Heta Strömberg

Anomaly Detection in IoT Data Streams

Master's Thesis
in Information Technology
February 1, 2024

University of Jyväskylä

Faculty of Information Technology

Kokkola University Consortium Chydenius

Author: Heta Strömberg
Contact information: heta.stromberg@gmail.com
Phonenumber: 040 700 9223
Title: Anomaly Detection in IoT Data Streams
Työn nimi: Poikkeavuuksien havainnoiminen IoT-datavirroissa
Project: Master's Thesis in Information Technology
Page count: 93+7
Abstract: Since the interest to IoT systems is constantly increasing, it is vital to recognize that the IoT data streams contain anomalies. Anomalies can be caused by system failure, network issues or malicious attacks and can lead to misinterpreted results if they are not found and handled properly. There are different ways to find the abnormal values from IoT data streams. The approach varies based on different aspects such as the IoT architecture and type of the anomaly. This Master's thesis presents a scalable procedure to detect anomalies from IoT data streams on the fly. As an artifact of design science it was created a procedure diagram and checklist to find the appropriate solution for each project of detecting anomalies from IoT data streams.
Suomenkielinen tiivistelmä: Kiinnostus IoT-järjestelmiin on selkeästi kasvussa ja sen myötä on entistä tärkeämpää tunnistaa, että IoT-datavirrat sisältävät poikkeavuuksia. Näitä voivat aiheuttaa järjestelmien tai tietoliikenteen toimimattomuus tai kyberhyökkäykset. Poikkeavuudet voivat johtaa vääriin johtopäätelmiin, jos niitä ei löydetä ja käsitellä ajoissa. Poikkeavuuksien löytämiseen IoT-datavirroista on erilaisia menettelytapoja ja menettelytavan valinta on riippuvainen erilaisista seikoista, kuten IoT-arkkitehtuurista ja poikkeavuuden tyypistä. Tämän pro gradun aiheena on luoda skaalautuva menettelytapa poikkeavuuksien havaitsemiseksi lennosta IoT-datavirroista. Suunnittelutieteen artifaktana esitellään prosessikaavio ja tarkistuslista, joiden avulla löydetään parhaiten sopiva menettelytapa IoT-datavirtojen poikkeavuuksien havaitsemiseksi.
Keywords: Big data, IoT, anomaly, ADE, machine learning, statistic, classification, clustering
Avainsanat: Big data, IoT, poikkeavuus, ADE, koneoppiminen, tilastotiede, luokittelu, ryhmittely

Glossary

ADE	Anomaly Detection Engine is being used for detecting anomalies from data streams.
AI	Artificial Intelligence is machines, especially computer systems, making intelligence decisions.
ANNs	Artificial Neural Networks are a branch of machine learning models. ANN is based on an assemble of connected nodes (artificial neurons), which resemble the neurons in human brain.
Autoencoders	AE is a type of AI where it learns the input and output data.
Big Data	Is gathering, storing , sharing, searching, analyzing and presenting very huge datasets that are unorganized and continuously increasing.
CNN	Convolutional Neural Network is an artificial neural network mostly used with images.
CoAP	Constrained Application Protocol is a specialized protocol for Internet applications that require constrains such as sensors that are constrained because of the limited size.
Collective anomaly	A sequence of data points that differ from the rest of the data.
Contextual anomaly	An occurrence that may be anomalous in some context but in some other context the same occurrence is considered as normal.
DBSCAN	Density-based spatial clustering of applications with noise is a clustering algorithm.
Decision Tree	Is a supervised learning algorithm which has a tree structure and the leafs present the possible outcomes.

Deep learning	Is one part of artificial intelligence where machine is learning with artificial neural networks with representation learning.
Discriminative model	A class of supervised machine learning models which make predictions by estimating conditional probability.
High-dimensional data	Data in which the number of variables observed are close to or larger than the number of observations.
Hub	A connection point, which is used to connect devices in a network.
IoT	Internet of Things consists on devices communicating and sharing information with each other. Devices in the system can be remotely controlled and followed through network.
K-means clustering	Is used in unsupervised learning where each observation belongs to the cluster that has the nearest mean.
KNN	K-Nearest Neighbor is a supervised learning method. K-NN is used for classification and regression.
Loss function	A machine learning method that evaluates how well your algorithm models your dataset.
Machine learning	Is one part of artificial intelligence where machine is learning independently from data without any instructions.
Multivariate time series	A series with multiple time-dependent variables. Each variable has dependency on other variables and depends on its past values also.
Naïve bayes	NB is statistical-based approach for classification.
Point anomaly	Abnormal data point which is far from the other data.
SVM	Support Vector Machine are learning models that use classification and regression.
Time series	A series of data points in chronological order. Data points can be indexed, listed or graphed.

Univariate time series	A series with a single time-dependent variable.

Contents

1 Introduction

The volume of real-time network data is constantly increasing because of the ever-growing number of Internet of Things (IoT) devices and their systems that utilize them [18]. The data from IoT data streams are used for decision making and hence it is crucial that the data do not contain irregularities. Abnormalities in the data can be false or missing values or they can be malicious attacks. No matter which kind of anomaly appears in the data, they can lead to misinformation and compromise the validity of decision making. Because the data volume is growing, the need to detect abnormalities from the IoT data streams is getting progressively important all the time.

This Master's thesis aims at outlining guidelines for detecting anomalies from IoT data streams automatically on the fly, with the smallest amount of human interaction, in a real-life case. More precisely, the aim is to find *a procedure* to detect empty values from IoT data streams. For this reason, specific anomaly detection platforms will no be analyzed in detail. Instead, attention is paid in providing a procedure diagram to present the steps that are required when planning the system for detecting anomalies.

Internet of Things (IoT) consists of devices connected through internet without human intervention [7]. IoT is a constantly increasing field, which means that the amount of IOT data is constantly increasing as well. It is challenging to process the data and gather them from IoT environments. [31] There may occur various kinds of anomalies in IoT data streams and the variety of reasons for the anomalies is similarly wide. In some cases the reason can be, for example, an error with the equipment, while in other cases the seasonality can cause data to be interpreted as anomaly. [13] Research on the anomaly detection is quite rare but despite of the small amount of the research, it is obvious that discovering data anomalies from IoT systems cannot be done with one solution that would fit for all the IoT systems. [34]

Sensor data take usually the form of time series, which means that observations are recorded at a specific point in time [5]. Generally used way to gather data from IoT devices is in the form of time series [42]. Typically, data gathered from IoT devices are used for monitoring. Data from IoT devices are also used for predictions.

Predictions are utilized when there is a need to know about the behavior of environments. Both monitoring and predictions can be in the form of time series. Time series can be divided into univariate and multivariate time series data. Detecting anomalies from multivariate time series needs much more attention since it cannot be done with only one chosen method. [7] The case depicted in this thesis utilizes univariate time series.

Anomalies are always present in the data due to the device defects and adverse environment [52]. Thus, it is important to recognize anomalies from IoT data streams. Common tools for anomaly detection are machine learning and deep learning [11]. Statistical methods are not always effective enough for anomaly detection, especially in real-time, since handling huge amounts of data may become challenging [42]. Still statistical techniques are being used together with machine learning in some cases. Machine learning has proved to be reliable in anomaly detection [48]. Machine learning can provide help when analyzing big data, consisting of huge amounts of diverse data that is constantly changing [14].

The system investigated in this Master's thesis already has a capability of detecting anomalies in IoT data that has been transferred into a storage. This anomaly detection has shown that there are some empty values coming from the sensors. The problem with this solution is that it is not happening real-time and it takes lots of resources to execute it. In practice, empty values are detected by running a SQL query on the database. The aim of this Master's thesis is to answer to a question "What kind of methods there are to detect anomalies from IoT data streams?". The aim is to create a procedure which will provide the guidance for finding the right things to consider when detecting anomalies from IoT data streams on the fly. The target organization also requested that this procedure should be scalable and as automated as possible.

Anomalies caused by cyber attacks are not considered in the scope of this thesis. The cause for anomalies can be cyber attacks but since the attack attempts are not supervened in the case system, they will not be included in this study. Other kind of anomalies, like a missing value or an inaccurate value, may lead to misconception when analyzing the data and hence lead to false conclusions [9]. Furthermore, finding the false or missing data points can save resources [13].

Design science is used as a research method within this Master's thesis. Design science aims to create innovative products called artifacts, that serve human purposes. Artifacts can be methods, models, constructs, and implementations. Two basic components are found from design science: build and evaluate. In the build phase the artifact is constructed, and in the evaluation part the artifact will be reviewed to see how well it works. [30] Feedback about the created artifact is important and for this reason iterations when creating the artifact play a crucial role. The usability of the product describes the quality of the product. [3]

This thesis has been divided into six different sections. Chapter two after introduction is about IoT data and the facts that are making the analysis of IoT data complex. These are the main challenges that can be seen with big data as well. Within this chapter it is also explained what are the different types of anomalies there can occur and the difficulties to find anomalies from IoT data streams. The third chapter is presenting the different kinds of methods that have been used when detecting anomalies. The methods in third chapter are divided to statistical and artificial intelligence approaches. Both of these methods have their benefits and limitations and they are presented in the third chapter. When it comes to artificial intelligence and machine learning, it is always necessary to decide whether to use supervised or unsupervised learning and those are presented in chapter three. Chapter four explains details about design science and why it was a natural choice to use as a research method in this Master's thesis. Also, the constructing of the procedure diagram and benefits of using checklist are presented in chapter four. Chapter five introduces the real-life case and the iterations for the process. In chapter six there are conclusions and the things to consider in the future regarding to detecting anomalies from IoT data streams.

2 Internet of Things and data streams

2.1 Internet of things

Internet of Things (IoT) consists of devices and sensors that are communicating and sharing data through network. It has been applied to several different sectors, for example in industry, healthcare and transportation [4], [51]. IoT has spread widely to various areas of life. Many people and companies find it useful for several processes because of the automated data collection and monitoring. [42]

The basis of IoT is to collect data from heterogenous devices and transmit the data to processing centers for analysis. The connections between sensors and platforms are wireless. [52] The commonly used IoT sensors measure physical surroundings such as temperature, pressure, and humidity. Sensors can also measure gas, motions, and there are optical sensors and Radiofrequency Identifiers (RFID) as well. The limited size of the sensors constrain the IoT systems. Limited size affects to computing power, power of the battery, memory and storage space, and networking capability. These are also the reasons why sensors are in danger of attacks, failures and breakdowns which are causing the losses of the data. [46]

Several sectors, such as health care, households, industry, and traffic, use IoT widely in automation [29]. A typical IoT system consists of sensing devices that generate the data, edge computing for processing the data, networks transmitting the data, platforms organizing the data and sending it to applications, and applications processing and analyzing the data [1]. Cloud computing provides storage, computation power that is adaptable and economical, and software services to help build applications [29].

Every node in IoT network connects with a server, and the nodes maintain a persistent connection. This leads to congestion in the network of the devices. Packet retransmission or loss, latency, and extended energy consumption, are normal when there is congestion in the network. These may overload the computational resources and for this reason there are specialized tools to make this lighter. These tools deallocate resources. For this reason, it is vital to use standards and communication protocols specifically designed for IoT systems. [48] Networks currently used meet the prerequisites for the employed IoT equipment sufficiently. In the future, when the existing communication infrastructure will increase, network will though require further improvements. For example, mobile networks relying on 3G and 4G technology must be enhanced with the wider implementation of 5G solutions. [12]

2.2 Architecture of IoT

Several applications have been introduced in order to improve the measuring the environment by the means of IoT. Examples of such applications are smart cities, intelligent transport systems and smart healthcare. The wireless sensor network plays crucial part in these applications and their popularity has grown since the wireless network can be expanded ad hoc, and for infrastructure there is no specified requirements. Sensor network can be deployed randomly, and it has capability of self-organizing. [25] In IoT networks, the primary objectives are devices sensing the environment (sensors), the scoping of system signals, and systems that obtain data to help in making decisions. Without infrastructure requirements, the sensors are capable of establishing an ad hoc network. Sensors can be positioned quite freely. [46]

A typical IoT architecture is three-layered. It consists of physical layer, network layer, and application layer. Hardware equipment and devices are in the physical layer. Devices such as sensors and actuators collect the data systematically. Devices also send and receive the data to and from the network layer. Communication is done by standards and protocols. Examples of these standards are Bluetooth and ZigBee, while examples of protocols are Wi-Fi and 6LowPAN. Network layer verifies the routing of data packets, and it acts as the gateway layer. Network layer is using standards like 4G, 5G, and IPv6. The controlling of communication between physical and application layers is done on the network layer. The application layer, or also known as the software layer, processes the data visualization for end users applications. The communication in application layer is from machine to machine (M2M). The protocols used in the software layer are, for example, the Constrained Application Protocol (CoAP) and Data Distribution Services (DDS). [1] The basic architecture is presented in figure 2.1.

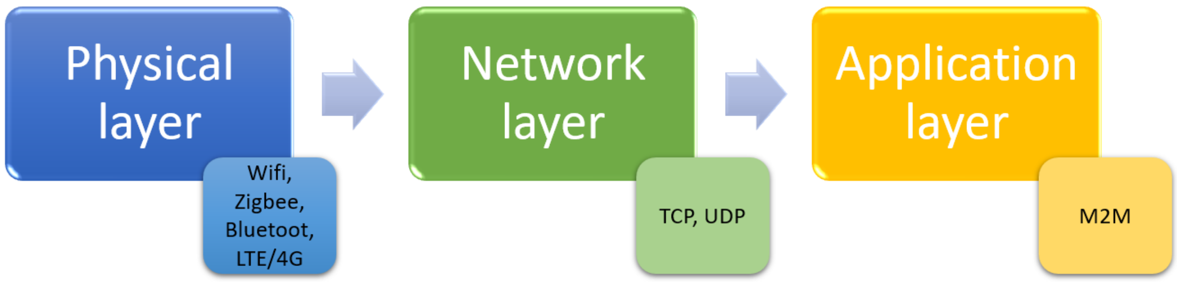


Figure 2.1: Basic architecture of IoT.

Huc and Trcek [22] present an IoT network model with four layers: application layer, transport and networking layers, physical and link layers, and sensing layer. The sensing layer consists of sensors, while the network layer has protocols, for example, to provide energy efficient routing. In transport layer UDP protocol is often used for connection. For sensors' further interactions via physical and link connections, hubs are deployed. Hubs are able to run the IoT stack protocols. [22] IoT network infrastructure is being described by Protogerou et al. [39] with three layers: IoT devices, Fog Nodes and the Cloud. Commonly, the IoT management applications are executed on Fog Nodes, when they process with the time-dependent data. [39]

Wireless communication protocols and LTE/4G mobile technologies are used for IoT sensor device communications. [46]The main protocols used at the transport layer are transmission control protocol (TCP), which is more suitable when the reliability of the connection is preferred, and user datagram protocol (UDP), which suits better in situations where real-time data is needed. Application layer contains more diverse protocols for machine to machine (M2M) communication. Constrained application protocol (CoAP) is one example of the communication protocols, which is compressed and customized version of the HTTP (hypertext transfer protocol). [48]

The IoT data processing architecture consists of the devices that are collecting the data, the data processing and data analysis layer. Imputing of the missing data and data aggregations are done in the data processing layer. In the data analysis layer the platforms play the main role. [25] The data fusion layer aims to collect the data from several IoT devices and combine it for further analysis. [46] By data integration or fusion Krishnamurthi et al. mean that data is gathered from multiple sensors. [46] Data processing phases are presented in figure 2.2.

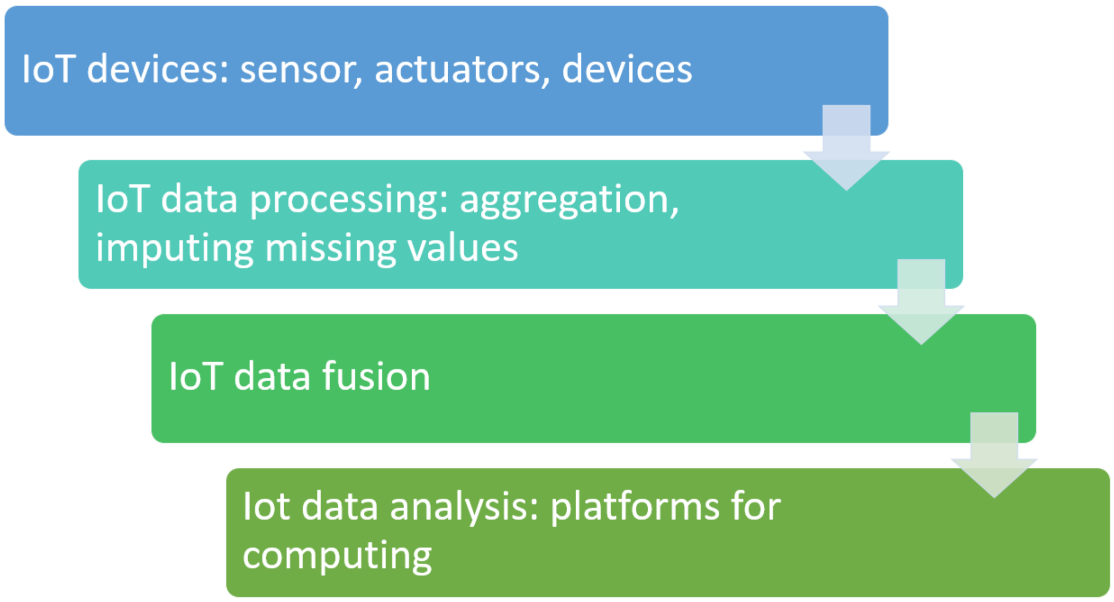

IoT devices: sensor, actuators, devices
IoT data processing: aggregation, imputing missing values
IoT data fusion
Iot data analysis: platforms for computing

Figure 2.2: Data processing in IoT data streams.

The layered structure of IoT architecture helps in accomplishing the requirements for low-cost and low power consumption of IoT devices. Several protocols and techniques, such as WiFi, Bluetooth, Zigbee and Near Frequency Communication (NFC), have been developed for helping the manufacturers to reduce the costs and power consumption of sensors. Still, there can occur issues, such as data loss, redundancy of data, or problems with data generations. [46]

2.3 Characteristics of data in IoT

2.3.1 Big data and IoT

Big data has been described with three features that are constantly present: volume, variety and velocity. The main purpose for collecting big data is to help with decision making in various different sectors. IoT data can be classified as big data once it has been gathered from the environment. This classification can be done because in this phase IoT data has the main features of big data: volume, variety and velocity. [31] The voluminous IoT data needs further analysis and, in most cases, in real-time. The sensor data has high velocity and huge volumes. Furthermore, it is highly heterogeneous. In order to procure useful information for decision making purposes, data generated by sensors need gathering, aggregating, analyzing, and visualizing. Data can take various formats, for example boolean, binary, featured values, continuous data, or numeric values. A generic representation of the data is as a small-sized tuple that contains information in structured format. [46]

Krishnamurthi et al. [46] note that sensor data is complex by character. The causes of the complexity are large volumes, the dynamic nature of data, real-time updating, and constant aging of data. Data sources also have interdependence between each other. [46] While IoT has spread widely, the analyzing of big data has become more challenging when the data is being gathered from various different sensors and objects. Even though both IoT and big data are used in many systems, the development of both of these is still in the early phases. [31] Machine learning and deep learning are considered as advanced analysis techniques. Recently, these techniques have been employed for IoT big data analysis [6].

The main characters of big data (volume, variate and velocity) require powerful infrastructure for storing, processing, and analyzing the big data. Cloud computing models are popular for this purpose. [6] IoT data can provide more valuable information and better services by exploiting cloud computing and big data technology [52]. Even though cloud provides excellent option for storing and processing the data, it is still good to keep in mind that networking capacities need to be thought as well. More bandwidth is needed when transmitting huge amounts of data to the cloud, and it also consumes a lot of energy [43].

Automation in different fields is an important factor and IoT has provided a valid solution for the automation. With IoT systems it is possible to have real-time information about the environment almost from anywhere. If automation is capable to provide data about changes in parameters or environment, it can be used via IoT systems [43]. This feature has been widely accepted for daily usage not only in business and industry but also at homes. The data from environments is gathered via IoT systems but it also needs to be analyzed also. IoT systems often rely on the analytic power of big data, and that is why big data technologies are one of the key components for IoT [6].

Big data requires special technologies to collect, process, analyze, and store the huge amounts of data. Examples of these technologies, especially for real-time data, include Spark, Hadoop, Hive, Hbase, Flume, Kakfa, Strom, Samza, and S4. Vulnerabilities, velocity and volume of the data are the features that create challenges to detect anomalies on real-time. [18]

2.3.2 Time series data

Popularity of IoT has grown widely even though it is rather new computing paradigm. Data gathered from IoT devices are generally used for monitoring purposes. Also, predictions of the behavior of systems and environments are done based on the gathered data. This kind of data is usually structured as time series data. [7]

The generation and transmission of the time series data has high velocity. Devices that sense the environment, are generally the primary source for data visualizations and data mining. [42] A time series can be continuous or discrete. In continuous time series values are recorded continuously for a window of time, and in discrete time series observations are measured at fixed points in time. [5] The essence of continuous time series is that the data is constantly changing [7]. Data samples may also be random and scarce. This means that real-time data from time series is usually non-linear, non-stationary, and contains high noise. [54]

Time series can be classified to univariate or multivariate. A multivariate time series is a multi-dimensional sequence that contains numerical values from multiple sensors at discrete time instants. Anomalies in multivariate time series data cannot be classified as simply as in univariate time series, where anomalies can be categorized to point, contextual and collective anomalies. In multivariate time series all variables need to be considered simultaneously. [7] Most statistical methods can handle only univariate data which means that the methods complete predictions only from one value. Deep learning methods, on the other hand, can also handle multivariate time series data. [28] One approach to detect multivariate time series anomalies is to compress these time series to a lower dimension and reconstruct it back to the original state. [7]

Special methods for IoT anomaly detection are needed since data from sensors usually is in the format of time series. Time series data is temporal, and also presented as sequences. [52] The temporal nature of time series data is hard to handle even with machine learning approaches. Seasonality and trends are seen in time series, and these produce some difficulties if the data is not preprocessed correctly. [28] Preprocessing the data is an essential part of machine learning methods. Preprocessing of the data is presented in chapter four more thoroughly.

Sensors can collect data continuously, discrete or as triggered by an external event. [46] Trigger might be attached to temperature, for example, and when the temperature changes over selected range, the sensor will send the value to dataset. The real-life case of anomaly detection depicted in this Master's thesis needs to be run against this kind of triggered time series data and also, the kind of data that has fixed timestamps when collecting the data. The format of timestamps can vary in different datasets. Both of these types of time series are containing only univariate time series data. These time series are presented in the chapter five.

2.4 Data quality

Data quality is crucial in IoT because the decisions based on the data aim to be reasonable [53]. Several things in IoT systems can cause the values to be incorrect. Problems may be caused by human errors, system failures or network related issues. Environment has impact on the quality of the data since altered factors in the environment may cause failures for the sensor values. Reasons for the failures should be recognized and fixed. Furthermore, the anomaly detection methods should be transparent for ensuring the quality of the data. When using machine learning methods, the data needs to be always preprocessed and it is important that these actions are transparent as well.

Integrity and accuracy of the data are in main role when defining the quality of an IoT service [52]. IoT is applied for example in smart city, healthcare and biology, smart car and transportation, and various industry applications. This brings not only possibilities, but also responsibility to confirm that the collected data is accurate and efficient. [51] IoT systems consist of several physical objects in the environment and the main task for objects is to constantly sense, gather, and compute the data while communicating with each other. The systems are suffering from noisy, faulty and missing data that are mostly caused by loss of battery power, communication errors, and malfunctioning devices. Missing sensor data is a very common problem in IoT. Missing data can be caused by issues with network or synchronization, and sensors or other equipment may be unreliable. [17]

External factors may have a huge impact on the generated IoT data and may affect negatively to the quality of the data. Miscellaneous sensors are usually deployed in the ascetic environment. Weather, for example, can cause problems with sensing of the devices. [52] Accurate and fast data transformations between devices also increase the reliability of the IoT system. The common reasons for low data quality are faulty connections, sensor failures, and security attacks. [2] No matter what the cause for sensor failure is, it is important to recognize the risk. [52]

Values that diminish the quality of data can be divided into two categories: abnormal values and missing values. The quality can be improved when the number of missing values is reduced. Missing values can be replaced by predictions when pattern of the data is known. [53] Even though missing data can be eliminated, this could lead to incorrect analytical results since the imputed values are just predictions instead of actual values [17].

Real-time data is commonly wanted and IoT systems can contain several sensors and other objects. It is important to recognize the uncertainties of the data before the data is analyzed. For further data analysis the uncertainties must be recognized and removed. Uncertainties can be removed by reducing the noise of the data, detecting anomalies, imputing missing values, and aggregating data. [46] When the number of devices is constantly increasing, this can burden the network extremely, and cause delays [43].

When detecting anomalies from data, it is crucial from the data quality perspective to make the detection process as transparent as possible. Huc and Trcek [22] have discovered that even with the same dataset the results can be different when detecting anomalies, if the data is being presented differently or preprocessing of the data has been done in different ways. Therefore, the discovery by Huc and Trcek [22] is that transparency with not only the data but also with machine learning algorithms are important components for anomaly detection. [22]

Data quality is an important thing to consider with machine learning as well. Machine learning uses algorithms to learn historical patterns from the dataset and makes predictions based on the patterns. With big data, machine learning provides analyzing even when the datasets are large, diverse, and constantly changing. One thing to highlight here is that whenever using machine learning, the data must be handled appropriately. This means that the cleansing and processing need to be done thoroughly. Learning algorithms are iterative when building models. [14]

2.4.1 Preprocessing the data

In order to have reliable results with artificial intelligence and anomaly detection, the data needs to be suitable for the chosen algorithm. This is an important part of the data quality. Preprocessing is a phase before the actual detection of the possible anomalies, where the data is prepared to be understandable for the machine learning algorithm. Basic steps for preprocessing the data before machine learning are cleansing the data, removing noise and reducing the dimensionality. [48].

Machine learning cannot work with raw data [48]. Hence it is important to transform raw data so that machine learning algorithms understand the data. Because anomalies in the data are extremely rare, it is important not to lose the information of abnormal data when preprocessing the data. The small amount of abnormal data may also lead to imbalanced training of the machine learning model which causes challenges for most of the machine learning approaches. [50]

One way to handle imbalanced data is to increase the size of a minority class. When the dataset is equally rebalanced, it forces the algorithm to notice both of those classes. Synthetic minority over-sampling technique (SMOTE) is an algorithm that creates new data for the underpopulated class. SMOTE uses the points nearest neighbors by choosing the minority sample at random, and selecting from the minority samples one of its k-Nearest Neighbors. Then SMOTE creates a new point by interpolating between the two points along each dimension separately. [50]

In machine learning algorithms some of the features in the dataset can determine the general behavior of the sample. At the same time other features may not bring any redundant information. Therefore, it is necessary to have knowledge on the dataset. Based on the knowledge, it is easier to determine the relevant features for the machine learning algorithm. [48] Feature selection means that appropriate features need to be selected to point issues that may occur, such as repetitive, insignificant, and noisy features. This can be done manually. Since lots of features might interfere the clustering process, it is important to select the appropriate features for detecting anomalies. [18] Feature selection also requires evaluation of the data types in the dataset. If the selected machine learning method obliges arrays to have integers as input data, the data needs to be transformed into numerical format. [48]

Time series have high dimensions and some of the algorithms do not work well with high-dimensional data. That is why it is crucial to reduce the dimensions. When reducing the dimensions, it is also important to keep in mind that necessary information still needs to remain. For anomaly detection in the time series, most of the machine learning algorithms needs to be transformed into two-dimensional if it already is not. [52]

2.4.2 Evaluation of the machine learning model

Part of the data quality is to know how well the machine learning model performs. This can be measured. Measuring gives the actual result if the machine learning model has been created in a correct way. There are some methods to perform the measuring. The methods vary based on what kind the learning problems are (classification or clustering). The most used metrics for this purpose is accuracy, F1 Score, precision and recall [19]. These metrics can be used together to get best understanding about the model performance. [7] [16]

Evaluation of the classification problems often uses accuracy as one of the metrics to see how well the model has performed, although the models performance can be only partially evaluated with accuracy. [19] Even though accuracy has been used with different algorithms, it is usually giving the similar kinds of values [11]. In binary and multi-class classification problems, accuracy is extensively used evaluation metric. Accuracy gives the ratio of the subset of accurate predictions over total observations. [48]

In clustering problems, it is harder to define the quality of the algorithm. This applies especially to clustering algorithms where several clusters exist. Some of the methods for clustering evaluation are statistical. [35] Accuracy as a measure is for classification methods but it is used with clustering once the data is clustered. This means that the clustering problem eventually needs to be treated as a classification problem in order to measure the accuracy. [7] [16] Accuracy as a measure tells about the results of the clustering, and how well the algorithm has recognized certain cluster points to be in a correct cluster [18].

Another evaluation metric is F1 and this works especially with binary classification. F1, precision and recall depict if the instances belong to positive or negative class. Precision gives the positive predictive value as a ratio of positive instances being predicted correctly. [19] [48] Recall gives the actual positive rate. [19]. Recall will give high value when the false positives are low [14].

False positive or false negative are errors and they mean that some of the predictions have gone wrong while using machine learning algorithm. These may lead to false conclusions and can cause some extra costs for the business. [14] Hence it is vital to understand the importance of the evaluation of the machine learning models.

The equations for accuracy, F1, precision and recall are as follows:

$$Accuracy = \frac{True\ positive + True\ negative}{True\ positive + True\ negative + False\ positive + False\ negative} \quad (2.1)$$

$$F1\ Score = \frac{2 * True\ positive}{2 * True\ positive + False\ positive + False\ negative} \quad (2.2)$$

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \quad (2.3)$$

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \quad (2.4)$$

As can be seen, the equations are based on the ratio of false positives and negatives, and true positives and negatives. The F1 formula (equation 2.4) gives weighted average of the precision. [19] These evaluations tell about the errors in anomaly detection. If the outcome for the evaluation is poor, parameters need to be redefined. Hyperparameter tuning is something that is done when trying to find the best parameters of an algorithm. For clustering this is quite often needed. [35]

2.5 Anomalies

Anomalies are abnormalities that differ from the predicted pattern of the data. [39]. Anomalies may appear when there are experimental error or an anomaly in the measurement [5]. Data from the devices may have anomalies for different reasons. They can be caused by malicious attacks or sensor failures, for example. [4] Anomalies in IoT systems may result in deviated predictions in data analysis [43]. Suspicious points must be detected from time series so that the analysis of the data will be correct [5]. Anomaly detection is a technique that discovers unusual behavior different to predicted patterns [4].

Since IoT data is used for decision making, it is important that the data is reliable. Protegorou et al. [39] have raised amongst other research that there is an urgent need to mitigate or hinder anomalies as well as enhance already existing techniques. This will not only sustain the global stability but also improve the consumption of resources. [39]

2.5.1 Different types of Anomalies

Various sources have presented that there occurs three types of anomalies: point anomaly, which is a single data point that differs from the other data points, contextual anomaly, which at one viewpoint can be normal but in another viewpoint it may be abnormal, and collective anomaly, which requires more knowledge about the data itself because the abnormality can be in sequences [13][42][15]. This classification divides the anomalies to different categories based on the characteristics of the abnormal data. The detection of these anomalies differ from each other since the abnormal data has specific features based on its type.

Point anomaly is abnormal data point which is far from the other data points. This type can occur as a sudden peak in the graphs and it is usual that this will deform the data especially when aggregating it. These exceptions need to be transformed before going further in analyzing the data. Point anomaly can be caused because there is something wrong with the sensor or when reading the sensor. [13]

In contextual anomaly, the data seems to be normal from one point of view but then at a another scenario it may be abnormal. [13] Contextual anomaly can be related to time. For example if there is certain amount of traffic on the streets in the early morning, the traffic may seem extremely large at noon. [42] Hence contextual anomaly requires knowledge of the context in order to recognize abnormal data [13].

Collective anomaly is the third type of anomalies. In collective anomalies the data needs to be analyzed before classifying it to normal and abnormal because the abnormality occurs in sequences instead of points. Any exception can lead to collective anomaly. After the normal pattern has been identified and analyzed, the abnormal behavior can be recognized. For analysis it is not enough to analyze only one point of observations but the sequence of the data need to be analyzed. [13][42] These kinds of anomalies need to not only be recognized, but they also need to be either removed or manipulated in some way [13].

Zhang et al. [54] have suggested that anomalies can be categorized into three categories: point anomalies, subsequence anomalies, and anomaly time series. In this categorization the subsequence anomaly is described to be the same as contextual anomaly. The time series anomalies are presented so that the entire time series is suffering from anomalous data. In order to detect these kinds of anomalies, the input data needs to be a multidimensional time series.[54]

2.5.2 Missing data in IoT data streams

Missing data point values are one of the main issues in IoT systems. Missing data can happen because there is a malfunction in a device, there can be issues related to device identification or there may be human failures or something has been neglected in a system. [34] The data is considered incomplete if it contains missing values. Data containing missing values can produce inaccurate or unreliable results. Vangipuram et al. [46] are presenting that for the data quality it is crucial to impute the missing values. [46]

Datasets containing data from IoT environments usually have empty values [46], [47]. This can be due to unreliable links or unexpected damages. [47] In order to find missing values, there are three steps. The first step is to find the reason for missing values. Missing values can be caused by various reasons, such as issues with network, damaged sensors, environmental circumstances, and issues with synchronization. The second step of finding missing values is to use approaches to search patterns of the missing data. The last step is to approximate the value of the missing data by using the approach of recognizing the missing value patterns. Classification tasks are impossible to apply if there occurs missing values in the dataset. Missing values must be either removed or replaced if classification tasks are needed. Imputation is filling missing values from datasets. [46]

If the node fails automatically because of environmental factors or depletion of energy, the data may be lost forever. Data recovery is an important factor in these cases. [47] There are traditional imputation algorithms but they are not recommended for IoT data. [46] Imputation technique is the most familiar technique to handle missing data. In imputation technique, the empty data points are imputed with the calculated values. Regression imputation and mean imputation are examples of these techniques. The problem with these techniques is that they have poor performance when it comes to the large dimension of missing data. [41] Vangipuram et al. [46] have used algorithms called F-Kmeans and K-means for imputing missing values. With those algorithms the first step is to identify the distance between two sensors (or other devices) to determine the nearest and most appropriate neighbor. Next steps require computing to find appropriate patterns and for clustering the data. This leads to finding similarities of the patterns and later from the instances. [46]

In most cases, missing data is characterized through three benchmark patterns based on if the data is missing random. Missing completely random (MCAR) means that there is a sequence of missing data, and the sequence is not dependent of the variables of the dataset. Missing at random (MAR) means that missing data and variables of the dataset are indirectly related. Missing data and variables are calculated by using probability. Missing not at random (MNAR) means that missing data are directly dependent on the variables of the process. MCAR is the most common in this categorization in wireless sensor network systems. [41]

2.6 Anomaly detection

One of the main factors when it comes to IoT technology is to detect the anomalies. Anomalies appear continuously in real-life systems. Devices can be more reliable and secure when anomalies are detected. It can also help to avoid the waste of resources. There are techniques that have been successfully used to discover anomalies from time series data. [42] The techniques used in anomaly detection in IoT are statistical, machine learning and hybrid models.

Most anomaly detection methods work with the same form: data is collected from the devices and the collected data is evaluated if it matches to any known patterns that contain anomalies. There is a predefined range for similarities in this comparison, and if the similarity is out of this range, data is considered as anomalous. With this approach there is a problem if the anomaly is previously unknown and hence the predefined pattern does not recognize it. In these cases, the anomaly might be interpreted as normal. The second known problem with IoT data stream anomaly detection is that the anomalies should be recognized in real-time if possible. This cannot be done on the cloud computing level by analyzing the already known patterns and doing the comparison. Traditional anomaly detection methods are not able to help with either of these problems. The latter problem increases the need for cloud computing capacity. Traditional methods also suffer the lack of flexibility and scalability. [51]

The method that compares the given data to the new data and tries to search for values that differ from the given data. is called Anomaly Detection Engine (ADE). In order to create any ADE, it is crucial to recognize the variety of anomalies that may occur. Mohamudally et al. [34] are presenting that there are three relevant elements in the development of ADE. First is the data quality, second is the transformations of the time series, and third one is where analytic operations are accomplished. [34]

2.7 Problems when detecting anomalies from IoT data streams

It has been recognized that there are still unsolved problems when it comes to detecting anomalies from IoT data streams. A literature view by Sgueglia et al. [42] represent that even though the interest on IoT devices is increasing, there is still a lack of trustworthy methods that can be used when detecting anomalies. Especially contextual anomaly detection is suffering from this. [42] The problems highlighted in several studies are shown in table 2.1.

Table 2.1: Problems when detecting anomalies from IoT data streams.

	Problem when detecting anomalies	Consequence
1	Multivariate time series	Needs more studies
2	Segmented abnormalities in data	Data is very close to normal pattern
3	Malicious attacks	False positives
4	Dynamic and continuous data	Model is not able to adapt to changing data quickly enough
5	AD on device level	Needs more studies
6	Energy consumption	Limitations for algorithms
7	Abnormal data is rare	Unbalanced data

As shown in table 2.1, one known issue is that it is easier to detect anomalies from univariate time series than it is from multivariate time series. [7]. With the existing algorithms, it is hard to detect anomalies when the data come from multiple sources. [51] More research is required on finding anomalies in multivariate time series data. [7] The second known problem is that when anomalies occur within a single data point, there are ways to find it but in time series data, the data is mostly in segments. For this reason data collected within time series is causing challenges for anomaly detection. Typically, classic anomaly detection aims to find single abnormal data points. Compared to finding anomalies from time series data, there is a need to find segments that are abnormal compared to the other data. This leads to the result that finding anomalies from time series data allows to find also the kind of single data points that are very close to normal when the pattern is abnormal. [42]

Third known issue is the false positives in malicious attacks. For malicious attacks there has been successful anomaly detection when analyzing the normal behavior of the data. Still the current systems are suffering from high rate of false positives. The other concern is that there is still a need for huge amounts of transactions from humans, when interpreting the data generated from IoT devices. This is a problem when the number and scale of IoT systems are constantly growing. [5]

Since the field of IoT applications and systems is constantly expanding, it means that IoT ecosystems have become more and more complex. In addition, IoT data streams are dynamic and continuous. This makes anomaly detection even more difficult. [24] This is recognized as the fourth known issue and it can be seen when the model in machine learning is not able to adapt to the continuously changing data quickly enough. Usually training data for machine learning is originating from historical data. In real-life scenarios IoT data streams are dynamic and continuous. This will lead to a problem that there will not be enough samples for abnormal data to train a model which can cause some imbalance of data streams. [24] Although classification is commonly used when detecting anomalies from IoT data streams, classification is not able to handle the constantly changing, dynamic data. [24]

The fifth problem regarding to IoT anomaly detection is anomaly detection on different layers. One major shortcoming in the field of IoT anomaly detection is that even though anomaly detection can be done on different levels of architecture, still all of those levels have not been tested adequately so that the different levels could be applied for anomaly detection. Anomaly detection can be done on a device level but there are not many studies regarding this. DeMedeiros et al. [11] have pointed out that the majority of machine learning approaches did not consider this option at all. Anomaly detection that is focused on attacks are the most common topic of studies. [11] On the device layer, it is hard to analyze data in a comprehensive way since there are computational limits with IoT devices. Device Identification (DI) is another problem that occurs in IoT networks and device computing in addition to the energy consumption issue. Edge computing makes it possible to filter unwanted data. This means that filtering of the data is done before running analytics on a cloud. [34]

Key thing in IoT networks is that they should work with the least possible amount of energy. This also sets requirements for computing, that also cannot take lots of resources. The sixth known issue in IoT anomaly detection is energy consumption. Because of these specific requirements, there are limitations on the selection of algorithms to be used with anomaly detection [22]. Deep learning has been suggested in several research papers as a feasible method for anomaly detection. However, with deep learning it must be noticed that some of the methods are ambiguous and require capable servers when running the models. [23]

Abnormal data is always rare in a IoT dataset compared to normal data and this is the seventh issue recognized with IoT anomaly detection. Abnormal data is the minority in the dataset. This can lead unbalanced data which in turn can lead to problems if the subset of the data contains only few or even none of the instances that represent the minority class. Most likely the performance of the anomaly detection model is poor in these kinds of cases. [48] In IoT network intrusion system it has been found out that unbalanced data is a common problem because the proportion of abnormal data is so small. [48] This is the case with other anomaly detection as well.

Besides the well-known issues regarding to detecting anomalies from IoT data streams, there are several other problems highlighted in different studies based on the testing of various detection approaches. These problems include the lack of proper real-world datasets for testing, the training of the machine learning models, which requires lots of time and resources, unsuitability of the tools for monitoring the network traffic in context of IoT data streams, and the challenges that originate from the variety of IoT protocols that are in use. Deep learning has also been investigated to be used with anomaly detection and it has gained some good results. However, a number of things need to be taken into account when utilizing deep learning.

Security based approaches for anomaly detection have traditionally been used to observe network traffic. Even though these approaches have given good results, in IoT systems they are not that efficient because of the huge number of devices and volumes of the data. Especially when the expected anomalies have new or mutated forms, these approaches may fail to detect those values. [1] Furthermore, IoT datasets use various protocols, and this is one thing that causes anomaly detection to be challenging. [48]

Linardatos et al. are presenting the downsides when using deep learning methods: large amounts of training data is required, and deep learning approaches tend to overfit the models. Sometimes the results cannot also be explained. [28] Guansong et al. [15] highlight the following regarding to deep learning:

- Deep learning approach is not explored enough,
- In deep learning one issue is that their functions are not enhanced for anomaly

detection,

- Most deep learning approaches for anomaly detection focus on point anomalies,
- Currently used deep learning approaches are mostly detecting anomalies on single data sources.

3 Methods for detecting anomalies in IoT data streams

Anomaly detection is a technique that discovers unusual behavior that is different to predicted patterns.[4] The techniques presented in this study are statistical, machine learning and deep learning. The latter ones are branches of Artificial Intelligence (AI). In order to take advantage of AI, there is always a need to teach models. Teaching can be categorized into three different categories based on the learning method: supervised, unsupervised and semi-supervised[5]. Those will be presented more thoroughly later on. Based on the research it is very clear that there does not exist a single model for detecting anomalies that would fit in all situations [34]. This is because of different architectures, different forms of data, and different types of abnormalities which are all handled in different ways when detecting anomalies from IoT data streams.

Statistical methods and machine learning have been accomplishing anomaly detection for a long time in time series data [7]. Statistical techniques have been mainly focusing on using mean or mode and regression [53]. Machine learning and deep learning both provide the possibility to be used with large amount of data.[42] But since IoT networks have specific requirements such as limited energy consumption and rigorous computing resources, it is necessary to find such machine learning approaches for anomaly detection that will accommodate to the restrictions.[22] Because of the limitations of energy consumption, it needs to be considered carefully what are the gains for the anomaly detection and hence all the actions needed. Imputing missing values is one the things that are causing more energy loss and the accuracy is another. The constantly changing data is setting requirements for the clustering if the data is dynamic. [47]. This means more effort for the maintaining of the anomaly detection system and also more energy loss.

3.1 Different approaches

Detecting anomalies in IoT data stream is challenging. The known challenges are IoT system architecture, dimensionality, noise, stationary data, time constraints, need for domain knowledge, and anomaly reporting methods. Key techniques for anomaly detection are pattern matching, clustering, distance-based, probabilistic, and ensemble. [40] In addition to these techniques Xiang et al. [51] present that observing network traffic is also one of the methods for detecting anomalies from IoT data streams [51]. There are several algorithms that check from a network, if the input data matches to the output data. [8] These approaches have different mechanisms and the anomaly detection engine runs in different stages based on which approach is being used.

The methods that are the most used in anomaly detection compare the data to known patterns and this way decide whether the data is abnormal. Both machine learning and statistical methods are using this kind of approach. Another often used approach is based on distance where two data points has a distance between them and the distance is calculated. If the data with the nearest neighbor is similar, then the data is considered to be normal. K-Nearest Neighbor (kNN) is one of the examples used this way. [51]

Anomaly detection techniques rely mostly on machine learning [40] and those have proven to be efficient. [49] Traditional machine learning methods have been proved to have the capability to learn from complex data with little or no human intervention. [28] Deep learning methods provide the possibility for the model to learn from the data itself instead of human expertise to guide them. [28] Many existing solutions for anomaly detection are problem specific and they use supervised approaches. Supervised approach requires domain knowledge and tremendous data labeling efforts. [16]

While the anomalies of the data can be caused by different reasons, there are many ways to detect the anomalies from different stages of the data streams. Detecting irregular network traffic patterns, for example, from IoT data streams the Artificial Intelligence (AI) and especially Machine learning (ML) have proved to have the capacity to work with those kinds of challenges. Neural Networks (NN) aim to classify abnormal occurrences by detecting the patterns of the data. [39]

Ariyaluran et al. present a framework for real-time anomaly detection. This framework uses big data technologies and it is a specifically designed framework which uses several machine learning algorithms. The main aim for the research is to have a good accuracy for anomaly detection but also to concentrate on time spent and resources taken when running the algorithms. [18] The research showed good results but more studies on the field of using big data technology with anomaly detection was not to be be found.

Conventional machine learning methods for anomaly detection can be divided to statistical-based, probability-based, similarity-based, and prediction-based methods. Statistical methods compare historical data to current data and by using standard deviation these methods are detecting anomalies. Even though these methods are simple to use, the accuracy of the methods is quite low. Probability-based methods like Hidden Markov models and Bayes nets assume that normal observations abate to a specific probability frequency distribution. In anomaly detection, they use probability as the metric. These methods are complex and time-consuming. [52]

3.2 Limitations

Computational and energy consumption are things that are always present when talking about IoT systems. It has a huge impact on the performance, at what architectural layer the computational engine is running [34]. Some methods have shown that detecting anomalies from industrial sensors on cloud computing centers has casually increased the pressure of transmission in bandwidth and the load of computing on cloud center. [51] As a comparison there are methods on anomaly detection on edge nodes.

Anomalies can be detected by monitoring the IoT network as well. One of these kind of methods is to pick the number of packets from network traffic, and also sources of IP addresses per minute. These are used as detection indicators. These kinds of algorithms are quite simple to use but because they rely on threshold they are not stable enough. [24] IoT devices generate large amount of data, and transmission of the data is done in small packets. Hence network traffic monitoring and analyzing (NTMA) is a necessity. Dymora et al. [12] are presenting that with Hursts exponent and by analyzing the network traffic, it is possible to detect attacks or other anomalies from IoT data network. To be able to use the network traffic anomaly detection efficiently, it is vital to find common patterns for the devices in given category. [12] This method gives good results when detecting anomalies. However, following of the network traffic has been ruled out from this Master's thesis since the amount of the actual data in the real-life case fluctuates so much that observing the patterns is not an option for anomaly detection in this case.

It is known that when training AI models, there is often a lack of access to relevant data. Other issues related to AI approaches are their cost, time consumption, and error-proneness. The process is time consuming, because training AI models requires vast amount of labeled data, and preparation and evaluations of the data. [26] Furthermore, especially real-time analysis suffers from the limitations of computational performance. [18] Regarding several algorithms used in anomaly detection, the computational resources have not been disclosed in the studies or it has been taking too long to run the algorithms.

3.3 Process when selecting the method

When deciding which technique to use in IoT anomaly detection, the whole process needs to be designed properly. There are several things to consider and one of the main things is to decide where the anomaly detection engine should be running. [34] Computational resources are the main focus when selecting the technique. Resources will affect not only to the design of the anomaly detection engine but also which machine learning algorithm to be used. [53]

When developing Anomaly Detection Engine (ADE) it is important to understand and identify the architectural layer, on which to run the engine that detects IoT data stream anomalies. Elements that need to be considered when making the decision are computational resources available, and the need for cleaning or filtering the data before analyzing it. [34] Mohamudally et al. [34] are presenting an example of ADE framework within their article [34]. The suggestion is as follows: first raw data will be presented in time series, then time series will go through transformation. After these steps the modeling of the time series follows. Evaluation of the data is done next and this will lead to labeling of the data. Finally, an alert will be generated if anomalies are detected.[34] The picture of the presented framework is in figure 3.1. This framework has been one of the structures that have been used when creating the artifact of this Master's thesis.

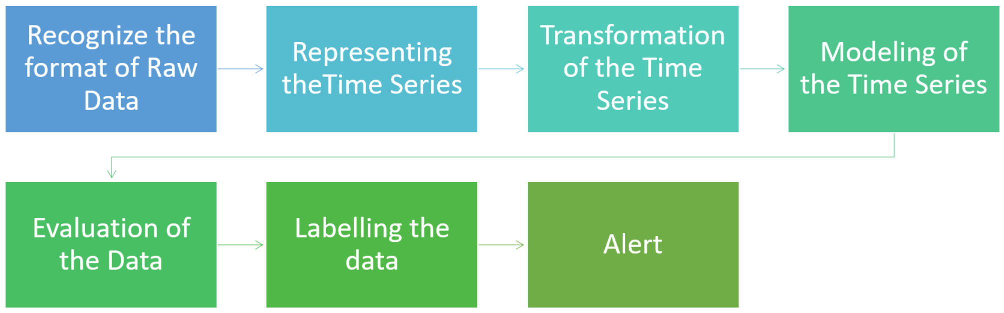

Recognize the format of Raw Data
Representing theTime Series
Transformation of the Time Series
Modeling of the Time Series
Evaluation of the Data
Labelling the data
Alert

Figure 3.1: ADE framework by Mohamudally et al.[34]

There are two critical factors to be considered before deciding which machine learning technique would be the most suitable: estimation accuracy and computational efficiency. Training and construction process will need computational power. Exception for this is kNN technique which is the most computationally efficient. [53] The research by DeMedeiros et al. [11] shows that most of the algorithms used in machine learning and deep learning when detecting anomalies are giving the same kind of results for accuracy. Although there are some minor differences, the results are approximately the same for all the algorithms. What makes it a bit hard to interpret the results is that different studies use different datasets when analyzing the results. [11] More computational power is also needed if there is a wish to replace the missing values. The higher the accuracy of imputing missing values is, the more computational power is needed [53].

In figure 3.2 the headlines about the machine learning as a process are represented. This is another framework that has been used when creating the artifact. Figure 3.2 shows the general tasks that are needed to be done when utilizing machine learning algorithm. Within this figure, labeling and training are included, so this is a general view of a supervised machine learning method where the data is known. With unsupervised data the data is unknown and there is no labeling. The preprocessing of the data and evaluation of the model are mandatory with both supervised and unsupervised methods.

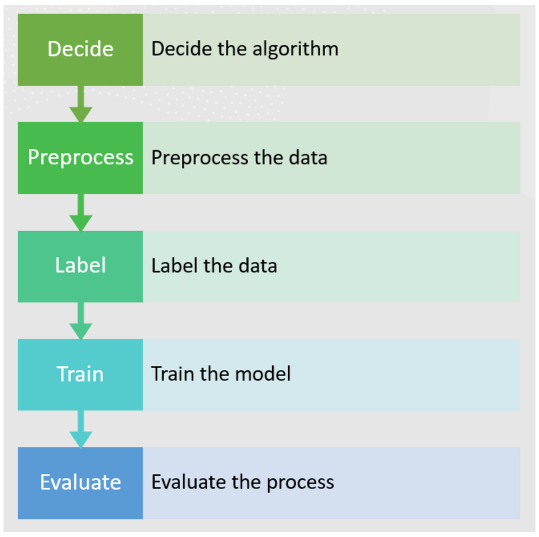
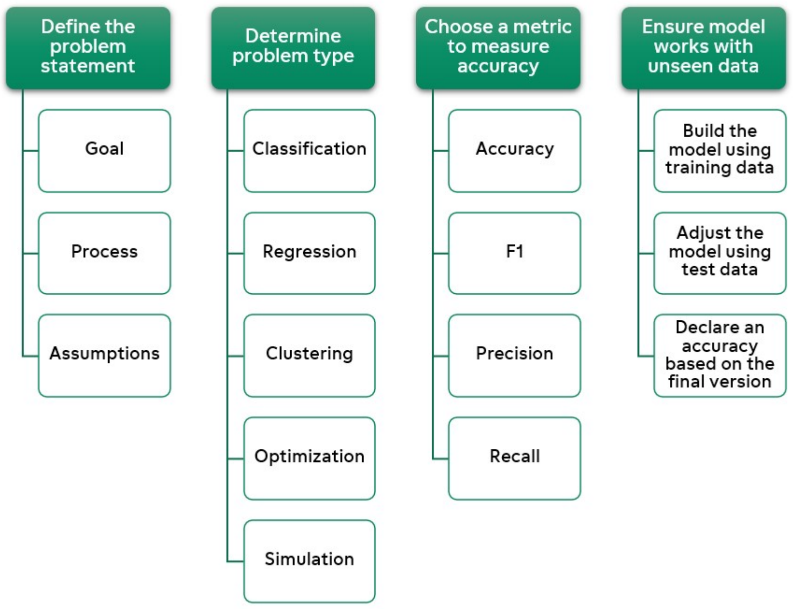


Figure 3.2: Actions that need to be done with ML algorithm.

In order to determine the algorithms to be used, the whole problem needs to be defined in depth. Figure 3.3 shows the decisions that are mandatory in order to select the correct algorithm to be used. Figure 3.3 outlines the problem statement that consists of goal, process and assumptions. After these the problem type needs to be understood. Problem types are classification, regression, clustering, optimization and simulation. Once the problem type is clear, the next thing is to choose a metric for measuring the results. The last step about ensuring that the model works with unseen data is emphasized on supervised learning since there are training and testing phases included. Framework presented in figure 3.3 was also used to create the artifact.



Define the problem statement
Goal
Process
Assumptions
Determine problem type
Classification
Regression
Clustering
Optimization
Simulation
Choose a metric to measure accuracy
Accuracy
F1
Precision
Recall
Ensure model works with unseen data
Build the model using training data
Adjust the model using test data
Declare an accuracy based on the final version

Figure 3.3: Process to define method.

3.4 Statistical methods

Statistical learning as a field of predicting data is older than machine learning. Machine learning methods are also often implementing statistical techniques. [14] Statistics in anomaly detection is used generally to define the acceptable fluctuation range. The problem with statistical approach is that with statistical models there are assumptions from the given data, and the results are highly reliant on the validity of these assumptions. [54] Statistical approach needs data to be collected, cleansed, and managed properly before implementing it. Statistical learning aims to have valid conclusions compared to machine learning which is about predictions. [14] The statistical methods for anomaly detection in this Master's thesis are presented in table 3.1. The table 3.1 shows that there are limitations when using statistical methods to detect anomalies from IoT data streams. Most often computational costs are high or the data needs to be certain type or manipulated in specific ways. The statistical methods presented here are used together with machine learning algorithms.

One statistical method to detect anomalies from time series is regression method. Regression method is used when there is normal data available for training. Normal

Table 3.1: Statistical methods for AD.

Method	**Main focus**	**Disadvantages of the method**
Regression	Imputing missing values	Requires the data to be linear
Autoregression models	Finding highly abnormal values	High computational needs
Expected maximization (EM)	Finding missing values	High computational needs
Control charts	Finding anomalies	Needs a presumption of probability distributions of the data
Spectral Analysis	Identifies trends and seasonal elements from time series	High computational needs, and needs a presumption of probability distributions of the data

data is compared to collected data, and the anomaly score is the difference between these values. The threshold is set to define if the point is normal or abnormal. [54] Logistic regression depends on the quality of the dataset. Regression is an example of a discriminative model. [19]

Autoregressive models make predictions of the system's future situation by using historical values. This can be utilized with anomaly detection if the deviation of the actual value has been set up so that the abnormal value can be recognized based on the predicted value. Anomaly detection for univariate time series are usually using autoregression (AR) and moving average (MA).Variations from these are autoregressive moving average (ARMA), and autoregressive integrated moving average (ARIMA). Multivariate time series are using the vector autoregression (VAR) model. This model is presenting linearly dependent values of itself and other variables, and besides these there is a random error term that calculates the elements that are not explicable by the past values. These require lot of computational resources with huge datasets. [7]

For estimation, Expected maximization (EM) algorithm is one method that has been used. With EM there are multiple models that can be used but the most used is Gaussian distribution. Gaussian distribution data points are presented as an assembly of Gaussian models and the aim for the models is to calculate probabilities. This has problems which are related to computational resources. What it basically means when detecting missing values is that this will require several steps to fit this with dimensional data. [17]

One of the statistical process control (SPC) tools are control charts. With control charts in time series data, the mean and variance shift are monitored. These can detect anomalies if the fluctuation of the system's data is noisy, or caused by an external aspect. The latter one is abnormal and hence it is important to detect those variations. Most used control charts with anomaly detection for univariate time series data are the Shewchart X-chart, the cumulative sum (CUSUM), and the exponentially weighted moving average (EWMA). With multivariate time series data, the most used control charts are Hotelling T2 control chart, multivariate cumulative sum (MCUSUM), and multivariate EWMA (MEWMA). The problem with control charts is that they depend upon a presumption of probability distributions of the data. [7]

Spectral Analysis is a statistical technique that combines attributes, and approximates the data. This technique requires the data be in low-dimensional shape. Most used methods in this technique are principal component analysis (PCA) and singular spectrum analysis (SSA). PCA is being used to decrease dimensionality of the data, and this way enhance the efficiency in storage usage and computational resources. SSA identifies trends, and seasonality elements from the time series. It has the capability to separate noise from the actual components of the time series data. For multivariate time series there is multidimensional singular spectrum analysis (MSSA) that is being used. Again, SSA is computationally expensive and PCA requires assumptions of specific distributions. [7]

3.5 Artificial Intelligence

In Artificial Intelligence (AI) data are generally used to train models, make the decisions, and perform tasks such as classification. Machine learning, deep learning and artificial neural networks are branches of Artificial Intelligence (AI). Machine learning is about building models that learn and make independent predictions based on the provided data. Artificial neural networks (ANNs) mimic the human brain which include perceptrons or "neurons" to transmit and process the data. ANNs are part of machine learning. [26] The models in ANNs mimic the structure of biological neural networks and these belong to the class of pattern matching techniques. [14] Deep learning provides good approach for processing and analyzing large and complex datasets because it uses multiple layers of interconnected neurons. [26] Figure 3.4 shows how AI, machine learning, deep learning and ANNs are related to each other. AI is the main branch for all the terms and machine learning, deep learning and ANNs are smaller ensembles that are on top of each other.

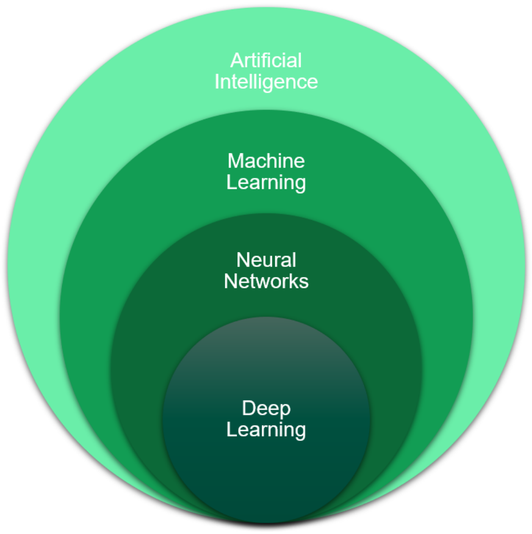


Figure 3.4: Artificial Intelligence.

Machine learning is a popular method for detecting anomalies. Machine learning is usually divided into three branches: supervised, unsupervised learning and semi-supervised. [5] This categorization is done based on the knowledge of the domain of the data. Labeled data is used with supervised learning, unlabeled data with unsupervised learning and semi-supervised learning assumes that the data contains both labeled and unlabeled data. [49] Learning from data is widely available in the IoT environment but class labels are rarely available. This leads to a challenge to create a model that is reliable. [4] Deep learning is one field in machine learning. Deep learning is a technique to build more convoluted neural networks with semi-supervised learning and hence it is operating with datasets that have small amounts of labeled data. [14]

The tasks for machine learning algorithms in anomaly detection are detecting, identifying and classifying anomalies [4]. In machine learning there are three datasets from the given data that are isolated into training, validation, and testing sets. For example, in training dataset classifier is built based on the data samples, in validation dataset data samples are verified against the classifier, and in testing dataset data samples help estimate the performance of the classifier. [14]

Alghanmi et al. [4] have presented that machine learning models are using four different key components: dataset, the learning algorithm, feature selection and evaluation techniques (figure 3.5). Dataset gathers the data from IoT devices. The data is structured in rows and columns where rows are observations, and columns are attributes. The learning algorithm is categorized for supervised or unsupervised data. Selecting subset of attributes or features to build a model is called feature selection. Evaluation techniques are utilized for validating how well the chosen machine learning algorithm performs.[4]

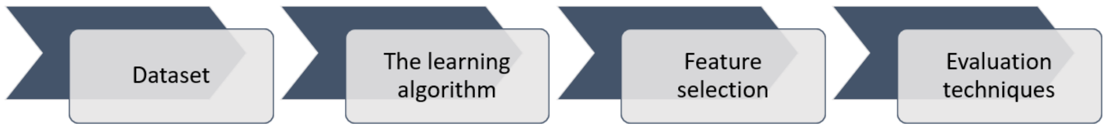


Figure 3.5: Key components for machine learning models.

3.5.1 Labeled and unlabeled data

Machine learning techniques can be categorized to supervised, unsupervised and semi-supervised learning based on the data labels and if they are available. [49] All of these methods have their benefits and limitations. Data can be labeled or unlabeled. Labeling the data means that, for example, a photo needs to have details of what it consist of, like animal or forest. Most of the time human interactions are needed when labeling the data. Usually labeled data is expensive and limited. Supervised learning is operating with known expectations hence it needs labeled datasets. On the contrary, unsupervised learning algorithms use unlabeled data. Unlabeled data consists, for example, from sensor data and there is no explanation for the data itself. Usually, it is in the format of raw data. Unsupervised learning uses clustering thus there is no specific target in mind at the beginning. [14] Semi-supervised learning is something between supervised and unsupervised learning. It can be used when the data can be divided into both labeled and unlabeled data. Semi-supervised data has achieved great success providing the answer for model training using partially labeled data. [37]

Semi-supervised learning has proven to be advantageous for anomaly detection in the research by Islam et al. [23]. However, Guansong et al. [15] present that semi-supervised learning requires both labeled and unlabeled data and if there are not enough both, the results are poor. [15] Semi-supervised learning is presented briefly in this section since it has proven to be efficient. But since the original request regarding this work was to have an automated anomaly detection for IoT data streams that has the minimum maintenance, the semi-supervised learning is not an option in this real-life case. Features for unsupervised and supervised learning are presented in table 3.2.

Table 3.2: Supervised and unsupervised learning.

	Supervised	**Unsupervised**
Data labels needed	x	
Preprocessing needed	x	
Can handle raw data		x
Small amount of data is enough		x
Expensive to label the data	x	
High accuracy	x	
Detects known anomalies	x	
Detects unknown anomalies		x

With supervised learning there is an assumption that both labels of data are available: normal and abnormal. The problem with supervised data is that it requires sufficient number of both these labels. Supervised method is not effective with unknown data. Anomalies need to be known before labeling hence unknown anomalies cannot be detected with supervised learning approach. [49] Supervised learning has higher classification accuracy, but this requires more resources when the training data needs to be manually labeled. [45] With supervised learning there are two steps: training and testing. Training means that a model is built based on the labeled data. Testing predicts the class labels which can be anomaly or normal and testing needs to be done for different data than in training set.[4]

With supervised learning method it is important to understand that the data needs to be labeled. Also, with time series data it must be kept in mind that since the data is constantly changing, there is a need to relabel the data from time to time. Actions for supervised learning are presented in figure 3.6. It is highlighted here that human interactions are needed when using supervised algorithm as a method.

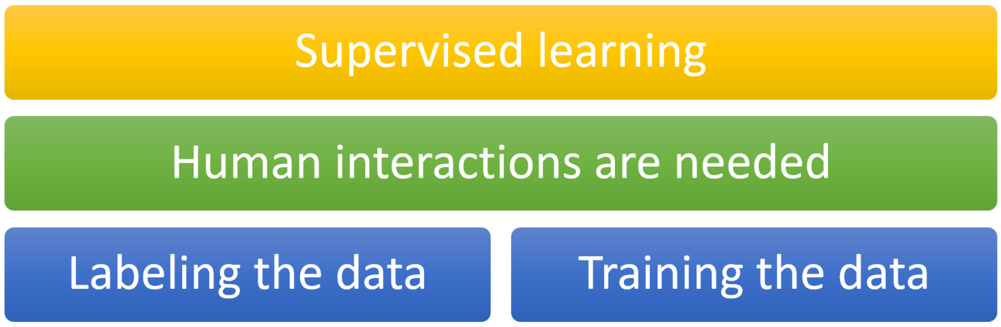


Figure 3.6: Actions for supervised learning.[34]

Unsupervised datasets contain mostly normal data [5]. Unsupervised learning does not need labeled training data. This method relies on the fact that abnormal data is more rare than normal data. [49] Unsupervised learning approach is preferable for anomaly detection compared to semi-supervised or supervised learning approaches for two main reasons. First, training data with supervised and semi-supervised data is usually imbalanced which leads to problems with classification of the data. Second, labeling is often done manually by humans which is expensive and may not always be accurate. [16] Unsupervised learning operates with unlabeled data and searches similar data to group them [4]. Also, for unsupervised data, there is not a need to separately train and test the data. [18]

With unsupervised learning algorithms the data is unknown. The unsupervised algorithms find the patterns for the data and labeling is not needed with unsupervised algorithms. It is widely used approach to detect anomalies with unsupervised learning by using clustering. In clustering the data is grouped based on how similar the data is compared to its neighbours. Actions in unsupervised learning are presented in figure 3.7.

Figure 3.7: Actions for unsupervised learning.[34]

Semi-supervised learning methods offer the possibility to detect anomalies when only normal data is available. These methods compare the similarity between the given data and normal data. [49] When anomalies can be labeled, supervised learning methods are usually better than semi-supervised learning methods. Semi-supervised learning has been used in cases when there are only normal samples available. [49] Semi-supervised anomaly detection needs labeled and normal data for training. This may lead to poor results if any incorrect labeling has been done to the data. Poorly labeled data can be a bigger issue when it comes to noisy instances because the methods can interpret noisy data as an anomaly. [15]

Semi-supervised and unsupervised learning do not need labeled data for training and hence they have gained popularity amongst anomaly detection. The limitation though is that supervised learning detects known anomalies better than semi-supervised and unsupervised methods. [49] Unsupervised algorithms are popular when there is only a little number of samples of the data. Also, with unsupervised algorithms the structure of the data can be unknown, which is the case in several real-life scenarios. [8] In anomaly detection unsupervised learning effectively identifies the clusters of anomalies. Supervised learning is recognized to be more efficient compared to unsupervised learning when the normal and abnormal data samples are similar. [50]

Based on the research, semi-supervised learning algorithms have a few weaknesses compared to supervised and unsupervised learning algorithms. As described previously, using semi-supervised learning method may lead to poor results if there is any incorrect labeling done. On the other hand, there are more studies about using supervised or unsupervised learning methods with anomaly detection than there are about using semi-supervised learning method [15]. Hence it has been ruled out within this Master's thesis to use semi-supervised learning algorithms when detecting anomalies from IoT data streams.

Supervised learning methods for anomaly detection are often impractical, because they assume that there are both normal and abnormal data available as a training data. Unsupervised methods do not have that kind of knowledge of the data. Unsupervised methods rely on the assumption that there will be anomalies to a certain extent. It does not need labeled data. However, it is recommended to use as much labeled data as possible. [15] Decision Tree (DT), Naïve Bayes (NB), and K-Nearest Neighbours (KNNs) are examples of the supervised learning algorithms[4], while examples of unsupervised learning include K-means clustering, Isolation Forest and Density-based spatial clustering of applications with noise (DBSCAN) [34] [5].

3.5.2 Learning problems

Learning problems can be divided into five different categories and these categories are presented in table 3.3. The categories are classification, clustering, regression, optimization, and simulation. [14] IoT anomaly detection is often treated like static data classifications problems.[24] It is common to consider anomaly detection as a classification task since anomalies are defined as instances that deviate from the normal patterns. However, there are not always labels to be used with machine learning algorithms. Fortunately there are ways to detect anomalies without knowing the data and data being unlabeled. Islam et al. [23] are describing that frame reconstruction and clustering methods, and methods that predict future, are effective in anomaly detection when training sets includes unlabeled data. [23]

Classification helps to identify the data behavior patterns by grouping the data. Clustering is also grouping the data but compared to classification, clustering does not have a specific target in mind at the beginning. Regression is needed with forecasting and prediction when there is not enough historical data to be used for predictions. With regression a graph is plotted between the parameters so that equation fits most of the data points. Whenever the data points do not fit the curve, they are removed. Simulation can be used when the data has many uncertainties. Optimization has more information about the data compared to simulation; with optimization there is access to data, its dependencies, and relationships between data attributes. Optimization is trying to make something better. [14] Optimiza-

Table 3.3: Learning problems in machine learning.

Category	**Used for**
Classification	Identifying the data patterns by grouping
Clustering	Grouping the data without any specific goal
Regression	Forecasting when not enough historical data
Simulation	Used when the data has many uncertainties
Optimization	Aims to make something better

tion methods are often not used alone but instead for optimizing machine learning algorithms when detecting missing values. [17]

When detecting the anomalies, classification and clustering are the most used approaches [34]. Classification is normally used with supervised data but there are approaches that can be used with unsupervised data as well. Classification problems can be divided to binary classification and multi-class classification problems [44]. A common approach with unsupervised learning is to use one-class classifier with normal data. [16] Most of the studies state that the accuracy often suffers when multi-class classification is used for anomaly detection. For example multi-class classification is not ideal on edge sensors because of the computational complexity [44]. Also,it is time consuming and requires resources not only from the system but also human interaction is needed [48]. One-Class Support Vector Machine (SVM) can be used when there is only one specific class and other classes are not available. But it has been stated that usually this is more difficult approach than ordinary classification problem. [45]

Clustering has two clear benefits: it groups the patterns of the data, and it also compresses the data. [18] In clustering, it is usually quite important that the learning representations are such that anomalies clearly deviate from the normal values. [15] Like in classification methods, there are several cluster-based methods and some of those methods can work even if there is no clear vision what kind of anomalies are involved. Dynamic data is causing more actions since clustering can be done only once if the data is static but in case the data is dynamic, clustering should be done periodically [47].

3.5.3 Deep learning

Deep learning is one of the approaches for automated anomaly detection [52], and it has gained success in the field of finding anomalies [15]. Deep learning has capabilities in learning representations of not only from complex or graphical data, but also from temporal and spatial data, which are the characters presented in time series data. In anomaly detection, deep learning uses neural networks to learn feature depictions. The methods have been shown to have a better performance than the conventional methods in real-life applications. [15] Despite the popularity, there still are several unsolved issues when using deep learning methods for anomaly detection [15]. Deep learning gives a good performance but with deep learning techniques there is a latency that comes from the training phase because deep learning requires more data for training. [42] The benefits and limitations of deep learning methods with anomaly detection are presented in table 3.4.

Benefits	Limitations
Learns expressive representations of complex data.	There is latency because of training phase.
Better performance compared to conventional methods.	Tailored for specific purposes.
Different methods give roughly equal results.	Detects mostly point anomalies.
Can be trained with raw and unlabeled data.	Lack of research for collective anomalies.
	Training time varies based on the structure.
	Time consuming.
	Requires lot of resources.

Table 3.4: Benefits and limitations of deep learning.

Many deep learning methods are tailored for specific situations and that is why there is not a solution that would fit to all cases. [7] In addition to the above mentioned latency and the need of more data for training, most of the deep learning methods are created to detect point anomalies. Methods used for point anomaly detection cannot be used with other anomaly types. [15] It has been recognized that one of the areas that lacks research is collective anomalies. Collective anomalies still lack proper way for detection. Deep neural networks might be the solution for that. [7] Generally, the different deep learning methods give roughly equal results [49].

One of the most popular machine learning methods is Artificial Neural Network (ANN) [2]. ANN is a machine learning technique that can be trained with raw data. It is a complex structure because it has large number of parameters. [19] ANN is a complicated model that includes several steps on different layers based on which type of ANNs are used: for example layered networks, recurrent networks, and convolutional networks [14]. There are also solutions that use hybrid models where ANN is part of those methods. Generative Adversarial Networks (GANs) is one those approaches. GANs use unsupervised data together with artificial neural networks. Time Series Anomaly Detection with Generative Adversarial Networks (TAnoGAN) is a variation from GAN and is being used with time series anomaly detection together with LSTMs. LSTM is an abbreviation of Long Short Term Memory, and it is an extension of Recurrent Neural Network (RNN). RNN has been showed to be efficient when it comes to predicting sequential and time series data. In deep learning training time depends for example on the architecture of the network, channels and batch sizes that are used and other hyper-parameters. [17][7]

Time series data puts up more challenges for the anomaly detection. Firstly, the real-time analytics requires resources on computational and on time levels. Secondly, the dimensionality needs to be considered if the dimensionality should be reduced. The real-time anomaly detection is very important in many cases. Real-time anomaly detection is a challenge since it requires evolving data streams and the cost for related computational is crucial [7]. Deep learning and deep neural networks (DNN) are effective with anomaly detection especially with high-dimensional time series. With DNN there is an automatic learning capability which means that no labeling is needed. This is one example of unsupervised algorithms. Convolutional neural networks (CNN) are mostly used with images and processing multidimensional time series. CNN has been shown to have higher performance, and more effective training by extracting patterns from the time series data. Temporal Convolutional Networks (TCNs) are variants of CNNs. TCNs are competent to handle data that has sequences. [7]

If the dimensionality should be decreased, it can be achieved by auto encoders

(AE) that are neural networks designed for projecting high-dimensional data into lower-dimensional space [45]. Autoencoders are popular deep learning methods for anomaly detection. An AE consists of an encoder and a decoder. It is a neural network that copies an input to its output in the network. The decoder tries to check the representation, if it is similar to input state. Different variations of AE have been introduced, aiming at improving the results AE. [49]

It has been shown that deep learning provides solutions for anomaly detection. For example, deep learning algorithms provide excellent results when detecting anomalies from images, speech signals, and natural language processing. The methods giving great performance are convolutional neural network (CNN), and recurrent neural network (RNN). [52] However, issues remain when using deep learning for anomaly detection. They are computationally expensive and might require continuous learning. CNN, for example, requires continuous learning to perform efficient anomaly detection [1]. Generative Adversarial Networks (GANs), on the other hand, are computationally expensive since they require non-linear optimization during the search of a latent sample [8].

3.5.4 Most used machine learning algorithms

There is a group of most used algorithms in machine learning approach. Weinger et al. [50] are enumerating machine learning algorithms that are known to be used in IoT anomaly detection: deep autoencoders, clustering, logistic regression, support vector machines, decision trees, and random forests. [50] In addition to these methods, it is very common to use hybrid algorithms in deep learning when detecting anomalies [7]. As mentioned before, when detecting anomalies there are several steps to apply before the anomalies can be found. Data preprocessing is one of those steps and that is why in anomaly detection there can be used several algorithms instead of just one. In this subsection there are presented the methods that are most used. ANNs and deep learning are presented in previous subsection.

The most used machine learning methods can be categorized based on whether there is labeled or unlabeled data available. Mohamudally et al. [34] have listed the most used methods based on this. Supervised model is used with the following methods, for example: Artificial Neural Networks (ANN), Decision Tree, Random Forest, K-Nearest Neighbour, Support Vector Machine (SVM), Deep Learning, and Naïve Bayes. The popular algorithms that are being used with unsupervised learning are: K-means clustering and DBSCAN [34]. Isolation Forest is also a good unsupervised approach for IoT anomaly detection [5]. Belay et al. [7] have highlighted in their study that even though some of the algorithms work better than the others, the accuracy is on the same level with all of the most commonly used algorithms. [7] The most used algorithms in IoT anomaly detection are presented in figure 3.8.

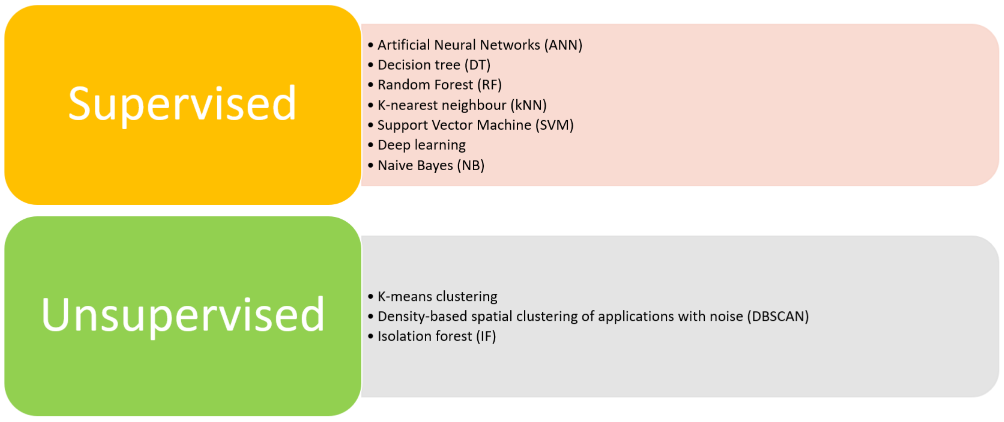

Supervised
• Artificial Neural Networks (ANN)
• Decision tree (DT)
• Random Forest (RF)
• K-nearest neighbour (kNN)
• Support Vector Machine (SVM)
• Deep learning
• Naive Bayes (NB)
Unsupervised
• K-means clustering
• Density-based spatial clustering of applications with noise (DBSCAN)
• Isolation forest (IF)

Figure 3.8: Unsupervised and supervised learning algorithms.

Decision Tree shows the possible outcomes of choices that are related to each other. It weighs benefits, costs, and probabilities and their actions against each node. Random forest is a supervised classification algorithm, and it builds the forest from many decision trees. Random forest is known from the high speed of executions. It has better performance than a single decision tree. [19] Tree-based methods are highly efficient with Gaussian data, and data that has high-dimensionality. In this category, the Isolation Forest (IF) and its variants are mostly used. IF processes random sub-sample data in a binary isolation tree. By selecting randomly the features, it can be recognized that samples far away from the original branch are probably normal, when shorter branches denote abnormalities. IF and Random Forest have many similarities. IF is computationally efficient and it is easy to implement.[7] Isolation Forest can be used to isolate anomalous objects instead of profiling normal objects. Unlike Decision Tree and Random Forest, Isolation Forest is an unsupervised learning method. [45]

K-Nearest Neighbor (kNN) is a classification algorithm. It is a good method for finding missing values particularly if dimensionality in the dataset is high and the amount of observations is low.[17] KNN is a distance-based approach. This method is widely explored but it has extensive computation. Also, the data needs to be brought to low-dimensional space before implementing.[15] KNN Is popular with statistic as well. The basic idea of kNN is to compute the distance between test and training points and show the points that are similar. With the help of regression, the average of the similar data points can be given as the desired value. [26] The problem with kNN is that training needs large amount of data. [41]

Support vector machine (SVM) is commonly used algorithm for classification of the data. Weakness of the SVM is high complexity in the computation aspect. [46] SVM is an example of discriminative model in machine learning. It is a supervised learning model, and it can be used for classification, regression, and detection of outliers. SVM is good for non-linear data. [19] SVM is known as one-class classification-based measure, and it is one of the most popular anomaly detection methods [15]. One-class SVM is an example of prediction-based method and the predictions are based on regression. These methods seem to be unsuitable for unpredictable time series. But instead, time series with periodicity or seasonality can use these methods for anomaly detection. [52]

Naïve Bayes is used as a classifier. It requires earlier knowledge of the data. Naïve Bayes uses the samples of the data to calculate the probability. [29] Naïve Bayes is widely used when text needs to be analyzed or filtered for spams, and used for sentiment analysis. As a classifier it assumes that features are independent. Data is categorized based on this presumption. [26] Naïve Bayes is based on Bayes theorem with the naïve assumption. Bayesian learning is a supervised learning method and the goal for the method is to build a model of the class labels. In Bayesian learning there is a need to have concrete definition of the target attribute. The thing that makes this approach naïve, is that Naïve Bayes assumes that one value does influence to another value. [14]

One of the most popular unsupervised learning approaches is k-means clustering. It is easy to use and the implementation is uncomplicated. [14] K-means algorithm is widely used clustering method. The benefits in using K-means are the simplicity and adaptability. K-means is also initializing other clustering algorithms. DBSCAN and the widened versions of DBSCAN, and spectral clustering are examples of algorithms that are initialized by K-means. Problems with K-means are the time and storage usage which can be quite high. [18] Methods that are based on similarity, rely on computing the similarity and distance between normal and anomalous data. In anomaly detection, K-means utilizes distance between unlabeled and labeled data. It is suitable for low-dimension data and quite simple to execute. [52]

Clustering algorithms with anomaly detection take more time and need more memory. The advantage is that they can use unlabeled data. One example of unsupervised machine learning is clustering. Clustering aims to group data points so that similar points are gathered while minimizing the similarities between the different clusters. DBSCAN is one unsupervised clustering algorithm. This requires only little domain expertise, and DBSCAN and its variants are generally used unsupervised learning algorithms for anomaly detection. [7] Alghamni et al. [4] have highlighted the results of DBSCAN in their overview of anomaly detection where DBSCAN proved to be able to avoid successfully false positives, and also DBSCAN was able to identify anomalies. [4] The problem with DBSCAN is that it requires relatively large amount of data [7].

3.6 Finding and imputing missing values

Based on literature, different methods can detect abnormal values from IoT data streams. Missing values are considered to be abnormal values in IoT data streams [46]. The main problem with missing values is that they can deform the results of the data and hence diminish the data quality. To reduce the impact of the missing values there are two main approaches that have been used: statistical and machine learning techniques. In the recent years, machine learning has been widely used to impute missing values. [53] Especially in fragile environments, it is crucial to detect the missing values and impute those to maintain the quality of the data. [2]

One way to impute missing values is to use regression with machine learning models. For example, the following techniques are famous: K-Nearest Neighbor (kNN), Artificial Neural Network (ANN), Support Vector Machine (SVM), and Random Forest (RF). [53] The basic approach of these methods are presented in sections ”Deep learning” and ”Most used machine learning algorithms”. Other popular algorithms for detecting missing values from IoT data are Gaussian Mixture model, spatial and temporal correlation, and Probabilistic Matrix Factorization [46].

Gaussian Mixture model (GMM) is considered to be a clustering algorithm, but it has other steps as well when imputing missing values: after clustering there will be classification, and after classification the distance is measured, and finally the data is refilled. When the closest occurrence in the cluster is found, the missing value can be replaced with the mean value. When finding missing values, spatial and temporal correlation has more steps. By estimating the time series it selects the series that is most important. By weighing different steps and computing them, it finally fills the missing value by using the value of the sensors nearby equivalent to the time. Probabilistic Matrix Factorization starts by using the K-means algorithm for clustering the data. The K-means algorithm compares the similarity of the sensors and groups them into distinct classes based on this. The missing data is replaced with the value based on analyzing the patterns of sensors that are neighbors. [46]

Data re-transmission is one of the methods that can be utilized when dealing with missing data. This approach transmits the data until the transmission is successful. The problem with this kind of solution is that energy consumption will increase. It can also cause some latency within the data being transferred. More energy-efficient solution could be an approach, in which data is being recovered and sparse sampling is done. Data recovery can be done with the previously recorded data, and statistics is a simple method for this. The problem with statistical methods is that they cannot provide prediction and therefore the data recovery can be inaccurate. [47]

Different approaches for recovering data are spatial and temporal approaches. Difference between these two approaches is that temporal methods use recorded data from the same sensor, and spatial methods compare data from sensors that have a specified range. For interpolation the commonly used approach is linear regression model together with k-Nearest Neighbor. [47] Regression modeling is a statistical method that has been used when considering imputation of missing values. The other statistical methods used with missing values is Expected maximization (EM). [17] They both need historical data to be able to predict the given data by calculating predictions. Limitations are high computational needs and the large amount of data needed.

3.7 Specific methods for time series

Since IoT is widely spreading to everywhere, it has been recognized that the investigations on anomaly detection need to catch up with the increasing amounts of data. Some of the methods presented in previous sections have been widely used and the results are well-known with those methods. Because those methods have some limitations such as heavy computational usage or they are time consuming, there are studies that have been trying to tackle these known issues by testing new or hybrid methods. This section presents some of the new methods that are trying to make the anomaly detection from time series data even more efficient.

There are some popular time series models for anomaly detection in IoT. First one is Autoregressive model which require the data to be linear so that the predictions of the next data point can be done by investigating the previous data points. Second one is Symbolic TSA and this requires the data to be in bits and bytes. With bits and bytes it is possible to utilize Information Theory from Shannon. The third popular ADE solution for IoT is Seasonal Trend Loss (STL) Decomposition. In STL Decomposition data points are evaluated during one period with the noise or with multiple datasets. Anomalies can be detected by decomposing and analyzing data points.[34]

Xiang et al. [51] have presented the Edge Computing based Anomaly Detection Algorithm (ECADA). ECADA detects anomalies on time series which have one or more data sources. As the name of the method refers, this algorithm is running on edge instead of cloud. Xiang et al. [51] are also presenting that the importance of separating anomaly detection from cloud center and edge nodes should be recognized. [51] Anomaly detection running on edge has a lack of enough research and with this Master's thesis the architecture of the real-life case does not include the option that the anomaly detection algorithms could run on edge. For these reasons, it is only mentioned here that there are some possibilities to use edge for this purpose as well.

Unsupervised learning is a good approach for hubs, since they are not able to store large flows and must learn on the fly as much as possible. If anomaly detection is done on a cloud level, there are better options for detecting anomalies by using more computationally demanding approaches such as DNN. The detection can also be done on edge gateways which have the better possibilities for storing huge amounts of data that training requires. Cloud and edge computing can be used together since by combining these could lead to a better final result. [22]

In machine learning, there is a model created for each device based on its local data. Federated learning (FL) works as a coordinator which combines the models and aggregates the local models to be global models. Federated learning is known to have various performance problems. For example, if the data types are different between different data models, this may impact overall performance. [50]

Sivapalan et al. [44] have proposed that instead of using multi-class classification, abnormal values could be classified with binary-classification in order to decrease power consumption required by IoT anomaly detection. The method they suggest is made for ECG sensors that are sensing arrhythmia from heart beats. Previous methods in ECG anomaly detection have been using multi-classification by classifying the heart beats into several different states. This new approach would have benefits for power consumption since heart beats are classified only to normal or abnormal. [44] This could be implemented in the case where one value can contain several errors but since the error values are known (blanks, nulls, zeros), it could be considered if the binary classification could be enough for the anomaly detection.

DNN-based approaches for detecting anomalies from time series use regression to learn from past values to predict future values. It contains prediction error to resolve if the value being predicted is abnormal. Example of DNN-based algorithms is Recurrent Neural Networks (RNN). RNN are commonly used for time series. RNN do not have a good performance when modeling long-range temporal correlations. Long Short-Term Memory networks (LSTM) and Gated Recurrent Units (GRU) are also DNN-based, and they have been developed to avoid that kind of complexities. LSTM and GRU are being applied for anomaly detection in time series, because they have capabilities of prediction and time-dependent modeling. [7]

CNN and RNN are under investigations by combining these two together in order to have better classification and prediction in data that is time-dependent. RNN memory units are commonly using the long-short term memory (LSTM) and gate current unit (GRU). LSTM and GRU are helping to overcome memory loss for long term sequences. Tests by Yin et al. show that using CNN and RNN together is highly effective approach for discovering anomalies but the parameters and architecture for this model require a lot from hardware resources. This means that for IoT anomaly detection this model needs to have more trials. [52]

4 Design science as a research method

Design science aims to create valuable and innovative products to serve human purposes. Design science creates new features that can be implemented for business or industrial purposes. Products that will be completed as a result of design science are artifacts. These artifacts can be constructs, models, methods, and implementations. More specifically, the product can be a physical implementation intended to perform certain tasks. [30] [3]

Design science is technology oriented [30]. Design science has its roots in engineering, and it is fundamentally a paradigm that solves problems. The final product, which can be an idea, practice or some technical capability, is defined after thorough analysis, design, and implementation. [3] There are two basic components in design science: building and evaluating. In the build phase an artifact is created and in the evaluation phase the artifact will be reviewed to see how well it works. [30]

The artifact can offer not only problem-solving solutions for humans, but also some intellectual and computational tools for organizations. Once the existing problem in the organization has been identified, design science attempts to solve the problem by creating and evaluating the artifact. The design process is a continuous task that aims to develop both the design process and artifact by iterating the solution with building and evaluating. Feedback on the created product (artifact) is important since it provides valid information about the feasibility of the product. The usability of the product describes the quality of the product. [3]

When building and evaluating the artifact, it is crucial to identify the real business needs and base the construction on those. Business needs are defined by the goals, tasks, problems, and opportunities in the organization, and they are defined by the people. They can be evaluated by comparing them to existing business processes, and organizational strategies, culture and structure. In addition to these values there are other things that will define how the artifact should be created. These values include the capabilities of the development, existing applications and infrastructure of the technology, and communication architectures. [3]

The artifact can be a methodology or set of procedures, in which case it defines the processes in real-life scenarios and provides guidance on how to solve certain problems. For example, the methods implemented during design science iterations can be mathematical solutions, or they can be textual descriptions of best practices or a combination of mathematical and textual descriptions. Method as an artifact will help the target organization in problem solving, but also the researcher to learn about the real world. Learning from real world helps also to see how the artifact has an impact to it. Computational and mathematical approaches are mainly applied when evaluating the quality and validity of artifacts in design science. In addition to these empirical techniques may also be used. [3]

Problems with design science can occur if the dependencies of the artifact on the environment where it is implemented and utilized is not adequately understood. In some cases, the environment can cause side-effects that are unwanted. The side-effects should be evaluated and taken into consideration so that they can be avoided. [30] One of the known problems when using design science is that design research must be separated from routine designing. Routine designing can be the utilization of existing knowledge to organizational problems. Key aspects of design science are innovation and unique way of solving certain problems. This is why the identification of the problem needs to be defined thoroughly. [3] However, it must be remembered that design science will rely heavily on existing theories or knowledge.

Hevner et al. [3] have enumerated other known problems with design science. First problem is related to undefined environmental context which can lead to unstable requirements. Second known problem is about sub-components of the problem and its solutions, which can result in complex interactions between the components. Third issue that has been recognized is that the design processes are not flexible to be changed if needed. The last problem presented are the human errors that hamper the efficient use of the constructed product. The human errors in using the product can be related to cognitive or social abilities. [3]

There are seven main characteristics in design science described by Hevner et al. [3]: first one is that there needs to be a creation of an innovative and purposeful artifact. Second one is that this artifact needs to be for a specified problem domain. Third main feature is evaluation of the artifact during the process. Fourth aspect of the design science process is to differentiate traditional designing from design science where the main purpose is to create an innovative solution for the problem. Fifth aspect regards defining the artifact to be presented, defined, coherent and internally consistent. Sixth element is the requirement that the solution must be effective. Finally, the solution must be communicated properly to the audience. [3]

The main characteristics of design science are presented in table 4.1.

Table 4.1: Main characteristics in design science.

1	Create innovative and purposeful artifact.
2	Make sure the artifact is for specified problem domain.
3	Evaluate the artifact during the process.
4	Differentiate traditional designing from design science.
5	Artifact needs to be well presented, defined, coherent and internally consistent.
6	Artifact must be effective.
7	Artifact must be communicated to the audience properly.

4.1 Defining the artifact

Definition of the artifact is the most relevant thing when creating an innovative solution for real-life case. In this Master's thesis the definition was created together with the Fortum IT Data & Software Development team. The real need was to create a procedure that detects empty values from IoT data streams on real-time. Fortum IT Data & Software Development team had already discovered that there are some empty values within the IoT data streams since they had run SQL queries after the data had been transformed to data storage. The problem with the existing solution is that there is a latency for detecting the empty values since the data is analyzed after the data is transferred into data storage.

In addition to the real-time anomaly detection, the objective was to define a procedure that runs automatically and sends alerts when empty values have been discovered. Furthermore, it was requested by the target organization that the procedure should be able to be maintained with the smallest amount of human interactions and it should be scalable for other IoT data streams. Based on these definitions the goal for this Master's thesis is to find an answer for a question: "What kind of methods there are to detect anomalies from IoT data streams?"

Design science aims at creating not only an innovative but also a purposeful solution to be used in real-life. To create a purposeful procedure for detecting anomalies, it is important to define the existing problem and create the artifact based on that. The definition from Fortum IT Data & Software Development team was very precise and it can be seen that there are five main aspects in the definition:

1. Create procedure that detects anomalies automatically.
2. The procedure needs to discover anomalies in real-time.
3. The procedure includes suggestion what to do when anomalies are detected.
4. The final procedure is maintained with the smallest amount of human interaction.
5. The procedure is scalable for other IoT data streams.

The original request from Fortum IT Data & Software Development team was to have this procedure as a format of procedure diagram. During the creation process it became quite clear that when this solution needs to be scalable for other IoT data streams, there is not a single procedure diagram that can be followed to create an anomaly detection method that is scalable for any kinds of Iot data streams. Because anomaly detection methods vary so much based on the IoT architecture, the anomaly type and the type of the time series, there are lots of questions that need to be answered before using the procedure diagram. Hence the final artifact in this Master's thesis includes both a procedure diagram and a checklist that are used together in order to define all the needed components to solve the real-life problem, which is to detect anomalies on the fly from any IoT data streams.

4.2 Procedure diagram

A presentation in pure textual format can cause misunderstanding for readers. When it comes to instructions, studies have shown that the best practice is to integrate text and visual elements. Diagrams help users identify the required steps to take, and that is why they are an effective way to communicate. Flowchart is a well-established and widely used type of diagram. It has been shown that providing visual information leads to better understanding from users perspective.[36] Synonym for flowchart is a process flow diagram. [33]

Flowcharts are widely used in software development, engineering design, and scientific experimentation [38] [33]. Studies have shown that flowcharts are being used successfully in production and manufacturing processes. Flowchart is a common way to describe the logical processes in engineering and software processes. [38] Simpson and Genovese [33] have listed the main reasons to use process flow diagrams. Firstly, the diagrams help in understanding the process better and share knowledge with colleagues. Second reason for using process flowcharts is to help to understand the complexities of the work. Other reasons to use process flow diagrams are to make the work visible and assess alternative ways of working. [33]

One of the benefits from using the flowchart is that the diagram will help to solve the problem step by step. It also forces to pursue a strict way of the process without skipping any steps. The aim of a flowchart is to show the actions behind the basic theory. A flowchart shows the material that should be used in problem solving. [27] Flowcharts also help users to identify relevant and irrelevant content of the matter [36] [33].

Process flow diagram is an essential tool for analyzing systems, procedures, and processes. [33] There can be different symbols inside the process flow diagram. [33] In this thesis, different symbols and different colors for the symbols are used in order to understand better what the needed actions or alternatives are. The symbols and colors are explained in chapter five. One of the regularities in flow diagrams is that they have clear inflow and outflow relationship. One node in a flow diagram can have multiple inputs or multiple outputs. [38]

The original request from Fortum IT Data & Software Development team was to have a flowchart that would show step by step how the anomaly detection from IoT data streams should go as a process. It was decided that flowchart would be the best approach in this case. However, during the creation process, it was quite clear that flowchart alone is not enough to complete the process since there are important things to decide before going into further steps. Simple yes or no questions would not be enough in these cases. Hence, it was decided that there could be a pattern of questions before going using the flowchart. Furthermore, the flowchart as a structure was not suitable since there are no clear yes or no answers for the procedure. Process diagram has better possibilities to describe the flow of the process. Both the procedure diagram and pattern of questions (checklist) are presented in chapter five.

4.3 Checklist

In IT world there are not much research regarding the usage of checklists when implementing any new procedures of work. Checklists are widely used in healthcare and with flight attendants and aviators, though. In a flight environment the purpose of a checklist is to prevent errors happening in the first place. [20] There are many other benefits of using checklists. Evidence show that the regular and proper usage of the checklists helps to accomplish the goals. [21] It has also been shown that a checklist can serve as documentation for the project as well. [32]

Hernandez et al. [21] have investigated the usage of checklist with catheter-associated urinary tract infection. The findings from Hernandez et al. [21] were that when using a checklist there is a need to have education in both the usage of the checklist and also for the task that is being accomplished. Other findings suggest regular auditing of the checklist usage and proper documentation. [21] Middle [32] is also presenting the importance of auditing the checklist. Auditing could improve the usage of the checklist over time. [32]

The goal of creating a checklist by Clapper [10] was to improve the documentation, reduce errors, and increase patient safety. [10] When creating a checklist, it also seems to be crucial that the components of the checklist are carefully chosen and their order is well-thought. Selecting the components for the checklist is vital. This gives the headlines for the checklist and hence the guidance for the project. [32] Clapper [10] claims that it makes a difference in which order the steps form the checklist are presented. In health care the logical order of the steps is the chronological order. [10]

Before creating a checklist, it is vital to define what are the errors to be avoided. [20] Hendarko et al. [20] have introduced a checklist based on the failure analysis. [20] In this Masters thesis it is not possible to first get the failure analysis and then create the checklist. But in the future, this could be a good aspect when considering the auditing of the checklist. It could be determined whether these components of the checklist have been used or not and whether they have helped with the guidance when starting a new process of IoT anomaly detection. For the documentation purposes it is vital to write down the failed features so that these can be used when developing the checklist to a next step.

In the case of this work, the objective of using checklist is to help choosing the proper procedure to follow when detecting anomalies on the fly. The whole procedure of anomaly detection on the fly is a new thing to Fortum IT Data & Software

Development team. Part of the artifact is to create a checklist to help getting started with the procedure. The other aim with the checklist is to create a procedure that is scalable for other anomaly detection situations as well. Since anomaly detection is dependent on the IoT architecture, the type of anomalies, and the availability of resources, it is impossible to offer one solution that would fit to all situations. Hence the artifact of this Masters thesis is to create both procedure diagram and checklist to help when deciding which kind of approach is the most suitable for diverse IoT anomaly detection system.

4.4 Iterations when creating the artifact

During the creation of the artifact there has been joint sessions with the Fortum IT Data & Software Development team members Petri Lehtonen and Rajeev Shunmugam Shyla. The original request was to have procedure diagram as a procedure to detect empty values from IoT data streams on the fly. Team had their own idea about the platforms that they want to be using. There are some limitations about the platforms because of the existing solutions. This is the reason why this Master's thesis will not be presenting any platform related solutions but instead a procedure that could be implemented despite what platforms are in use.

There was altogether five iterations during the time when creating the artifact. Before the iterations there were three meetings where the definition of the artifact was defined. Mostly the discussions were via Teams and all of the cases I sent the meeting invite with the material and questions that needed decisions to be made to go further with the process.

Meeting invites for Teams meeting were sent a week or two weeks before the actual meeting. The reserved time for the meetings varied from 30 minutes to one hour. Some of the 30 minutes meetings were arranged because there was not better suitable time to go through the next steps. Although after one meeting that had only 30 minutes time reserved, it was obvious that this is not enough to ask questions on both sides. Therefore it was agreed that the meetings should have time reserved for at least 45 minutes. The iterations were kept fortnightly and within the two weeks there was enough time to do changes for the artifact and create new questions for the next iteration.

Every time with the meeting invite, there was agenda created by me. When the Master's thesis started to progress, a copy of the Master's thesis was also sent as an attachment. Invite itself included the agenda and also, the questions related to the thesis and artifact.

The real-life case and the results of each iteration are presented more thoroughly in chapter five. The original request was to create a procedure diagram as a method. With procedure diagram the aim was to follow steps to come up with the solution that is needed for detecting empty values from IoT data streams. The final decision making should be based on the theory presented within this thesis.

In chapter six there is the final result split into several different sections which can be found in figures. The artifact was presented to IT Data & Software Development in Teams meetings with these figures so that the team could easily understand what are the things to consider when deciding the best method for detecting anomalies. These figures were gone through one by one and the team told their opinion whether the figures were the actual things that should be included to the final result or not.

Timeline for the iterations in this process is presented in figure 4.1. Agenda for the iterations can be seen in the figure 4.1 and more thoroughly explanation of the agenda and iterations is in chapter five.

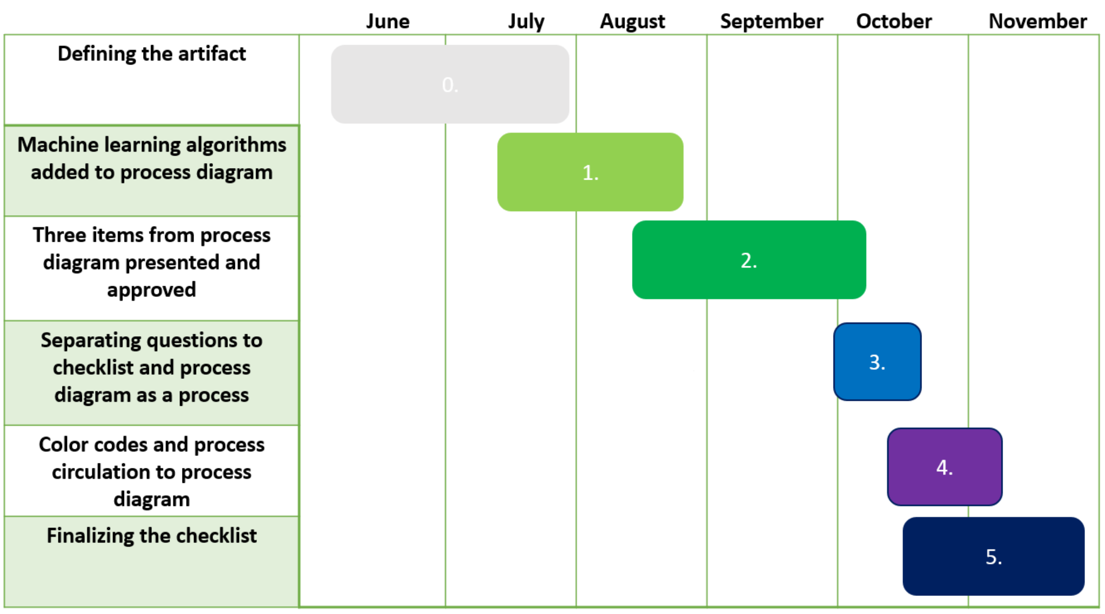


Figure 4.1: Timeline of the process.

5 Real-life case for anomaly detection in IoT data streams

The main objective for this Master’s thesis is to create a procedure on how to detect anomalies in IoT data stream in real-life setting. This request comes from IT Data & Software Development team in Fortum. The team lacked a procedure for detecting empty values in IoT data streams automatically with the smallest amount of human interactions. Also, the aim is to introduce a procedure which is detecting anomalies on the fly before the data is transferred into storage. There are various types of IoT data streams, so the detection method should be scalable and adjustable to other datasets.

The current solution for anomaly detection is to collect the data and discover empty values by analyzing the data after the data is stored in database. This method has shown that there indeed are some empty values occasionally in the IoT data streams. The procedure currently used is presented in figure 5.1. First, sensors are collecting the data and the data is transferred into data storage. After the data is in the database, the data is analyzed by performing a number of SQL queries on it. Queries are used to identify the empty values in the data and to create visualizations on the data,

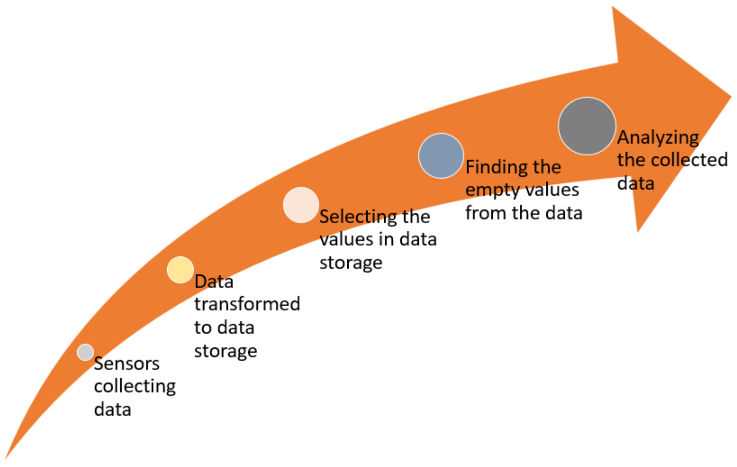

Sensors collecting data
Data transformed to data storage
Selecting the values in data storage
Finding the empty values from the data
Analyzing the collected data

Figure 5.1: Current detection method.

Even though the current solution is capable of identifying the empty values within the data, there are some disadvantages with the method. Firstly, the analysis is performed after storing the data in database, so it cannot provide real-time alerts. Secondly, since SQL queries are consuming lots of server capacity in the platform, they cannot be run continuously. Therefore, anomalies can occur when they are not actively being observed, and the data quality can suffer due to possible aggregations done before the actual analysis and identification of the anomalies. Because of the large capacity need of the SQL queries when detecting empty values, the SQL queries have not been triggered automatically, but instead they have been executed manually. As a summary, an improvement for the current situation would be to have the anomaly detection 1) automated, 2) done on the fly before the data is in the storage, and 3) maintained with the smallest amount of human interactions.

The state of will is to have step-by-step guidelines on how to achieve these improvements in the anomaly detection process. Part of this Master's thesis is to search for an automated solution that will detect the empty values on the fly before the data will be transferred into the data storage. The future procedure is presented in figure 5.2.

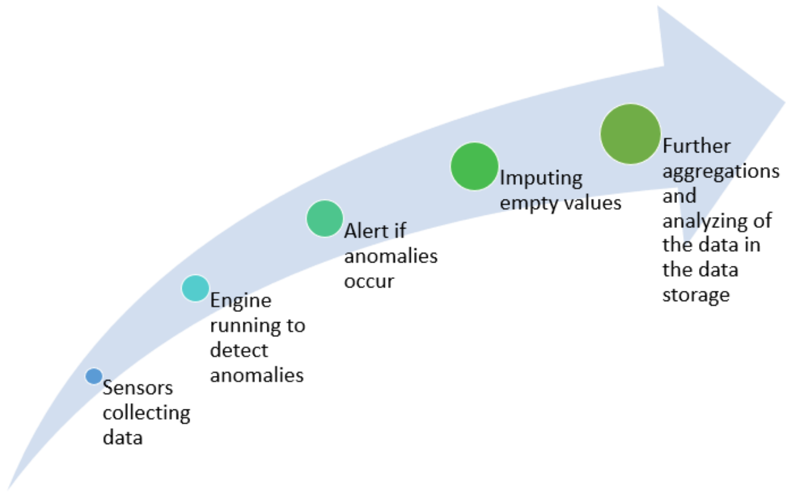
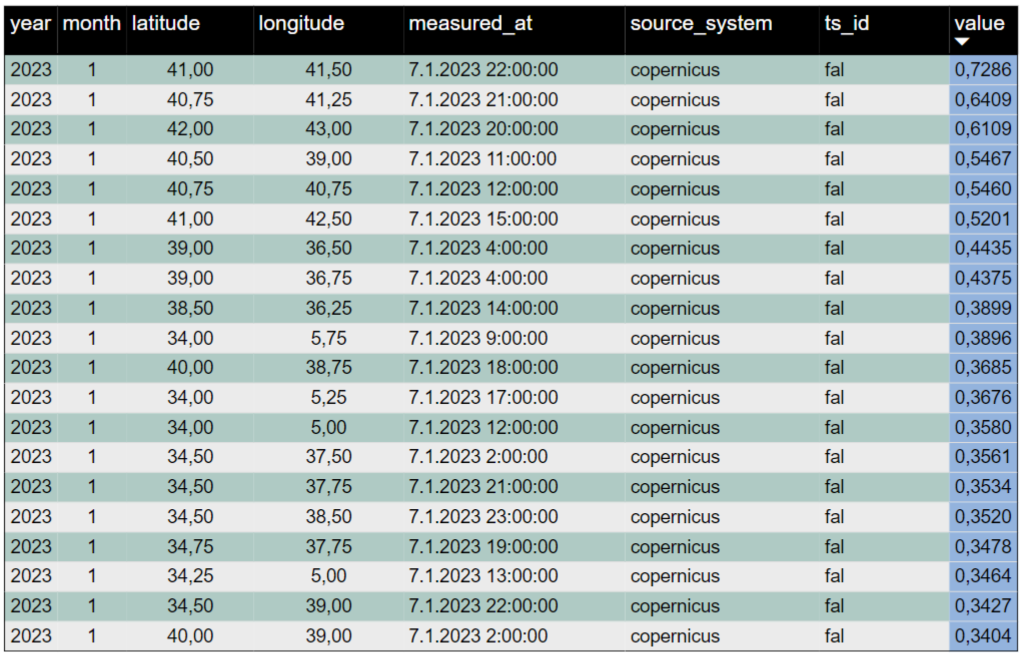

Sensors collecting data
Engine running to detect anomalies
Alert if anomalies occur
Imputing empty values
Further aggregations and analyzing of the data in the data storage

Figure 5.2: Future detection method.

The procedure presented in this Master's thesis is requested to be scalable for other than the actual case. Thus, the solution presented in this work tries to outline the information that is needed when starting a new project of detecting anomalies from IoT data streams instead of providing a single detailed process model for the case at hand. Setting up this kind of procedure is quite straightforward for point anomalies, but detecting other kinds of anomalies, such as contextual or collective anomalies, may become much more problematic.

5.1 The real-life IoT data stream

There are several scenarios defined by the Fortum IT Data & Software Development team about the cases in which the detection of anomalies would be applied to. The dataset represented here gives an example of what kind of data needs to be analyzed in case of anomalies. This dataset is publicly available for everyone and the basic concept for the other IoT datasets is very similar than the example here. The attributes of the dataset are "latitude", "longitude", "measures_at", "value", "ts_id", "year", "month", "arrived_at", "loaded_at", "rsrc" and "source_system". This dataset is containing data starting from the year 1950. In figure 5.3, an example of the contents of the data is presented. Blue-colored column named "value" is emphasized in the figure, and this is the attribute of interest from which possible anomalies should be observed.



year	month	latitude	longitude	measured_at	source_system	ts_id	value
2023	1	41,00	41,50	7.1.2023 22:00:00	copernicus	fal	0,7286
2023	1	40,75	41,25	7.1.2023 21:00:00	copernicus	fal	0,6409
2023	1	42,00	43,00	7.1.2023 20:00:00	copernicus	fal	0,6109
2023	1	40,50	39,00	7.1.2023 11:00:00	copernicus	fal	0,5467
2023	1	40,75	40,75	7.1.2023 12:00:00	copernicus	fal	0,5460
2023	1	41,00	42,50	7.1.2023 15:00:00	copernicus	fal	0,5201
2023	1	39,00	36,50	7.1.2023 4:00:00	copernicus	fal	0,4435
2023	1	39,00	36,75	7.1.2023 4:00:00	copernicus	fal	0,4375
2023	1	38,50	36,25	7.1.2023 14:00:00	copernicus	fal	0,3899
2023	1	34,00	5,75	7.1.2023 9:00:00	copernicus	fal	0,3896
2023	1	40,00	38,75	7.1.2023 18:00:00	copernicus	fal	0,3685
2023	1	34,00	5,25	7.1.2023 17:00:00	copernicus	fal	0,3676
2023	1	34,00	5,00	7.1.2023 12:00:00	copernicus	fal	0,3580
2023	1	34,50	37,50	7.1.2023 2:00:00	copernicus	fal	0,3561
2023	1	34,50	37,75	7.1.2023 21:00:00	copernicus	fal	0,3534
2023	1	34,50	38,50	7.1.2023 23:00:00	copernicus	fal	0,3520
2023	1	34,75	37,75	7.1.2023 19:00:00	copernicus	fal	0,3478
2023	1	34,25	5,00	7.1.2023 13:00:00	copernicus	fal	0,3464
2023	1	34,50	39,00	7.1.2023 22:00:00	copernicus	fal	0,3427
2023	1	40,00	39,00	7.1.2023 2:00:00	copernicus	fal	0,3404

Figure 5.3: Example from the dataset containing data of IoT data streams.

The value attribute contains the actual measured data from sensors. If there is an error when fetching the data from a sensor or delivering the data to a database, the column called "value" will have either an empty value or otherwise wrong data. This may affect to the results of data analysis when aggregating the data and hence lead to wrong conclusions. Currently, it is known that there can occur different values that are considered abnormal. Values ″0″, null or empty are considered abnormal. The problem is that whenever there is a value ″0″, it can be either a normal value or an abnormal value depending on the environmental state. This anomaly type is known as contextual anomaly. Research have shown that collective and contextual anomalies in IoT data streams are not that well-handled and there is not enough knowledge to provide exhaustive solutions for these anomaly types. This Master′s thesis includes the point anomaly detection method options for the Fortum case. The challenges regarding to contextual and collective anomalies are presented though as a reminder that there are ways to detect those types as well but it requires more effort and resources. Since there is not an existing solution to detect anomalies on the fly currently, it is as a good practice to start the process by detecting point anomalies because they are easier to find.

In figure 5.4, a thousand values extracted from the IoT data that is sent to the database are being evaluated based on the variation of the measures. It is shown in the figure that the minimum value of the data points is 0.0595 when the maximum value of the data points is 0.7286. If there is null values in the dataset, it will affect to data analysis when aggregating the data. The impact on the results will be significant especially if the analysis is done in a narrow data window.

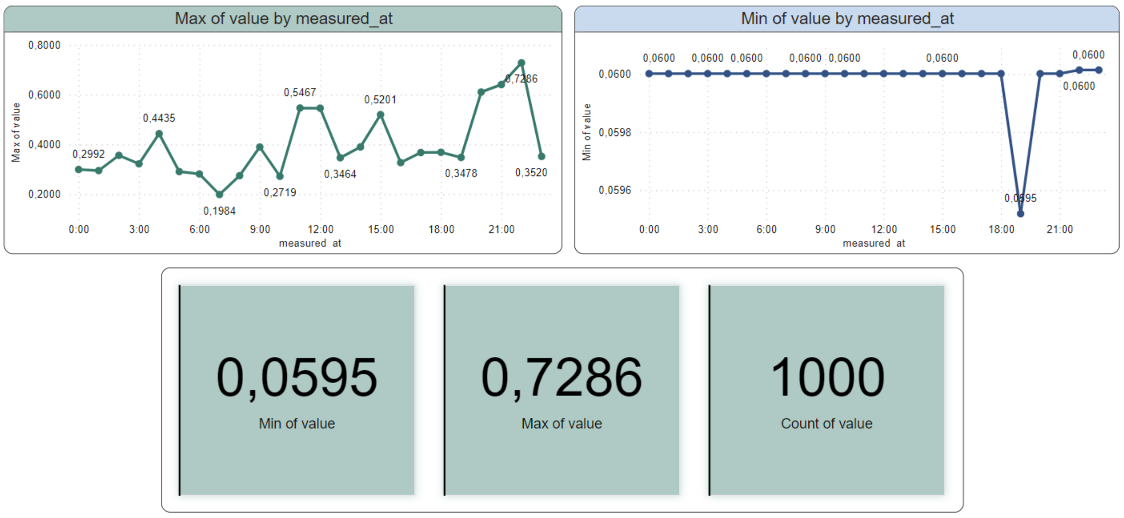


Figure 5.4: Variation of the measurement values in an example dataset.

5.2 Iteration of the artifact

Iterations of the artifact were done fortnightly with the Fortum IT Data & Software Development team via Teams. Agenda for each meeting was delivered to participants along with the Teams invite. The agenda for each iteration was defined separately based on the changes from previous iteration and new ideas regarding the creation of the artifact.

5.2.1 First iteration

Based on the theory, it seems obvious that machine learning algorithms are needed when detecting anomalies on the fly. The first thing to change with the original assignment was to introduce possible machine learning algorithms in the procedure diagram. Originally, only a process was going to be described but since the research strongly suggested utilization of machine learning techniques in finding anomalies from IoT streams, it was agreed that machine learning should be used to achieve the real-time detection. Hence the most used algorithms along their benefits and limitations are presented in this work and suggestions of the algorithms are included in procedure diagram.

5.2.2 Second iteration

During the second iteration, a partial procedure diagram containing three steps was represented for commenting. The idea was to present these steps for the team in order to gain better understanding of all the needed aspects for the procedure. By presenting these steps separately it was easier to understand the specific requirements for the procedure diagram. The three steps presented were

- **How to get started with the procedure**

  Define what needs to be analyzed

- **Which layer to be used**

  Device, network, edge computing

- **Type of the anomaly**

  Point, contextual or collective

In figure 5.5 the first questions that are mandatory when getting started with anomaly detection are presented. First question is the key thing to decide when considering what are the required results from the anomaly detection: whether the detection algorithms will try to find empty values or other abnormalities. The IoT data requiring anomaly detection consists purely of time series data, thus this picture reminds that aspect to be taken into account in the analysis.

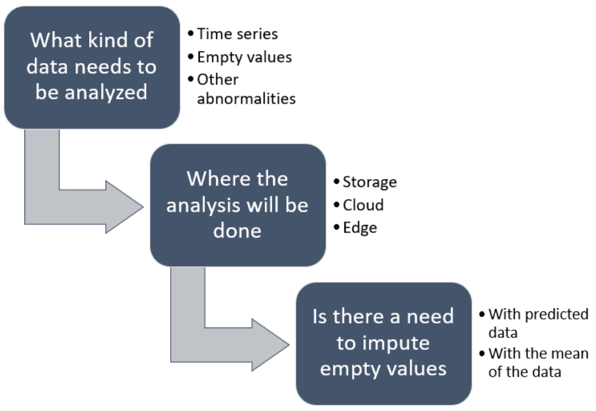

What kind of data needs to be analyzed
• Time series
• Empty values
• Other abnormalities
Where the analysis will be done
• Storage
• Cloud
• Edge
Is there a need to impute empty values
• With predicted data
• With the mean of the data

Figure 5.5: How to get started

The second step depicted figure 5.5 is about where analysis should be done. The analysis is done currently on a storage level and this should be included in the procedure diagram as an option, because it has been a working solution. The current IoT network architecture does not have a separate edge computing layer. This is because the overall architecture has been kept as simple as possible. The existing platform (cloud) is presenting the edge computing component and that is where the data processing happens. Thus, it was decided that the algorithms that are specifically designed to work on edge, will not to be included with the final procedure diagram.

The third step in figure 5.5 is asking whether the empty values should be imputed or not. The original request was only to detect the anomalies and there is no need for the team to impute the empty values at this moment. However, because it has been shown that empty values are causing problems with classification algorithms, the imputing of empty values are included to the procedure diagram. This way it can be decided later if the imputing is needed.

In this iteration, the team raised an issue that IoT data streams can contain other abnormalities than empty values as well. It was emphasized by the team that they hope to find empty values that can take different embodiments in the IoT data streams. The empty values can show up as blank values, as ″null″ values or as ″0″ values. It was described that some of the values that are ″0″ (zero) are, in some cases, actually valid values, but abnormal in some cases. This kind of anomaly type is contextual and there are completely different algorithms to be used with contextual anomalies compared to point anomalies. A decision was made that this

Master's thesis should concentrate only on the point anomaly detection. Contextual anomaly detection will be considered in further development of the artifact.

The second step (figure 5.6) shown to the team was about which layer in IoT architecture should the anomaly detection engine run on. Device layer is not recommendable since there are not that many studies regarding anomaly detection on a device level. [11] Furthermore, it is hard to analyze data in a complex way on the device layer since there are considerable computational limitations with IoT devices. [34] With the existing IoT architecture solution, the edge computing is done in the cloud which is working as an edge computing component. When deciding the appropriate algorithms for the anomaly detection, it should be carefully considered, on which architectural level the detection engine should run, or if there is a need to have multiple algorithms on different layers in order to gain wanted results.

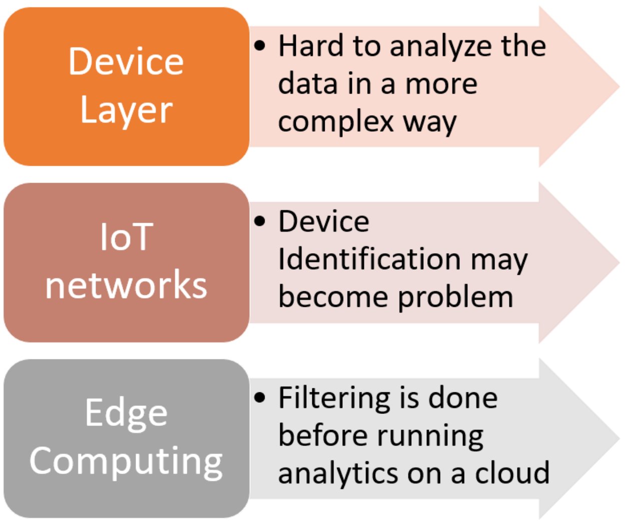

Device Layer
• Hard to analyze the data in a more complex way
IoT networks
• Device Identification may become problem
Edge Computing
• Filtering is done before running analytics on a cloud

Figure 5.6: Analysis can be done on different layers

The third step (figure 5.7) was about the identification of anomaly type. Different types of anomalies were explained to the team, and shortly explained the differences between point, contextual and collective anomalies. Each of these types have different mechanisms when detecting anomalies. There is a lack of research regarding finding contextual and collective anomalies in IoT data streams. Some methods for detecting these types are presented briefly in this thesis, but it was proposed to the team to start anomaly detection by searching only for point anomalies. This was approved by the team.

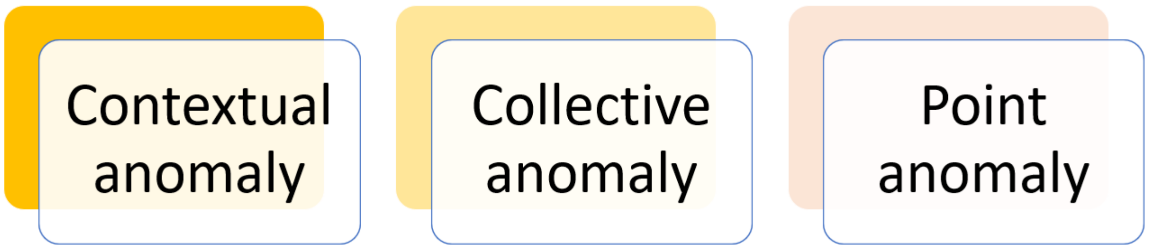


Figure 5.7: Anomaly types.

Questions that emerged from the team within second iteration were about the scalability of the procedure diagram and about the effort that is required for the maintenance. As it has been pointed out, there is no solution for IoT anomaly detection that would fit to all the cases [34]. Even though the scalability is very important, it is almost impossible to create a procedure diagram which covers all the needed aspects regarding to detecting anomalies from all types of IoT data streams on the fly. It was decided that scalability would be addressed in the subsequent iterations and attention would be paid on how the scalability could be accomplished with this artifact.

5.2.3 Third iteration

Creating a procedure diagram for detecting anomalies on IoT data streams turned out to be a challenge that cannot be easily solved, since there are so many things to take into account before the actual anomaly detection can begin, and especially since the scalability should be ensured. For these reasons, in third iteration, I suggested that instead of one procedure diagram, there could also be a checklist that would sum up all the needed actions before proceeding with the procedure diagram, which in turn would guide through the process. This way it would be possible to determine the goals for the task first, and then decide on how the achieve these goals. This kind of approach is necessary especially when targeting for a scalable solution.

The first draft of the list of needed actions is presented on figure 5.8 This list was assembled from the general machine learning guidelines. The general machine learning guidelines were modified to match the needs for the case at hand. First questions for definition of the problem concern what the model should predict and when the model is successful. When defining what the model should predict it is crucial to define whether they are empty values or some other kind of anomalies that should be recognized from the normal patterns of the data. It is also important to define the rules that determine the successfulness of the model. This helps to understand the real goal of the process. Second question is about where the anomaly detection engine should run. This should be decided before using the procedure diagram since there are many things to take into account when considering what kind of resources there are available for the anomaly detection. Third action is to prepare the data. This means that data should be cleaned and transformed properly before creating a machine learning model for it. After the data clean up, fourth action is to train the machine learning model. Algorithm to be used needs to be chosen and decided whether there is only one algorithm to detect the anomalies or should there be more than one algorithm. After choosing the correct algorithm, the hyperparameter values are decided. Fifth and final action is to generate insights so that the requested predictions can be seen.

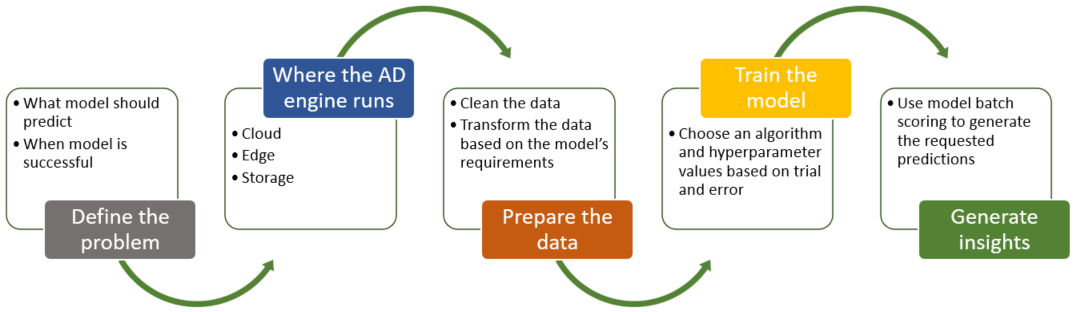

• What model should predict
• When model is successful
Define the problem
Where the AD engine runs
• Cloud
• Edge
• Storage
• Clean the data
• Transform the data based on the model's requirements
Prepare the data
Train the model
• Choose an algorithm and hyperparameter values based on trial and error
• Use model batch scoring to generate the requested predictions
Generate insights

Figure 5.8: Machine Learning guidelines.[34]

After the first draft of the checklist was created, it was enhanced to include questions about the whole process and not only about machine learning. These questions are general questions that guide to the right direction when deciding the proper approach for the anomaly detection.

1. What kind of time series is analyzed: univariate or multivariate?

2. What is the format of the time series data?
3. Where to run the detection engine?
4. What are the point anomalies to be discovered?
5. What is the learning algorithm to be used?
6. Which are the features to be used?
7. How to clean the data for the machine learning algorithms?
8. How to evaluate the results?
9. Where the alerts should come?
10. Is there a need to impute the empty values?

The first question in the checklist considers what kind of time series data will be analyzed in anomaly detection. It needs to defined if the time series is multivariate or univariate. With univariate time series only one algorithm is adequate for detecting anomalies but with multivariate time series more than one algorithm are required. The next question is about the format that is used to present the data. It can be json or binary format, for example. This is important to recognize in order to properly clean the data before the machine learning algorithms are used. The third question asks where the anomaly detection engine should run. This is a decision that needs to be made every time there is a different dataset to be analyzed. The fourth question instructs the user user to define what is the purpose of the particular anomaly detection. The algorithms to be used differ from each other depending of the anomalies that are sought for. For example, value ″0″ is most likely contextual anomaly instead of point anomaly since the value varies based on the environmental state whether it is abnormal or normal value. On the other hand, empty values are considered as point anomalies since there is a presumption that a value should occur but instead there is no value. Point anomaly detection is easier compared to finding contextual anomalies.

After these questions there are some guidelines about which algorithms could be used based on the answers. So the fifth question concerns the algorithm should be used. Before making this decision it is crucial to understand the nature of the data so that it can be decided if the data should be analyzed with supervised or unsupervised algorithms. Third option is semi-supervised algorithms, but that is not recommended within this context, since the research show that semi-supervised algorithms in anomaly detection need further investigations.

Once the algorithm for the anomaly detection is decided, the hyperparameters need to be defined. It requires more resources if this done to supervised algorithms. Based on the decisions whether to use supervised or unsupervised algorithms and which hyperparameters are going to be used with the algorithms, the cleaning and other preparations for the data can be done. Next, the question about how to evaluate the correct result needs to be answered. Before doing any executing with the algorithms, it must be decided how the results will be evaluated. With machine learning algorithms, it needs to be understood that there are always false positives and false negatives, and that the hyperparameters can regulate the final result, for example.

After defining the machine learning algorithms to be used, a plan of the platform that delivers the alerts about abnormal values, and recipients of the alerts needs to be outlined. Finally, in this thesis it is presented that after the alert is triggered, it should be decided if there is a need to impute empty values if they occur. Empty values can cause misleading results for the data analysis. Thus, the empty values should be not only recognized but also imputed. As the question concerning empty values is the last one in the checklist, it acts also as a reminder that there are ways to improve the data quality when imputing the empty values.

Changes made after the proposal

The Fortum IT Data & Software Development team concluded that the checklist for the IoT data stream anomaly detection is a great idea for supporting the procedure diagram. This improves the scalability of the final result for other IoT anomaly detection cases as well. However, modifications will be done based on the needs of real-life case anomaly detection and the changes are presented in this section. The first question was formatted based on the real-world needs and question number two was removed. Other questions were approved the way suggested.

1. What kind of time series is analyzed: break point pr fixed interval time series?

2. ~~What is the format of the time series data?~~

The Fortum IT Data & Software Development team have IoT data streams that are gathering univariate time series data. Hence the first question whether the data is univariate or multivariate is not needed here. Instead there was a proposal from the team that the first question could be more specific to better match their needs. The team has IoT data that is presented with so called ″breakpoint time series″ and ″fixed interval time series″. The difference between these two is that ″breakpoint timeseries″ sends values to cloud and storage only if the value changes. For example, these time series will not send anything if temperature remains the same but once the temperature changes, then the sensors are sending the data. ″Fixed interval time series″ sends data at fixed time points. The interval can be anything from one minute to one week but the essence of these time series is that the data will be sent on fixed time points that are defined based on the needs of the particular application.

The other thing to be changed within the checklist is that there is no need to define the format of the IoT data when starting the anomaly detection process. The platform in use in the real-life cases uses Apache Spark which can digest any kind of data not dependent of the format of the data. After Apache Spark has digested the format of the data, it goes forward from that point, and the anomaly detection starts only after Apache Spark has ingested the data. Hence it needs to be considered whether there is any reason to keep this question within the checklist.

For the next iteration, questions about supervised and unsupervised data were raised. Based on the shared knowledge regarding the data gathered from IoT, it must be considered if there is need to use both supervised and unsupervised algorithms. Supervised algorithm could be better with the ″fixed interval time series″ since there are values that can be predicted to appear in the system. On the other hand, unsupervised algorithms might be better to use with ″breakpoint time series″ since the prediction can be more challenging to accomplish. Clustering with unsupervised learning could discover the anomalies with higher accuracy in break point time series.

5.2.4 Fourth iteration

The agenda for fourth iteration was to represent the last version of the procedure diagram. Biggest changes of the procedure diagram were changing the colors and creating a circle for the procedure diagram after the alert. Colors are representing the options for the procedure which means that the different alternatives for the procedure are coded with different colors. The circle in the procedure diagram describes the continuous process where the evaluation needs to be done after every iteration.

Colors were changed to match Fortum theme colors but besides this, the steps were coded with colors. White boxes are always points to stop and think about the answer for a question. For example, after choosing what kind of anomaly type is being looked for, it needs to be decided whether anomaly detection needs to be on real-time or not on real-time.

Light green and light purple ovals in the procedure diagram mean that you have two options where to go from there. For example, you can choose to run machine learning algorithm on cloud and after deciding on that you need to prepare the data. Before preparing the data it is necessary to know whether you are dealing with known data or unknown data. With known data supervised learning algorithm is the one to be used and this leads to classification before training the model. With unknown data you should follow the unsupervised learning path, which leads to using clustering algorithms.

Yellow ovals are options that are not targeted at this point when detecting anomalies in real-life case. However, they are included for the procedure diagram as a reminder since the original request was to create a scalable procedure for the anomaly detection. In this case, the contextual anomalies and collective anomalies are left out from this thesis since they require more research how to implement anomaly detection to those. The procedure diagram can be extended to target those types of anomalies as well. For the same reason, ”Edge” as an anomaly detection engine is not needed currently, but might be needed later. The same concerns the semi-supervised learning that currently it is not recommended but later, when there is more research regarding to semi-supervised learning used with IoT anomaly detection, the procedure can be widen to include semi-supervised learning methods as well.

The final step in the procedure diagram is the alert. After the alert there should always be evaluation of the machine learning model. Evaluation methods are presented in chapter three. Once the evaluation is completed, the path creates a circle in the procedure diagram. This circle describes how the process should be evaluated all the time and how the retraining of the model needs to be done regularly.

The procedure diagram describes one process with evaluating and retraining. If the procedure diagram is used for another process, the process should always start from the beginning. The checklist is useful also for starting a new project, since it reminds the basic questions about the process and helps with the decision making.

Changes made based on the fourth iteration

The procedure diagram was accepted as suggested. The checklist on the other hand was requested to have minor changes. Request from Fortum IT Data & Software Development team was to add a question to the checklist about the actions to take once the anomalies are detected. The other important question to be added to the checklist concerns how an anomaly is defined with the machine learning model. The major thing that the team is worried about is that will there be an alert if one blank value occurs or can it be defined that if the same blank value occurs several times in 10 minutes or so, is it only then considered as an anomaly. These were the things to be brought for the last iteration.

5.2.5 Fifth iteration

For the fifth and final iteration the checklist was finalized. In this iteration, the checklist was presented highlighting the headlines of the checklist. Instead of using questions, it was more clear to specify the actions by creating the headlines for the checklist. Headlines were accepted, meaning that we were able to find the key things in chronological order and present them by selecting the correct headlines for the checklist. The completed checklist can be found as appendix two. Altogether it was decided to have nine components as headlines in the checklist:

- Define the type of time series,
- Define the type of delivery,
- Define the type of anomaly,
- Location of the anomaly detection engine,
- Define the wanted results,
- Decide the algorithm,
- Preprocess the data,
- Actions if anomalies are detected,

- Evaluation.

Based on the previous iterations on the artifact it was quite clear to define the key things as headlines and they are presented in the final version of the checklist in chronological order. Chronological order helps to follow the process step by step. The chronological order starts by defining the type of the time series which in this real-life case the options are break point time series or fixed point time series. In fixed point time series it needs to be defined what is the gap between the data delivery and this is added to checklist as a reminder as well. The anomaly detection is dependent on the data delivery gap. The Fortum IT Data & Software Development team wanted the type of the delivery to added here, too. This is crucial for them to define the type of the delivered time series. The options are ″batch″ and ″continuous″. This is affecting how often the anomaly detection engine should run.

The third section of the checklist is to define the type of anomaly. Here, the procedure diagram gives the point anomaly as an only option. But since the request was to have scalable solution, later the options can include contextual or collective anomalies as well. The fourth section is about where the anomaly detection engine should run. The options are storage or cloud in the real-life case. As a reminder, it is added here that this can be done on edge layer, but since currently there is no edge layer in the IoT architecture, it was ruled out for more investigations.

Fifth section of the checklist defines the wanted results. It means that before deciding which algorithm to be used, it must be decided if the anomalies that are being looked for are blank values, presented as value ″0″ or as value ″null″. At this point it needs to be defined whether all the anomalies should raise an alert, or if the anomalies are wanted to be alarmed of only when there are a certain amount of them.

Previous decisions have an impact to the next section. The sixth section is to decide the algorithm to be used. In this real-life case it is recommended to use supervised or unsupervised algorithm and the most used options are presented within the checklist. Chapter three represents the benefits and limitations for both of these methods and there are short explanations on how to use these algorithms to get the best result.

After the algorithm is chosen, the seventh section is about preprocessing the data. The preprocessing requires more effort, if the chosen method is supervised learning. There are some basic guidelines about the required steps to be used before running the supervised algorithm and those are added to the checklist. With unsupervised learning it can be decided more freely how much the data needs to be preprocessed.

The eight section of the checklist is what to do when anomalies occur. Options are to impute the missing values, do more research of the network or check if the sensors have some kind of failure. Imputing missing values can be done with machine learning algorithms but investigations of the network or the sensor failure must be done by using some other methods.

The last section is the evaluation. It means that there are measures that can be used with machine learning that can give the estimations how well the algorithm has performed. Evaluation process is always part of machine learning algorithms but since it gives the kind results that will affect to the next round of running the algorithm, it has been separated here to highlight the importance of choosing the correct measures for the evaluation. This is a step to be used with the existing solution where the anomalies are detected via SQL queries and hence it is found as a step in procedure diagram as well.

6 Conclusions

Since the field of IoT is constantly evolving, it is quite clear that anomaly detection is needed. Several different approaches have been tested in research, and some promising results have been achieved. However, the IoT anomaly detection needs more investigation, because of the special features of IoT systems: architecture, huge volume of data, protocols, real-time analytics, and energy consumption. These all bring their own challenges to finding a solution for detecting anomalies from IoT data streams.

The real-life case depicted in work was to find an answer to the question "What kind of methods there are to detect anomalies from IoT data streams?" More precisely there was a request to create procedure that will meet these requirements:

1. Create procedure that detects anomalies automatically.
2. The procedure needs to discover anomalies on real-time.
3. The procedure includes suggestion what to do when anomalies are detected.
4. The final procedure is maintained with the smallest amount of human interactions.
5. The procedure is scalable for other IoT data streams.

Design science was chosen for research method since this request was about creating a new procedure. Design science aims to gain knowledge on an issue through a creation of an artifact that will serve human purposes and provide an innovative solution to a certain problem. As an artifact it was created a procedure diagram and checklist to be used together. The procedure diagram provides a general view of the artifact while the checklist reminds and explains more about the details that need to be considered when deciding the method for anomaly detection.

6.1 The final artifact

The final version of the artifact is presented in appendices. Appendix one represents the procedure diagram to be used when implementing anomaly detection in

IoT data stream. Appendix two includes the checklist to be used before going to procedure diagram. The checklist is giving more detailed information regarding things that need to be considered before running any machine learning algorithms. Procedure diagram can be used more generally when the items from the checklist are more familiar. Since this is a new procedure for Fortum IT Data & Software Development team to detect anomalies from IoT data streams on the fly, the checklist provides details that are required to understand the items in the procedure diagram.

6.1.1 First part of the artifact: procedure diagram

The complete version of procedure diagram can be found in appendix one. The diagram is presented here in smaller parts in order to see why specific steps are crucial to be included for the diagram. The steps can be divided into five different components based on the factors that guide choosing the correct procedure for IoT anomaly detection. The components are type of the time series, anomaly type, anomaly detection engine, SQL query on data storage and machine learning method on cloud.

The first section of the procedure diagram is to define which type of time series is being observed in detecting anomalies. This step is shown in figure 6.1. These types of time series are the ones that are in use in Fortum IoT data streams. Even though they are both referring to same direction in procedure diagram, it is crucial to define which one is being used when deciding the method for anomaly detection. It needs to be tested out whether these types of time series can have the same algorithm for anomaly detection. Furthermore, the definition of the parameters for the machine learning algorithm needs to be carefully considered to get the best result. For example, in fixed point time series blank values can be part of the normal data, the break point time series does not have this same quality since the data is being sent only when value changes.

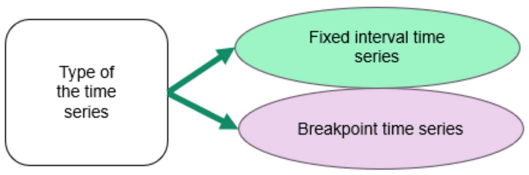


Figure 6.1: Type of time series.

The second section of the procedure diagram concerns the type of anomaly. As mentioned in chapter five, the ovals are color coded so that the light green or light purple are the ones that are recommended to be used. The yellow ovals are the ones that are included as reminders that different anomalies can also be detected but it requires more resources. At this point it is vital to understand the difference between the different anomaly types, and for this reason these yellow ovals remind that results should be defined as point anomalies instead of contextual or collective anomalies.

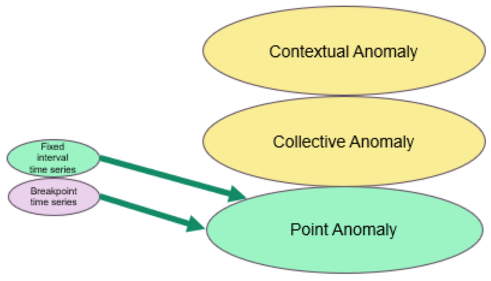


Figure 6.2: Anomaly types.

Third section of the procedure diagram defines where the anomaly detection engine runs (figure 6.3). In case there is no need for real-time anomaly detection, the process can be done as it has already been previously: with running SQL query once the data is in storage. However, if the anomaly detection is required to provide real-time analysis, then the engine should run on cloud. It is suggested to run machine learning algorithms on the cloud. Edge is the third option to run anomaly detection but it is marked as a yellow oval in the procedure diagram since it is not currently applicable because of the existing IoT architecture solution. Edge is included as a reminder here in scalability perspective.

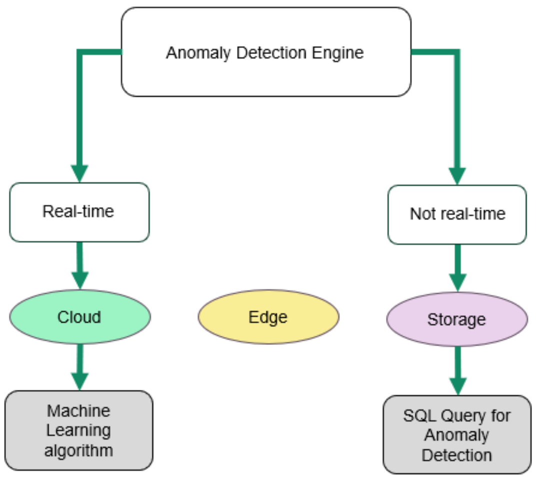


Figure 6.3: Anomaly detection engine.

Figures 6.4 and 6.5 illustrate options concerning real-time and delayed (non-real-time) solutions in more detail. In figure 6.4, SQL query is run based on the data that is already stored in the database. If anomalies are found, an alert is sent. After sending the alert, there will be evaluation how the method has worked. In case no anomalies are found, the query still needs an evaluation before running it again.

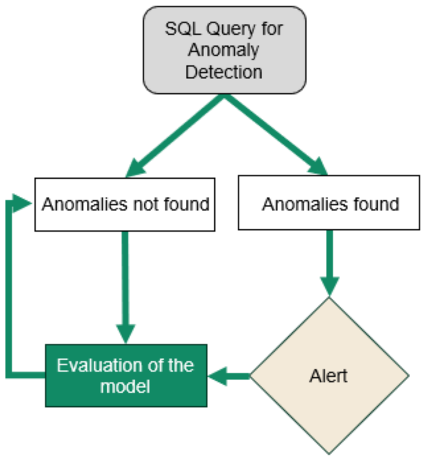


Figure 6.4: SQL query on data storage.

Finally, the last section, shown in figure 6.5, describes the issues to be decided when running a machine learning algorithm in cloud. The options are that we know the data, or the data is unknown. Based on this selection the known data leads to using supervised algorithm and the unknown data leads to unsupervised algorithm. Unsupervised algorithm has fewer interphases since after clustering there is no need to train and test the model. Supervised algorithm uses classification and for this the data needs to be labeled. After labeling, the model needs to be trained and then the model is tested with unseen data. Unsupervised algorithm uses clustering for detecting anomalies and supervised algorithm classification. As in SQL query, finding anomalies leads to an alert. Evaluation of the model needs to be done regardless which algorithm is chosen and whether anomalies are found or not.

With both SQL query and machine learning algorithms, the process starts from the beginning based on the chosen method. This is a continuous and iterative circle that needs to be run again once the previous is done. In the checklist, there are more precise descriptions about how often these loops are run and about the interphases that are needed with machine learning algorithms.

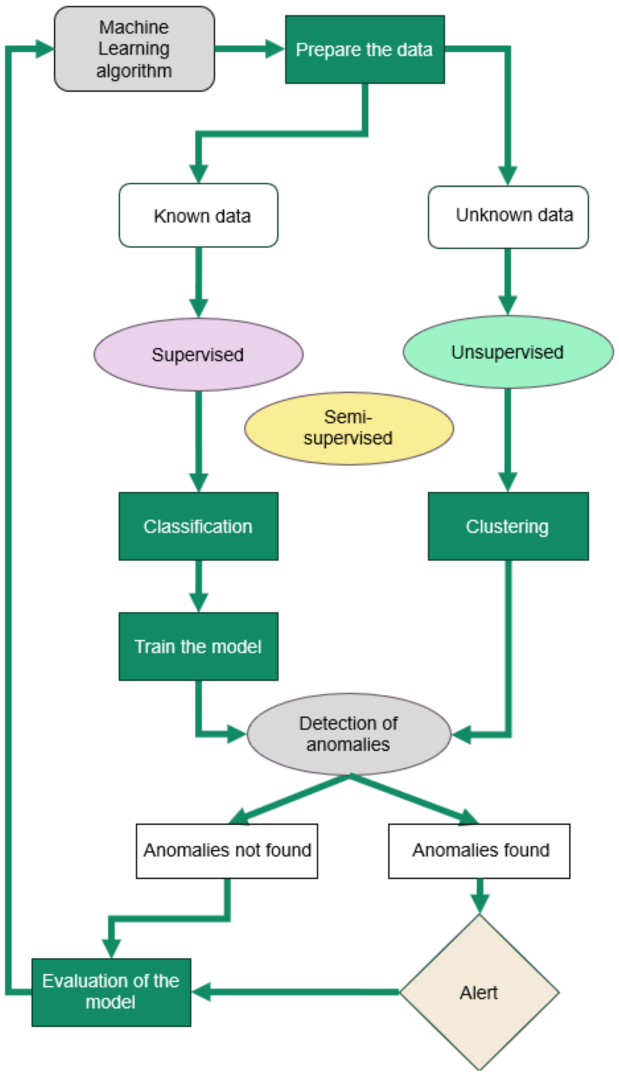


Figure 6.5: Machine learning on cloud.

6.1.2 Second part of the artifact: checklist

The checklist is shown in appendix two. The checklist was divided into nine different sections in chronological order as follows:

1. Define the type of time series
2. Define the type of delivery in time series
3. Define the type of anomaly
4. Location of the anomaly detection engine
5. Define the wanted results
6. Decide the algorithm
7. Preprocess the data
8. Actions if anomalies are detected
9. Evaluation of the model

Since the checklist can also be a part of the documentation created for each anomaly detection project, it has the same components than the procedure diagram. Compared to the procedure diagram, checklist provides more detailed information about the needed components. The method of using the checklist involves mostly selecting the correct checkbox for each situation, but the there are also sections where an answer to a specific question needs to be written down. For example, with fixed point time series the interval of the sent data needs to be defined for documentation purposes.

The checklist starts by defining the type of the time series. The options in this real-life case are fixed interval time series and breakpoint time series. Breakpoint time series sends values only if they change. Fixed interval time series sends data at fixed time points and in that case the interval needs to be determined. By the request of Fortum IT Data & Software Development team, the type of delivery of the time series was included here which is crucial when deciding how often the anomaly detection engine is run.

Next, the checklist guides to define the anomaly type. For scalability, the point anomaly, contextual anomaly and collective anomaly can be chosen here. This section provides short explanations about how these anomaly types differ from each other. This section works as a reminder since it is vital to understand the diverse types of anomalies. Diverse types of anomalies are detected by different methods.

The checklist continues with the question of location of the anomaly detection engine. The options in the checklist include only cloud and data storage since they are the ones to be used in the real-life case at this point. Edge is added here as an option as well for scalability even though currently edge is not used within the IoT architecture.

Determining the desired results is the next section. Under this headline there are four questions to be answered to: what the model should predict, when the model is successful, what are the results that will lead to an alert and where the alert should be delivered to. Before testing this artifact Fortum IT Data & Software Development team was not sure if they wanted all the point anomalies to trigger an alert or should an alert be sent only when a certain number of anomalies have occurred. Thus, the definition of the rules and mechanism of alerting was added to the checklist.

After determining the objectives for the data model, choosing an appropriate algorithm becomes possible. In order to select correct algorithm, there are things that need to be understood on the data domain level. If the data is known, supervised algorithm is recommended, but if the data is unknown, the unsupervised algorithm should be used. Semi-supervised algorithm is here for scalability but before there is more research on semi-supervised algorithm finding anomalies, it cannot be recommended for the real-life case. For supervised and unsupervised algorithms, there are some suggestions in the checklist about the most used algorithms in IoT anomaly detection. Furthermore, the feature selection is part of this section where the algorithm needs to be decided.

Preprocessing the data is the next section. It has different options that are not mandatory all to be fulfilled: clean the data, remove the noise, reduce the dimensionality, change data types, balance the data, label the data and train the model. For example, labeling and training data is required only if supervised algorithms are used. Furthermore, it is not mandatory in all to reduce the dimensionality or change the data types, if the data types are already in a format that machine learning algorithm can understand. For scalability all the options are though added here.

The second last section is defining the actions to be done once anomalies are detected. Options are imputing empty values, replacing false data, ensure if there are network issues and ensure if the system has errors. All of the boxes can be ticked, or based on the real-life case, only some of the boxes can be ticked.

Finally, there is the last section concerning evaluation. For evaluation, it is recommended to use predefined metrics in order to have a good understanding whether the model has performed correctly and is it actually giving the expected values. Evaluation metrics in the checklist are mostly for supervised learning, but can be used for unsupervised learning when treating unsupervised learning eventually as a classification problem. Other suggestions for unsupervised learning are complicated and hence they are not recommended at this point since the solution was requested to be simple rather than having more algorithms to be maintained.

6.2 Results of the process

The procedure diagram and checklist were both approved to be used when detecting IoT anomalies in IoT data streams. During the implementation there were a total of five iterations together with Fortum IT Data & Software Development team. Iterations were helpful and with them it was easier to design the kind of artifact that can be used in real-life case. Together the procedure diagram and checklist are able to fulfill the original requirement: to have an automated and scalable procedure that detects anomalies in real-time with the smallest amount of human interactions and provide suggestion what to do when anomalies are detected.

Since design science is about creating an innovative and purposeful artifact, it is crucial to have the iterations during the process. All the iterations during this process were very beneficial and helped to achieve the goal. The definition of the goal was defined by Fortum IT Data & Software Development team, and it was very precise. This helped to create the kind of artifact that is really useful in real-life as well. Iterations helped to create the whole picture of the existing problem. Each iteration was useful and gave more information about the existing data so that the methods for the final procedure diagram could be evaluated more thoroughly.

Iterations helped both sides to better understand what are the things to consider with anomaly detection in IoT data streams. I presented the results based on the research and the Fortum IT Data & Software Development team provided with their special knowledge regarding the real-life case needs. The further the process proceeded, the more clear picture was formed about the real-life data. With this knowledge sharing, it was easier to create the procedure that is actually purposeful for the company itself. The definition of the artifact was very clear from the beginning and the iterations were good approach to validate how the artifact should work.

It has been shown that the accuracy for real-time anomaly detection can be quite poor. Real-time analytics requires higher memory and the execution time is long. [18] One thing to keep in mind is that the existing solution gives the information already from the empty values. The problem with the solution was that it was not on real-time. Running the SQL query also requires some resources, but so will the machine learning methods require resources, and probably even more than the SQL query. This was the reason why in the procedure diagram and in the checklist, the existing solution is included as well as one option. It should be carefully considered which are the cases need the real-time anomaly detection and which are the cases where running the SQL query once the data is in the storage could remain as a solution.

The procedure diagram and checklist do not give recommendations whether to use supervised or unsupervised algorithm when selecting the correct method. They both have their benefits and limitations. With supervised learning there are more human interactions to be done, but on the other hand, the accuracy might be better. Unsupervised learning does not need labeled data but instead it can be used with raw data. Dynamic and constantly changing IoT data will give challenges for both methods in order the methods to succeed with their classification or clustering tasks.

The artifact created in this Master's thesis is the checklist and procedure diagram to be used together when deciding what kind of method to be used when detection anomalies from IoT data streams. The checklist provides more guided instructions about the chronological order to use with the anomaly detection. The procedure diagram is also in chronological order but the procedure diagram can be understood better after the checklist has provided more detailed information. As a conclusion it is better to spend some time and resources when deciding the correct method than to rush into running algorithms that can in the worst scenario give false data if the whole concepts is not understood. The things to give extra attention are the architecture of IoT system, the anomaly type and the type of the time series. Which algorithm to be used for anomaly detection can be discovered after testing.

6.3 Future research

The most important thing for the future is to test the artifact and collect feedback from the user perspective. The artifact most likely will be modified after it has been tested a few times when it has been discovered what are the good things and what could be modified. Hence the artifact should not only be tested, but also evaluated by the feedback from the users.

During the process of creating the procedure for detecting empty values, it was discovered that the empty values can actually occur as three different kinds of values. These values may occur as completely blank values, they might be nulls or they can be presented as "0" values. The procedure diagram presented in the thesis gives guidelines to a procedure to discover empty and null values since these values are classified as point anomalies. When it comes to values that are "0", the "0" can be either a normal value or an abnormal value. This can vary for different reasons, for example, because of the season and hence it cannot be always classified to be as an abnormal value. These kinds of anomalies are classified as contextual anomalies since it depends on the time and environmental state whether it is normal or abnormal. Anomaly detection has divergent methods when it comes to contextual anomalies compared to point anomalies. The methods require more resources and most of the times they cannot be handled with only one machine learning method. Also, human interactions are most likely needed when detecting contextual anomalies from IoT data streams.

On iteration number two it was agreed to rule out the contextual anomalies from this Master's thesis. This was decided together with the Fortum IT Data & Software Development team. Since contextual anomalies require more resources and more primary working by humans, it was decided to make this first version to be simpler procedure than what is needed for the contextual anomaly detection. But because anomaly detection is crucial for data quality and decision making, it is important to notice that detecting contextual anomalies should be further investigated in the future.

The other thing ruled out from this thesis was multivariate time series data. Again, detecting anomalies from multivariate time series requires more resources. It is a good approach to start the anomaly detection from univariate time series since it is more straightforward and requires less resources. This way the procedure can be tested with univariate time series and in case the detection methods are needed for the multivariate time series, it can be widened later. Multivariate time series anomaly detection requires more than one machine learning method. Thus, it cannot be implemented directly without additional investigation.

Based on the research, imputing the empty values is quite often necessary to maintain the quality of the data. At this point it was requested from Fortum IT Data

& Software Development team not to include machine learning methods for imputing empty values to procedure diagram. The original request was to detect empty values so that the possible network issues, or problems with devices or the maintenance of the devices could be discovered and fixed. In the final result, the machine learning methods for imputing empty values were still mentioned, but their usage is optional. Using them requires again more resources, and it is understandable that it is not the main priority when starting the process. However, in the thesis, it has been stated that some of the statistical methods could be used together with the methods that discover the empty values. This requires more resources and hence it is recommended but not mandatory to be used together with the methods that detect empty values.

Even though imputing missing values need resources, it also improves the quality of the data. The methods presented here are machine learning methods. When detecting anomalies from Iot data streams, it could be considered if the detection itself needs to be run on the fly but could it reduce the resources if the imputing was done in the data storage before analyzing the data further.

Deep learning is presented in this thesis because it has been proven to provide good results when detecting anomalies in IoT data streams. However, the final solution does not recommend its usage because of its high demand of resources and time consumptiveness. For future research, it would be beneficial to study this aspect more carefully, because it provides a good option for anomaly detection as well. Especially, in the future when deep learning might not need so much resources, using this method could be considered.

References

[1] AHMAD, . Z., KHAN, A. S., ZEN, K., AND AHMAD, F. Ms-ads: Multistage spectrogram image-based anomaly detection system for iot security. *Wiley* (2023), 1–21.

[2] AL-MILLI, N., AND ALMOBAIDEEN, W. Hybrid neural network to impute missing data for iot applications. *Researchgate* (2019), 1–6.

[3] ALAN R. HAVNER AND SALVATORE T. MARCH AND JINSOO PARK AND SUDHA RAM. Design sciense in information systems research. URL `https://www.researchgate.net/publication/201168946_Design_Science_in_Information_Systems_Research`, viitattu 3.10.2023.

[4] ALGHANMI, N., ALOTAIBI, R., AND BUHARI, S. M. Machine learning approaches for anomaly detection in iot: An overview and future research directions. *Wireless Personal Communication 122* (2021), 2309–2324.

[5] APOSTOL, E.-S., TRUICA, C.-O., POP, F., AND ESPOSITO, C. Change point enhanced anomaly detection for iot time series data. *MDPI Water 13* (2021), 1–19.

[6] BANSAL, M., CHANA, I., AND CLARKE, S. A survey on iot big data: Current status, 13 vs challenges, and future directions. *ACM Computing Surveys 53*, 6 (2020), 1–59.

[7] BELAY, M. A., BLAKSETH, S. S., RASHEED, A., AND ROSSI, P. S. Unsupervised anomaly detection for iot-based multivariate time series: Existing solutions, performance analysis and future directions. *Sensors 2844* (2023), 1–24.

[8] BERGMANN, P., BATZNER, K., SATTLEGGER, D., AND STEGER, C. The mvtec anomaly detection dataset: A compehensive real-world dataset for unsupervised anomaly detection. *International Journal of Computer Vision 129* (2020), 1038–1059.

[9] CHATTERJEE, A., AND AHMED, B. S. Iot anomaly detection methods and applications: A survey. *Internet of Things 19* (2022).

[10] CLAPPER, T. C. Proposing a new debrief checklist for teamstepps® to improve documentation and clinical debriefing. *SAGE Journals 47*, 6 (2016), 710–719.

[11] DEMEDEIROS, K., HENDAWI, A., AND ALVAREZ, M. A survey of ai-based anomaly detection in iot and sensor networks. *MDPI: Sensors 1352* (2023), 1–33.

[12] DYMORA, P., AND MAZUREK, M. Anomaly detection in iot communication network based on spectral analysis and hurst exponent. *MDPI: applied sciences 5319* (2019), 1–20.

[13] FAHIM, M., AND SILLITTI, A. Anomaly detection, analysis and prediction techniques in iot environment: A systematic literature review. *IEEE 7* (2019), 81664–81681.

[14] GOLLAPUDI, S. *Practical machine learning*. Packt Publishing, eBook, 2016.

[15] GUANSONG, P., CHUNHUA, S., LONGBING, C., AND HENGEL, A. V. D. Deep learning for anomaly detection: A review. *ACM Computing Surveys 54* (2021), 1–38.

[16] GUO, Y., JI, T., WANG, Q., YU, L., MIN, G., AND LI, P. Unsupervised anomaly detection in iot systems for smart cities. *IEEE Transactions on Network Science and Engineering 7*, 4 (2020), 2231–2242.

[17] GUZEL, M., KOK, I., AKAY, D., AND OZDEMIR, S. Anfis and deep learning based missing sensor data prediction in iot. *Wiley* (2019), 1–15.

[18] HABEEB, R. A. A., NASARUDDIN, F., GANI, A., AMANULLAH, M. A., HASHEM, I. A. T., AHMED, E., AND IMRAN, M. Clustering-based real-time anomaly detection a breakthrough in big data technologies. *Wiley 23*, 1 (2020), 1–27.

[19] HASAN, M., ISLAM, M., ZARIF, I. I., AND HASHEM, M. Attack and anomaly detection in iot sensors in iot sites using machine learning approaches. *Internet of Things 7* (2019), 1–14.

[20] Hendarko, T., Indriyanto, S., and Maulana, F. Determination of uav pre-flight checklist for flight test purpose using quantitative failure analysis. . *IOP conference series. Materials Science and Engineering 352* (2018), 12007–12015.

[21] Hernandez, M., King, A., and Stewart, L. Catheter-associated urinary tract infection (cauti) prevention and nurses checklist documentation of their indwelling catheter management practices. *Nursing Praxis in New Zealand 35* (2019), 29–42.

[22] Huc, A., and Trcek, D. Anomaly detection in iot networks: From architectures to machine learning transparency. *IEEE* (2021), 1–10.

[23] Islam, M., Dukyil, A. S., Alyahya, S., and Habib, S. An iot enable anomaly detection system for smart city surveillance. *MDPI: Sensors 2358* (2023), 1–14.

[24] Jiang, J., Liu, F., Liu, Y., Tang, Q., Wang, B., Zhong, G., and Wang, W. A dynamic ensemble algorithm for anomaly detection in iot imbalanced data streams. *Computer Communications 194* (2022), 205–257.

[25] Krishnamurthi, R., Kumar, A., Gopinathan, D., Nayyar, A., and Qureshi, B. An overview of iot sensor data processing fusion, and analysis techniques. *MDPI: Sensors 6076* (2020), 1–23.

[26] Kufel, J., Bargie-aczek, K., Kocot, S., Kozlik, M., Bartnikowska, W., Janik, M., ukasz Czogalik, Dudek, P., Magiera, M., Lis, A., Paszkiewicz, I., Nawrat, Z., Cebula, M., and Gruszczynska, K. What is machine learning, artificial neural networks and deep learning? examples of practical applications in medicine. *MDPI: Diagnostics 13* (2023), 2582–2604.

[27] Li, J. Using flowchart to help students learn basic circuit theories quickly. *MDP Sustainability* (2022), 1–12.

[28] Linardatos, P., Papastefanopoulos, V., Panagiotakopoulos, T., and Kotsiantis, S. CO_2 concentration forecasting in smart cities using a hybrid arima-tft model on multivariate time series iot data. *Scientific reports 13*, 17266 (2023), 1–23.

[29] MANIMURUGAN, S. Iot-fog cloud model for anomaly detection using improved naïve bayes and principal component analysis. *Journal of Ambient Intelligence and Humanized Computing* (2020), 1–10.

[30] MARCH, S. T., AND SMITH, G. F. Design and natural science research on information technology. *Decision Support Systems 15* (1995), 251–266.

[31] MARJANI, M., NASARUDDIN, F., GANI, A., KARIM, A., HASHEM, I. A. T., SIDDIQA, A., AND YAQOOB, I. Big iot data analytics: Architecture, opportunities, and open research challenges. *IEEE 5* (2016), 5247–5261.

[32] MIDDLE, S. A documentation checklist for (linked) humanities data. *International Journal of Digital Humanities 1*, 1 (2023), 1–19.

[33] MIKE, S., AND ANDREA, G. *Case Study: The Mortgage Advisor: Process Flow Diagrams*. eBook Academic Collection - Worldwide, eBook, 2016.

[34] MOHAMUDALLY, N., AND PEERMAMODE-MOHABOOB, M. Building an anomaly detection engine (ade) for iot smart applications. *Procedia Computer Science 134* (2018), 10–17.

[35] PALACIO-NINO, J.-O., AND BERZAL, F. Evaluation metrics for unsupervised learning algorithms. URL `https://arxiv.org/pdf/1905.05667.pdf`, viitattu 23.10.2023.

[36] PASSERA, S. Flowcharts, swimlanes, and timelines: Alternatives to prose in communicating legal-bureaucratic instructions to civil servants. *Journal of Business and Technical Communications 32* (2018), 229–272.

[37] PENG, D., BRUZZONE, L., ZHANG, Y., GUAN, H., DING, H., AND HUANG, X. Semicdnet: A semisupervised convolutional neural network for change detection in high resolution remote-sensing images. *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING 59* (2021), 5891–5906.

[38] PENG, Z., WENZHANG, D., AND HUAPING, L. Flowcharts, swimlanes, and timelines: Alternatives to prose in communicating legal-bureaucratic instructions to civil servants. *Scientific reports* (2023), 1–14.

[39] PROTOGEROU, A., PAPADOPOULOS, S., DROSOU, A., TZOVARAS, D., AND REFANIDIS, I. A graph neural network method for distributed anomaly detection in iot. *Evolving Systems 12* (2019), 19–36.

[40] Ray, P. P., and Dash, D. Iot-edge anomaly detection for covariate shifted and point time series health data. *Journal of King Saud University Computer and Information Sciences 34* (2022), 9608–9621.

[41] Sankaranarayanan, S., Swaminathan, G., Radhakrishnan, T., and Sivakumaran, N. Missing data estimation and iot-based flyby monitoring of a water distribution system: Conceptual and experimental validation. *Wiley* (2019), 1–10.

[42] Sgueglia, A., Sorbo, A. D., Visaggio, C. A., and Canfora, G. A systematic literature review of iot time series anomaly detection solutions. *Future Generation Computer Systems 134* (2022), 170–186.

[43] Shanmuganathan, V., and Suresh, A. Lstm-markov based efficient anomaly detection algorithm for iot environment. *Applied Soft Computing 136* (2023), 1–11.

[44] Sivapalan, G., Nundy, K. K., Dev, S., Cardiff, B., and John, D. Anneet: A lightweight neural network for ecg anomaly detection in iot edge sensors. *IEEE Transactions on Biomedical Circuits and Systems 16* (2022), 24–35.

[45] Tien, C.-W., Huang, T.-Y., Chen, P.-C., and Wang, J.-H. Using autoencoders for anomaly detection and transfer learning in iot. *MDPI: Computers 10* (2021), 1–14.

[46] Vangipuram, R., Gunupudi, R. K., Puligadda, V. K., and Vinjamuri, J. A machine learning approach for imputation and anomaly detection in iot environment. *Wiley 37* (2020), 1–16.

[47] Vedavalli, P., and Ch, D. A deep learning based data recovery approach for missing and erroneous data of iot nodes. *MDPI: Sensors 23* (2023), 1–16.

[48] Vigoya, L., Pardal, A., Fernandez, D., and Carneiro, V. Application of machine learning algorithms for the validation of a new coap-iot anomaly detection dataset. *MDPI: Applied Sciences 4482* (2023), 1–25.

[49] Vu, L., Cao, V. L., Nguyen, Q. U., Nguyen, D. N., Hoang, D. T., and Dutkiewicz, E. Learning latent representation for iot anomaly detection. *IEEE Transactions on Cybernetics 52*, 5 (2022), 3769–3782.

[50] WEINGER, B., KIM, J., SIM, A., NAKASHIMA, M., MUSTAFA, N., AND WU, K. J. Enhancing iot anomaly detection performance for federated learning. *Digital Communications and Networks 8* (2022), 314–323.

[51] XIANG, Y., XIANFEI, Y., QINGJI, T., CHUN, S., AND ZHIHAN, L. An edge computing based anomaly detection method in iot industrial sustainability. *Applied Soft Computing 128* (2022), 1–9.

[52] YIN, C., ZHANG, S., WANG, J., AND XIONG, N. N. Anomaly detection based on convolutional recurrent autoencoder for iot time series. *IEEE Transactions on Systems, Man, and Cybernetics: Systems 52* (2022), 112–122.

[53] ZENGYU, D., GANG, M., SALVATORE, C., YIXUAN, L., AND NENGXIONG, X. Comparison of estimating missing values in iot time series data using different interpolation algorithms. *International Journal of Parallel Programming 48* (2018), 534–548.

[54] ZHANG, R., CHEN, J., SONG, Y., SHAN, W., CHEN, P., AND XIA, Y. An effective transformation-encoding-attention framework for multivariete time series anomaly detection in iot environment. *Mobile Networks and Applications* (2023), 1–13.

A Appendix 1: Procedure diagram.

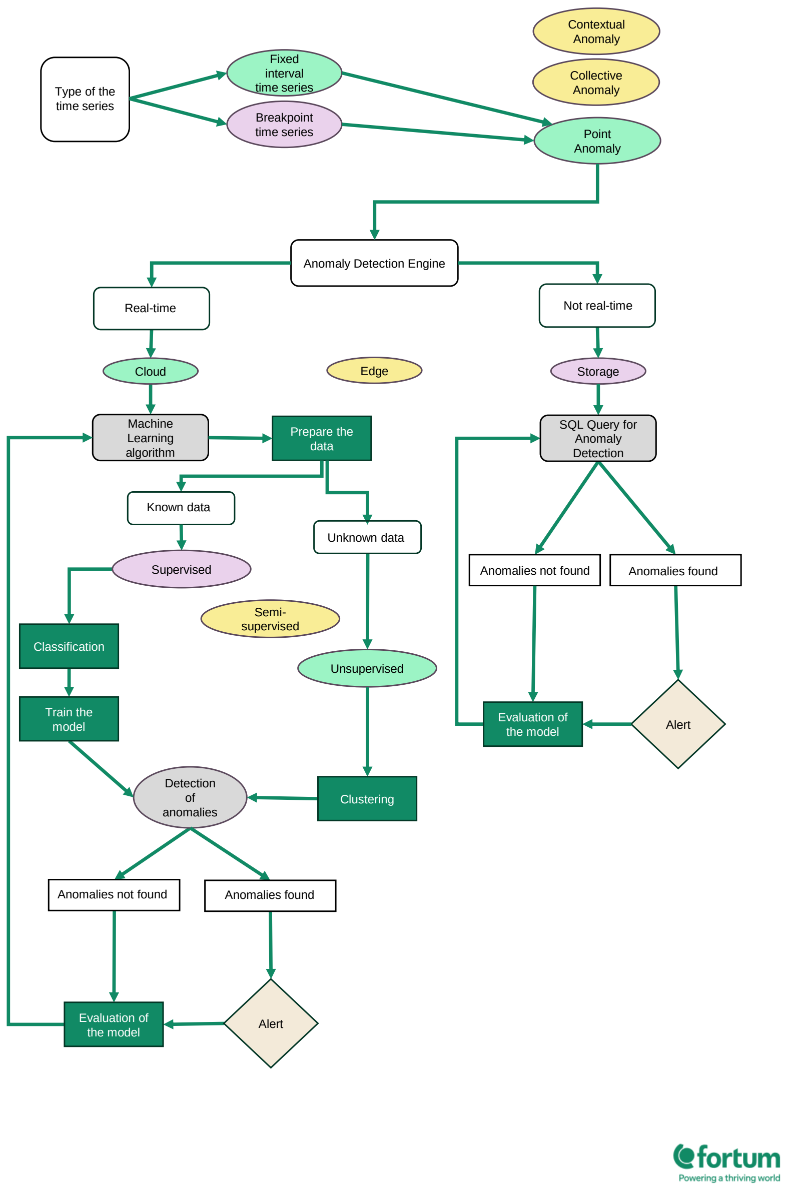
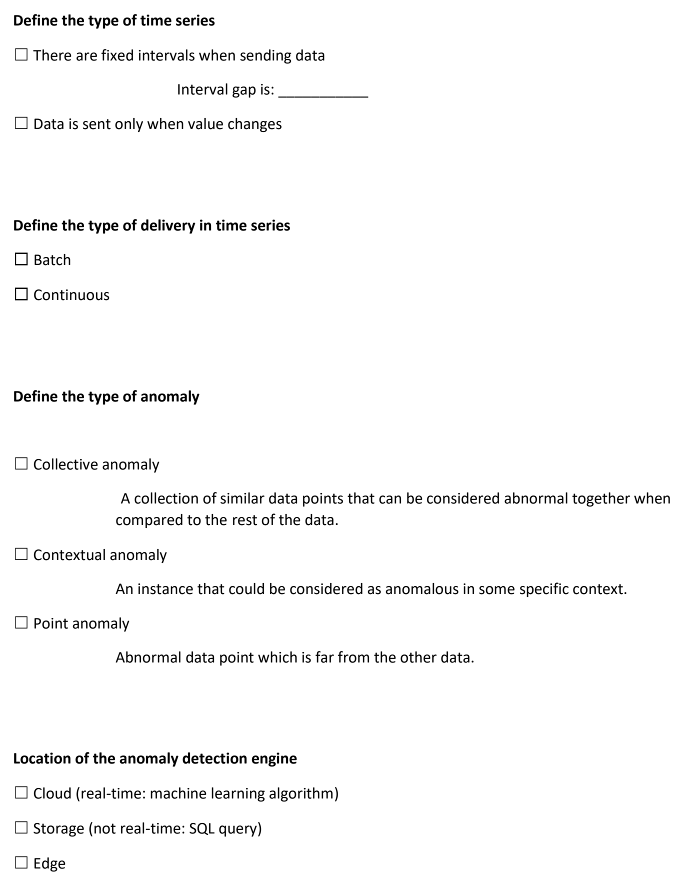


Figure A.1: Procedure Diagram

B Appendix 2: Checklist.



Define the type of time series

☐ There are fixed intervals when sending data

Interval gap is: ___________

☐ Data is sent only when value changes

Define the type of delivery in time series

☐ Batch

☐ Continuous

Define the type of anomaly

☐ Collective anomaly

A collection of similar data points that can be considered abnormal together when compared to the rest of the data.

☐ Contextual anomaly

An instance that could be considered as anomalous in some specific context.

☐ Point anomaly

Abnormal data point which is far from the other data.

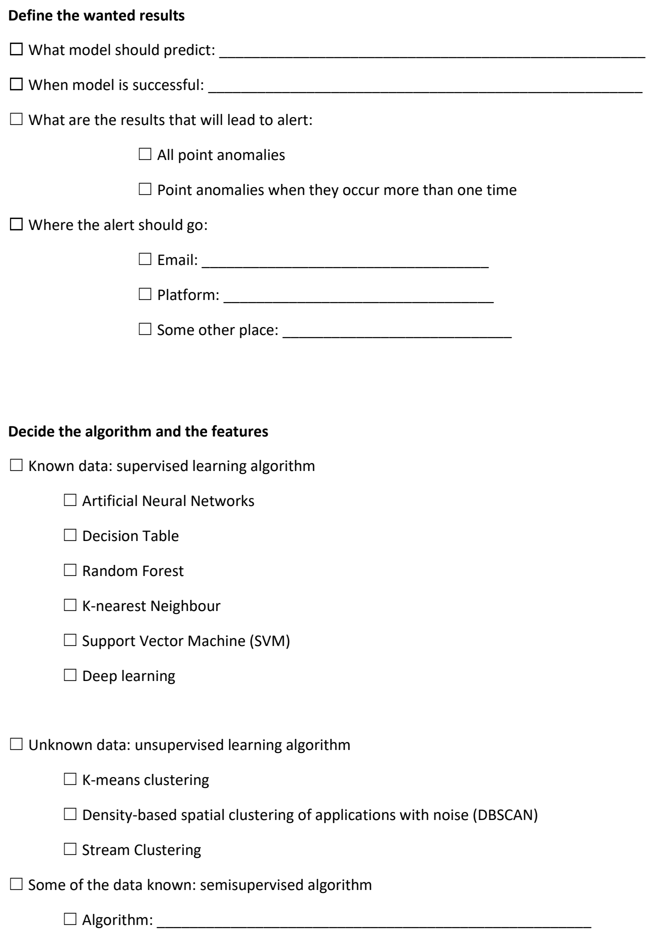
Location of the anomaly detection engine

☐ Cloud (real-time: machine learning algorithm)

☐ Storage (not real-time: SQL query)

☐ Edge

Figure B.1: Checklist page 1.



Define the wanted results

☐ What model should predict: ____

☐ When model is successful: ____

☐ What are the results that will lead to alert:

- ☐ All point anomalies
- ☐ Point anomalies when they occur more than one time

☐ Where the alert should go:

- ☐ Email: ______________________________
- ☐ Platform: ____________________________
- ☐ Some other place: ________________________

Decide the algorithm and the features

☐ Known data: supervised learning algorithm

- ☐ Artificial Neural Networks
- ☐ Decision Table
- ☐ Random Forest
- ☐ K-nearest Neighbour
- ☐ Support Vector Machine (SVM)
- ☐ Deep learning

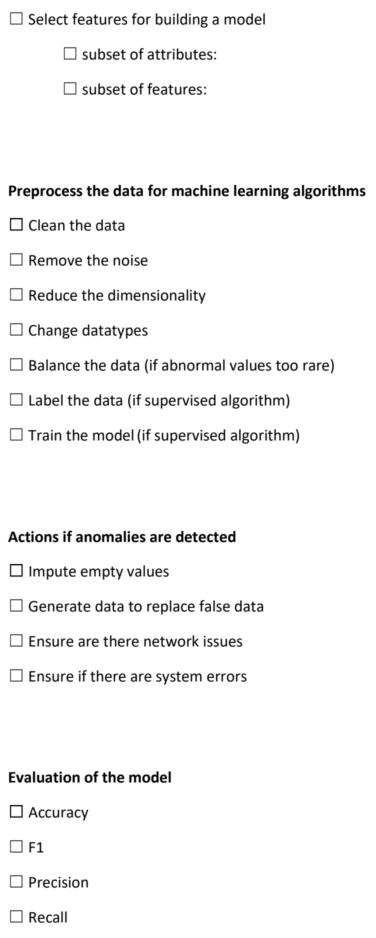
☐ Unknown data: unsupervised learning algorithm

- ☐ K-means clustering
- ☐ Density-based spatial clustering of applications with noise (DBSCAN)
- ☐ Stream Clustering

☐ Some of the data known: semisupervised algorithm

- ☐ Algorithm: ____

Figure B.2: Checklist page 2.



☐ Select features for building a model

- ☐ subset of attributes:
- ☐ subset of features:

Preprocess the data for machine learning algorithms

☐ Clean the data

☐ Remove the noise

☐ Reduce the dimensionality

☐ Change datatypes

☐ Balance the data (if abnormal values too rare)

☐ Label the data (if supervised algorithm)

☐ Train the model (if supervised algorithm)

Actions if anomalies are detected

☐ Impute empty values

☐ Generate data to replace false data

☐ Ensure are there network issues

☐ Ensure if there are system errors

Evaluation of the model

☐ Accuracy

☐ F1

☐ Precision

☐ Recall

Figure B.3: Checklist page 3.