve as a basis for anomaly detection and root cause analysis.

From a NORIA-O vocabulary perspective, building the classifier involves learning the relational model on events near the resource that is reported in a given incident (i.e. walking the graph and computing embeddings), and then linking the relational model to a category:

$$\begin{aligned}&\Big\{EventRecord.logOriginatingAgent\big(Resource(i)\big)\\&\qquad .logOriginatingAgent\big(Resource(i)_{neighbors}\big)\Big\}\\&\sim TroubleTicket.\Big(relatedResource\big(Resource(i)\big)\\&\qquad \sqcap problemCategory\Big)\end{aligned} \tag{3}$$

where `noria:problemCategory` is an attribute describing the final nature or technical impact of a `noria:TroubleTicket` entity. This attribute is part of the key fields used in the trouble ticket system to fully qualify incident resolution upon closure[1]. It is an `owl:ObjectProperty` with values from the `kos/TroubleTicket/trouble-category` controlled vocabulary, a SKOS Concept Scheme defining 9 concepts.

We make use of the pyRDF2Vec library [13] and the *scikit-learn* library [38]. pyRDF2Vec[2] is a Python implementation of the RDF2Vec algorithm [40], which captures the context of RDF graph nodes (properties and neighboring nodes) as latent feature vectors and is inspired by node2vec and word2vec. The data processing steps for our approach are the following: *1)* we create graph walks (i.e. sequence of vertices) by traversing the RDF dataset with `noria:TroubleTicket` entities as the starting points, and filtering out the `noria:problemCategory` property as it is used for classification; *2)* we compute embeddings for the `noria:TroubleTicket` entities by training a Word2Vec model on the graph walks; *3)* we train a random forest classifier with the embeddings as training input samples, and `noria:problemCategory` ranged entities as target values. The choice of the random forest classifier is based on its intrinsic interpretability as it is using decision trees.

A potential interest with this approach is that for any new trouble ticket matching a certain set of characteristics, projecting the anomaly model onto the rich graph representation of the network would be equivalent to circumscribe the search space in the network's parameter space (i.e. assets and features of interest to further scrutinize for anomalies). This meets the circumbscribe assets need raised for network and security operations (use case #1 in Table 1). With network administrators' commands also logged as `noria:EventRecord` entities, projecting the model could also recommend remediation and repair actions to perform (Eq. 4):

$$\begin{aligned}TroubleTicket_{similar} \times TroubleTicket_{actual}\\ \rightarrow \{Resource, Action\}_{actual}\end{aligned} \tag{4}$$

3.2 Model-Based Anomaly Detection

In addition to statistical learning, we develop an anomaly retrieval approach using SPARQL queries. The Listing 1 presents such a query, derived from expert panel interviews (Section 2), to assert a risk alert for *Applications* with k out-of n (50%) `Resources` in alarm (`EventRecord`).

A limited set of queries is available beforehand due to the lack of comprehensive knowledge about the situations to be detected. We posit that similar queries apply for similar trouble tickets. Thereon, we use the following analysis scheme on a user-provided dataset to identify the form of the queries and gain additional insight on similarities: *1)* for each trouble ticket, display it and provide an expert

[1] Other fields are `noria:problemResponsibility`, `noria:troubleTicketCause` and `pep:forProcedure`.
[2] See INK [11] for an alternative to pyRDF2Vec.

analysis about the SPARQL query to implement for anomaly detection; *2)* retrieve the k most similar trouble tickets based on their embeddings using a cosine distance; *3)* display these k tickets with all known attributes (e.g. creation date, services or resources involved, etc.) and provide an expert analysis as whether we could consider these tickets indeed related to the origin ticket and according to which dimension. We analyze the reciprocal alignment between incident tickets grouped according to the `noria:problemCategory` attribute and grouped according to the clusters obtained from a similarity graph on the embeddings (Algorithm 1).

```
CONSTRUCT { ?App noria:atRisk "K out-of N (50%)" . } WHERE {
  SELECT ?App
  (COUNT(DISTINCT ?Res) AS ?ResTotal)
  (COUNT(DISTINCT ?ResImp) AS ?ResWithImpact)
  WHERE { ?Res a noria:Resource ; noria:resourceForApplication ?App .
    OPTIONAL {
      ?Event a noria:EventLog ;
      noria:eventLogOriginatingManagedObject ?Res .
      BIND (?Res AS ?ResImp) } }
  GROUP BY ?App HAVING ( (?ResWithImpact / ?ResTotal) >= 0.5) }
```

Listing 1: SPARQL query for model-based anomaly detection.

Algorithm 1 Similarity graph of entities embeddings

$E \leftarrow$ embeddings entities
$k \leftarrow$ number of entities for similarity
$SG \leftarrow \emptyset$ ▷ Empty graph
for all $e \in E$ **do**
  $SG \leftarrow e$ ▷ Add vertex
  $SIM \leftarrow MostSimilar_{cosine}(e, E, k)$ ▷ Similarity on embeddings
  for all $e_{sim} \in SIM$ **do**
    $SG \leftarrow e_{sim}$ ▷ Add vertex
    $SG \leftarrow (e, e_{sim})$ ▷ Add edge
  end for
end for
$SG \leftarrow P_{\text{Louvain modularity}}(SG)$ ▷ Node partitioning
$SG \leftarrow R_{\text{Centrality}}(SG)$ ▷ Node ranking

The potential interest for this approach is twofold. First, it brings to enumerate query patterns for anomaly models, including for trouble tickets describing complex situations like a network outage impacting multiple applications. Second, it enables to explore how the embedding space is correlated with the semantic similarity [19] based on the logical form of the anomaly model (i.e. the SPARQL query).

4 EXPERIMENTS AND RESULTS

This section details the experiments conducted based on the approaches described in Section 3 and analyzes their results.

4.1 Dataset

For our experiments, we use a RDF dataset generated using the NORIA knowledge graph construction platform [28]. The input data of the platform is based on 15 tables distributed across 10 sources such as: trouble tickets, change requests, logs & alarms monitoring, network topology, applications, teams, users, etc. The size of the resulting RDF dataset is approximately 4 million triples for 400K entities, including streamed events spanning over 111 days. The Table 3 provides an overview of the dataset.[3]

[3]Due to confidentiality, this dataset is not made public.

Table 3: Dataset overview

Class names are provided in the $\langle prefix \rangle : \langle Class \rangle$ form. Percentage is the ratio of the entities count for a given class over the total number of entities.

Class name	Entity count	Percentage
noria:Resource	236'318	54.535
noria:EventRecord	89'606	20.678
foaf:Person	26'879	6.203
noria:CorporateUserIdentifier	26'879	6.203
noria:Locus	22'662	5.230
noria:ApplicationModule	9'314	2.149
noria:ProductModel	4'306	0.994
org:OrganizationalUnit	3'677	0.849
noria:Application	3'170	0.732
bot:Storey	2'869	0.662
noria:Room	2'869	0.662
bot:Building	1'656	0.382
bot:Site	1'374	0.317
org:Organization	366	0.084
noria:NetworkInterface	346	0.080
prov:Activity	324	0.075
noria:ChangeRequest	190	0.044
noria:NetworkLink	181	0.042
noria:TroubleTicket	150	0.035
noria:TroubleTicketNote	110	0.025
noria:AnomalyMode	75	0.017
pep:Procedure	9	0.002
TOTAL	433'330	

4.2 Multiclass Classifier with Graph Embeddings

Computing embeddings and training the model. Prior to computing embeddings, we generate 9 sets of walks with a random walk strategy [12], walk depth $WD \in \{4, 8, 10\}$ (vertices) and walk counts $WC \in \{10, 20, 30\}$ (per entity). Then, the Word2Vec training for embeddings holds on 10 epochs for each set of walks. These sets of walks are referred to as *WDxx/WCyy* in the Table 5.

We use the random forest algorithm as the classifier. The input values of the model are the embeddings. Target classes are values from the `noria:troubleTicketCategory`[4] property. The Table 4 presents the possible values and their distribution in the dataset. We use a stratified fixed-split strategy to build the training dataset while taking into account the target class imbalance, with a proportion of 25% of the dataset to include in the test split. Tuning the model's hyper-parameters relies on a grid-search heuristic, with parameters: the number of trees $\in \{10, 20, 30, 50, 70, 100\}$, split criterion $\in$ $\{\text{gini}, \text{entropy}\}$, maximum depth of the trees $\in$ $\{3, 5, 10, \text{pure leaves}\}$, and feature selection weight $\in$ $\{sqrt(\#features), log_2(\#features), (\#features)\}$. We use a weighted F1 score for model selection.

Evaluation & discussion. Table 5 reports on the classifier performance with respect to the weighted F1 score and the model parameters for each *WDxx/WCyy* set of walks. The WD08-WC30 shows the best performance for the classification task with a 0.81 weighted F1 score. Table 4 reports on the per class weighted F1 score for the best model (WD08-WC30) in order to discover if some classes are harder to predict than others, regardless of their frequency.

We observe from Table 5 that the model performance globally increases with the walk counts (WC) parameter. The performance does not appear to increase proportionally to the walk depth (WD)

[4]https://w3id.org/noria/ontology/troubleTicketCategory

Table 4: Target class distribution
Class labels relate to the NORIA-O skos:Concept skos:prefLabel for the noria:troubleTicketCategory property. Percentage is the ratio of the noria:TroubleTicket entities count for a given class over the total number of entities. F1 weighted and Support are classification performances for the WD08-WC30 random forest model over the test data.

Class label	Entities	Percentage	F1 weight.	Support
Interrupted service	77	55.8	0.97	19
Degraded QoS	22	15.9	0.75	5
No service impact	22	15.9	0.62	6
Defect to be qualified	13	9.4	0.57	3
Equipment failure	4	2.9	0.00	1
TOTAL	138	100.0	0.81	34

Table 5: Classifier performance
F1 weighted score and random forest best model parameters as a function of graph walks parameters. WC = Walk Count, WD = Walk Depth, model parameters in the form *<criterion>-<max depth>-<max features>-<n estimators>*.

	WC10	WC20	WC30
WD04	0.64 gini-05-SQRT-030	0.59 gini-05-SQRT-020	0.73 gini-05-SQRT-030
WD08	0.49 gini-05-SQRT-100	0.75 gini-05-SQRT-050	**0.81** gini-05-SQRT-020
WD10	0.52 gini-05-SQRT-020	0.60 gini-05-SQRT-020	0.76 gini-05-SQRT-020

parameter. However, the F1 score reaches a peak at $WD = 8$. In-depth analysis of the dataset to better understand the phenomenon shows that the available context for trouble ticket entities is not systematically consistent. For examples, some `noria:TroubleTicket` entities refer to `noria:Resource` entities out of the scope of the knowledge graph construction process, hence the context from the network neighborhood is absent from the embeddings. Similarly, the time frame of some `noria:EventRecord` entities (e.g. alarms, device logs) does not overlap with the creation date of the `noria:TroubleTicket`. We also observe some non standard values for the `noria:troubleTicketCategory` (i.e. values absent from the NORIA-O controlled vocabulary), hence the $150 - 138 = 12$ delta between the number of `noria:TroubleTicket` from Table 3 and Table 4.

Overall, we conclude that the classifier works relatively well but that the dataset is too small (for some classes in particular) and inconsistent for generalizing and tackling the circumbscribe assets need (use case #1 in Table 1). The typical approach to overcome this issue is to improve the knowledge graph construction stage with broader data sources and data quality assessment.

4.3 Model-Based Anomaly Detection

Qualitative analysis of trouble tickets. To identify query patterns for anomaly models, we first retrieve all attributes values associated with `noria:TroubleTicket` entities from the RDF dataset with a SPARQL query. Next, we employ the model-based anomaly detection analysis scheme developed in Section 3. In a second step, we compute the similarity graphs (Algorithm 1) over the WD08-WC30 embeddings with parameter $k \in \{3, 4, 5\}$, and then compare the overlap of query patterns with the partitions resulting from the Louvain community detection algorithm. We use the Szymkiewicz-Simpson coefficient for analyzing the overlap.

Results & discussion. From the qualitative analysis step, we identified 12 query patterns over 139 trouble ticket entities. The details of

Table 6: Retrieval patterns distribution and overlap coefficients
"C&O" stands for Count & average Overlap coefficient. Columns $[C0, \ldots, C6, None]$ reports on the pattern count and overlap coefficient for the WD08-WC30 / $k = 3$ similarity graph. $None$ stands for entities that were rejected by the Algorithm 1 due to syntax issues.

Pattern name	C0	C1	C2	C3	C4	C5	C6	None	C&O
AlarmState	2 0.13	2 0.13	1 0.06	1 0.07	25 0.96	15 0.88	3 0.14	 0.00	**49** 0.30
AuthError	1 0.11	 0.00	4 0.44	 0.00	 0.00	 0.00	2 0.22	2 0.22	**9** 0.13
CoFailure	3 1.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	**3** 0.13
Complex	7 0.47	 0.00	6 0.40	1 0.07	 0.00	 0.00	 0.00	1 0.08	**15** 0.13
Debug	 0.00	1 0.17	1 0.17	2 0.33	 0.00	 0.00	 0.00	2 0.33	**6** 0.13
ErroneousRes.	 0.00	 0.00	2 0.22	6 0.67	 0.00	1 0.11	 0.00	 0.00	**9** 0.13
HeartBeat	 0.00	11 0.85	 0.00	 0.00	 0.00	 0.00	2 0.15	 0.00	**13** 0.13
Overbilling	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	**6** 0.00
RecurringFai.	 0.00	1 1.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	**1** 0.13
RequestForIn.	1 0.06	 0.00	1 0.06	 0.00	1 0.06	1 0.06	5 0.29	8 0.62	**17** 0.14
RiskPreventi.	2 0.13	 0.00	1 0.06	4 0.29	 0.00	 0.00	10 0.59	 0.00	**17** 0.13
RMA	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	 0.00	**10** 0.00
C&O	16 0.16	15 0.18	16 0.12	14 0.12	26 0.09	17 0.09	22 0.12	13 0.10	**139** 0.12

these patterns are detailed below with indications on the involved classes and properties and examples of corresponding situations[5]. Table 6 reports on the overall distribution of the patterns, and how they were captured by the community detection algorithm for the $k = 3$ similarity graph. Running the Algorithm 1 with $k \in \{3, 4, 5\}$ led to generate $|P(SG, k)| = \{7, 6, 5\}$ partitions respectively.

We observe that a significantly lower number of patterns emerge from the dataset compared to the number of tickets considered ($12/139 \simeq 0.09$ reduction factor). Furthermore, it appears that some patterns, such as "AlarmState" and "HeartBeat", can capture diverse situations while remaining very specific by using restrictions on the objects and values of properties. This provides valuable insights on the detection and implementation of new patterns, in the perspective of an increase in the size of the dataset. However, as discussed in Section 4.2, the inconsistency of the data prevents us from directly validating the queries and their relevance on the dataset. As a consequence of this, we are currently not able to establish a correspondence between the patterns, the incident categories reported by the classifier, and the relevant anomaly models. Despite this, we can observe from Table 6 a connection that, although not entirely clear, suggests that there might be n-to-1 or n-to-m relationships between the patterns and the trouble ticket categories.

AlarmState: Alarm state w.r.t. `noria:EventRecord.*` and (`noria:Resource` or `noria:Service`). Examples: Service disruption on Optical Network Terminal (ONT). Unable to access http://example.org

AuthError: User role conformance w.r.t. `noria:EventRecord.type()` and `noria:EventRecord.logOriginatingAgent()` and `org:OrganizationalUnit`. Examples: Authentication error. User does not have access to the 'xxx' role. Please check my rights.

[5] See https://w3id.org/noria/dataset/ for query examples.

CoFailure: Alarm state w.r.t. `noria:TroubleTicket.troubleTicketRelatedResource()` and `seas:connectedTo` and `noria:EventRecord.logOriginatingManagedObject()` and `noria:EventRecord.type()`. Example: Co-occurring alarm in a network device neighborhood and creation of a parent/child relationship between trouble tickets for Service Level Agreement (SLA) tracking.

Complex: Requires further expertise for providing a pattern.

Debug: Non-relevant trouble ticket entity, present for debugging purposes of the ticketing system.

ErroneousResourceInOperationPlan: Non-existing resource in operation plan with respect to `noria:EventRecord.type()`. Example: The processing flow references a resource that does not exist.

HeartBeat: Value conformance and event frequency w.r.t. `noria:EventRecord.type()` and `noria:EventRecord.alarmMonitoredAttribute()` and `noria:EventRecord.logText()`. Examples: The number of failed calls has increased significantly. No response to SNMP polling and Ping. Agent not running or cannot communicate. Extreme slowness or even unavailability of the service when opening and closing documents on the platform.

Overbilling: Value conformance w.r.t. `noria:EventRecord.logOriginatingManagedObject()` and `noria:EventRecord.type()` and `noria:EventRecord.alarmMonitoredAttribute()` and `noria:EventRecord.logText()` and `foaf:Person` and `org:OrganizationalUnit`.

RecurringFailure: Repeated situation w.r.t. `noria:TroubleTicket.troubleTicketRelatedResource()` and `noria:TroubleTicket.problemCategory()`. Example: Repeated occurrence of the same type of failure on a device within a short period of time.

RequestForIntervention: Resource or service w.r.t. `noria:ChangeRequest.changeRequestPlannedStartTime()` and `noria:ChangeRequest.changeRequestStatusCurrent()`. Examples: Please decommission the 'xxx' system. The Customer is calling about the Request For Change (RFC) status.

RiskPreventionNotification: Presence of en event w.r.t. `noria:Resource` and `noria:OperationPlan`. Example: Automated deployment flow triggered on resource.

RMA: Alarm type w.r.t. `noria:EventRecord.*` and `noria:Resource.resourceProductModel()`. Example: Return Merchandise Authorization (RMA) for redundant Power Supply Unit (PSU).

5 RELATED WORK

This section briefly reviews related works from the perspectives of anomaly detection, knowledge graphs and IT management. To the best of our knowledge, our work is the first of its kind to address incident management over ICT systems through the combination of RDF knowledge graphs and graph embeddings. However, there exist works on specific aspects that are close to our research, notably in terms of graph representation or the application domain.

Close to the field of IT management, [47] proposes to speed up the triage of trouble tickets by using Natural Language Processing (NLP) techniques to provide an *a priori* categorization of the tickets. Using RDF knowledge graphs for anomaly detection: [14] presents a solution for mapping Failure Mode and Effect Analysis (FMEA) data with ontologies, which allows for the detection of anomalies and the derivation of their underlying causes through reasoning; [10] uses the APriori algorithm [42] for mining association rules on graph triples depicting user activities.

In the field of cybersecurity, various research trends coexist around RDF knowledge graphs, with a strong emphasis on ontology implementation. A first trend, led by forensic experts, links Indicators of Compromise (IoCs) to attack typologies through knowledge base construction [34, 50, 52]; knowledge extraction techniques from incident reports or through expert opinions aggregation are prominently at play in this context. A second trend focuses on modeling and classifying attack scenarios, with applications in detection tools using reasoning techniques [7, 35], knowledge management [9, 41, 49], and risk assessment based on the combination of infrastructure descriptions and a vulnerability repository [8]. Ultimately, these trends and projects could converge under the guidance of an ontology describing the management and response to cybersecurity incidents, with vocabulary alignment to standard cybersecurity repositories, as discussed in [16, 41]. We observe that many of cybersecurity-related projects have remained at the intention stage or have been developed without public sharing.

Without considering RDF knowledge graphs, [25, 39] review the literature related to anomaly detection with graph representations, including some of which are related to the IT/telecommunication domain. The works mentioned mainly refer to inference techniques based on a statistical model of the graph structure (relational learning) or graph traversal techniques. For example, [48] presents three Graph-Based Anomaly Detection (GBAD) algorithms for identifying abnormal sub-structures through modifications (vertex label or edge label different than expected), insertions (unexpected vertex or edge), and deletions (vertex or edge absent) compared to normative sub-structures. The detection principle is based on the idea that malicious behavior is close to normal behavior, which, in mathematical terms, corresponds to a percentage of isomorphism between the considered sub-structure and the normative sub-structure. Similarly, [51] provides assistance in analyzing cybersecurity incidents by applying a community detection algorithm to attack reports linked by similarities. In these two examples, the graph construction process implicitly incorporates the domain knowledge to be analyzed.

6 CONCLUSION AND FUTURE WORK

In this work, we aimed to simplify incident management activities related to broad scale Information and Communications Technology (ICT) systems, using knowledge graphs and learning an explicit representation of the context of each incident. We notably tackle the emblematic "common cause failures" and "alarm spreading phenomenon" cases since they require considering simultaneous situations that are seemingly unrelated as a whole.

Providing knowledge about the potential relationships between events and incidents occurring in networks is a way to simplify the diagnostic phase. Based on this idea, we firstly hypothesized that relating situations requires a common language to describe them, both in their variety and in the context that groups them together. We posit that RDF knowledge graphs are adequate knowledge representation formalism as they bring an abstraction level for standard interpretation and logical reasoning over heterogeneous data. We also consider that learning the relational structure for each type of incident, to a greater or lesser extent, would allow us to gain an explicit understanding of the complex phenomena that occur in network operations.

In a first analytical phase, the formal analysis of the incident management process led us to identify three families of anomaly detection techniques (model-based, process-mining, statistical learning) and how they could be implemented as a set of reasoning services. In a second step, we developed a dual statistical learning/model-based approach for modeling anomalies to overcome the lack of exhaustive knowledge of the situations to be addressed. The statistical learning approach led to the development of a graph embeddings-based classification method for categorizing incident tickets using a random forest model. For our experiments, we used a RDF dataset generated using the NORIA knowledge graph construction platform [28] and making use of the NORIA-O conceptual model [27]. Based on our evaluation of the classifier, we have determined that while it performs reasonably well (0.81 weighted F1 score), the dataset is too small and inconsistent (particularly for certain classes) to effectively address advanced inference services, such as projecting back the anomaly models onto the knowledge graph for guiding support teams on similar incidents. The model-based approach led to the identification of 12 SPARQL query patterns for anomaly detection based on a qualitative analysis of 139 incident tickets from the RDF dataset. The rather small number of patterns and the versatility of some of them provided valuable insights on the detection and implementation of new patterns, notably in the perspective of an increase in the size of the dataset. We also explored the limits of the model-based approach and the complementarity of the statistical approach through the projection of the query patterns onto the embeddings. We observed that there might be n-to-1 or n-to-m relationships between the patterns and the trouble ticket categories. This should be further investigated as we found some data inconsistency during the classifier evaluation step.

Future work will first focus on addressing data quality issues to improve the performance of the classification model and continue our research efforts in connecting query patterns to the latent space of embeddings. We also plan to continue exploring ways to improve the capture of the context of incidents at the knowledge graph embeddings stage. Hence, we aim to automatically process the description of incidents or the description of recovery interventions written in natural language. Indeed, traditional knowledge graph embeddings techniques just go through object properties to build the embeddings of a node since a datatype property breaks the graph (i.e. a literal cannot be a subject). Alternative knowledge graph embeddings approaches exist to handle literals [2]. However, this gives rise to further challenges, including the requirement to discretize the literals (for example, determining a general criterion for discretizing dates, numbers, etc.). An alternative consists in annotating datatype properties with semantic entities using a language model. Next, we aim to explore additional sampling and walk strategies [21]. This could notably allow us to guide the extraction of network topology data from the knowledge graph according to infrastructure or time criteria. Finally, we would like to test our approach using dynamic graph generation tools [31] to open it up for comparative studies, particularly regarding scalability. Ultimately, future research could explore automating IT support functions by reasoning on these shared explicit models of infrastructure and service behavior and past incidents management.

REFERENCES

[1] 2018. *ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering.* Technical Report. https://doi.org/10.1109/IEEESTD.2018.8559686

[2] Agustinus Kristiadi, Mohammad Asif Khan, Denis Lukovnikov, Jens Lehmann, and Asja Fischer. 2019. Incorporating Literals into Knowledge Graph Embeddings. In *International Semantic Web Conference (ISWC).* https://doi.org/10.1007/978-3-030-30793-6_20

[3] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2020. Knowledge Graphs. arXiv:2003.02320 [cs]

[4] Alistair Cockburn. 2012. *Writing Effective Use Cases.* Addison-Wesley.

[5] Alistair Miles and Sean Bechhofer. 2009. *SKOS – Simple Knowledge Organization System Reference.* W3C Recommendation. W3C.

[6] Amina Annane, Nathalie Aussenac-Gilles, and Mouna Kamel. 2019. BBO: BPMN 2.0 Based Ontology for Business Process Representation. In *20th European Conference on Knowledge Management (ECKM).*

[7] Andreas Ekelhart, Fajar J. Ekaputra, and Elmar Kiesling. 2021. The SLOGERT Framework for Automated Log Knowledge Graph Construction. In *European Semantic Web Conference (ESWC).* https://doi.org/10.1007/978-3-030-77385-4_38

[8] Andrei Brazhuk. 2020. Security Patterns Based Approach to Automatically Select Mitigations in Ontology-Driven Threat Modelling. In *Open Semantic Technologies for Intelligent Systems (OSTIS).*

[9] Andrei Brazhuk. 2021. Threat Modeling of Cloud Systems with Ontological Security Pattern Catalog. *International Journal of Open Information Technologies* (2021).

[10] Bin Jia, Cailing Dong, Z. Chen, Kuo-Chu Chang, Nichole Sullivan, and Genshe Chen. 2018. Pattern Discovery and Anomaly Detection via Knowledge Graph. In *21st International Conference on Information Fusion (FUSION).* https://doi.org/10.23919/ICIF.2018.8455737

[11] Bram Steenwinckel, Gilles Vandewiele, Michael Weyns, Terencio Agozzino, Filip De Turck, and Femke Ongenae. 2022. INK: Knowledge Graph Embeddings for Node Classification. *Data Mining and Knowledge Discovery* (2022). https://doi.org/10.1007/s10618-021-00806-z

[12] Bram Steenwinckel, Gilles Vandewiele, Pieter Bonte, Michael Weyns, Heiko Paulheim, Petar Ristoski, Filip De Turck, and Femke Ongenae. 2021. Walk Extraction Strategies for Node Embeddings with RDF2Vec in Knowledge Graphs. In *Database and Expert Systems Applications (DEXA) Workshops.*

[13] Bram Steenwinckel, Gilles Vandewiele, Terencio Agozzino, and Femke Ongenae. 2023. pyRDF2Vec: A Python Implementation and Extension of RDF2Vec. In *Extended Semantic Web Conference (ESWC).*

[14] Bram Steenwinckel, Pieter Heyvaert, Dieter De Paepe, Olivier Janssens, Sander Vanden Hautte, Anastasia Dimou, Filip De Turck, Sofie Van Hoecke, and Femke Ongenae. 2018. Towards Adaptive Anomaly Detection and Root Cause Analysis by Automated Extraction of Knowledge from Risk Analyses. In *9th International Semantic Sensor Networks Workshop (SSN).*

[15] Taco Cohen. 2022. Towards a Grounded Theory of Causation for Embodied AI. In *Workshop on Causal Representation Learning.* https://openreview.net/forum?id=K9PI5ewchUY

[16] Dave A. Mundie, Robin Ruefle, Audrey J. Dorofee, John McCloud, Samuel J. Perl, and Matthew L. Collins. 2014. An Incident Management Ontology. In *Semantic Technologies for Intelligence, Defense, and Security.*

[17] Dave Reynolds. 2014. *The Organization Ontology.* W3C Recommendation. W3C.

[18] Guillaume Brogi and Valerie Viet Triem Tong. 2016. TerminAPTor: Highlighting Advanced Persistent Threats through Information Flow Tracking. In *8th IFIP International Conference on New Technologies, Mobility and Security (NTMS).* https://doi.org/10.1109/NTMS.2016.7792480

[19] Hongyu Ren, Mikhail Galkin, Michael Cochez, Zhaocheng Zhu, and Jure Leskovec. 2023. Neural Graph Reasoning: Complex Logical Query Answering Meets Graph Databases.

[20] Ines Chami, Sami Abu-El-Haija, Bryan Perozzi, Christopher Ré, and Kevin Murphy. 2020. Machine Learning on Graphs: A Model and Comprehensive Taxonomy.

[21] Jan Portisch and Heiko Paulheim. 2022. Walk This Way! Entity Walks and Property Walks for RDF2vec. https://doi.org/10.48550/arXiv.2204.02777

[22] Jan Portisch, Nicolas Heist, and Heiko Paulheim. 2022. Knowledge Graph Embedding for Data Mining vs. Knowledge Graph Embedding for Link Prediction - Two Sides of the Same Coin? *Semantic Web* (2022). https://doi.org/10.3233/SW-212892

[23] Josh Chessman. 2020. *Magic Quadrant for Network Performance Monitoring and Diagnostics.* Technical Report G00463582. Gartner.

[24] Kelly Kavanagh, Toby Bussa, and Gorka Sadowski. 2018. *Magic Quadrant for Security Information and Event Management.* Technical Report G00348811. Gartner.

[25] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2014. Graph-Based Anomaly Detection and Description: A Survey.

[26] Lionel Tabourier, Alina Stoica, and Fernando Peruani. 2012. How to Detect Causality Effects on Large Dynamical Communication Networks: A Case Study. In *Communication Systems and Networks (COMSNETS).* https://doi.org/10.1109/

COMSNETS.2012.6151301

[27] Lionel Tailhardat, Yoan Chabot, and Raphaël Troncy. 2022. NORIA-O: an Ontology for Anomaly Detection and Incident Management in ICT Systems. *Semantic Web Journal* (2022).

[28] Lionel Tailhardat, Yoan Chabot, and Raphaël Troncy. 2023. Designing NORIA: a Knowledge Graph-based Platform for Anomaly Detection and Incident Management in ICT Systems. In *4th International Workshop on Knowledge Graph Construction (KGC)*.

[29] Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. 2020. BOT: The Building Topology Ontology of the W3C Linked Building Data Group. *Semantic Web Journal* (2020).

[30] Manish Thapa, Jose Espejo-Uribe, and Evangelos Pournaras. 2019. Measuring Network Reliability and Repairability against Cascading Failures. *Journal of Intelligent Information Systems* (2019). https://doi.org/10.1007/s10844-017-0477-0

[31] Maria Massri, Zoltan Miklos, Philippe Raipin, Pierre Meye, Amaury Bouchra Pilet, and Thomas Hassan. 2023. RTGEN++: A Relative Temporal Graph GENerator. *Future Generation Computer Systems* (2023). https://doi.org/10.1016/j.future.2023.03.023

[32] Maxime Lefrançois. 2017. Planned ETSI SAREF Extensions Based on the W3C&OGC SOSA/SSN-compatible SEAS Ontology Patterns. In *Workshop on Semantic Interoperability and Standardization in the IoT (SIS-IoT)*.

[33] Maxime Lefrançois, Jarmo Kalaoja, Takoua Ghariani, and Antoine Zimmermann. 2016. *SEAS Knowledge Model*. Deliverable 2.2. ITEA2 12004 Smart Energy Aware Systems.

[34] Nidhi Rastogi, Sharmishtha Dutta, Mohammed J. Zaki, Alex Gittens, and Charu Aggarwal. 2020. MALOnt: An Ontology for Malware Threat Intelligence. https://doi.org/10.13140/RG.2.2.16426.64962 arXiv:2006.11446

[35] Noam Ben-Asher, A. Oltramari, R. Erbacher, and Cleotilde González. 2015. Ontology-Based Adaptive Systems of Cyber Defense. In *STIDS*.

[36] Pankaj Prasad and Josh Chessman. 2019. *Market Guide for IT Infrastructure Monitoring Tools*. Technical Report G00450400. Gartner.

[37] Paul Cichonski, Tom Millar, Tim Grance, and Karen Scarfone. 2012. *Computer Security Incident Handling Guide: Recommendations of the National Institute of Standards and Technology*. Technical Report NIST SP 800-61r2. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-61r2

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* (2011).

[39] Peizheng Huang, Shulin Liu, Kuan Zhang, Tao Xu, and Xiaojian Yi. 2022. Overview of the Application of Knowledge Graph in Anomaly Detection and Fault Diagnosis. In *4th International Conference on System Reliability and Safety Engineering (SRSE)*. https://doi.org/10.1109/SRSE56746.2022.10067308

[40] Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. 2019. RDF2Vec: RDF Graph Embeddings and Their Applications. *Semantic Web* (2019). https://doi.org/10.3233/SW-180317

[41] Peter E. Kaloroumakis and Michael J. Smith. 2021. *Toward a Knowledge Graph of Cybersecurity Countermeasures*. Technical Report. The MITRE Corporation.

[42] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules. In *20th International Conference on Very Large Databases (VLDB*.

[43] Scaled Agile, Inc. 2022. SAFe – Story. https://scaledagileframework.com/story/.

[44] Serge Romaric Mouafo Tembo, Sandrine Vaton, Jean-Luc Courant, Stephane Gosselin, and Michel Beuvelot. 2017. Model-Based Probabilistic Reasoning for Self-Diagnosis of Telecommunication Networks: Application to a GPON-FTTH Access Network. *Journal of Network and Systems Management* (2017). https://doi.org/10.1007/s10922-016-9401-0

[45] Stefan Kempter. 2007. IT Process Maps – Incident Management. https://wiki.en.it-processmaps.com/index.php/Incident_Management.

[46] Masato Uchida. 2015. Recent Trends and Some Lessons for Serious Network Failures in Japan. In *International Conference on Intelligent Networking and Collaborative Systems (INCOS)*. https://doi.org/10.1109/INCoS.2015.31

[47] Wenting Sun and Alka Isac. 2020. Resolve trouble tickets with machine learning. https://www.ericsson.com/en/blog/2020/10/how-to-resolve-trouble-tickets-machine-learning.

[48] William Eberle and Lawrence Holder. 2007. Discovering Structural Anomalies in Graph-Based Data. In *7th IEEE International Conference on Data Mining Workshops (ICDMW)*. https://doi.org/10.1109/ICDMW.2007.91

[49] Wim Muskee, Marcus Rohrmoser, and Tony Stein. 2015. Unified ICAS ontology. https://github.com/twosixlabs/icas-ontology.

[50] Yoan Chabot. 2015. *Construction, Enrichment and Semantic Analysis of Timelines : Application to Digital Forensics*. Thesis. University of Burgundy.

[51] Zach Kurtz and Samuel J. Perl. 2017. Measuring Similarity between Cyber Security Incident Reports. In *Forum of Incident Response and Security Teams (FIRST Conference)*.

[52] Zareen Syed, Ankur Padia, M. Lisa Mathews, Tim Finin, and Anupam Joshi. 2016. UCO: A Unified Cybersecurity Ontology. In *AAAI Workshop on Artificial Intelligence for Cyber Security*.