Automatic alarm prioritization by data mining for fault management in mobile networks

Antonio J. García[a], Matías Toril[a], Pablo Oliver[a], Salvador Luna-Ramírez[a], Manuel Ortiz[b]

[a] *Department of Communication Engineering, University of Málaga, Bulevar Louis Pasteur, Málaga 29071 ES*
[b] *Ericsson Spain, Severo Ochoa 55, Málaga 29590 ES*

Abstract

Network management systems play an important role to deal with the large size and complexity of current mobile networks. Thus, operators and vendors focus much of their efforts on developing new techniques and tools for this management. One of the most critical processes in network management is fault management, since a failure in a network element might have a strong impact on user satisfaction due to service degradation. Unfortunately, mobile networks generate thousands of alarms daily, which have to be checked manually by operator personnel. With the latest advances in big data analytics, different methods for reducing the number of alarms to be monitored have been presented. In this work, an automatic method for prioritizing alarms based on the need for specialized personnel is presented. The core of the method is a model built with supervised machine learning techniques that estimates the probability that an alarm generates a trouble ticket. For this purpose, the model is trained with trouble ticket data from the network operation center. The model is tested with a real alarm and trouble ticket dataset taken from a live mobile network. Results show that the proposed model correctly flags those alarms that need further analysis by the operator and filter out those alarms that do not have impact on network performance.

Keywords: Network management, automation, fault management, data mining, machine learning

1. Introduction

Over the last years, the number of users and services in mobile networks has increased dramatically. By 2021, a tenfold increase of mobile traffic is expected and around fifty billion of devices will be connected to mobile networks (Cisco Systems Inc., 2017a). Likewise, the deployment of new radio access technologies (e.g., 5G) in the coming years will pave the way for new use cases (5G Infrastructure Association, 2016). The price to be paid is an increase of network heterogeneity due to the co-existence of multiple radio access technologies, base stations of very different ranges and disparate devices (Hossain & Hasan, 2015). Such diversity will increase the complexity of mobile networks, creating new problems in network management.

To deal with network complexity, mobile operators and vendors have focused their efforts on developing automation techniques to manage their networks (NGMN, 2015). These efforts have resulted in Self-Organizing Networks (SON), standardized by the Third Generation Partnership Project (3GPP) (3rd Generation Parthnership Project, 2012), which provide intelligence inside the network and network adaptability by simplifying configuration and optimization procedures. SON techniques can be divided into three main categories, depending on the stage in

*Corresponding author
Email addresses: ajgp@ic.uma.es (Antonio J. García), mtoril@ic.uma.es (Matías Toril), pob@ic.uma.es (Pablo Oliver), sluna@ic.uma.es (Salvador Luna-Ramírez), manuel.ortiz@ericsson.com (Manuel Ortiz)

the network life cycle, namely self-configuration, self-optimization and self-healing (Aliu et al., 2013). Self-configuration defines the process whereby Base Station (BS) configuration parameters are automatically set when a new base station is deployed. Once the system has been correctly configured, self-optimization guarantees optimal network performance by continuous monitoring and tuning of system parameters to cope with changes in the environment. In parallel to self-optimization, self-healing is triggered whenever a fault or a failure is detected to diagnose the cause (i.e., root-cause analysis) and execute proper compensation mechanisms (Klaine et al., 2017).

In a market where networks and services are quite similar between operators, a differentiating factor is the way operators manage their networks to increase user satisfaction. A key goal in this process is to reduce the impact of network failures, which is the role of Fault Management (FM) (Cisco Systems Inc., 2017b). FM comprises problem recognition, problem notification, problem diagnosis, launch of corrective actions and restoration of initial settings after fixing the problem (Ramiro & Hamied, 2011). Some of these tasks are labor intensive. FM starts by collecting performance statistics from network devices and links in real-time, notifying failures and/or service degradation by different sequences of alarms. Once a fault is identified, network administrators try to fix the failure remotely. Traditionally, this process has been performed by a group of experts, located in the Network Operation Center (NOC), that use their experience and intuition to troubleshoot, localize and solve faults by checking all alarms collected in the different network segments. Thus, FM is currently the most time and expertise demanding process of all network management areas (Awad & Hamdoun, 2016). The size and complexity of mobile networks, where equipment from different vendors and technologies coexist, result in a vast amount of alarms (tens of thousands per day). Besides, most of these alarms do not provide useful information for network administrators, being only a low percentage of them related to active operation problems. This makes FM extremely unreliable when done manually, causing that it can no longer be effectively managed by human administrators (Bouillard et al., 2013).

With recent advances in information technologies, it is now possible to process massive volumes of information by big data analytic (BDA) techniques (Zheng et al., 2016). 'Big Data' refers to data that cannot be processed by traditional means due to its volume, velocity and variety (e.g., alarms). By analyzing this data, operators are more aware of the current state of the network, allowing corrective actions in a proactive way (Gupta & Jha, 2015). One of the most pursued goals is to make data available in (near) real time to reduce reaction time (Bange et al., 2015). Thus, an increasing number of frameworks have emerged for real-time BDA (e.g., Hadoop Online, Storm, Spark, ...) (Liu et al., 2014). These frameworks take advantage of the huge and reliable computing power provided by cloud computing. In parallel, specifying BDA assets and systems has become a priority for standardization bodies (ITU-T, 2016).

FM systems are critical in providing valuable information to minimize physical and economic loss in response to abnormal situations (Mehta & Reddy, 2014). However, typical FM systems have proven to be far away from the desired performance in respect of number of managed alarms (VanCamp, 2016). Thus, the design of effective FM systems has become a crucial goal not only for mobile network systems but for all domains. Researchers have focused on designing methods to reduce the number of alarms handled by FM processes. The most extended approach is *alarm correlation* (Jakobson & Weissman, 1993). A single failure can result in multiple alarms referred to the same issue, which are notified to the management operators for inspection. To avoid overloading management personnel, alarms can be correlated based on the initial cause and condensed, thereby reducing the final number of alarms to be monitored (Wietgrefe et al., 1997).

In this context, BDA seems to be a powerful process providing different mechanisms for knowledge discovery. It encompasses several interrelated disciplines, from the simplest data visualization steps in Exploratory Data Analysis (EDA) to the most sophisticated Machine Learning (ML) algorithms. The aim here is to build models for characterizing and classifying alarms for which the most important features must be selected. Classification models can be based on heuristic rules or automatic and complex algorithms based on unsupervised (e.g., k-means) or supervised learning (e.g., support vector machines, neural networks). However, selecting the most proper algorithm supposes a challenge. There is no a general classifier that can deal with all kind of problems. Generally, classifiers show strengths and weakness depending on the scenario. In order to solve this, ensemble modeling is nowadays a common process to reduce the generalization error and the accuracy of predictions obtained by different models. Ensemble modeling has been discussed in different works along the years. In (Parvin et al., 2013a), an innovative classifier ensemble methodology based on subspace learning by applying genetic algorithm to improve the performance of the classification is proposed. In (Parvin et al., 2013b), a new technique for clustering ensemble by boosting sampling of original data is introduced. The effectiveness of bagging techniques, comparing the efficacy of sampling with and without replacement, in conjunction with several consensus algorithms are discussed in (Minaei-Bidgoli et al., 2014). In this work, non-adaptive and adaptive resampling schemes for the integration of multiple independent and dependent clusterings are proposed. In (Parvin et al., 2015), authors present a novel method for ensemble creation. In this work, different base classifiers are previously partitioned by using a clustering algorithm and, finally, the proposed method produces a final ensemble by selecting one classifier from each cluster.

On the other hand, due to the nature of data FM systems have to deal with, these traditional algorithms can result in an inaccurate performance. Generally, dataset collected in mobile networks consists of a mixture of numerical and categorical features which presents limitations for some algorithms like k-means. In the literature, there are different works that try to treat with this limitation. In (Ahmad & Dey, 2007), a clustering algorithm based on k-mean paradigm that works well for data with mixed numeric and categorical features is proposed. On the other hand, authors in (Zhang et al., 2015) present a novel method to transform categorical data to numerical representations, in order to open the possibility of exploiting the abundant, numerical learning algorithms. Besides, this dataset consists of a large number of features that, on one hand, may increase the complexity of this data processing and, on the other hand, it may lead to worse performance. For this reason, feature selection is a process that has been mostly used in data mining projects. In (Dy & Brodley, 2004), different issues involved in developing an automated feature subset selection algorithm for unlabeled data are analyzed. (Minaei-Bidgoli et al., 2011) improves one of the existing feature selection algorithm based on fuzzy entropy concept by using ensemble methods.

In the literature, there is a wide variety of proposals for alarm correlation systems. Generally, alarm correlation methods are divided into three categories: 1) similarity-based algorithms, generating clusters of similar alarms based on simple association rules or complex machine learning techniques (Smith et al., 2005); 2) knowledge-based algorithms, relying on causal relationships or Causal Relation Graphs (CRGs) based on Bayesian networks to fuse alarms, (Zali et al., 2012); and 3) statistical-based algorithms, whereby a categorization of alarms is performed by detecting statistical similarities (Ren et al., 2010).

Over the years, many FM systems based on alarm correlation have been developed in the industry. In (Liu et al., 2003), a system is described for suppressing valueless alarms based on their repeatability and providing advisory information in process plants. In (Noda et al.,

2011), a data-mining method to detect statistical similarities among discrete occurrence of alarms and then group them in an ethylene plant is presented. In (Folmer & Vogel-Heuser, 2012), an algorithm for identifying the most frequent alarms and those causal alarms generating alarm sequences is proposed. The alarm correlation system introduced in (Wang et al., 2015) combines similarity analysis and causal relationship detection for root alarm analysis. In (Lai et al., 2017), a novel online pattern matching based on reduction of incoming alarm floods is presented. More recently, a novel framework for alarm causality analysis based on smart data analytics is proposed in (Hu et al., 2018).

Alarm correlation in telecommunication networks is also discussed in several earlier works. In (Frohlich et al., 1997), different alarm modeling approaches for explaining monitored resources and associated alarms in cellular networks are discussed. In (Mannila et al., 1997), the use of frequent episode discovery is proposed for finding sequential rules in a single sequence alarm for several domains. In (Wietgrefe et al., 1997), neural networks are used to find the root cause alarm by correlating alarms in a cellular network. In (Julisch, 2003), a novel alarm-clustering method is proposed to group similar alarms based on root cause. (Bellec & Kechadi, 2007) introduces the problem of analyzing, interpreting and reducing the number of alarms before localizing the fault in cellular networks. A programming language for alarm applications is proposed in (Duarte Jr et al., 2008). In Makanju et al. (2012), an alert detection system that learns from confirmed anomalies and detects future errors is proposed. In (Çelebi et al., 2014), a novel alarm correlation, rule discovery and significant rule selection method based on sequential rule mining with a time-confidence parameter is proposed for mobile networks. A big data framework to analyze network alarms is presented in (Man et al., 2016).

In spite of these research efforts, none of them have already succeeded in reducing the number of alarms to a single alarm per incident (Mirheidari et al., 2013). Even if this is case, a manual inspection of the remaining ones must still be done to prioritize those alarms requiring an action to restore the service. In this process, NOC staff needs to identify those alarms derived by an incident that require some corrective action and subsequently generate a ticket (a.k.a. trouble ticket), consisting of a collection of data about the incident with enough information to initiate the corresponding repairing activity. Several works have stressed the role of trouble tickets in FM, as these are related to actual incidents identified as network failures (Dreo & Volta, 1995; Medem et al., 2009). In (Salah et al., 2019), an automatic method for incorporating semantically rich information in tickets into the alarm correlation process is presented. The proposed correlation approach is based on the time alignment of events (alarms and tickets) that affect common elements in the network. However, the heuristic model used to correlate alarms and trouble tickets is simple and depends on previous human knowledge. Moreover, it has the same limitations in terms of alarm reduction capability as previous works.

In this work, a new model to identify and prioritize alarms based on the need for generating a trouble ticket is presented. Such a model allows to reduce the number of alarms to be manually checked by NOC staff, which can focus on those alarms that are really the cause of a network failure . Similarly to (Salah et al., 2019), trouble tickets and alarm data is crossed to improve the incident resolution process. However, in this work, this data is not used for improving the alarm correlation process by reducing the number of alarms to a single alarm per incident. Instead, it is used to identify the most priority alarms that urgently require the generation of a trouble ticket to initiate some recovery action as soon as possible. Thus, the main contribution of this work is the propose of a supervised machine learning (ML) algorithm for automatic alarm classification by determining which alarms will result in the generation of a trouble ticket (i.e., it is

really associated to an incident) and, thus, it must be prioritized to be handled by NOC staff. Flagging alarms as high priority, the recovery process can be speeded up. The model is trained and tested with a real alarm and trouble ticket dataset taken from a live mobile network.

The rest of the work is organized as follows. Section 2 describes the fault management process in a mobile network. Section 3 discusses the benefits of BDA and ML to empower fault management. Section 4 introduces the data mining project developed in this work for alarm prioritization. Section 5 shows the results obtained by the developed model in a real scenario. Finally, Section 6 presents the main conclusions of the work.

2. Fault Management Process

In this section, the FM process in mobile networks is described. Firstly, a basic network management architecture is presented to understand system structure and components. Secondly, the main roles involved in FM are introduced. Finally, a generic FM methodology is presented.

2.1. Network Management Architecture

A mobile network management architecture basically consists of five components (Verma & Verma, 2009): manager, agent, network management protocols, management information base (MIB) and communication model.

1. *Manager*: A manager is a tool with a user interface, generally located in NOC, that allows to manage the different network elements in a mobile network. Its main functions are: a) to monitor managed devices and collect data reported by them, used for further analysis by the management staff; b) to request information from managed devices and receive their responses; and c) to configure managed devices by setting variables and thresholds.
2. *Agent*: An agent is a program typically embedded in a network element (e.g., router, switch, server ...) responsible for monitoring it and communicating with the manager. The agent provides information about the status of a network element to the manager, either asynchronously or after a query.
3. *MIB*: A MIB is a database that collects management information that describes the network element parameters. This information is provided by the agent and shared with the manager. Parameters in the dataset are generally a set of statistical and configuration parameters. In a fault management process, this information is usually requested for troubleshooting purposes and further analysis.
4. *Network management protocol*: The main role of a network management protocol is to define a standard language used by the agent and manager to exchange information across all network elements. The most widely adopted protocols for network management are (Ersue & Claise, 2012): a) the Simple Network Management Protocol (SNMP), used for the monitoring of fault and performance data and with its stateless nature (while SNMP also works well for status polling and determining the operational state of specific functionality (Case et al., 1990)); b) Remote Network MONitoring (RMON), which is an extension of SNMP defining a set of statistics and functions that can be exchanged between the controllers (Waldbusser, 2006); and, c) Common Management Interface Protocol (CMIP), more complex than the previous protocols and only used by some information technology service providers for network management purposes (ISO/IEC Information Technology Task Force (ITTF), 1998).
5. *Communication model*: The main schemes for exchanging information between the manager and the agent in a Network Management System (NMS) are poll and push. Poll model is based on request/response paradigm in which the manager requests data and the agent replies by sending the re-

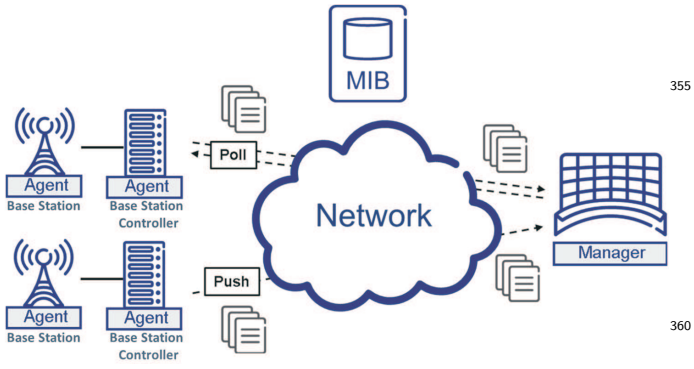


Figure 1: Centralized network management architecture.

quested information. Thus, the data flow is always initiated by the manager and the polling can be automatic or user-initiated. In a push model, data flow is initially configured by user and, then, agents individually take the initiative to push data to the manager via scheduler or asynchronously.

The above-described components can be distributed in different network management architectures, which can be categorized as centralized, distributed and hierarchical (Verma & Verma, 2009; El Brak et al., 2011). In a centralized architecture, shown in Fig. 1, there is only one manager controlling the entire network and one agent per managed device. This architecture is graphically illustrated. In contrast, distributed management splits the network into segments and a manager is deployed in each segment, without interaction between them. Alternatively, the hierarchical architecture combines both approaches in which each manager locally manages a subset of network elements being managed by a higher-level manager that act as central.

2.2. Roles in Fault Management

During the FM process, NOC staff is responsible for monitoring every network element managed by the NMS, as well as making decisions and performing corrective actions. This staff is usually composed of several actors and roles that work together to ensure optimal network performance and productivity. The main roles in a FM process are:

1. *Management staff*: It is the main role in the fault management process, responsible for solving network faults and restoring the service as soon as possible. Their main tasks are to analyze alerts to detect faults, identify and prioritize the faults, and trigger corrective actions to restore service. In turn, this staff, composed of engineers and technicians, may be categorized into different levels based on their expertise and problem-solving ability. A 1^{st} level (L1) technician (a.k.a. fault reporter) is responsible for monitoring alarms and discovering and reporting faults. In case a fault requires further inspection and corrective action, a trouble ticket for incident notification is created by L1 group, which is then escalated to a higher level technician (i.e., 2^{nd} level, L2, and 3^{rd} level, L3), who is responsible for troubleshooting the issue and carrying out the required resolution tasks, that usually result in the creation of a work order to perform some manual process in the network element. These higher level technicians are also responsible for checking the result of the applied solution, and hence their name of fault resolver and solution validator.
2. *Help desk staff*: It is a secondary role used by the customer as a point of contact for issues and queries. Help desk staff is focused on receiving and registering requests from customers or employees (via phone call or email) to notify an incident that may not be reported by an alarm but it is detected by the end user. Among its functions, help desk staff is responsible for logging and managing end users' calls, registering the incident by a trouble ticket that is not associated to any alarm, requesting technical support by means of a work order (if required) and updating the alarm and trouble ticket logs with the resolution state.
3. *Other roles*: Additional groups may be required for an effective FM process, such as dispatcher, responsible for taking the required corrective actions by send-

ing/closing work orders, and field engineers, responsible for doing on-site visits and implementing the required corrective actions that cannot be done remotely (e.g., replacing hardware, updating software ...)

2.3. Basic Fault Management Methodology

As shown in Fig. 2, FM is broken down into four steps: fault detection, fault notification, fault diagnosis and fault resolution.

1. *Fault detection*: Fault detection provides the NMS with the capability to detect and report faults. For this purpose, all network elements report their status to NOC. Two different fault detection modes can be configured in the NMS: passive and active (Shields, 2007). In the passive mode, agents notify the manager when a pre-defined condition, configured by management staff, is met. Note that if the agent stops working, no notification is generated and fault detection does not work. In contrast, in the active mode, the manager checks the status of each agent by sending request messages. Thus, if some agent does not provide the required information, the manager can handle it by generating the corresponding alarm to be investigated later on.
2. *Fault notification*: Once a fault occurs, fault information is forwarded by the NMS to the manager, which checks it by comparing it with a set of predefined rules. In case rules match, the manager generates a notification message to management staff that will be previewed in a console or sent by email and instant message. This notification is usually defined as an *alarm* (Jakobson & Weissman, 1993), consisting of a brief description of the fault in a specific format defined by the equipment vendor. Such a description includes the device/service generating the fault, plain text describing the issue , fault class (equipment, communication, environmental, quality of service ...), severity of alarm (notification, minor, critical ...) and information associated to the management process (fault identifier, fault creation time, resolution status...). All these alarms are shown in real-time and monitored by the management staff in the Alarm Management (AM) process. Then, alarms are categorized and prioritized so as to identify the most critical alarms. L1 technicians check a huge amount of alarms in real time to identify those requiring further analysis or corrective actions. In this process, alarm information is usually enriched with data from MIB about the managed device where the fault occurred. Once the most critical alarms are detected, a trouble ticket is generated and escalated to L2 group. At this point, the alarm has been isolated and a deeper inspection is required for the fault resolution.
3. *Fault diagnosing*: Once the trouble ticket is generated, L2 group starts a Ticket Management (TM) process, where a root cause analysis is performed to diagnose the cause of the fault. On the other hand, the remaining alarms that do not result in a trouble ticket generation generally do not affect service and are eventually restored by themselves.
4. *Fault resolution*: Once the root cause of the issue has been identified, corrective actions are initiated. In some cases, fault can be solved remotely and no more actions are required for service restoration. However, in some other cases, resolution may involve a physical action requiring an on-site visit by a field engineer. In these cases, L2 group generates a work order, handled by the Work Order Management (WOM) process. Once a work order is created, a dispatching notification is sent to the corresponding field engineer who should accept this notification and initiate the corrective action (e.g., replace a defective hardware component). Once the fault is solved and the service restored, work order information is completed with the actions performed by the field engineer and the

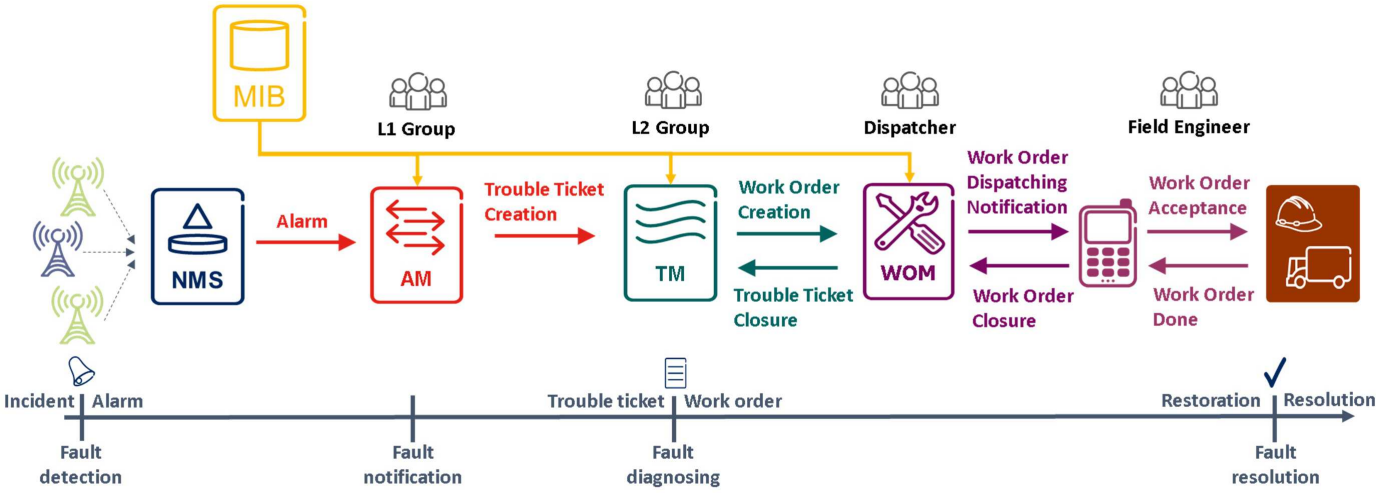


Figure 2: Workflow of the fault management process.

trouble ticket is closed.

3. Big-Data driven Fault Management

Mobile operators use commercial tools to collect and process the huge amount of raw data reported by the NMS. This data includes not only the information generated by network elements but additional information required for an effective network management (e.g., information stored in MIBs). Unfortunately, legacy tools suffer from limited scalability and analytical capabilities, which is the reason for the reactive maintenance approach adopted so far by operators. The newest big data techniques can help to obtain a better understanding of how the mobile network works, disclosing previously unknown patterns and correlations (Bi et al., 2015). Such a knowledge can be used to solve the lack of self-awareness and self-adaptiveness, which have been recognized as the main drawbacks of current SON approaches (Imran et al., 2014; Zheng et al., 2016; Han et al., 2017).

As a first step, the data collected by the NMS can be used off-line for knowledge discovery by data analytics. The following tasks in FM can be solved by ML:

1. *Classification*: Alarms can be grouped based on different fields, such as fault type, severity or network element. Nowadays, this is performed manually. However, this can be automated by models built with heuristic rules or automatic classification algorithms based on supervised learning (e.g., artificial neural networks, decision trees) or unsupervised learning (e.g., k-means clustering).

2. *Regression*: A model is needed to characterize a statistical parameter of a network element that produces a recurrent alarm when a threshold value is exceeded. Such a mapping can be derived from measurements collected by network elements and Key Performance Indicators (KPIs) obtained from the network. Then, model construction can be performed by classical regression techniques (e.g., generalized linear regression) or supervised learning algorithms (e.g., support vector machines, neural networks or Bayesian networks algorithms) (Harahap et al., 2010), which better approximate non-linear mappings between metrics and faults.

3. *Feature selection*: With the huge amount of information generated by a NMS, the most relevant fields affecting the network element functionality have to be identified. Reducing the number of variables in the models built with ML decreases the computational load, speeds up the learning process, improves generalization capability and makes interpretation easier

for the operator.

4. Model for alarm prioritization

A novel predictive model is presented here for prioritizing alarms based on the need for specialized personnel (i.e., the higher priority, the higher the need for a specialist). The input of the model is alarm data produced by faults in managed network elements. The output of the model is a prediction of whether an alarm would generate a trouble ticket, so that must be prioritized. During model construction and assessment, such an output is checked against real trouble ticket data generated by L1 technicians.

Model construction is based on the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology, consisting of six stages: business understanding, data understanding, data preparation, modeling, evaluation and deployment. Fig. 3 shows the different building blocks of the proposed model when implemented in a commercial data mining tool (International Business Machines Corporation (IBM), 2018). The model consists of interconnected nodes that cover the different stages of CRISP-DM methodology. Circles represents import nodes to retrieve input data, hexagons are nodes for basic operations, such as data merging, field derivation or data balancing, stairs are "*super-nodes*" used to combine basic nodes in more complex operations, pentagons are nodes for data analytic operations, such as feature selection, decision trees or artificial neural networks, gold diamonds (a.k.a "*model nuggets*" in this tool) represent the output of each model once trained, and squares are used for assessment purposes. More details of these operations are given below.

4.1. Business Understanding

A major constraint in FM is the need for manually checking a huge amount of alarms by the NOC staff. A model that automatically prioritizes and flags those alarms requiring the generation of a trouble ticket would significantly reduce the staff and time needed to handle alarms. This would lead to large savings for the network operator and improve the service offered to end users by reducing the time to fix faults.

4.2. Data Understanding

Understanding data involves a preliminary collection of data to identify attributes, explore data structure and check data quality. In this work, two main data sources are used: alarm and trouble ticket data. Both data sources are usually stored in databases that can be accessed by NOC staff. At this stage, each database has to be explored to identify the different fields and their nature. This task is performed in Fig. 3 by the nodes in the dashed red-lined box. In general, the alarm database consists of three types of data: a) categorical data with information related to alarm type (e.g., fault type, severity, summary with basic information about the fault...) and its origin (i.e., network element identifier), b) numerical data, specifying the number of occurrences of the alarm and numerical codes, generally with similar information to categorical fields, that may be mapped by NOC staff for further analysis, and c) date-time data to register the different status of the alarm (i.e., alarm creation, alarm notification, alarm closure and alarm deletion). Likewise, trouble ticket database shows similar data fields, but enriched with information introduced by L1 technicians. Specifically, in a trouble ticket database, the following information can be found: a) categorical data with information regarding the alarm associated to the trouble ticket (obtained from alarm database) and information regarding the trouble ticket (e.g., trouble ticket description, solution that is filled once the incident is solved, name of L2 group taking care of the trouble ticket ...), b) numerical fields with codes mapped by NOC staff and the associated alarm identifier, and c) date-time data to log different status of the trouble ticket.

From a preliminary analysis of real alarm and trouble ticket data, several conclusions can be drawn:

- A new record is generated in the database whenever

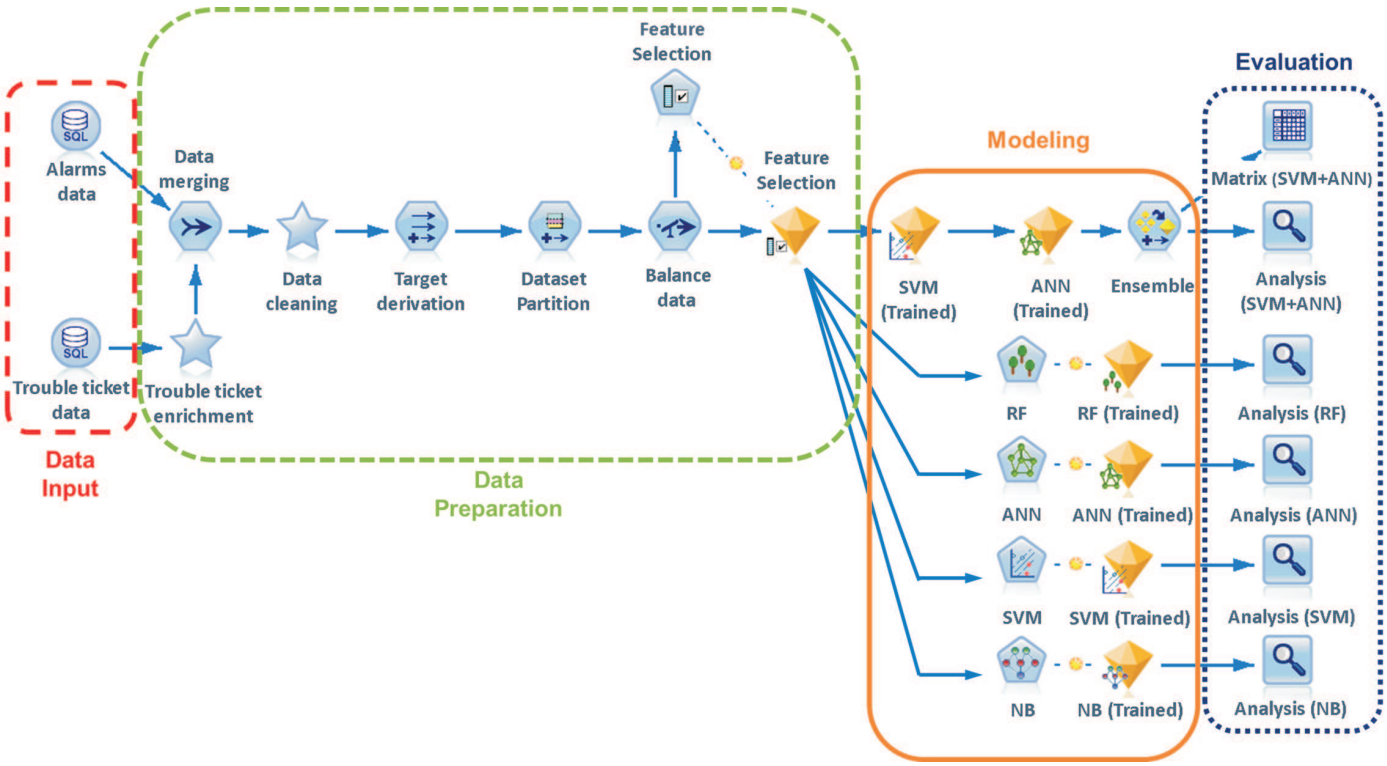


Figure 3: Alarm Prioritization model.

an alarm/trouble ticket status changes (e.g., open, cleared, closed, deleted) to add the new status. However, older records that contain the same basic information, but different status, are kept. As a consequence, both alarm and trouble tickets may be duplicate in the dataset. Thus, a preliminary filtering process is required to eliminate duplicates that can affect model training.

- One alarm that is not closed yet may generate a trouble ticket in the future and, thus, be considered as of a high priority. However, if this data is used to train the model, the latter would consider this alarm as not ticketed, since a trouble ticket has not been generated yet. As a result, the model may misclassify future alarms. Thus, it is necessary to filter those alarms that are still not closed to avoid incomplete data for model training.

- Eventually, some fields may have wrong data (i.e., corrupt data, missing data ...). To build a robust model, data quality must be as high as possible, for which only data that is correctly populated must be considered and, thus, corrupt records must be removed.

- Some trouble tickets may not be associated to an alarm (e.g., those created by help desk staff based on a customer complaint). Moreover, some trouble tickets may not show the alarms originating the trouble ticket due to malpractice of NOC staff. Therefore, a trouble ticket enrichment process is required in order to, firstly, remove those trouble tickets that are not associated to any alarm and, secondly, associate those alarms and trouble tickets that were not correctly associated due to malpractice.

4.3. Data Preparation

Preparation includes integrating, formatting, cleaning, constructing and selecting data. These tasks can be time consuming, but are critical for the success of data mining.

Firstly, trouble ticket information needs to be enriched, since there may be trouble tickets without an associated

alarm due to a customer request or due to malpractices in the way of using the NMS. In this process, referred to in Fig. 3 as *Trouble ticket enrichment*, trouble ticket data is explored to find trouble tickets associated to customer requests or trouble tickets associated to an alarm without an identifier. Trouble tickets associated to a customer request are out of scope and they are excluded from the dataset. In contrast, trouble tickets originated by an alarm, but with missing data, are enriched by crossing alarm and trouble ticket data to find matches (Salah et al., 2019). Fields used for this purpose are the network element identifier, site identifier and creation and closure times. Thus, if a trouble ticket was created for the same network element and site as an alarm and the life range of the trouble ticket (i.e., from creation to closure) is within the life range of the alarm, that alarm is candidate to be associated to the trouble ticket. This association is validated by exploring the alarm information field in trouble ticket (if available) with text mining techniques. It is important to clarify that this process is only focused on correlating alarms and trouble tickets data. Thus, if several alarms are produced by the same incident and it requires the creation of a trouble ticket, each alarm will be associated to the same trouble ticket (i.e., alarms will be treated independently).

After the enrichment process, alarm and trouble ticket data sources are merged to combine both in a single dataset used for training purposes . Then, a data cleaning process is performed to solve issues discovered during the data understanding process. Once the dataset is cleaned, the next step is to derive the target variable used to train the model. In this work, the target variable is a binary field identifying those alarms that generated a trouble ticket (hereafter named as *Ticketed*) and those closed without generating a trouble ticket (named as *Not Ticketed*). Later, the dataset is split into training and testing data for the modeling stage.

Before training any classification model, it is necessary to check how imbalanced the dataset is. FM datasets often contain very few cases of alarms that needed further actions. In this situation, some ML classifiers (e.g., random forest or support vector machine) tend to favor the class with the largest proportion of observations, leading to predictions that, even if correct in most cases, do not make the most of the values of attributes (Witten et al., 2011). This problem can be solved by changing the class proportion and the size of the dataset (He & Garcia, 2009). In this work, an under-sampling method is used to delete instances of the majority class to obtain a 50/50 ratio between classes (i.e., *Ticketed* and *Not Ticketed*).

Finally, field reduction is carried out by automatic feature selection. This process aims to reduce the number of variables in the model, identifying those fields (predictors) with the largest impact on the probability that an alarm generates a trouble ticket (dependent variable). In this work, a filter method based on pearson correlation is used for feature selection. Thus, relevant fields are selected based only on their correlation with the outcome variable. As a result, the same set of predictors is used for all ML algorithms tested.

4.4. Modeling

In live networks, a large set of labeled data including alarms and trouble tickets is often available for operators. Thus, supervised learning algorithms can be used to derive the model that predicts if a combination of alarms will lead to a trouble ticket. In this work, four ML classification algorithms are tested:

- *Random Forest (RF)* (Breiman, 2001): This ensemble learning method builds multiple decision trees and merges their outcome to get more robust estimations. Different decision trees are built by randomly selecting training subsets from the original dataset. Then, tree outputs are combined by taking the most frequent prediction of all decision trees. The resulting model is not easy to interpret, but, unlike classical decision

trees, RF prevents overfitting, which makes it one of the most powerful supervised learning algorithms.

- *Artificial Neural Network (ANN)* (Schmidhuber, 2015): A ANN is a collection of processing units (neurons) connected by links (edges). Neurons are aggregated into layers that perform different kinds of transformations on their inputs. Signals travel from the input layer to the output layer, possibly after traversing intermediate layers multiple times. Learning is carried out by adjusting the weight of neurons and edges to minimize the output error. Thus, ANN has the ability to model non-linear complex relationships and infer unseen relationships. Its main drawbacks are their black-box approach, the need for very large labeled datasets and its large computational load.

- *Support Vector Machine (SVM)* (Cortes & Vapnik, 1995): SVM performs classification by representing cases as points in a multidimensional space and constructing hyperplanes that separate cases in classes with a gap as wide as possible. New cases are then mapped into the same space and classified based on which side of the gap they fall. SVM can capture non-linear relationships by mapping data to higher dimensions with kernels (Witten et al., 2011). More importantly, SVM is defined as a convex optimization problem, for which there are very efficient solution methods.

- *Naive Bayes (NB)* (Witten et al., 2011): NB is a probabilistic binary classifier based on Bayes' theorem. It computes two types of probabilities from the training data, namely the probability of each class and the conditional probability for each class given each previous class value. Once trained, the algorithm makes new predictions by using Bayes Theorem. Although this algorithm considers the unrealistic assumption that each feature is independent, it is very effective on a large range of complex problems.

Each of the above-described algorithms results in a model adjusted with the training dataset (labeled as "trained" in Fig. 3). A fifth method is developed by combining the models with a higher accuracy. Ensemble learning is an approach that combines several algorithms into a single predictive model in order to decrease variance (bagging), reduce bias (boosting) or improve predictions (stacking) (Witten et al., 2011). To combine models, a stacking technique (Wolpert, 1992) is used. Unlike bagging and boosting (used, e.g., in RF), stacking is used to merge models of different type (e.g., SVM and ANN). Roughly, stacking discovers how best to combine the output of base models by using another learning algorithm (the metalearner).

For the construction model, SVM and ANN have been selected as base models for ensemble learning. As it will be shown later, these two models present a higher accuracy than the two other algorithm and, thus, their outputs are combined by stacking them based on an ensemble learning technique. Stacking defines a meta-model that is formed by SVM and ANN algorithms that is trained on the outputs of the base models as features.

4.5. Evaluation

Each model is evaluated to check its accuracy. In this work, a true positive (negative) denotes a case when the binary classifier correctly labeled the alarm as 'ticketed' ('not ticketed'). Conversely, a false positive (negative) denotes a case when the binary classifier incorrectly labeled the alarm as 'ticketed' ('not ticketed'). Based on these statistics, the following metrics are calculated:

- *Classification accuracy (a.k.a. accuracy)*: It is the ratio of the number of correct predictions to the total number of samples, computed as

$$Accuracy = \frac{TP + TN}{P + N}\,, \tag{1}$$

where TP and TN are the number of true positives and true negatives, and P and N are the total num-

ber of positive and negative values, respectively. This metric is a true reflection of accuracy only when the classes in the dataset are balanced (i.e., same percentage of positive and negative classes).

- *Sensitivity (a.k.a. probability of detection)*: It measures the proportion of positive values (i.e., *ticketed*) that are correctly predicted (i.e., the true positive ratio).
- *Specificity*: It measures the proportion of negative values (i.e., *not ticketed*) that are correctly predicted (i.e., the true negative ratio).
- *Receiver Operating Characteristic (ROC) curve* (Fawcett, 2006): It is a graphical plot representing the true positive ratio against the false positive ratio with different threshold parameter for the classifier. A perfect ROC curve takes the form of a unit step function.
- *Area Under Curve (AUC)*: It represents the probability that a randomly chosen positive sample ranks above a randomly chosen negative sample in a binary classifier, computed as the area under the ROC curve. The higher (i.e., the closer to 1), the better.

5. Performance assessment

In this section, the proposed model for alarm prioritization based on the need of generating a trouble ticket is assessed in a live mobile network. For clarity, the analysis set-up is first described and results are presented later.

5.1. Analysis set-up

Assessment is done in a real mobile network consisting of 7,758 sites covering a geographical area of 705,589 km^2 with different radio access technologies (GSM, UMTS and LTE). The dataset includes: a) alarm data generated by network elements in the whole network, taken from the NMS, and b) trouble ticket data generated by NOC staff. The data collection period is 3 months, during which a total set of 5,877,444 alarms and 44,637 trouble tickets were registered. These figures clearly show that only a very small share of alarms (i.e., less than 0.8%) generate a trouble ticket. During the analysis, the dataset is split in two exclusive subsets, the training dataset with the first two months and the testing dataset with the last month. The main input used for prediction is the alarm dataset. Each alarm includes 32 fields. After automatic feature selection, only 15 out of 32 fields are considered to be related to the outcome variable. Specifically, these fields provide information regarding: a) the alarm (type of alarm, severity or number of occurrences), b) the affected network element (type, vendor), and c) the domain where alarm was generated (radio access network, operations support systems, packet switch core). This subset of features is shared by all ML algorithms tested. Five binary ML classifiers are compared. Four of them are built with individual algorithms (RF, ANN, SVM and NB) and a fifth one is the ensemble of two (ANN+SVM). Model performance is evaluated with the metrics detailed in Section 4.

As mentioned before, IBM SPSS Modeler is used for the creation of the proposed model. SPSS Modeler is a data mining tool that allows users to build data mining algorithms without the need for programming them. It is based on a drag-and-drop methodology by using nodes with different functionalities that are connected to each other to perform operations.

5.2. Results

Table 1 shows the performance of individual models in the training and testing datasets. Results prove that ANN and SVM outperform the other methods in terms of classification accuracy. Specifically, ANN and SVM classify nearly 90% of cases correctly, showing similar values for 'ticketed' (sensitivity) and 'not ticketed' (specificity) alarms. Moreover, similar values are obtained with both datasets, showing their generalization capability (i.e., lack

of overfitting).

The same conclusions can be drawn from the analysis of ROC curves for the training and testing datasets, shown in Fig. 4 (a)-(b). In both figures, the x-axis represents the *False Positive Rate* (i.e., $1 - Specificity$), whereas the y-axis represents *True Positive Rate* (i.e., *Sensitivity*). The four lines represent the ROC curves of base models. From both subfigures, it can be concluded that ANN and SVM present the highest estimation accuracy, since their ROC curves are the furthest away from the reference line (dashed line), corresponding to a random model. Specifically, the AUC of ANN and SVM is 0.95 and 0.94, whereas the AUC of the next model is that of NB, with 0.90. A comparison of both figures shows that the ROC curve of each method has the same shape in both datasets, which is a clear indication of the absence of overfitting.

Table 1 also shows the same metrics for the ensemble method. It is observed that, by combining several models, a more accurate classification of alarms is obtained. Specifically, the accuracy of ANN+SVM is 91.45% compared to 89.12% and 85.69 for ANN and SVM, respectively.

Based on these results, the following conclusions can be drawn:

- The number of alarms in a live mobile network deemed to require further investigation after a preliminary analysis is really small (0.8%). Thus, an automatic alarm ticketing system would save much time and effort for NOC personnel.

- After feature selection, the number of predictors is still large (15). This is clear indication that deciding whether to generate a trouble ticket or not for an alarm is not a trivial task and it is based on complex rules that relate predictors to target variable, justifying the need for a data mining project.

- The sensitivity figure obtained by the best of methods indicates that only 7% of the alarms that required a trouble ticket are misclassified, and would not be prioritized with the automated approach. Likewise, specificity (i.e., the complement of the probability of false alarm) is 90%. Keeping this figure as high as possible is key to avoid triggering unnecessary actions by L2-L3 group personnel. Nonetheless, these false trouble tickets are more than compensated for by the large savings introduced by automatically discarding 89% of alarms that actually did not require a trouble ticket.

5.3. Implementation issues

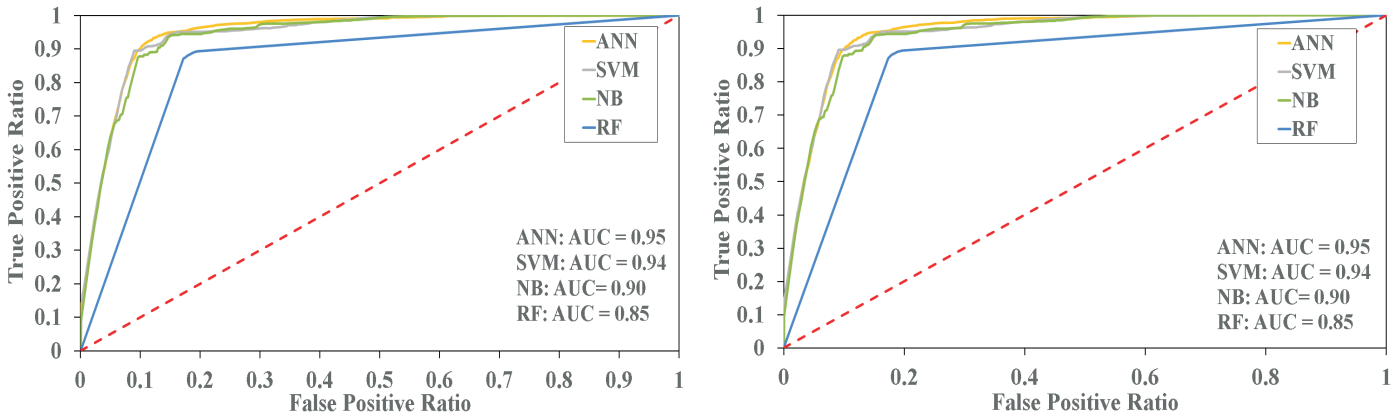
The model is designed as a centralized scheme that can be integrated in the NOC of a real mobile network. Despite its complexity, its computational load is relatively low. In practice, the most time consuming processes are data preparation and model training, which can be done once and offline. Specifically, the execution time for the considered dataset in a 2.6-GHz quad-core processor laptop is 7,000 s for data preparation, 2,035 s for ANN, 1,370 s for SVM, 1,820 s for RF model, 1,275 s for NB and 4055 s for ANN+SVM.

6. Conclusions

In this work, a supervised machine learning classifier for prioritizing alarms in a live mobile network based on the need for generating a trouble ticket has been described. Model inputs are alarm data available in the NOC. However, trouble ticket data is also required in a first training stage. The model has been tested with a real dataset taken from a live mobile network consisting of several radio access technologies. During assessment, different machine learning algorithms have been compared. Results have shown that, in the considered real scenario, neural networks and support vector machines slightly outperform random forest and naive Bayes classifiers in terms of estimation accuracy (89.12 and 85.69% against 83.92 and 81.15%, respectively). These figures are slightly improved

Table 1: Base models comparison.

Base model	Partition	Accuracy	Sensitivity	Specificity
ANN	Training	89.90 %	92.38 %	88.56 %
	Testing	89.12 %	92.31 %	88.39 %
SVM	Training	88.75 %	93.07 %	85.71 %
	Testing	85.69 %	94.03 %	85.51 %
NB	Training	85.30 %	88.05 %	82.28 %
	Testing	83.92 %	89.94 %	84.05 %
RF	Training	82.84 %	86.12 %	80.51 %
	Testing	81.15 %	85.66 %	80.01 %
ANN+SVM	Training	91.45 %	93.41 %	89.02 %
	Testing	89.90 %	94.19 %	88.45 %



(a) Training dataset

(b) Testing dataset

Figure 4: Base models comparison based on ROC curves

when neural networks and support vector machines are combined.

The proposed big-data driven approach can be used to reduce reaction time, which is a major issue of current self-healing schemes. Execution times are reasonably low, since model construction is done once and offline. Then, the resulting model can be used in production to ticket alarms in real time without human intervention, minimizing the time to repair and, ultimately, improving system availability. Equally important, the model can be shared by different network operators, provided that the same alarm information is available.

Acknowledgment

This work was funded by the Spanish Ministry of Economy and Competitiveness (TEC2015-69982-R) and Ericsson Spain.

References

3rd Generation Parthnership Project (2012). Telecommunication management; Self-Organizing Networks (SON); Concepts and requirements. In *TS 32.500, version 14.0.0 (Release 8)*.

5G Infrastructure Association (2016). *5G Empowering vertical industries*. Technical Report White Paper.

Ahmad, A., & Dey, L. (2007). A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering*, *63*, 503–527.

Aliu, O. G., Imran, A., Imran, M. A., & Evans, B. (2013). A survey of self organisation in future cellular networks. *IEEE Communications Surveys & Tutorials*, *15*, 336–361.

Awad, M., & Hamdoun, H. (2016). A framework for modelling mobile radio access networks for intelligent fault management. In *Basic Sciences and Engineering Studies (SGCAC), Conference* (pp. 43–49). IEEE.

Bange, C., Grosser, T., & Janoschek, N. (2015). *Big data use cases: Getting real on data monetization*. Research Study Barc.

Bellec, J.-H., & Kechadi, M.-T. (2007). Feck: A new efficient clustering algorithm for the events correlation problem in telecommunication networks. In *Future Generation Communication and Networking (FGCN)* (pp. 469–475). IEEE volume 1.

Bi, S., Zhang, R., Ding, Z., & Cui, S. (2015). Wireless communications in the era of big data. *IEEE Communications Magazine*, *53*, 190–199.

Bouillard, A., Junier, A., & Ronot, B. (2013). Impact of rare alarms on event correlation. In *Network and Service Management (CNSM), 9th International Conference* (pp. 126–129). IEEE.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Case, J. D., Fedor, M., Schoffstall, M. L., & Davin, J. (1990). *Simple network management protocol (SNMP)*. Technical Report No. RFC 1157.

Çelebi, Ö. F., Zeydan, E., Arı, İ., İleri, Ö., & Ergüt, S. (2014). Alarm sequence rule mining extended with a time confidence parameter. In *IEEE International Conference on Data Mining (ICDM)*.

Cisco Systems Inc. (2017a). *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update*. White Paper.

Cisco Systems Inc. (2017b). *Network Management System: Best Practices*. White Paper.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297.

Dreo, G., & Volta, R. (1995). Using master tickets as a storage for problem-solving expertise. In *Integrated Network Management IV* (pp. 328–340). Springer.

Duarte Jr, E. P., Musicante, M. A., & Fernandes, H. D. H. (2008). Anemona: a programming language for network monitoring applications. *International Journal of Network Management*, *18*, 295–302.

Dy, J. G., & Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of machine learning research*, *5*, 845–889.

El Brak, S., Bouhorma, M., & Boudhir, A. A. (2011). Network management architecture approaches designed for mobile ad hoc networks, .

Ersue, M., & Claise, B. (2012). *An Overview of the IETF Network Management Standards*. Technical Report No. RFC 6632.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*, 861–874.

Folmer, J., & Vogel-Heuser, B. (2012). Computing dependent industrial alarms for alarm flood reduction. In *Systems, Signals and Devices (SSD), 9th International Multi-Conference* (pp. 1–6). IEEE.

Frohlich, P., Nejdl, W., Jobmann, K., & Wietgrefe, H. (1997). Model-based alarm correlation in cellular phone networks. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)., Proceedings 5th International Symposium* (pp. 197–204). IEEE.

Gupta, A., & Jha, R. K. (2015). A survey of 5G network: Architecture and emerging technologies. *IEEE access*, *3*, 1206–1232.

Han, S., Chih-Lin, I., Li, G., Wang, S., & Sun, Q. (2017). Big data enabled mobile network design for 5g and beyond. *IEEE Communications Magazine*, *55*, 150–157.

Harahap, E., Sakamoto, W., & Nishi, H. (2010). Failure prediction method for network management system by using bayesian

network and shared database. In *Information and Telecommunication Technologies (APSITT), 8th Asia-Pacific Symposium* (pp. 1–6). IEEE.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, *21*, 1263–1284.

Hossain, E., & Hasan, M. (2015). 5G cellular: key enabling technologies and research challenges. *IEEE Instrumentation & Measurement Magazine*, *18*, 11–21.

Hu, W., Shah, S. L., & Chen, T. (2018). Framework for a smart data analytics platform towards process monitoring and alarm management. *Computers & Chemical Engineering*, *114*, 225–244.

Imran, A., Zoha, A., & Abu-Dayya, A. (2014). Challenges in 5G: how to empower SON with big data for enabling 5G. *IEEE Network*, *28*, 27–33.

International Business Machines Corporation (IBM) (2018). IBM SPSS software. URL: `https://www.ibm.com/analytics/spss-statistics-software[Lastaccessed:2018-06-22]`.

ISO/IEC Information Technology Task Force (ITTF) (1998). Information Processing Systems - OSI, ISO standard 9596-1: Common Management Information Protocol, part1: Specification.

ITU-T (2016). *Big data standardization roadmap*. Recommendation Y.3600 International Telecommunication Union.

Jakobson, G., & Weissman, M. (1993). Alarm correlation. *IEEE Network*, *7*, 52–59.

Julisch, K. (2003). Clustering intrusion detection alarms to support root cause analysis. *ACM Transactions on Information and System Security (TISSEC)*, *6*, 443–471.

Klaine, P. V., Imran, M. A., Onireti, O., & Souza, R. D. (2017). A survey of machine learning techniques applied to self-organizing cellular networks. *IEEE Communications Surveys & Tutorials*, *19*, 2392–2431.

Lai, S., Yang, F., & Chen, T. (2017). Online pattern matching and prediction of incoming alarm floods. *Journal of Process Control*, *56*, 69–78.

Liu, J., Lim, K. W., Ho, W. K., Tan, K. C., Srinivasan, R., & Tay, A. (2003). The intelligent alarm management system. *IEEE Software*, *20*, 66–71.

Liu, X., Iftikhar, N., & Xie, X. (2014). Survey of real-time processing systems for big data. In *Proceedings of the 18th International Database Engineering & Applications Symposium* (pp. 356–361). ACM.

Makanju, A., Zincir-Heywood, A. N., & Milios, E. E. (2012). Interactive learning of alert signatures in high performance cluster system logs. In *Network Operations and Management Symposium (NOMS), IEEE* (pp. 52–60). IEEE.

Man, Y., Chen, Z., Chuan, J., Song, M., Liu, N., & Liu, Y. (2016). The study of cross networks alarm correlation based on big data technology. In *International Conference on Human Centered Computing* (pp. 739–745). Springer.

Mannila, H., Toivonen, H., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, *1*, 259–289.

Medem, A., Teixeira, R., Feamster, N., & Meulle, M. (2009). *Determining the causes of intradomain routing changes*. Technical Report University Pierre and Marie Curie.

Mehta, B. R., & Reddy, Y. J. (2014). *Industrial process automation systems: design and implementation*. Butterworth-Heinemann.

Minaei-Bidgoli, B., Asadi, M., & Parvin, H. (2011). An ensemble based approach for feature selection. In *Engineering applications of neural networks* (pp. 240–246). Springer.

Minaei-Bidgoli, B., Parvin, H., Alinejad-Rokny, H., Alizadeh, H., & Punch, W. F. (2014). Effects of resampling method and adaptation on clustering ensemble efficacy. *Artificial Intelligence Review*, *41*, 27–48.

Mirheidari, S. A., Arshad, S., & Jalili, R. (2013). Alert correlation algorithms: A survey and taxonomy. In *Cyberspace Safety and Security* (pp. 183–197). Springer.

NGMN (2015). *5G White paper*. White Paper.

Noda, M., Higuchi, F., Takai, T., & Nishitani, H. (2011). Event correlation analysis for alarm system rationalization. *Asia-Pacific Journal of Chemical Engineering*, *6*, 497–502.

Parvin, H., Alinejad-Rokny, H., Minaei-Bidgoli, B., & Parvin, S. (2013a). A new classifier ensemble methodology based on subspace learning. *Journal of Experimental & Theoretical Artificial Intelligence*, *25*, 227–250.

Parvin, H., Minaei-Bidgoli, B., Alinejad-Rokny, H., & Punch, W. F. (2013b). Data weighing mechanisms for clustering ensembles. *Computers & Electrical Engineering*, *39*, 1433–1450.

Parvin, H., MirnabiBaboli, M., & Alinejad-Rokny, H. (2015). Proposing a classifier ensemble framework based on classifier selection and decision tree. *Engineering Applications of Artificial Intelligence*, *37*, 34–42.

Ramiro, J., & Hamied, K. (2011). *Self-Organizing Networks: Self-Planning, Self-Optimization and Self-Healing for GSM, UMTS and LTE*. New York, NY, USA: John Wiley & Sons.

Ren, H., Stakhanova, N., & Ghorbani, A. A. (2010). An online adaptive approach to alert correlation. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 153–172). Springer.

Salah, S., Maciá-Fernández, G., & Díaz-Verdejo, J. E. (2019). Fusing information from tickets and alerts to improve the incident resolution process. *Information Fusion*, *45*, 38–52.

Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks*, *61*, 85–117.

Shields, G. (2007). *The Shortcut Guide to Network Management for*

the Mid-Market. Realtimepublishers.

Smith, R., Japkowicz, N., & Dondo, M. G. (2005). Clustering using an Autoassociator: A Case Study in Network Event Correlation. In *Proceedings of the 17th IASTED International Conference on Parallel and Distributed Computing and Systems* (pp. 613–618).

VanCamp, K. (2016). Alarm management by the numbers. In *Chem. Eng. Essentials CPI Prof.*. Autom. Control.

Verma, D. C., & Verma, D. C. (2009). *Principles of computer systems and network management*. Springer.

Waldbusser, S. (2006). *Remote network monitoring management information base version 2*. Technical Report.

Wang, J., Li, H., Huang, J., & Su, C. (2015). A data similarity based analysis to consequential alarms of industrial processes. *Journal of Loss Prevention in the Process Industries*, *35*, 29–34.

Wietgrefe, H., Tuchs, K.-D., Jobmann, K., Carls, G., Fröhlich, P., Nejdl, W., & Steinfeld, S. (1997). Using neural networks for alarm correlation in cellular phone networks. In *International Workshop on Applications of Neural Networks to Telecommunications (IWANNT)* (pp. 248–255).

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, USA: Morgan Kaufmann.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Zali, Z., Hashemi, M. R., & Saidi, H. (2012). Real-time intrusion detection alert correlation and attack scenario extraction based on the prerequisite consequence approach. *The iSC international Journal of information Security*, *4*, 125–136.

Zhang, K., Wang, Q., Chen, Z., Marsic, I., Kumar, V., Jiang, G., & Zhang, J. (2015). From categorical to numerical: Multiple transitive distance learning and embedding. In *Proceedings of the 2015 SIAM International Conference on Data Mining* (pp. 46–54). SIAM.

Zheng, K., Yang, Z., Zhang, K., Chatzimisios, P., Yang, K., & Xiang, W. (2016). Big data-driven optimization for mobile networks toward 5G. *IEEE network*, *30*, 44–51.