Titre: Title:	Development of a Method for Anomaly Detection in Time Series Applied to Vehicle Monitoring
Auteur: Author:	Pablo Garcia Vega
Date:	2022
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Garcia Vega, P. (2022). Development of a Method for Anomaly Detection in Time Series Applied to Vehicle Monitoring [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10478/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/10478/
Directeurs de recherche: Advisors:	Bruno Agard, & Nicolas Saunier
Programme: Program:	Maîtrise recherche en génie industriel

POLYTECHNIQUE MONTRÉAL

affiliée à l’Université de Montréal

Development of a Method for Anomaly Detection in Time Series Applied to Vehicle Monitoring

PABLO GARCIA VEGA

Département de mathématiques et de génie industriel

Mémoire présenté en vue de l’obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie industriel

Août 2022

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Development of a Method for Anomaly Detection in Time Series Applied to Vehicle Monitoring

présenté par **Pablo GARCIA VEGA**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Michel GAMACHE, président
Bruno AGARD, membre et directeur de recherche
Nicolas SAUNIER, membre et codirecteur de recherche
Souheil-Antoine TAHAN, membre

DEDICATION

To my family and loved ones. We will climb up to the summits because we embrace the challenges.

ACKNOWLEDGEMENTS

I want to thank my director Bruno Agard and codirector Nicolas Saunier for giving me the opportunity to work in this project and for their trust and help throughout the process. I am very grateful to my colleges and friends of the Data Intelligence Lab (LID) who have been an inspiration and a great support to me.

RÉSUMÉ

L'objectif de ce projet a été de développer une méthode de détection des anomalies dans les séries temporelles, applicable à la surveillance des véhicules routiers. La détection de comportements anormaux dans la conduite des véhicules permet d'anticiper les problèmes mécaniques, les mauvaises pratiques et d'améliorer les performances des véhicules. Pour le présent projet, nous avons collaboré avec une entreprise montréalaise visant l'électrification partielle des camions. Les séries temporelles associées aux véhicules proviennent de la surveillance effectuée par leurs capteurs tout au long du processus de conduite. Ces informations sont stockées sous la forme de séries chronologiques multivariées.

La détection d'anomalies dans ces véhicules se heurte notamment à la présence de séries chronologiques multivariées, à la subjectivité de la définition de l'anomalie et au manque d’ensembles de données étiquetées.

Les techniques étudiées dans la littérature ont été classées en méthodes probabilistes, basées sur la distance, basées sur un modèle, basées sur la fréquence et basées sur la théorie de l'information. À partir de ces approches, une méthodologie itérative a été développée. Elle se compose de cinq étapes, brièvement décrites ci-dessous :

La première étape est appelée prétraitement. Ici, il faut s'assurer que les données d'entrée sont dans un format approprié. Certaines opérations de filtrage ou de normalisation sont effectuées à ce stade. La deuxième étape vise à extraire des sous-séquences de la série temporelle. Ces sous-séquences sont obtenues en recherchant les changements dans la dynamique de la série multivariée. Dans la troisième étape, un score d'anomalie est calculé pour chaque sous-séquence et un seuil d'anomalie est défini. À l'issue de la troisième étape, un lot de sous-séquences extraites de la série originale est défini comme des anomalies candidates. Les candidats sont regroupés en catégories en fonction de leur comportement dynamique dans la quatrième étape et la cinquième étape consiste à valider les catégories de candidats précédemment crées.

Cette méthodologie est dite itérative car les résultats de chaque itération sont utilisés pour le calibrage des paramètres de chaque étape, ainsi que pour identifier à chaque fois des anomalies différentes. La méthode décrite est appliquée dans un cas réel en utilisant les données d'un partenaire industriel. Ce partenaire installe des modules « Stop Start » (SSM) dans des camions afin de réduire leur consommation de carburant. Lorsque le moteur du camion est arrêté, la batterie intégrée au SSM est chargée d'alimenter les systèmes auxiliaires du camion (chariot élévateur, climatisation, etc.) et elle aide également à redémarrer le camion. Actuellement, la détection des anomalies est effectuée visuellement par l'équipement de diagnostic. La valeur perçue par le partenaire industriel consiste à automatiser la détection et le diagnostic des anomalies.

Dans le cadre du présent projet, les données recueillies par le SSM pour l'une des flottes de camions ont été utilisées. La base de données initiale contient un total de 206 923 observations de 279 variables différentes pour un camion. Au total, trois itérations de la méthode sont effectuées.

Au cours de la première itération, l'extraction des sous-séquences est effectuée en utilisant la série temporelle de la position de l'accélérateur comme référence. Le score d'anomalie est calculé en mettant en relation les courbes de vitesse et d'accélération du véhicule. Les candidats identifiés sont les séquences pour lesquelles la relation entre l'accélération et la vitesse est inhabituelle. Un total de 19 séquences anormales sont validées par le partenaire industriel dans cette itération.

La deuxième itération utilise une variable catégorielle, liée à l'état du SSM pour l'extraction des sous-séquences. En utilisant le même score d'anomalie que lors de la première itération, on obtient 82 anomalies validées par des experts. En outre, un patron d'anomalie est découvert dans cette itération, qui est utilisé dans l'itération suivante.

La troisième itération utilise les mêmes sous-séquences extraites lors de la deuxième itération, mais le score d'anomalie est lié à la distance par rapport au patron anormal détecté lors de la deuxième itération. Ainsi, après classification et validation, 103 sous-séquences sont validées comme anomalies.

Enfin, on obtient ainsi après trois itérations, en éliminant les anomalies qui se chevauchent entre les itérations, 122 séquences validées comme anomalies, 13 sous-séquences qui nécessitent une analyse plus approfondie pour être validées et 125 sous-séquences rejetées comme anomalies. Un outil visuel a été développé pour aider l'équipe de diagnostic lors du processus manuel de détection des anomalies.

La méthode employée s'est avérée efficace pour la détection des anomalies dans le domaine de la surveillance des véhicules. Une attention particulière a été accordée pour faciliter l'interprétation des résultats par l'équipe de diagnostic du partenaire. Cependant, elle présente des limites liées à

l'évolutivité, en raison de l'intervention humaine pour l'étape de validation, et à la mesure des performances, en raison du manque de données étiquetées. À l'avenir, l'incorporation de méthodes plus complexes et automatiques sera une option intéressante pour détecter de nouvelles anomalies, de même que le passage à un modèle hors ligne, tel que le modèle actuel, à un modèle en ligne dans lequel les données télémétriques des véhicules peuvent être analysées en temps réel.

ABSTRACT

The objective of this project was to develop a method for detecting anomalies in time series, applicable to vehicle monitoring. The detection of abnormal behaviors in the driving of vehicles allows to anticipate mechanical problems, bad practices and to improve the performance of vehicles. For the present project, we collaborated with a Montreal-based company aiming at the partial electrification of trucks. The time series associated with the vehicles come from the monitoring performed by their sensors throughout the driving process. This information is stored in the form of multivariate time series time series. Detecting anomalies in these vehicles is hampered by the presence of time series with multiple variates, the subjectivity of the definition of anomaly and the lack of labeled databases.

The search of the literature provided several perspectives on the problem of anomaly detection in time series. The techniques studied were classified into probabilistic, distance-based, model-based, frequency-based, and information theory-based methods. From the studied techniques, an iterative methodology was developed. The first step, preprocessing, aims at clean, filter and normalize the data. The whole time series is segmented in subsequences taking into account the dynamical changes of the time series in the second stage of the method. Then, for each subsequence, an anomaly score is calculated and, by defining an anomaly score threshold, some candidates are identified as potential anomalies. Candidates are grouped according to their dynamic behavior in the fourth step to then proceed to validation by the industrial experts.

Finally, the results after three iterations, eliminating overlapping anomalies between iterations, are 122 sequences validated as anomalies, 13 subsequences that require further analysis to be validated, and 125 subsequences rejected as anomalies. A visual tool was developed to assist the diagnostic team in the manual anomaly detection process.

The method used has proven to be effective for anomaly detection in the vehicle monitoring domain. Special attention was given to facilitate the interpretation of the results by the diagnostic equipment. However, limitations related to scalability and performance measurement have been identified. In the future, it will be of interesting to incorporating more complex and less interpretable methods will be for detecting new anomalies and move from offline to an online model capable of processing data in real time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ARIMA	Autoregresive Integrated Moving Average
ARMA	Autoregresive Moving Average
BV	Basic variable
CNN	Convolutional Neural Networks
DTW	Dynamical time wrapping
DTW	Dynamical Time Wraping
GMM	Gaussian Mixture Models
HMM	Hidden Markov Models
k-NN	k-Nearest Neighbours
LCSS	Longest Common Subsequence
LD	Levenshtein distance
LSTM	Long Short-Term Memory
NN	Nearest Neighbours
PCA	Principal Components Analysis
PIP	Perceptually Important Points
PLR	Piecewise Linear Representation
SAX	Symbolic Aggregate aproXimation
SSM	Stop-Start module
TSUA	Time Series Unit of Analysis

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

Repair and maintenance costs have a big impact in the automotive industry. Its global market accounted for US$565 billion in 2020 and is expected to reach US$678.4 billion in 2026 (Linker, 2022). Regarding truck fleets, maintenance costs represent more than 9% of total truck operation cost per mile in United States according to (McNally, 2022). The emerging technologies related to the Internet of Things are transforming the industry. Advances in data collection, cloud storage and accessibility are helping to make the shift into data driven fleet management. Smart vehicle management, remote diagnostics and predictive maintenance help to decrease maintenance costs of the fleets while improving the service and safety during operations.

Many times, data regarding vehicles' operation is stored in form of time series. The time series typically contain information about variables like acceleration, speed, oil temperature, etc. Meaningful insights can be extracted from the analysis of these time series of telemetry data. For instance, detection of anomalous operation behaviours and repetitive patterns are of great importance when it comes to identify, diagnose, and prevent faults in the vehicles.

The case study concerning this document is performed in collaboration with a small enterprise that installs stop-start modules (SSMs) in fleets of trucks. The module is installed in already in-operation trucks in order to save fuel. Truck monitoring data is recorded by the SSM and used by our partner to detect failures in their product, anomalous behaviour of the trucks and improve the functioning of the SSM. Like other small enterprises in the sector which do not have enough maturity or resources, the detection and diagnosis of anomalies is performed by visual inspection of telemetry data. This visual inspection often leads to slow wait times to get a proper diagnosis and a waste of human resources. More automatized methods involve time series data mining techniques. Hence, the main objective of the present project is to develop a method for anomaly detection that can be applied to vehicle monitoring, in order to support industrial experts in the detection and diagnosis of anomalies. Furthermore, the implementation of an automatized method for anomaly detection may lead into detection of anomalies that visual inspection cannot yet detect.

There are multiple methods in the literature addressing anomaly detection in time series. However, not all of them are suitable for vehicle monitoring applications. Challenges regarding the deployment of automated anomaly detection algorithms involve the lack of databases with labeled anomalies, the subjectivity in the definition of anomaly and the high complexity of some anomaly detection algorithms.

The proposed method addresses these challenges through a close collaboration with the industrial partner. Because the scope is to support experts in the diagnosis of anomalous behaviours, special attention is put in interpretability of the methods and algorithms from a mechanical and electrical point of view. In all, a compromise between the automatization of anomaly detection, effectiveness and ease of use needs to be achieved.

The rest of the document is organized as follows: In Chapter 2, a literature review will be presented covering the state of art of anomaly detection and its applications. A detailed explanation of the problematic will be presented in Chapter 3. Chapter 4 will present the proposed method for anomaly detection and its application to vehicle monitoring will be studied in Chapter 5. Finally, Chapter 6 will draw the conclusion of the project, expose its limitations and future work.

CHAPTER 2 LITTERATURE REVIEW

Temporal data represents the state of a system for a determined time. We are continuously collecting data: weather conditions, price of stocks, sensors readings, loggings in web servers, etc. This data is often stored in the form of time series. Time series analysis is a branch of data mining that seeks to find meaningful insights such as pattern discovery, classification, rule discovery, forecasting and anomaly detection (Fu, 2011). In the present literature review, anomaly detection in time series is addressed.

Anomaly detection in time series has been researched for a wide range of applications such as healthcare (Li *et al.*, 2021), information technologies (Cook *et al.*, 2019), text mining (Gomes *et al.*, 2014), stock analysis (Gupta *et al.*, 2014) (Fu-Lai *et al.*, 2004) and traffic management (Bawaneh *et al.*, 2019) among others. In the following lines, a brief introduction of anomaly detection is presented where key concepts are defined, an outlook of the actual methods is then proposed together with their main applications. A deeper study on vehicle monitoring applications is performed to put into context the case study presented in Chapter 4.

2.1 Definitions

The following definitions will be used in the remaining of the document.

A *time series* can be defined as a set of *n* records $X = \{x_t\}_{t \in T}$ that have been observed at a specific time $t \in T$. Commonly, each time instant where an observation is recorded is named time stamp. The values recorded in the series can be categorical (e.g., seasonal crops rotation) or numeric (e.g., Yearly Gross Domestic Product of Canada) (Hamilton, 2020).

A *subsequence S* is a contiguous set of *m* points $x_t, x_{t+1}, \dots, x_{t+m-1}$ extracted from a time series $X = \{x_t\}_{t \in T}$ of length $n > m$.

2.1.1 Dimensionality

Regarding the number of variables recorded, a time series $X = \{x_t\}_{t \in T}$ can be univariate or multivariate:

- *Univariate time series* consist of a single observation being recorded each time (e.g., data recorded from a temperature sensor over the time). In univariate time series x_t represents a measured value of a variable.

- *Multivariate time series* consists of multiple variables recorded each time stamp (e.g., sensors for temperature, flow and rotation speed all recorded over the same period of time). In multivariate time series, x_t represents a vector of k observations $x_t = (x_{t,0}, x_{t,1}, \dots, x_{t,k-1})$. In general, multivariate time series analysis is more complex to analyse than univariate and it usually requires transformations in data (Blázquez-García *et al.*, 2021).

- A *channel*, in the context of vehicle monitoring, refers to each variable recorded by the vehicle in form of time series. Engine speed, acceleration position, oil temperature are channels, or variables, from the multivariate time series representing the state of the vehicle at each observation.

2.1.2 Representation

Time series representation is another important aspect to bear in mind for anomaly detection because the way a time series is represented is going to affect the kind of algorithms that can be used for the analysis. Figure 2.1 shows different ways of representing the same time series. On the left, the raw time series is a represented by plotting the values observed at each time instant. In Figure 2.1b, an example of Piecewise Linear Representation (PLR) is showed. In few words, PLR divides a given time series into a finite number of straight segments.

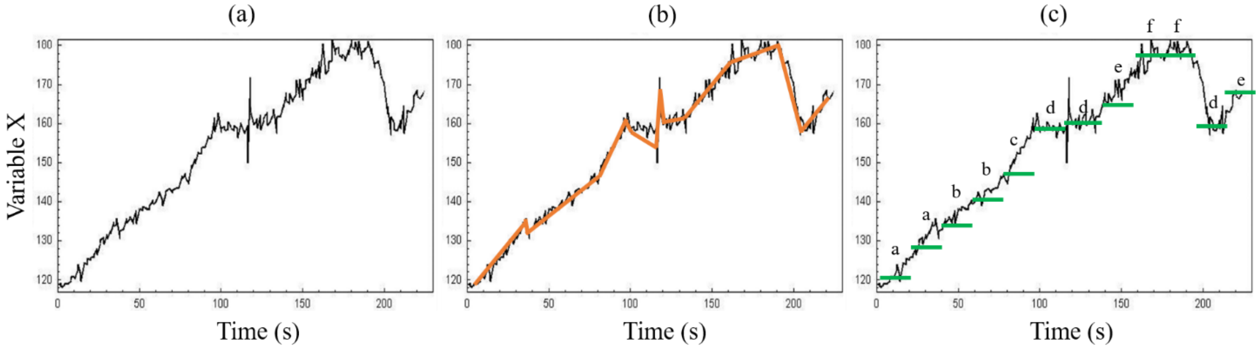


Figure 2.1 Different representations of a time series

The number of segments is determined by a maximum error calculated as the difference between a segment and the real curve. PLR is frequently used in time series segmentation because it reduces computation, storage and transmission costs while supports techniques for clustering and change point detection. (Keogh *et al.*, 2001).

Figure 2.1c shows the Piecewise Aggregate Aproximation (PAA) representation which is a technique consisting in reducing the dimensionality of the time series from n observations to w segments of equal length and constant value. The value of each segment is calculated as the mean of the x_t observations falling into it (Guo *et al.*, 2010).

Symbolic representation of time series is another technique usually employed to convert a numerical time series into a categorical one. An example of it is the Symbolic Aggregate approXimation (SAX). SAX uses the PAA representation to create an alphabet in which each segment is associated to a symbol (Gomes *et al.*, 2014). Hence, with the SAX approach, time series are represented as *words,* which are sequences of symbols (letters). In the Figure 2.1c, the raw time series has been transformed in the *word* (aabbcddeffde).

2.1.3 Anomaly detection

Anomaly detection has been studied for decades by many authors for different domains. Hence, there is not a universal definition of what an anomaly is. Little nuances are added by authors depending on the application. A classical definition of an outlier is given by (Grubbs, 1969) who treated an anomaly as an observation appearing to deviate from the other members of the set. A slightly similar definition is given in (Barnett *et al.*, 1979), where an outlier is defined as an observation (or subset of observations) looking inconsistent with the remainder of that set of data. Other definitions of anomalies include (Aggarwal *et al.*, 2001) who treated them as noise points lying outside a set of defined clusters. Other names given by authors to anomaly detection are outlier detection and novelty detection. Although novelty detection is more focused in data differing from the training set, it has practically used in literature as a synonym of anomaly detection (Pimentel *et al.*, 2014).

2.1.4 Type of anomaly in time series

An important aspect that the definition of Barnett *et al.* (1979) brings is the distinction of punctual and subsequent anomalies. The type of anomaly trying to be detected affects data preprocessing and determines the kind of algorithm to be used.

Punctual anomalies are defined as an aberrant point at a specific timestamp of the timeseries. As it is illustrated in the left side of Figure 2.2. Punctual or point anomalies are close to the definition provided by (Grubbs, 1969).

Subsequent anomalies are defined as a succession of values, a subsequence, whose collective behavior is abnormal. As showed in the right part of Figure 2.2, although each point by itself is not necessarily an anomaly, their joint comportment is aberrant (Cook *et al.*, 2019).

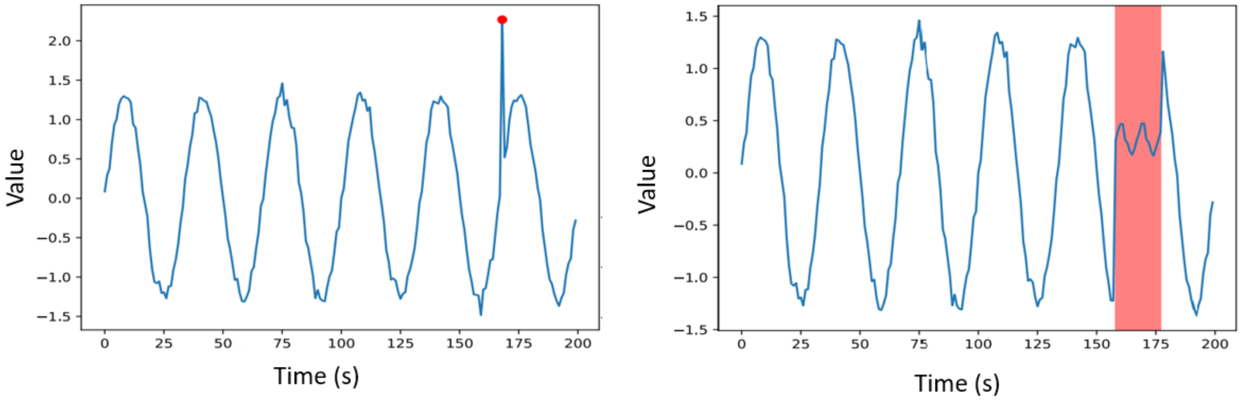


Figure 2.2 Punctual and subsequent anomalies (Lai *et al.*, 2021)

2.1.5 Degree of labeling

Another significant aspect of anomalies is whether they are labeled or not. Labeled anomalies are those that have already been identified as such and this information is included in the data set. Supervised machine learning techniques can be used with labeled data. When anomalies are not labeled, but there is a set of normal data identified, then it is considered as a semi-supervised process. With completely unlabeled data only unsupervised tools can be used. Unfortunately, anomalies in a data set are often not labeled and manual labeling or unsupervised learning algorithms are required (Hodge *et al.*, 2004).

2.2 Anomaly detection methods for time series

Approaches for anomaly detection in time series have been classified by authors into multivariable or univariate methods (Cook *et al.*, 2019), supervised, semi-supervised or unsupervised techniques (Hodge *et al.*, 2004), punctual or subsequent anomalies (Blázquez-García *et al.*, 2021) among others. The classification of methods in the present literature review are based on the works in novelty detection by (Pimentel *et al.*, 2014) and the review in outlier detection by (Blázquez-García *et al.*, 2021). The methods covered in this literature review are divided in the following categories: probabilistic, distance-based, model-based, frequency-based and information-theoretic.

2.2.1 Probabilistic methods

Probabilistic methods assume that observations can fit into a certain distribution. This approach is based on the idea of constructing a model of the data with a probability distribution and then defining a threshold of normality. Observations are compared to the model and, if they exceed the threshold, they will be considered as an anomaly.

Simple probabilistic methods include the box-plot graphic for numerical data. This method identifies the data distribution by 5 quantities: the minimum (min), first quartile (Q1), median (Q2), third quartile (Q3) and the maximum (max). The interquartile range (IQR) is defined as $IQR = Q3 - Q1$, minimum and maximum are then calculated as $\min = Q1 - c * IQR$ and $\max = Q3 + c * IQR$, where the coefficient c depends on each case but often 1.5 is chosen. Observations out of the interval (min, max) are considered as outliers. Some modifications of the box-plot technique are included in (Sun *et al.*, 2012), where this method is used for outlier detection and visualization of climate data.

Models for more complex distributions in which the data have several centers of high population density are parametric mixed models. Mixed distributions commonly used are gamma (Stein *et al.*, 2002), Poisson (Benkabou *et al.*, 2021), Student’s t (Grzesiek *et al.*, 2020) and Weibull (Aremu *et al.*, 2019), although the most popular is the Gaussian Mixed Models (GMM) (Mouret *et al.*, 2022). GMM’s popularity might be because they are more effective when little information is known a priori about the distribution of the elements. For each of the models, the parameters of the distribution are estimated from a dataset considered as normal. An anomaly threshold k is then determined and the probability of an observation to be out of the normal zone is calculated. The main drawback of this approach is the mixture model is the difficulty to define the anomaly threshold and the number of components in the model (Cook *et al.*, 2019).

2.2.2 Distance-based methods

Distance-based techniques rely on the definition of distances between elements to detect the anomalous ones. It includes the approaches of nearest-neighbour and clustering analysis which are also used in classification of time series. The assumption is that “normal” data are tightly clustered, while novel data occur far from their nearest neighbours or the clusters.

Nearest neighbour approach assumes that normal objects are close to their neighbours and, when elements are further located from the normal set, they are considered as an anomaly. The distance can be computed to the nearest neighbor (NN) or an average of the *k* nearest neighbors (*k*-NN) (Chen *et al.*, 2020).

Time series clustering is defined as the process of partitioning a set of time series so that similar time series are grouped using a similarity measure. The idea behind clustering is that, by grouping similar objects, outliers will remain isolated. Clustering algorithms can be divided in partitioning, hierarchical, model-based, grid-based and density-based. The last two categories have been little used for anomaly detection of time series most probably due to their high complexity (Aghabozorgi *et al.*, 2015). Partitioning and hierarchical methods are addressed in the following lines while model-based algorithms are explained in next section.

Partitioning algorithms distribute the elements into *k* groups of at least one element per group. Each group typically has a center that minimises the distances between all elements of the group. The most famous and used algorithm thanks to its simplicity of application is k-means, which computes the center of the cluster as a mean of all elements belonging to that cluster (Yu *et al.*, 2018). Modifications of k-means include methods like k-medoids. Instead of calculating the center of the cluster as an average point between objects, k-medoids sets the center of the cluster as the object that minimizes the sum of distances to the rest of the objects in the cluster. Although computing times are longer, it is advantageous for having real objects representing each cluster (Niennattrakul *et al.*, 2007). Partitioning clustering requires to define the representative point of each cluster and the number of clusters, which can result in variable quality of clustering.

Hierarchical clustering aims at building a hierarchy of clusters. There are two approaches named bottom-up or agglomerative and top-down or divisive. Agglomerative clustering constructs the hierarchy starting from the single objects. These are grouped according to the closest elements and the hierarchy is built by grouping clusters until a single cluster containing all elements is reached. Divisive clustering starts with a unique cluster containing all the elements and proceeds to construct the hierarchy by subsequently splitting the clusters until clusters are formed of single elements (Roux, 2018). Divisive methods are used in (Islam *et al.*, 2018) for the clustering of anomalies within an internet network. Agglomerative clustering is used in (Spiegel *et al.*, 2011) for pattern recognition and identification of abnormal situations in car driving.

For both clustering and nearest neighbour approaches, a similarity measure must be defined. The choice of the similarity or distance measure depends on the type of data (categorical or numerical), the length of the time series (same length or variable length), the number of variables (univariate or multivariate) and the computational resources needed. One of the most common distance measures is the Euclidean distance that can be used for univariate and multivariate time series with continuous data. Another example very popular for *k*-NN is the Mahalanobis distance. It computes the distance from every object to the centroid of the rest of the points. It is more expensive in terms of computation than the Euclidean distance because each time it has to go through the entire set of data (Pimentel *et al.*, 2014). Other options for continuous data include shape-matching distances like Dynamical Time Wrapping (DTW) (Thuy *et al.*, 2021) and Longest Common Subsequence (LCSS) (Soleimani *et al.*, 2020), which are distance measures that take into account the shape of the time series, regardless of their lengths, and find the best match between the time stamps.

Distance measures for categorical data include the overlap distance, which simply calculates the number of time stamps where the categorical attribute is the same (Boriah *et al.*, 2008), the Hamming distance (Safaei *et al.*, 2020) and its modifications (Mihailović *et al.*, 2018). Not far from categorical distances between time series are the similarity measures used in text mining. As SAX is used for time series representation and analysis, textual similarity measures such as Levenshtein (Tamura *et al.*, 2016) are also employed for categorical data. Briefly, this distance measure transforms a string of characters into another by editing individual symbols. Although it is not compatible with numerical data, it allows the comparation of series with different lengths (Petitjean *et al.*, 2012).

2.2.3 Model-based methods

The intuition behind model-based approaches is to create a model out of the existing data and flag anomalies if there is a significant difference between the prediction calculated by the model and the actual value. Models for univariate time series such as Autoregressive Moving Average (ARMA) (Kadri *et al.*, 2016) and other modified versions like Autoregressive Integrated Moving Average (ARIMA) (Alizadeh *et al.*, 2021) have been used for detecting point anomalies in stationary time series. In few words, it consists in performing a regression with the last *k* values of the time series and the errors of the last *n* predictions. Other methods such as Long Short-Term Memory (LSTM) have also been used for univariate time series. This algorithm uses an autoencoder[1] to transform the original time series into a vector with reduced dimensions to then reconstruct the original time series from the lower dimension representation. When the error between the reconstructed and original time series goes beyond a certain threshold, the time stamp is considered an anomaly (Dong *et al.*, 2021).

Among other techniques aiming at detecting subsequent anomalies in univariate and multivariate time series we find Convolutional Neural Networks (CNN), which will predict not just the next point, but a subsequence of points based on a sequence of previous values. Generally, the longer the subsequence is, the less accurate the model will be because predictions of future values are based on previously predicted values. This technique developed for image processing has been adapted for time series forecasting (Munir *et al.*, 2018).

Model-based anomaly detection approaches for multivariable time series usually require dimensionality reductions. Principal Component Analysis (PCA) has been used in (Hoang *et al.*, 2018) and (Long *et al.*, 2020) as techniques to reduce the number of variables. Then clustering is

[1] An autoencoder is a type of neural network that encodes input data into a lower dimensional representation by training the network to just keep significant data and then return a reconstruction with the same input dimensions (Martín Abadi, 2015)

performed to isolate the anomalous objects, as explained in the previous section. An alternative to dimensionality reduction for multivariate time series are the algorithms derived from the Markov principles, which are known for their complexity. An example of them are graphical models used in (Cheng *et al.*, 2009) to detect and characterise noise in multivariate time series. Moreover, in (Li *et al.*, 2017) and (Fuse *et al.*, 2017), a Hidden Markov Model (HMM) models the time series as a succession of states and transitions. Each transition between states has an associated probability that is calculated with the training data. Given a sequence, there is a certain probability that this sequence belongs to the HMM model. If this probability is lower than a threshold, then this sequence is an anomaly.

2.2.4 Frequency-based methods

Other methods for detecting subsequent outliers are based on frequency. To these methods, an outlier is defined as a subsequence that appears less frequently than expected. Because of the difficulty of finding two time series with the exact same shape in real-values, frequency-based methods are normally used with discrete representations of the time series like SAX (Rasheed *et al.*, 2013).

To perform the frequency analysis of the subsequences, a subsequence extraction process needs to be done before. Algorithms for subsequence extraction are mainly based on fixed-window lengths. The most employed one is the sliding window, which consists in dividing the whole time series into overlapping windows with a fixed length and a fixed overlapping gap. Among others, (Rasheed *et al.*, 2013) uses a sliding window followed by frequency analysis to discover anomalous patterns in discretized time series. Using sliding window subsequence extraction (Li *et al.*, 2021) may also detect outliers with a clustering approach.

Few works have used a variable window length for subsequent extraction. An example is found in (Lu *et al.*, 2020), who proposes an algorithm that subdivides a multivariate time series into non-overlapping subsequences based on dynamical changes. Those changes can be seen as variations in operation regime, changes in the working conditions or faults. Another case of variable length window for subsequence extraction is given in (Spiegel *et al.*, 2011), where the whole time series is divided in subsequences cutting through critical points. Those critical points are defined as the local extrema of the series (local maximums and minimums).

2.2.5 Information-theoretic methods

These methods use measures such as entropy, conditional entropy, or other related measures to find the outliers. Entropy represents the amount of information contained in a data set, which is also related to the degree of randomness of the source of information. The more randomness in the data points within a data set, the higher its entropy is. In information-theoretic methods, entropy or other related measures are firstly calculated for the whole set of data. Then, the elements that, when calculating again the measures without them, induce a big difference, are considered outliers. Applications of information-theoretic approaches are seen in (Marchetti *et al.*, 2016). This work detects anomalous subsequences in in-vehicle network messages by calculating their entropy. By defining an anomaly threshold, they can identify attacks in the network. In addition to entropy, other measures have been employed. The *infomax* principle is used in (Ruff *et al.*, 2019) and (Hjelm *et al.*, 2018): it is a metric that combines entropy and conditional entropy and tries to maximize the mutual information shared by two variables.

2.3 Anomaly detection in vehicle monitoring

2.3.1 Vehicle monitoring and predictive maintenance

With the implementation of the Controller Area Network bus (CAN bus) protocol and On-Board Diagnostics (OBD) in the 1980s (Bortolami, 2020), communication between subsystems in vehicles and recording of driving data improved notoriously. This data has been used for daily maintenance operations, telemetry and constant improvement of vehicles design. In the following decades, predictive maintenance techniques pioneered by C.H. Waddington, started to gain importance as they prevented accidents and decreased maintenance costs (Eye, 2018).

Nowadays, with the rise of the Internet of Things (IoT), smart vehicles and automated driving, more and more sensors are collecting data from vehicle driving. This data can be recorded and transmitted in real time, which is highly valuable for manufacturers, fleet managers and telemetry technicians. Vehicle aftersales business is still an important business for vehicle manufacturers. In 2018, aftersales business (maintenance, repair, wear, etc.) accounted for a revenue of 0.9 billion dollars for the world's original equipment manufacturers, representing a quarter of their total profit (Deloitte., 2020). Analysis of recorded and real time data from vehicles monitoring is key to reduce maintenance costs, extend vehicles life and improve user experience. In addition to experts' knowledge, data mining techniques leverage the possibility of detecting anomalies and patterns that visual inspection cannot detect. The main applications of anomaly detection in time series data to vehicle monitoring are addressed in the following lines.

2.3.2 Applications in vehicle monitoring

2.3.2.1 Classification of operating modes

Vehicle operation monitoring and anomaly detection is addressed in (Alizadeh *et al.*, 2021). The proposed method aims at detecting anomalous behaviours in different operating states. In their work, they perform a subsequence extraction based on rules that identify the dynamical changes on vehicle operation to classify them into four operation states. ARIMA model is then used to predict fuel rate, transmission oil temperature and vehicle speed channels and detect anomalies based on the deviation between predictions and real values. Similar to the previous work, (Poteko *et al.*, 2021) characterize operation modes of a vehicle based on decision trees and vehicle speed time series data. However, none of the previous had a validation set of data to evaluate the performance of their classification algorithms.

2.3.2.2 CAN bus internal cyberattacks

Anomalies in CAN bus messages to prevent cyberattacks in in-vehicle networks are a common application of outlier detection in the vehicle industry. To do so, LSTM has been a popular technique, as we can see in the works of (Narayanan *et al.*, 2016), (Qin *et al.*, 2021), (Zhu *et al.*, 2019) and less common methods for CAN bus outlier detection include entropy based methods (Theissler, 2014) and HMM (Wang *et al.*, 2018).

2.3.2.3 Mechanical and sensor fault detections

Other works focus on the detection of faults in vehicle sensors. Applications for Unmanned Aerial Vehicle (UAV) sensors are seen in (Khan *et al.*, 2019), which uses Principal Component Analysis to reduce dimensionality and isolate the anomalies. Their results showed in PCA graphics clearly isolate anomalies, but the authors were concerned about the lack of interpretability to help the diagnosis phase.

In contrast to the previous work, labeled databases are used in (Van Wyk *et al.*, 2019) and (Theissler, 2017). The former extracts subsequences with a fixed length from multivariate time series coming from sensors of Automated Vehicles to then use a CNN and a Kalman filter to detect four types of known sensor failures. The latter uses multiple classifiers to identify mechanical faults using eight vehicle channels in various driving conditions. It is remarkable of their work that, rather than extracting segments to detect subsequent anomalies, they firstly identify punctual anomalies and then they incorporate a sequencer that aggregates them into subsequent anomalies.

2.3.2.4 Abnormal human behaviour detections

The work presented in (Zhang *et al.*, 2017) aims at detecting unusual driving behavior. It proceeds with a fixed window segmentation to then use a graphic-based anomaly detection technique following Markov principles. Contextual subsequent anomalies for univariate time series and correlational abnormal behaviours for multivariable time series are detected.

Similar works targeting human behaviour anomalies are presented in (Negi *et al.*, 2020) and (Spiegel *et al.*, 2011). The first one uses a LSTM for detecting anomalies within acceleration, speed, and steering channels while the second one searches for complex driving maneuver patterns by the clustering of subsequences. It is worth highlighting the importance of preprocessing in this publication. A Savitzky-Golay filter is used to eliminate the noise and smoothen sensor signal before their subsequent extraction.

2.4 Literature review conclusion

In this chapter, a study of the state of the art of anomaly detection methods in time series has been carried out. A wide range of algorithms have been reported for punctual and subsequence anomaly detection. Probabilistic methods are easy to implement and identify anomalies in an effective manner. However, their performance decreases when there is not enough data for the training or when there are too many variables in the data set. In addition, some knowledge about the data set is needed to model the distribution. Methods based on distance, on the other hand, are useful when there is no prior knowledge. Nevertheless, they require the definition of a distance measure which can result into expensive calculation when treating high dimensional time series.

Unlike distance-based methods, model-based methods do require a priori knowledge of the data set. Model-based methods can be used for both univariate and multivariate time series and have a high performance when processing sequential discrete data. However, model-based methods fail to be quite complex and often add a high opacity, especially when neural networks are involved. This results in a lack of human interpretability if a manual diagnosis phase is needed after anomaly identification. In addition, labelled data is also required for training the neural networks.

Frequency-based and information-theoretic algorithms have been used for univariate time series for both numerical and categorical data, but no works were found for multivariate time series. Instead, when treating multivariate time series, dimensionality reduction is widely employed in the previous cited methods. Although there are methods for transformed time series, dimensionality reduction still results into a lack of human interpretability. To solve this issue, the use of univariate techniques for multivariable problems is an interesting approach proposed in (Blázquez-García *et al.*, 2021). It avoids dimensionality reduction and helps the diagnosis phase.

Some challenges remaining in anomaly detection of time series are the selection of an anomaly threshold and the length of the windows for subsequence extraction. Regarding the setting of an anomaly threshold, the reviewed works mostly set it manually and, in some cases, the use of statistical distributions is seen. For the window selection, fixed window techniques are still the most used. Although variable length windows seem to better adapt to the characteristics of subsequent anomalies, there are still few examples of methods for subsequence extraction using them.

Regarding applications of anomaly detection in vehicle monitoring, they are focused in detecting operating modes, in-vehicle networks, sensor faults and abnormal human behaviour. No works were found detecting anomalies with both human and mechanical causes. In addition, there is a general lack of human interpretation of the results regarding the diagnosis phase of the anomalies.

In the present work, an anomaly detection method is proposed. It aims at identifying anomalies in multivariate time series without resorting dimensionality reduction. A special emphasis is given to the interpretation of the results and the diagnosis phase. An application of the method for vehicle monitoring is then presented aiming at detecting anomalies caused by human and mechanical misbehaviours. The particularity of the application is that anomalies are not conceived as punctual aberrant points or subsequences of points significantly differing from its context, as explained in literature. Instead, an anomaly is a subsequence of points in which the vehicle or the driver behaves in an unexpected way. Unexpected behaviours are not characterized by deviation from regular values in the time series, but by irregularities in the relation between variables. A more detailed explanation is presented in Chapter 4.

CHAPTER 3 ANOMALY DETECTION METHOD

In this chapter, the method for anomaly detection is presented, explaining at each stage the algorithms and metrics used, as well as the input and output variables. The method is initially conceived for non labeled datasets of multivariable time series. Since interpretability is a priority, field expert intervention is required at some stages. Figure 3.1 graphically represents the method followed.

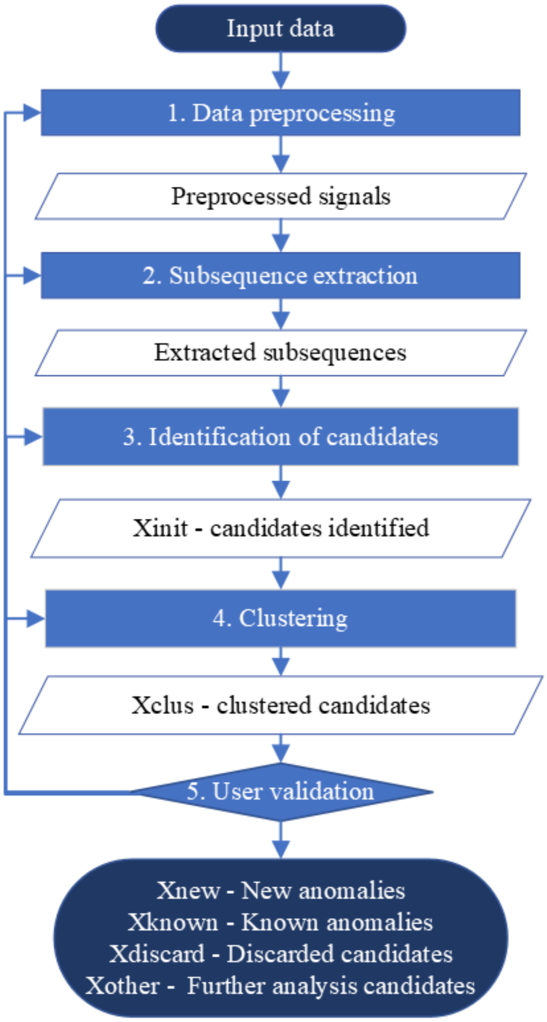

Input data
1. Data preprocessing
Preprocessed signals
2. Subsequence extraction
Extracted subsequences
3. Identification of candidates
Xinit - candidates identified
4. Clustering
Xclus - clustered candidates
5. User validation
Xnew - New anomalies
Xknown - Known anomalies
Xdiscard - Discarded candidates
Xother - Further analysis candidates

Figure 3.1 Method for anomaly detection

In few words, the method divides an entire time series into smaller subsequences. Then, for each subsequence, an anomaly score is calculated and used to identify potential candidates. All candidates are clustered and validated in the last two stages. It should be noted that it is an iterative method, where human expertise is involved. Results of stage 5 are used to improve the method and parameters in the previous stages in order to obtain better results, while discarding non relevant candidates. The proposed method is composed of five stages that will be addressed in the following lines: data preprocessing, subsequence extraction, identification of candidates, classification, and user validation.

3.1 Data preprocessing

Raw data often contains missing values, noise, and inconsistencies. Hence data preparation is usually the first step in any data mining application. Depending on the algorithms to be used and the goal of the project, data preprocessing tasks can be very different. For the interest of this project, elimination of outliers and inconsistencies in raw data are not part of the preprocessing tasks because those outliers can in fact be the anomalies that we try to detect. Instead, for the first step of the method, the focus is on data selection, filtering, and normalization.

3.1.1 Data selection

Vehicle sensors receive information of different natures such as temperature, tension, rotating speed or acceleration. At this stage it is necessary to understand, depending on the nature of the anomalies to detect, if the analysis is multivariate and univariate. Whereas univariate time series are simpler to treat, they may not contain enough information themselves for our purpose and multivariable analysis is then required. For this methodology, correlation analysis and industrial expert's advice are the main criteria to determine the relevance of the channels and to make a selection of variables to treat. Invariant channels or channels containing too many missing values are excluded from the analysis.

3.1.2 Filtering

Sensor data usually contain noise and missing values because of faults in transmission or in central units. Filtering is then recommended to mitigate the noise effects and smooth the signals.

The Savitzky Golay filter, based on least-squares smoothing of signals, has been used for vehicle speed and acceleration time series filtering as in (Spiegel *et al.*, 2011). This filter is dedicated for the continuous signals. The main parameters to be adjusted are the order of the polynomial fitting the samples and the length of the window. To adjust the parameters, it must be considered that, for a given window, the greater the order of the polynomial is, the less smooth the output will be.

3.1.3 Normalization

In case variables chosen for the analysis are not in the same range of values, normalisation is performed to standardize the range of the variables and prevent those with large ranges from having more effect in the results than the ones with a smaller range. To this purpose, the most common techniques are minimal-maximal normalization and decimal scaling. Minimal-maximal normalizes to the greatest and the lowest value providing a scaled range from 0 to 1 while decimal scaling normalizes between -1 and 1 dividing each value by a power of ten based on the maximum value. (ElAtia *et al.*, 2016)

3.2 Subsequence extraction

The next step of the method divides the original time series into a set of subsequences with reduced number of time stamps. This process is similar to the segmentation process often used for time series representation methods like PAA. They both aim at dividing the original time series into smaller subsequences. However, while the segmentation methods studied in the literature usually employ a representation using linear segments (Lovrić *et al.*, 2014), subsequence extraction aims at extracting the subsequences directly from the raw time series resulting in non-linear segments in most cases.

The extracted subsequences become the Time Series Unit of Analysis (TSUA) and its size should correspond to the size of the anomalies expected to be detected. The TSUA should contain the anomaly as a whole, hence, avoiding the split of one anomaly in two or more TSUA. Understanding the type of the anomalies to be detected is the first step of subsequence extraction. Next steps include defining the edges of the segments, to then perform the subsequence extraction.

3.2.1 Type of anomalies to be detected

As seen in the literature review, anomalies in time series can be classified in punctual anomalies or subsequential anomalies. Punctual anomalies are defined as an aberrant point in a specific timestamp of the timeseries. The size of a Time Series Unit of Analysis (TSUA), containing this kind of anomaly will be of one timestamp.

On the other hand, subsequence anomalies are characterized by having an anomalous behaviour for some period within the TSUA. In general, the length tends to differ from one anomaly to another. The proposed method is centered on the detection of subsequent anomalies.

The next step to discuss is the type of window to use. Division can be performed using a fixed width or a variable width window approach. In most cases seen in literature, a fixed window length is used, as done in Piecewise Aggregate Approximation (Tamura *et al.*, 2017). However, windows with fixed length are not adapted to the problem of subsequence extraction. Firstly, because anomalous patterns with meaningful information normally occur in gaps of time with different length. Secondly, when fixed window segment extraction is performed, there is a risk of missing subsequence anomalies that are cut by the fixed-length windows. Hence, it is more suitable to use a flexible approach which detects the dynamical changing points of the time series to proceed with the subsequence extraction (Fu, 2011).

3.2.2 Basic Variable definition

The multivariate time series is divided into subsequences called TSUAs. To this purpose, most of the multivariate time series segmentation techniques carry out the segmentation using dimension reduction such as Principal Component Analysis (PCA). However, dimension reduction has the drawback of a loss of interpretability in the results. Easier interpretable segmentation methods are those used for univariate series. Piecewise Linear Representation (PLR) and Symbolic Aggregate approXimation (SAX) are very popular in this field as it simplifies the time series into a succession of straight segments which are understandable to humans.

A hybrid method, in which the entire time series is divided based on the segmentation of a single variable called Basic Variable (BV), is conceived. Inspired by the univariate techniques used for multivariate time series seen in (Blázquez-García *et al.*, 2021), this division treats the whole multivariate time series as a univariate time series just for performing the division. To determine which variable should be the Basic Variable correlation matrix can be use to discard non pertinent variables or those giving the same information. Experts may be consulted as well. Figure 3.2 shows how this division is performed.

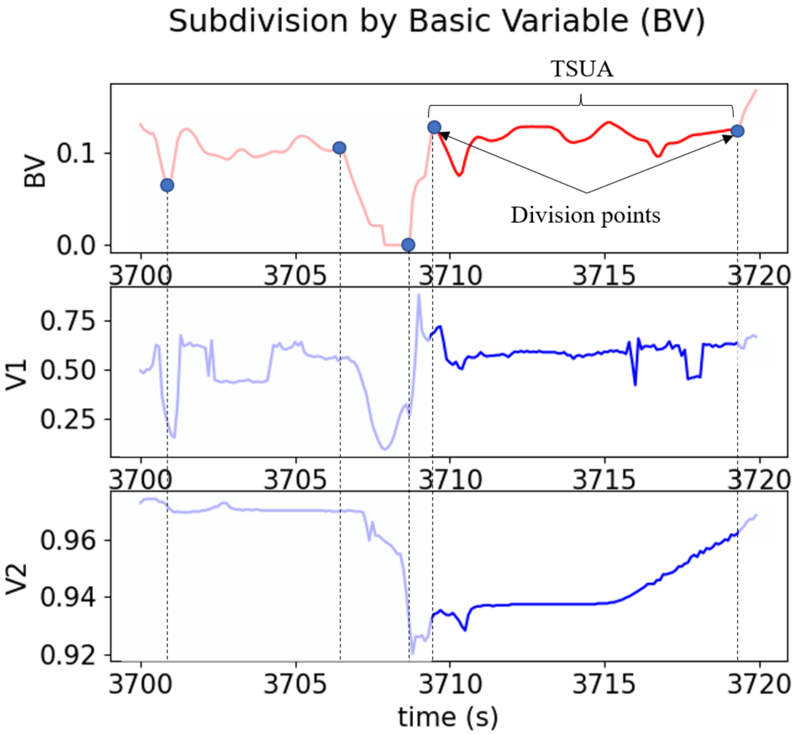


Figure 3.2 Subdivision by points defined for the Basic Variable

Division points are defined for just one variable (BV) based on dynamic information of this univariate time series. Then, the BV and the rest of the variables, V1 and V2, are all divided by those division points. In the example of the Figure there are just three variables, but the method can be applied for as many variables as needed. More details about the subdivision process are provided in following subsections.

For the definition of the variable length windows, the subdivision technique is inspired by (Lu *et al.*, 2020). They propose an algorithm to divide a multivariate time series into subsequences based on dynamical analysis. The objective is to subdivide the Basic Variable (BV) by defining a set of critical points taking into account the dynamical behaviour of the time series and then perform the subdivision by those points. Given a time series $X \ = \{x_i |\ i = 0, 1, \dots, n\}$, where n is the number of observations and x_i represents a vector of *m* components corresponding to the *m* variables of the time series and a set of critical points $Cp = \ \{Cp_j |\ j = 0, 1, \dots, m\}$, X is divided in m segments whose boundaries are two consecutive critical points. Perceptually Important Points (PIP) and points of interest such as local extrema or inflection points are proposed to find those critical points.

PIP is widely used in pattern recognition for stock values. Briefly, the algorithm firstly takes the first and last observation points as critical points and calculates the Euclidean Distance between them and the rest of the observations of the BV. The third critical point is the one that maximizes the sum of distances between it and the extremes of the time series. Following critical points are determined in the same way as the ones maximising the sum of distances between two adjacent critical points. For this algorithm to stop, a fixed number of points or a minimum distance threshold has to be defined (Chung *et al.*, 2001).

Other points of interest for finding critical points are local extrema, saddle points or inflection points which also represent a dynamical change in a time series. This approach is simpler, but effective in some cases (Spiegel *et al.*, 2011).

A set of critical points are defined for the BV and subdivision is performed by splitting the multivariate time series at these points. As a result of subdivision stage, the entire multivariate time series is divided into smaller units of analysis, TSUA. In next iterations of the method, different BVs can be used to detect different types of anomalies.

3.3 Identification of candidates

Once the time series has been broken down into TSUAs, an Anomaly Score (AS) is calculated for each of the TSUAs. The subsequences whose AS surpasses a defined threshold are identified as candidates. Thus, this phase consists of two steps: anomaly score definition and set of anomaly score threshold.

3.3.1 Anomaly Score definition

The anomaly score is defined through an iterative process in which one or several time series are involved. In the present methodology two ways are highlighted.

First, it is proposed to study the problem from a statistical point of view, calculating the correlations between variables of the time series. This approach is similar to the technique employed by (Jones *et al.*, 2014), where anomalies are detected using two time series that are nonlinearly related. Ratio between two variables that are highly correlated is calculated to point out the abnormal TSUAs. The choice of variables depends on the industrial application and, for this, expert advice is recommended.

Another way to proceed would be to use known anomaly patterns. AS in this case is defined as a measurement of the similarity between the TSUA and the given pattern. Distance definition needs to be compatible with the characteristics of the TSUAs but generally, for a shape-based distance measurement, Dynamical Time Wrapping is the most commonly used as shown in the literature review. Similarity measurements like Levenshtein or Hamming distance are adequate for time series with categorical data.

3.3.2 Anomaly threshold

The determination of an anomaly threshold is performed with the help of industry experts. Statistical methods based on standard deviation or z-score can be used to determine the anomaly threshold. Typically, a measure of three times the standard deviation is adopted in many cases. This method has the disadvantage that the data needs to follow a unimodal distribution and that the presence of many outliers would distort the normality of the data.

Graphical methods such as boxplot or a display of the AS histogram can be enough for determining the anomaly threshold, as well as industrial experts' advice. In successive iterations of the methodology, the threshold can be further refined.

As a result of the third stage of the method, a batch of candidates has been identified as potential anomalies. Next steps of the method are used to validate the potential candidates.

3.4 Clustering

The potential candidates obtained in the previous stage are then grouped into categories that will facilitate the interpretability of the anomalies and its subsequent validation. For time series clustering, partitioning, hierarchical and model-based algorithms were studied in the literature review. Depending on the specific application and the nature of the features to be grouped, one or the other may be used. For the purpose of this project, partitioning and hierarchical algorithms will be explored due to their higher interpretability and fast execution times (Aghabozorgi *et al.*, 2015).

For both, a similarity measure has to be defined and certain hyperparameters of the algorithm parameters need to be set.

3.4.1 Distance definition

Defining a distance between TSUAs is the first step for clustering. As seen in the literature, distance measures in time series depend, among other characteristics, whether values are continuous or categorical.

Shape-based distance measures for continuous numerical values include the classical Euclidean distance and Dynamical Time Wrapping (DTW). The first one is the most extended similarity measure for comparing equal length time series, while DTW has been widely used for variable length time series (Berndt *et al.*, 1994).

The difference between both distance measures is shown in Figure 3.3. Euclidean distance in the left matches each point of both time series occurring at the same time stamp while DTW matches points between time series minimizing the path between pairs of points.

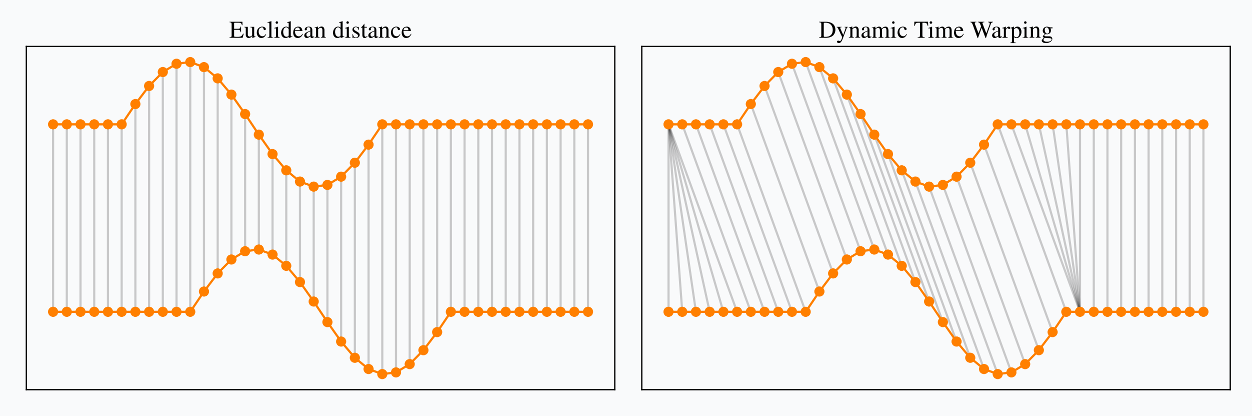


Figure 3.3 Euclidean distance and Dynamical Time Wrapping (Tavenard, 2021)

DTW is chosen as the distance measure between TSUAs because it allows comparing different length vectors, it is compatible with both univariate and multivariate time series and it is capable of recognising similar shapes in time series that are out of phase. Although DTW's computational cost is higher because it has to compare multiple path distances, there are techniques to make it more cost-effective and faster like FastDTW (Salvador *et al.*, 2007).

Data in time series can also be categorical. In fact, a discretization in continuous data can be performed transforming it into categorical as seen in the symbolic approaches reviewed in chapter 2. Among the distance measures for categorical data, Levenshtein distance (LD) seems adequate because it also allows comparing sequence with variable length sequences. In simple words, LD is defined as the minimum edit operations that must be made to transform one sequence into another. Edit operations include insertion, deletion, or replacement of a value.

3.4.2 Clustering algorithm

Proposed clustering algorithms use a distance-based measure of similarity among objects. The distance is calculated with DTW for numerical data and with LD for categorical data, as it was explained in the previous lines. Hierarchical and partitioning clustering methods are proposed for the clustering.

As it was seen in the literature, there are two different approaches to hierarchical clustering. On the one hand agglomerative, or bottom up, and on the other hand divisive, or top down. Although divisive methods usually give better results using the Goodman-Kruskal evaluation method, it is decided to use agglomerative clustering because of faster run times and easier implementation (Roux, 2018). Although run times are not very relevant in early iterations of the method with small amounts of data, when scaling to bigger data sets are used, computation time becomes critical from a business perspective.

Agglomerative procedure is said bottom up because in the beginning each object is considered as a cluster. Then, at each iteration, new clusters are formed by combining the two closest clusters together following the dissimilarity measure. There are four basic ways of computing the distance: single linkage refers to the minimum distance between two objects of the clusters, each belonging to one of the two clusters; complete linkage refers to the maximum distance between two objects of both clusters, each belonging to one of the two clusters; average linkage refers to the average of the distances between all pairs of objects within both clusters; centroid linkage refers to the distance between centroids of the two clusters. The selection of the linkage is discussed in the next lines.

Hierarchical algorithms are proposed due to their visualization capabilities. In fact, as a result of agglomerative clustering, a colored tree diagram is created. It contains all clustered objects linked accordingly to the distances between them. Starting from a single cluster grouping all objects, the dendrogram gradually branches into multiple clusters until it reaches the individual objects. It is a graphical tool that helps in the visualization of these clusters.

Partitioning clustering is also studied for grouping the candidates. The most used partitioning algorithm to this purpose is k-means as seen in the literature. However, in this section Partition Around Medoids (PAM) or k-medoids, a variation of k-means is proposed. For a population of N objects, this algorithm is developed in the following steps:

1. Definition of the number of clusters k
2. Initialization of the clusters using k random objects that will be the initial centers of the cluster or medoids.
3. Associate each individual to the nearest medoid using the given distance definition. In this case distance used is DWT or LD.
4. Calculate again the medoid for each group as the element minimizing the distance to the rest of the objects.
5. Repeat steps 3 and 4 until reaching convergence. Criteria for convergence is that medoids remain the same from one iteration to the next one or a pre-set number of iterations is reached.

(Chen *et al.*, 2017)

The reason why k-medoids is used instead of k-means or other partitioning methods is because it leads to a single existing element representing each cluster. It is useful for the validation stage to have a single element that is the most characteristic of a cluster because it simplifies the user validation process.

3.4.3 Set of parameters

Main parameters to set for the clustering are the number of clusters *k* and the type of linkage.

Determining the number of clusters, *k* is performed using the elbow method. Firstly, the distances between TSUAs are calculated (DTW or Levenshtein) and stored in a distance matrix. Then, clustering algorithm is tested for different values of *k* and the inertia (*I*) is calculated for each value of *k* as described in the following equation.

$$I = \sum_{j=1}^{k} \sum_{i=1}^{n_k} D_{ij}\left(x_{ij} - c_j\right)^2 \qquad 3.1$$

The number of elements in a cluster k is denoted by n_k; x_{ij} denotes the TSUA i in cluster j; c_j represents the center or medoid of cluster j and d is the distance between time series. Inertia represents the sum of squared errors of all the elements from the cluster and it is seen as a quality parameter for clustering. The elbow graphic shows the evolution of the inertia for an increasing number of clusters. Inertia decreases as the number of clusters increases. Usually, this graphic has a marked elbow shape in which there is a point where the increasing of *k* corresponds to a slightly decrease of inertia. Figure 3.4 shows an example of this graphic. Selection of *k* is made by looking for this elbow point. In the given example, the *k* meeting the elbow criteria is 4.

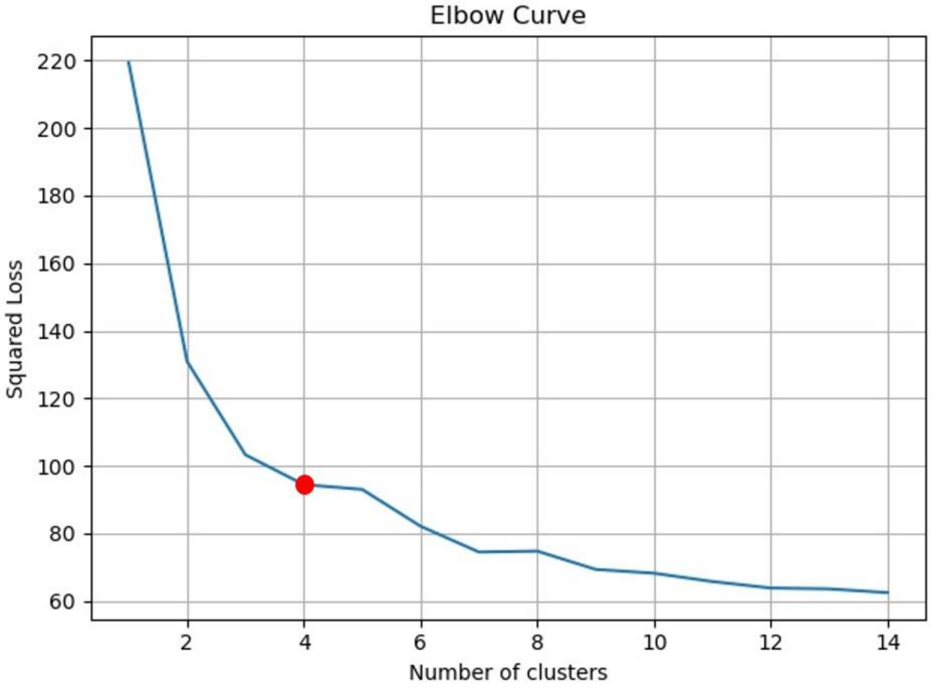


Figure 3.4 Example elbow method

Regarding the type of linkage for the agglomerative clustering, often single linkage is often used for outlier detection (Jiang *et al.*, 2001), but it depends on the specific application and the data set. It is suggested to test different linkages and find a graphical way of comparing the results. For example, comparing the dendrograms or the distribution of population of the clusters.

After setting the clustering parameters and performing the clustering, the output of this stage are *k* categories of candidates to anomalies with similar characteristics. In further iterations of the methodology loop, once clustering has been done for a considerable number of TSUAs, and categories have been validated, new TSUAs, can be automatically assigned to a class by simply computing the distance to the centroids of the clusters without the need of repeating the whole clustering process.

3.5 User validation

Validation is the last stage of the methodology. The user of the detection method (industrial partner) will validate the results obtained in the previous stages. This step necessary to determine which classes are relevant and evaluate the performance of clustering as a classification method. Candidates after validation stage are divided in:

- *Xknown* detected anomalies that have similar behaviour to anomalies previously studied.
- *Xnew* identified anomalies that have different characteristics to the already known ones.
- *Xdiscard* candidates that are discarded from being an anomaly.
- *Xother* candidates that might need further analysis before determining whether they are an anomaly.

In the proposed method, where human interpretability is a priority and expert knowledge is involved, user validation is carried out in two different ways: visual inspection and rule-based validation.

3.5.1 Visual inspection

In early iterations of the method, where little information is known about the anomalies to detect, visual inspection carried out by industry experts is performed to validate the candidates to anomalies. As far as possible, candidate by candidate validation is recommended in the first iterations. However, if it is not possible in terms of time or human resources, a less accurate technique can be employed extracting a validation set from each cluster so that just a percentage of the total candidates is inspected. For visual inspection, experts can use graphics of plotted candidates. Graphical libraries such as *matplotlib* or *plotly* for python language are just two examples of graphical tools to use.

The categories of candidates validated by the experts as relevant continue in the analysis and the rest of the categories are disregarded. The output of visual inspection is used to adjust the parameters of the algorithms and metrics used in stages 3 and 4 and change the Basic Variable used for subsequence extraction in stage 2.

3.5.2 Rule-based validation

In advanced iterations, a more autonomous and less expensive validation technique is proposed. Rule-based validation consists in the definition of certain rules characterizing the dynamical behaviour of the candidates that categorizes the TSUAs so to create categories of behaviours regarding, for example, the length of the TSUA or the maximum value reached by a certain variable. Then, the behaviours are labeled as: *Xnew, Xknown, Xdiscard* and *Xother*. The validation rules are defined in collaboration with the industry experts and automatically categorize the candidates into the four categories previously mentioned. It is important that the workflow of the validation process ensures that one label is assigned to each candidate. Figure 3.5 illustrates the rule validation process.

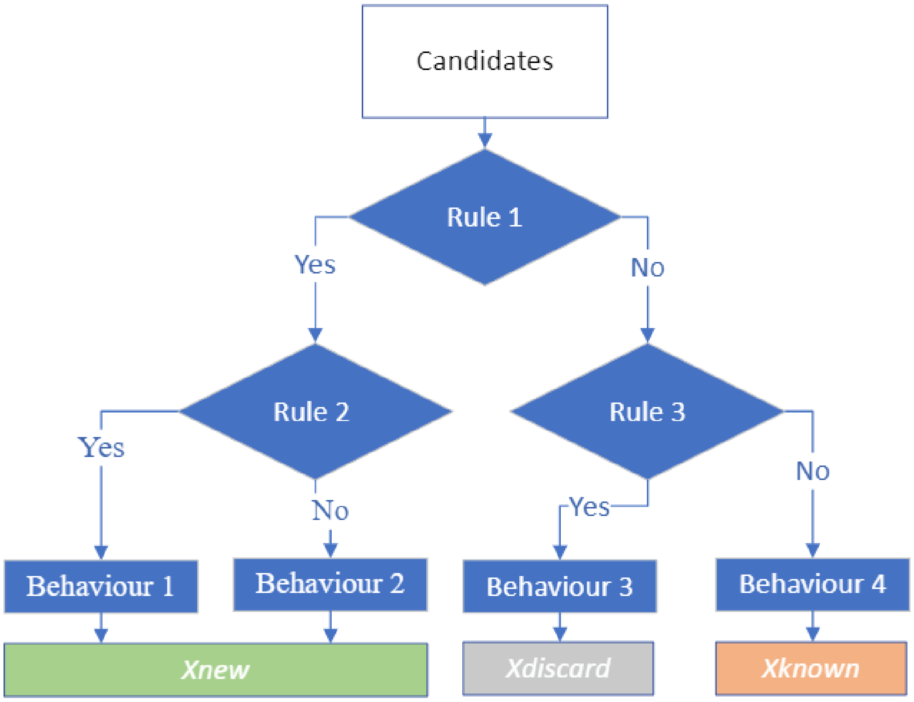


Figure 3.5 Rule-based validation schema

Stage 5 of the method is user validation. The input in this stage is a batch of categories of candidates to anomaly. The output is a labeled set of validated anomalies (*Xnew* and *Xknown*), potential anomalies (*Xother*) and discarded candidates (*Xdiscard*). As a result, a confusion matrix is created comparing the categories of behaviours created by clustering and the categories of behaviours validated by the rules. Metrics like accuracy, f-score and recall are calculated to evaluate the performance of clustering.

CHAPTER 4 CASE STUDY

The following chapter details the application of the method proposed in Chapter 3 to a case study using real data from an industrial partner. First, the case study is put into context, explaining the industrial situation. Then, each of the stages of the proposed method is covered, including successive iterations. Finally, the results are presented and discussed.

4.1 Context

The industrial partner with which the project was carried out is situated in the heavy vehicle sector. It is a Montreal-based small to medium enterprise (SME) founded in 2006 with the aim of reducing the environmental impact of heavy-duty vocational trucks. To this end, they developed various technologies aimed at the total or partial electrification of commercial fleets. The present project is developed for a pioneer Stop Start Module (SSM) installed in already-in-operation trucks that can save up to 30% of fuel consumption and reduce by half engine run time. Typical industrial partner's customers for those modules are companies that manage a fleet of vehicles for garbage collection or container movement within a port. These are trucks that are often idling while the engine is still running and have great potential for fuel savings with the use of SSM. Due to the novelty of the product and the growth phase in the industrial partner's current stage, the after-sales service becomes extremely important, and the objective of this project will be to detect possible abnormal behaviours of the SSM. By doing this, predictive maintenance strategies could be deployed and a better service is delivered to the client.

The next lines include a deeper explanation of the SSM and its installation, the type of anomalies that the industrial partner usually faces with its clients, a brief explanation of the anomaly detection process, current diagnosis process and the problem faced by the industrial partner.

4.1.1 Stop Start Module

The SSM is installed in already-in-operation vehicles. It shuts down the combustion engine and switches to electric mode while the truck is idling (e.g., the operators are loading the containers on a truck). It consists of three subsystems.

1. Electric motor-generator. Electrical engine that supports engine restart after shut-down and electrical generator that recharges the ultracapacitor.

2. Power-pack. Ultracapacitor storing the energy for restarting the engine and feeding auxiliary systems.
3. Electrical PTO. Enables the engine auxiliary systems to be used during shutdown.

Figure 4.1 shows the installation of the module consisting of the three main parts previously described.

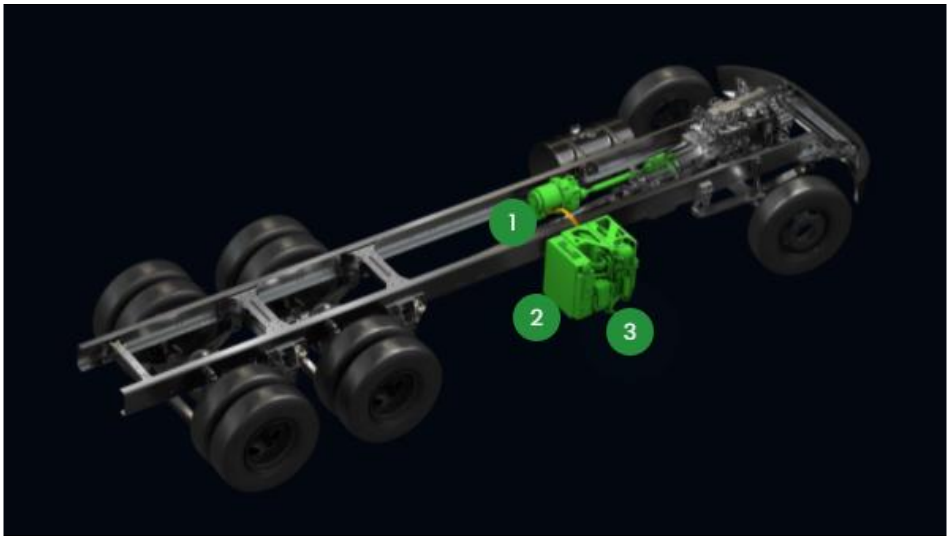


Figure 4.1 Stop-start module installation

For SSM to work, it is necessary to know at each time the state of the vehicle. The information of the state of the vehicle (acceleration pedal position, brake’s pressure, engine speed, etc.) is continuously captured by the multiple sensors installed on the truck and transmitted through the Controller Area Network bus to the SSM. Each variable is transmitted through a channel of the CAN bus and is saved in a log file. In the following, variables or channels would refer to the signals transmitted by the sensors of the vehicle.

4.1.2 Type of anomalies

The anomalies to be detected are categorized as mechanically caused anomalies on the one hand and human-caused anomalies on the other. Examples of anomalies with a mechanical cause are a delay in starting the engine, restarting in an unusual gear or an undesired shutting down of the engine. These anomalies are detected using the data collected by the SSM. An example is shown in Figure 4.2. A driver pressed the acceleration pedal (red) but it took about 4 seconds for the truck to gain in speed (green). Engine speed (pink) shows the shutdown and restarting process that impeded the truck from moving during those 4 seconds. Engine speed went down to zero (shutdown) and increased again (restart) before the truck gains in speed. This anomaly affects not just driving experience, but also results in an unnecessary stop start operation of the engine that discharges the ultracapacitor.

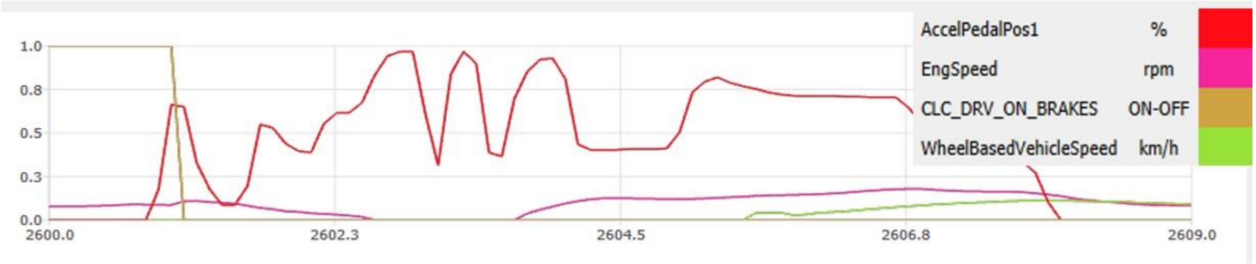


Figure 4.2 Example of mechanical anomaly

The human-caused anomalies seek to find unusual or undesired behavior on the part of the driver. The interest of detecting them lies in the improvement of SSM efficiency and the detection of bad practices. An example of this bad practice would be the forced restart of the combustion engine when the truck is stopped using the electric mode. This forced restart is showed in Figure 4.3, where the driver presses the acceleration pedal (red) while brakes (brown) are pressed. The press of the acceleration pedal forces the restart of the engine, but there is not an actual intention of the driver to move since brake pedal is pressed at every time. This type of behavior unnecessarily increases engine running time, increasing gasoline consumption, and emitting more exhaust fumes in cities.

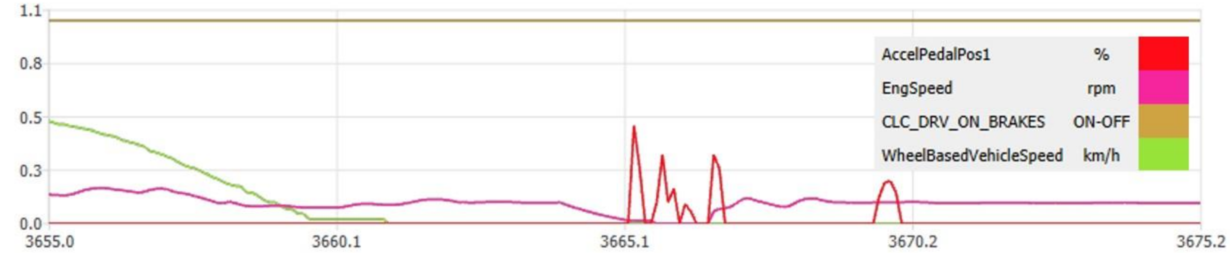


Figure 4.3 Example of anomaly caused by the human driver

The anomalies described by the industrial partner do not seamlessly enter in the punctual or subsequent definition of anomalies described in the literature. The first reason is the subjectiveness of anomaly definition for our industrial partner. Throughout the project, several employees of the partner company have collaborated and their criteria for defining anomalies have been different. Moreover, an unexpected behaviour of the truck for the driver might not be anomalous for the company and vice versa. Secondly, anomalous behaviours are not related with values of the time series that differ from its context, but with abnormal correlations between channels, as described in the given examples.

In the absence of a labeled anomaly base and a homogeneous criterion for the definition of anomaly, the criterion of one of the company's experts with whom we have worked the longest has been used.

4.1.3 Actual anomaly detection process

A total of 200 to 300 channels provide the necessary monitoring information for proper operation of the SSM. All operating data is stored in a large cloud database. When undesired events occur during driving, the client contacts our industrial partner, and the after-sales service retrieves the log data file from the specific vehicle and day to start the diagnostic process. Figure 4.4 illustrates the fault detection process.

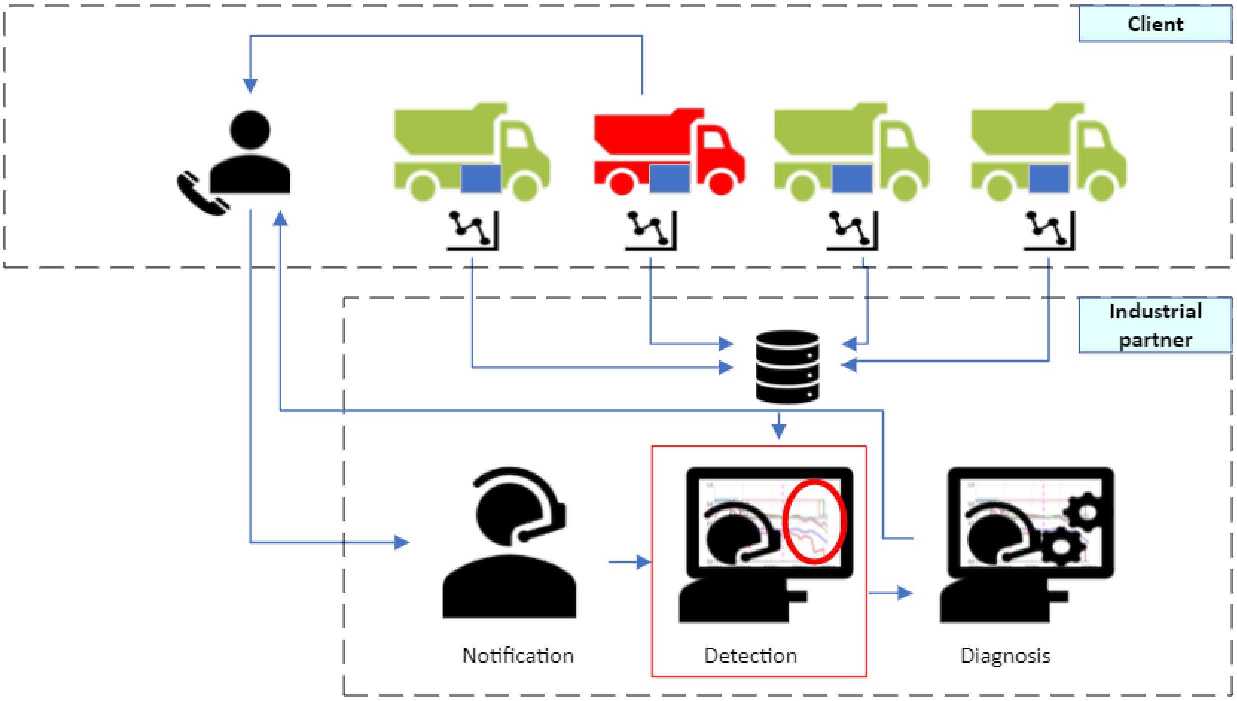


Figure 4.4 Fault detection process

Using a visualization program, the after-sales team is responsible for detecting abnormal behavior. The display program shows the evolution over time of the 200 to 300 signals retrieved from the log file. There are some error codes that help this process, but these error codes only show punctual failures due to missing sensor data but do not give information about abnormal behaviors during driving. An exhaustive manual analysis of the log files is often necessary as the customer's descriptions are not very accurate and a lot of time is spent in the process, as each file may contain approximately 200,000 timestamps recorded every 0.1s, for a total of 5.5h of driving.

After detection, a diagnostic phase follows in which the causes of the problem are sought. The same data visualization tool is used in this case, together with mechanical or electrical test benches. The results of the diagnosis are communicated to the customer to offer a solution. For this project, the focus is made on the detection stage. However, the proposed method takes into account a future diagnosis stage.

4.1.4 Problem description

The SSM already shows some error codes that detect specific failures in the SSM or in the truck due to missing sensor data. The problem arises when the anomalies are perceived subjectively by the user. For example, a driver complaint may be a delay in restarting the engine. Even though there has been no error code, the driver feels that the truck has not responded as usual and it is important for the industrial partner to understand whether the reason for this behavior is related to the SSM or is due to a mechanical problem with the truck itself.

Anomaly detection is a manual process, based mostly on the knowledge of the company's experts. To date, there is no labeled anomaly database containing a detailed description of abnormal behaviors and the situations in which they have occurred. The objective of this project is therefore the development of a method that automates the anomaly detection process while keeping the human in the loop. As a result, the performance of the SSM can be improved, costs for the industrial partner and its customers related to the diagnosis process can be reduced as well as the environmental impact of heavy trucks.

4.2 Application of the methodology

4.2.1 Description of input data set

The input data in this case are files that record the status of the truck throughout a working day. The status is monitored by sensors with a frequency of 10 Hz. The result is a matrix in which each column corresponds to a channel (‘Time’, ‘AccelPedal’, ‘ParkingBrake’), or variable, and each row to a record with a timestamp. There are approximately 250 variables per file and trucks from the same client usually have the same channels. The number of rows varies from one file to another and from one client to another because driving times per day are not fixed, but usually range from 150,000 to 300,000 timestamps (4 to 8 driving hours). The size of each file goes from 10Mb to 20Mb approximately.

For this case study, a file is chosen from a given contractor with a total of 279 channels and 206923 timestamps recorded every 0.1s, equivalent to 5h and 45min of driving. Table 4.1 shows an outlook of raw input data.

Table 4.1 Raw input data

		Channels					
		Time	AccelPedal	ParkingBrake	...	...	EngSpeed
Timestamps	0	0.00	0	1	...	...	0
	1	0.10	0	1	...	...	0
	2	0.20	0	1	...	...	0
	3	0.30	0	1	...	...	0
	…	…	…	…	...	...	…
	…	…	…	…	...	...	…
	206921	20692.10	0	1	...	...	0
	206922	20692.20	0	1	...	...	0
	206923	20692.30	0	0	...	...	0

4.2.2 Data preparation

4.2.2.1 Data selection

The preparation of the input data begins with the selection of the relevant channels. A first filter is applied to eliminate empty channels, which are those containing only values equal to zero due to a sensor error or missing data. Other filtered channels are the invariant ones, whose value is constant throughout the file, as they will not provide information of interest for anomaly detection. This first filtering eliminates 91 channels from the initial 279 ones.

Selection of variables is commonly performed with correlation coefficients as studied in (Saidi *et al.*, 2019) and (Liu *et al.*, 2020). In this case study, a correlation matrix between the 188 remaining channels is calculated using the Pearson linear correlation coefficient for the numerical variables. The results are presented visually in the form of a heat map in appendix B.

Industrial experts are then consulted to reduce the number of relevant channels and a final 13 channel selection is done. A final correlation heatmap of the selected channels is showed in Figure 4.5. It is noteworthy that there is a direct correlation of more than 0.5 for 13 channel pairs of channels and of less than -0.5 for 10 pairs of channels. Correlations between channels are important at this point because they can be a way of finding anomalous behaviours. For example, a sudden change in the correlation between variables could be an anomaly.

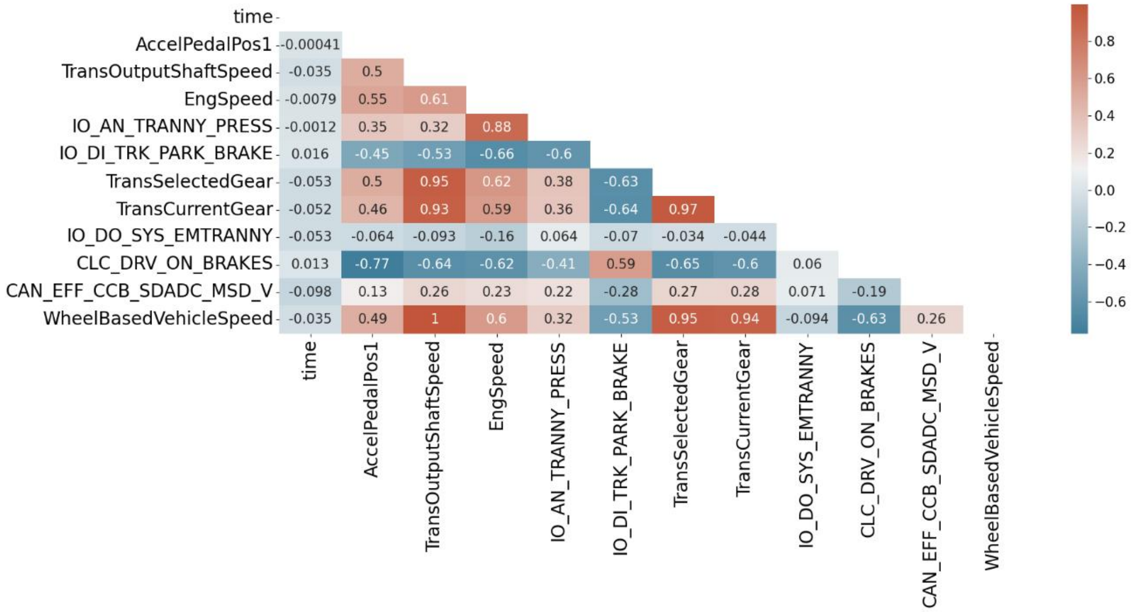


Figure 4.5 Correlation heatmap for the relevant channels

4.2.2.2 Filtering

The selected channels are then pretreated for missing data. In the case study, the missing data of each channel is manifested by values that deviate greatly from the standard deviation of the values in that channel. The value for an error code varies from one channel to another and the error code dictionary was not shared for the project. Hence the standard deviation Z-score is calculated for each timestamp of every channel, with $z_i = \frac{x_i - \overline{x}}{\hat{\sigma}}$, where x_i is the value of a given channel for time stamp i, $\overline{x}$ is the average value of the same channel and $\hat{\sigma}$ is the standard deviation. If the Z-score z_i exceeds 4, then it is considered as an error code for missing values. Isolated missing values are replaced by the mean of the two adjacent non-missing values. Successive error codes are removed from the initial data set and are analyzed separately if considered relevant by the industrial partner.

Then, in the first iteration, a Savitzky Golay filter is used for smoothing the continuous signals, such as acceleration. Figure 4.6 shows the output of the filtered signal (red) versus the original signal (blue). After several tests, with different window lengths and polynomial orders, it is decided to keep a window of 5 timestamps and a polynomial approximation of first order. All sharpening points are well smoothed while the shape of the curve is kept.

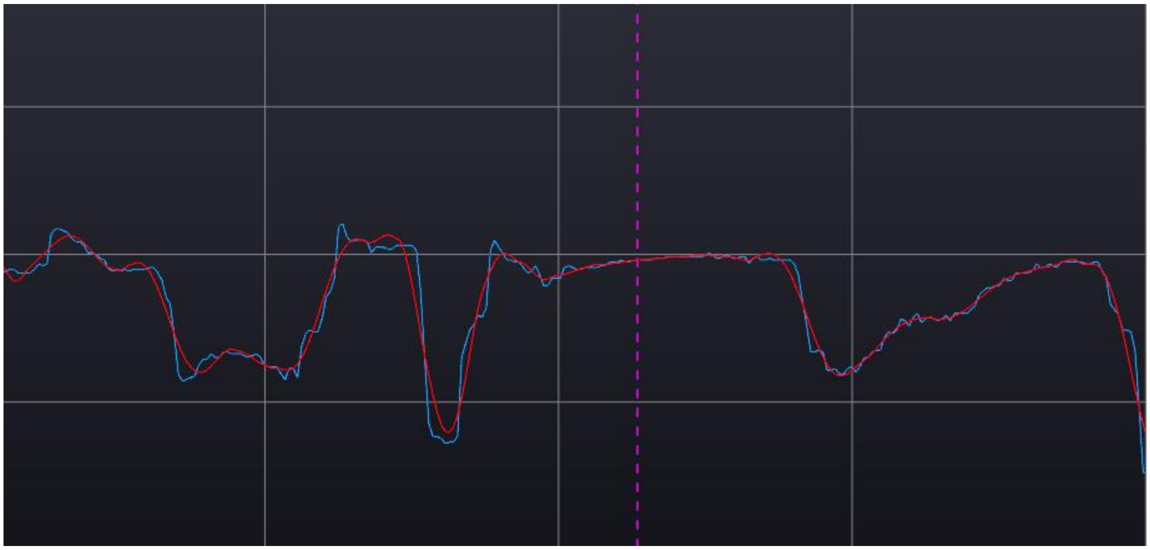


Figure 4.6. Savitzky Golay filter for acceleration pedal channel

4.2.2.3 Normalising

Channels not ranging from 0 to 1 are then normalized using the minimal maximal scaler $x_{i\,scaled} = \frac{x_{max} - x_i}{x_{max} - x_{min}}$, where x_i is an observation, x_{max} and x_{min} are the maximal and minimal values of the time series for a given channel. After minimal maximal scaling all channels of the time series range from 0 to 1.

4.2.3 Subsequence extraction

After pre-processing the data, subsequences are extracted from the multivariable time series. The purpose is to divide the whole time series into smaller TSUAs, as seen in Chapter 3. Division is done with a variable length window approach.

4.2.3.1 Length of the anomalies

The size of the TSUA is adapted to the characteristics of the anomalies to be detected. The type of the anomalous behaviors described in ection 4.1.2 corresponds to subsequent anomalies, rather than punctual aberrant timestamps. These anomalies are identified by a succession of timestamps in which each one of them is not an anomaly by itself, but the occurrence of one after the other is unusual.

It is also important to highlight the multivariate nature of the anomalies since a normal behavior for one variable may not be consistent with the evolution of another variable. For example, an increase in vehicle speed is a normal sequence but, if this increase in speed is produced along with an increase in brake pedal pressure, it becomes an anomalous behavior.

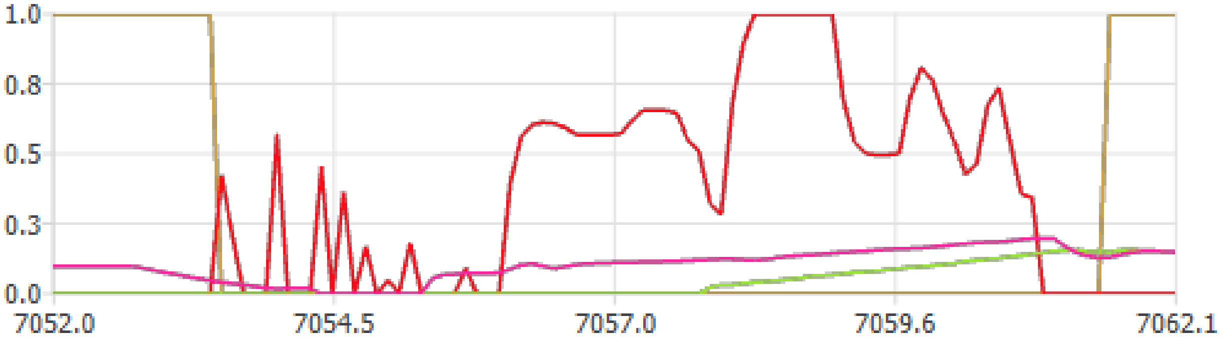


Figure 4.7 Example of anomaly caused by the mechanical issue

Examples of mechanically caused anomalies are given in Figure 4.2 and **Error! Reference source n ot found.** (anomalies) and of an anomaly caused by a human in Figure 4.3. The duration of this sort of anomalies range between 2 and 6 s. However, its boundaries are not clearly defined.

4.2.3.2 Subsequence extraction

The first iteration of the method uses the position of the accelerator pedal (AccelPedalPos1) as the BV. The selection of this channel is based on expert advice from the industrial partner. The accelerator pedal is a continuous channel that receives information directly from the driver's behavior. It is also one of the channels most frequently used by the after-sales service team for the visual detection of anomalies. Once the BV is defined, segmentation by critical points is performed as explained in the methodology.

In a first attempt, a simple local extrema detection technique is used to determine those critical points in the BV. Although the AccelPedalPos1 signal has already been smoothed with Savitzky Golay filter, local extrema lead to small length TSUAs, which are not able to cover the truck's restart processes.

Extraction of perceptually important points (PIPs) is then performed to extract the critical points from the BV. The number of PIPs is tested through several iterations and checked with the industrial partner until an average of 1 PIP point every 10 timestamps. Results are showed in Figure 4.8.

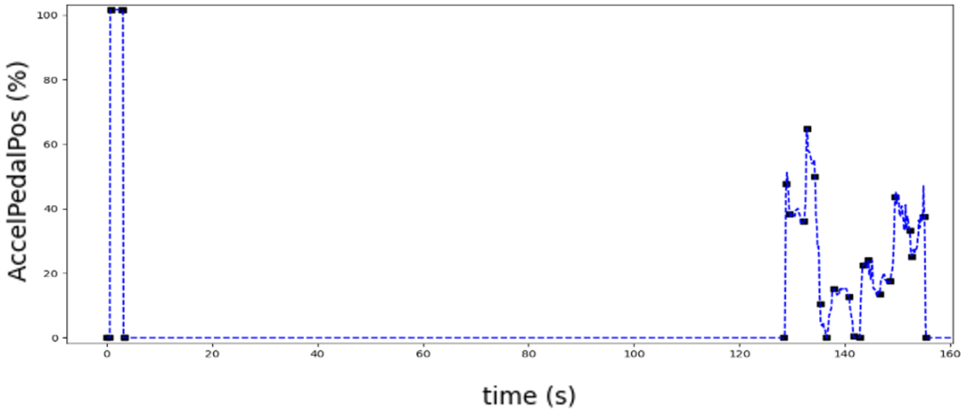


Figure 4.8 PIP segmentation of acceleration pedal channel

Although gaining in mechanical interpretability due to the shape-based PIP allocation of the points, the algorithm takes too long for allocating the points in a single time series. For the test data base of 206923 timestamps, it took 110 s.

Due to the unsatisfactory results obtained with the extreme local and PIP techniques, a critical point location method is proposed that balances computational time and good human interpretability of the results. Those critical points aim at detecting the dynamical changes of the BV. The critical points are defined by the timestamps at which the BV channel goes from zero to non-zero. Those timestamps are defined in the following way:

$$x_i \in Cp \leftrightarrow ((x_{i-1} = 0) \cap (x_{i+1} \neq 0)) \cup ((x_{i-1} \neq 0) \cap (x_{i+1} = 0)) \qquad 4.1$$

Where x_i is the value of the BV in the timestamp i.

Graphically, critical points are showed in Figure 4.9, represented by the blue dots.

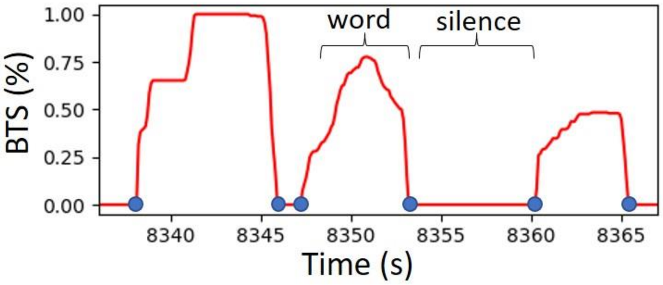


Figure 4.9 Time series segmentation

The BV is divided in segments where $x_i = 0$ and other segments where $x_i \neq 0$. Similarly to the symbolic approach (Li *et al.*, 2015), if a time series is seen as a long phrase, the first group of segments will be the *silences* and the second will be the *words*. For the purpose of this project, the *silences* ($x_i = 0$) will be excluded from the analysis and the TSUAs will be the *words*, where $x_i \neq 0$. For ease of reading, the term *word* and TSUA will be used interchangeably, referring to the unit of analysis into which the initial time series has been segmented.

The first iteration transforms the 206923 timestamps into 876 *words* based on acceleration pedal segmentation. In further iterations, other channels are used as BV and they are segmented using the same technique. The use of different channels as BV aims to increase the range of anomalies that can be detected. It should be ensured that the value 0 for those channels represents a period of inactivity.

For the second and third iterations, the channel called EMPTO_STATUS is used as BV and a different anomaly score is used. This channel's range, unlike the acceleration pedal, has 14 discrete values indicating states of the SSM ranging from 0 to 13. Since state 0 indicates inactivity of the SSM, by keeping the *words* and eliminating the silences from the analysis, just the relevant information about restarting process is kept in the *words*.

Table 4.2 Subsequence extraction of the time series

Iteration	**Basic Variable**	**Timestamps**	**Critical points**	**TSUAs/words**
1st	AccelPedalPos1	206923	1752	876
2nd	EMPTO_STATUS	206923	1256	627
3rd	EMPTO_STATUS	206923	1256	627

In the second and third iterations, the same number of subsequences are extracted because the BV used in both iterations is EMPTO_STATUS. However, the anomaly score will differ. Through subsequence extraction, the time series is transformed into a table of words with attributes as shown in Table 4.3.

Table 4.3 Sample of the word data set

word_id	**Basic variable**	**length (timestamps)**	**word time start (s)**	**word time end (s)**	**…**
0	EMPTO_STATUS	10	0	0.9	⋯
1	EMPTO_STATUS	74	21.6	28.9	⋯
2	EMPTO_STATUS	518	336.4	388.1	⋯
3	EMPTO_STATUS	114	391.2	402.5	⋯
4	EMPTO_STATUS	40	680.6	684.5	⋯

4.2.4 Identification of candidates

4.2.4.1 Definition of Anomaly Score

The anomaly score is used to identify the abnormal *words* that have been extracted from the original time series. As proposed in the methodology, the anomaly score (AS) is firstly addressed by studying correlations between various channels of the time series. By definition, the average acceleration (a) is the increase in velocity (Δv) over the corresponding duration (Δt) $a = \frac{\Delta v}{\Delta t}$, hence an intuitive way of identifying anomalous behaviours will be to exploit this correlation even though acceleration pedal position channel does not measure itself the acceleration expressed in the previous formula.

In the first iteration of the method, the AS is defined for each *word* as:

$$AS = \frac{\sum_{i=t_s}^{t_e} AccelPedalPos1_i}{\sum_{i=t_s}^{t_e} WheelBasedSpeed_i} \quad 4.2$$

Where $AccelPedalPos1_i$ refers to the scaled value of accelerator pedal position channel for timestamp i of the *word*, $WheelBasedSpeed_i$ refers to the scaled value of the transmitted speed to the shaft of the truck for timestamp i of the *word*, t_s refers to the *word* starting time and t_e refers to the *word* ending time.

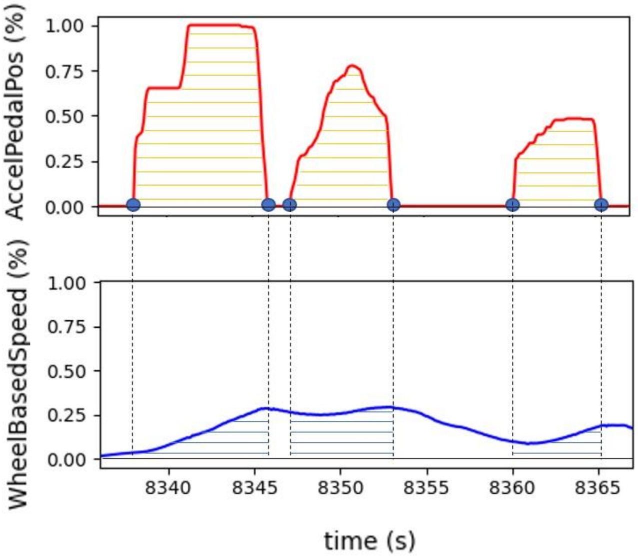


Figure 4.10 Graphical representation of the Anomaly Score for the first iteration

Figure 4.10 shows graphically the definition of AS, seen as the ratio between the area under the acceleration pedal curve and the area under the velocity channel curve. Although the acceleration pedal position and wheel-based speed initially had a Pearson’s correlation coefficient of 0.5, the calculated areas increase the correlation coefficient to 0.93.

In the second iteration of the method, using the EMPTO_STATUS *words*, the same definition of AS is applied. In this iteration, a specific sequence of EMPTO_STATUS values is observed that is found repeatedly in the identified candidates. This sequence is taken as an anomaly pattern and is used in the third iteration for candidate identification. More detailed explanations about this pattern are given in the user validation stage.

For the third iteration, in which the EMPTO_STATUS *words* are used again, the AS is defined by the distance to the pattern identified in the second iteration. The distance from each *word* to the pattern is based on categorical similarity measures and is calculated using the Levenshtein distance.

$$AS = lev_{dist}(word_{ES}\ , pattern_{\mathrm{ES}}) \qquad 4.3$$

where $word_{ES}$ refers to the vector of values of EMPTO_STATUS channel of the $word.$, $pattern_{ES}$ refers to the vector of values of EMPTO_STATUS channel of the pattern and lev_{dist} refers to the Levenshtein distance measure between the two sequences.

4.2.4.2 Anomaly threshold

The setting of an anomaly score threshold in the first iteration is done by using visual statistical methods. Figure 4.11 shows the distribution of anomaly score observations of the extracted subsequences through a boxplot.

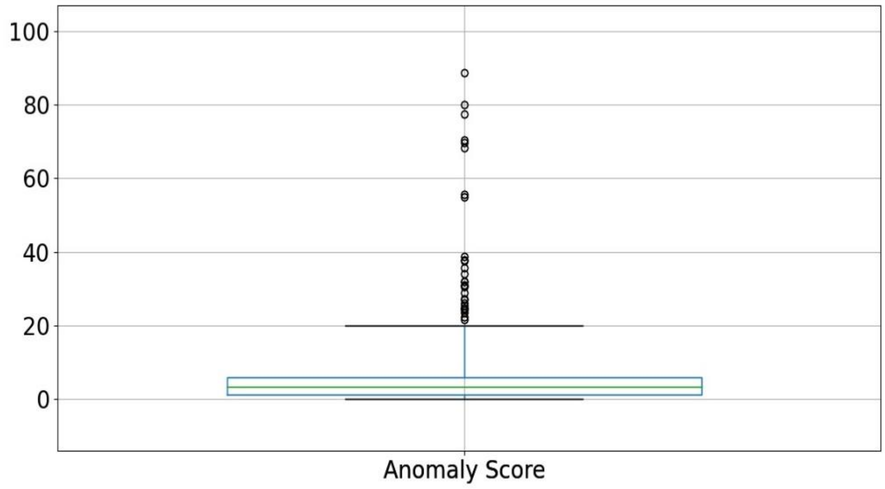


Figure 4.11 Boxplot anomaly score first iteration

In view of the graph, the anomaly threshold value 20 is used in the first iteration: it corresponds to the upper limit of the whisker, defined as Q3+1.5IQR, where Q1 and Q3 are the first and third quartiles respectively and the interquartile range IQR is equal to Q3-Q1. Visual methods such as boxplot graphics or a display of the AS histogram can be enough for determining the anomaly threshold, as well as industrial experts’ advice.

For the second iteration, a similar boxplot is obtained, and the AS threshold is set to 40, based on the upper whisker as seen in Figure 4.12.

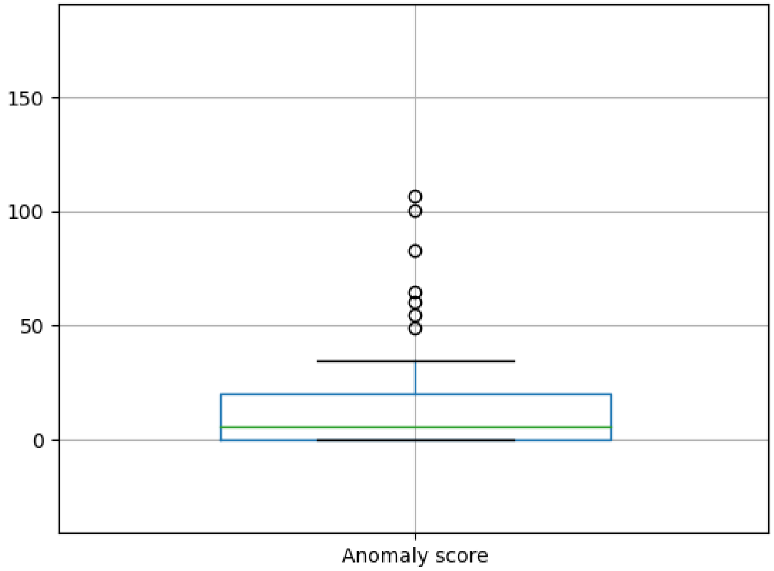


Figure 4.12 Boxplot anomaly score second iteration

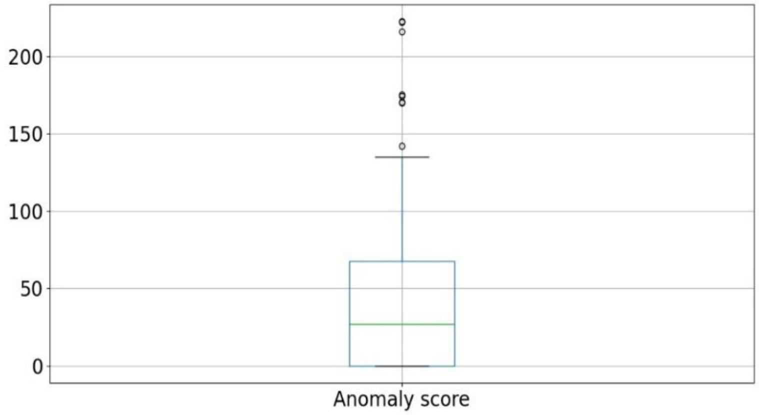


Figure 4.13 Boxplot anomaly score third iteration

For the third iteration, the EMPTO_STATUS words are employed and distance to a given pattern is used as AS. Aiming at determining the *words* that best fit the pattern, selection of the AS threshold is chosen so that the candidate *words* are those minimising the AS. Figure 4.13 graphically represents the AS calculated for the third iteration.

The lower whisker and first quartile of the boxplot is situated at zero, meaning that at least the first fourth of the *words* have the exact same shape as the pattern. Therefore, the AS threshold is situated at 0 (included) for the third iteration so that only the *words* following the exact same pattern are counted as potential candidates.

As a result of the candidate identification, a batch of TSUAs have been extracted as potential anomalies. Table 4.5 summarises the candidate identification stage.

Table 4.4 Candidate identification for the case of study

Iteration	**BV**	**Number of words**	**AS formula to identify candidates**	**Number of word candidates**
1st	AccelPedalPos1	876	$AS = \frac{\sum_{t_s}^{t_e} AccelPedalPos1_i}{\sum_{t_s}^{t_e} WheelBasedSpeed_i}$	142
2nd	EMPTO_STATUS	627	$AS = \frac{\sum_{t_s}^{t_e} AccelPedalPos1_i}{\sum_{t_s}^{t_e} WheelBasedSpeed_i}$	133
3rd	EMPTO_STATUS	627	$AS = lev_{dist}(word_{ES}\ , pattern_{\mathrm{ES}})$	143

4.2.5 Clustering

Clustering stage aims at grouping the candidates with similar characteristics to extract meaningful insights from the abnormal patterns and to divide into different categories the abnormal behaviors. Clustering also helps the user validation stage, where different categories are validated or not as anomalies. In the following lines, the distance definition for the study case is addressed as well as the setting of clustering parameters and algorithms.

4.2.5.1 Distance definition

DTW and LD have been set as the distance measure between *words* for numerical and categorical time series respectively. For *N* candidates, a distance matrix of *N x N* elements is then calculated for the channel chosen. Therefore, each element of the distance matrix for a given channel ($DM_{channel}$) is defined as:

$$DM_{ij} = dist(word_{i\,channel}\ ,word_{j\,channel}) \qquad 4.4$$

$word_{i\,channel}$ is the vector of values of a given channel for the *word* i, DM_{ij} is the element of the distance matrix located at row *i* and column *j* and *dist* is the distance chosen for that channel: for continuous numerical channel, DTW and for categorical channels LD.

In the first and second iterations, the channel for the distance matrix used is the same channel used for defining the *words*. First iteration corresponds to AccelPedalPos1 and second iteration to EMPTO_STATUS.

For the third iteration, other channels are introduced to calculate a multivariate distance. A multivariable distance seeks to obtain a categorization that can be better interpreted by the diagnostic team, as it relies on the variables most frequently used by the team for the manual classification of anomalies. Multiple distance matrices are calculated for multiple channels of the *words.* A global distance matrix (GDM) is then computed as a weighed sum of the multiple distance matrixes:

$$GDM = \sum_{i=1}^{M} w_i \times DM_{channel_i} \qquad 4.5$$

$DM_{channel_i}$ refers to the distance matrix calculated as showed in the previous paragraph, w_i refers to the weight of channel i, and M refer to the number of channels used to calculate GDM. The channels chosen to calculate the GDM are: AccelPedalPos1, which refers to the position of the acceleration pedal (%); CLC_DRV_ON_BRAKES, which refers to the pressing of the brakes (0/1); TransSelectedGear, which refers to the selected gear by the truck driver and Wheelbasedspeed[2], which refers to the speed of the truck measured with the wheel's rotation speed. For instance, each channel used has the same weight w_i, but further iterations will include an iterative process to maximise the performance of clustering based on the validation stage.

[2] Wheelbasedspeed in some files from the industrial partner is named as TransOutputShaftSpeed.

4.2.5.2 Set of parameters

A distance matrix is given as input for the clustering algorithms, but some parameters have to be defined to perform the clustering, as it was mentioned in Chapter 3. The most critical parameter for the chosen clustering algorithms is the number of clusters *k*, which is determined by the elbow method. Figure 4.14, Figure 4.15 and Figure 4.16 show the evolution of inertia or squared loss with the increasing number of clusters for all the three iterations of the method following the k-medoids algorithm. The red dotted line in each figure represents the *k* chosen for each iteration following the elbow shape of the curves. The chosen *k* for the k-medoids algorithm is also used for agglomerative clustering, so that both clustering algorithms outputs can be comparable.

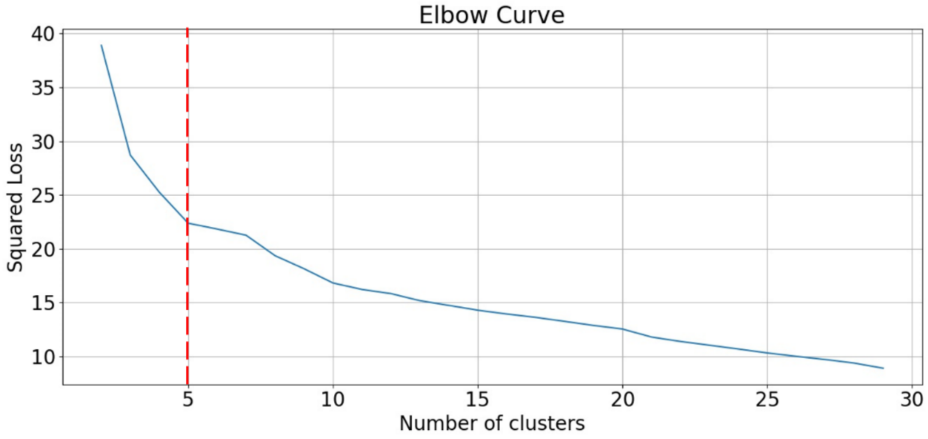


Figure 4.14 Elbow curve first iteration

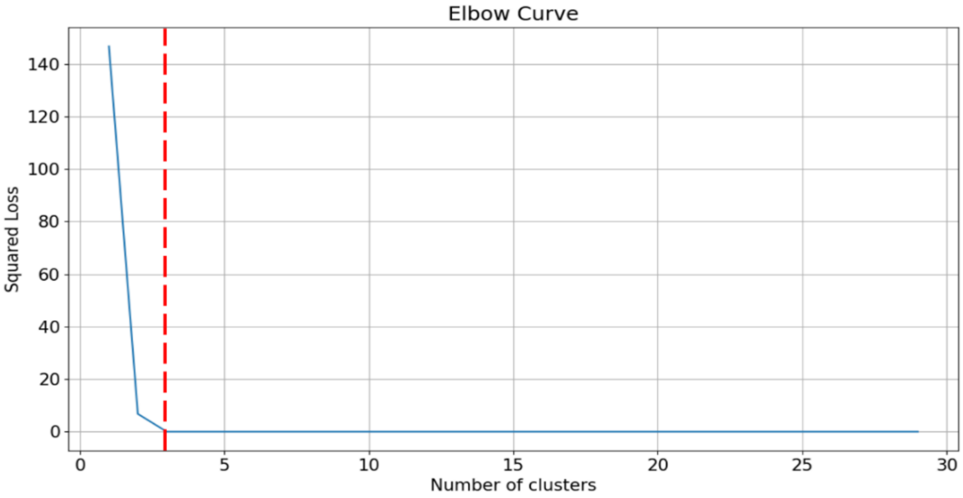


Figure 4.15 Elbow curve second iteration

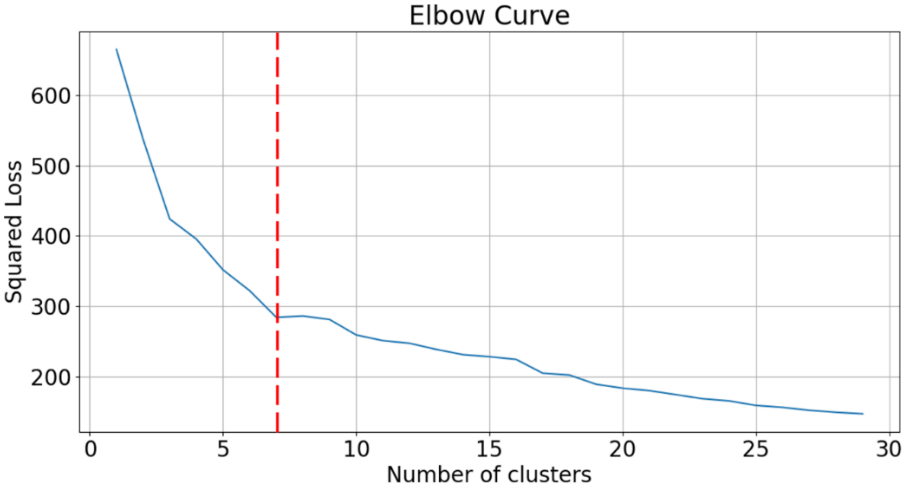


Figure 4.16 Elbow curve third iteration

It is remarkable that for the second iteration, the elbow curve stays at zero from *k* equal to 3 clusters. Meaning that there are just three categories of words for iteration two. A priori, a clustering with this elbow has little sense since there are just three categories clearly identified. Hence there is no need of clustering. Analysing more in depth the shapes of EMPTO_STATUS curve in user validation stage, one can extract the pattern leading to an anomaly. A more detailed explanation is presented in Section 4.3.

Regarding the agglomerative clustering algorithm, another parameter to set is the type of linkage. Tests with single, complete, and average linkage are performed. The number of instances in each cluster in first iteration is shown in Figure 4.17.

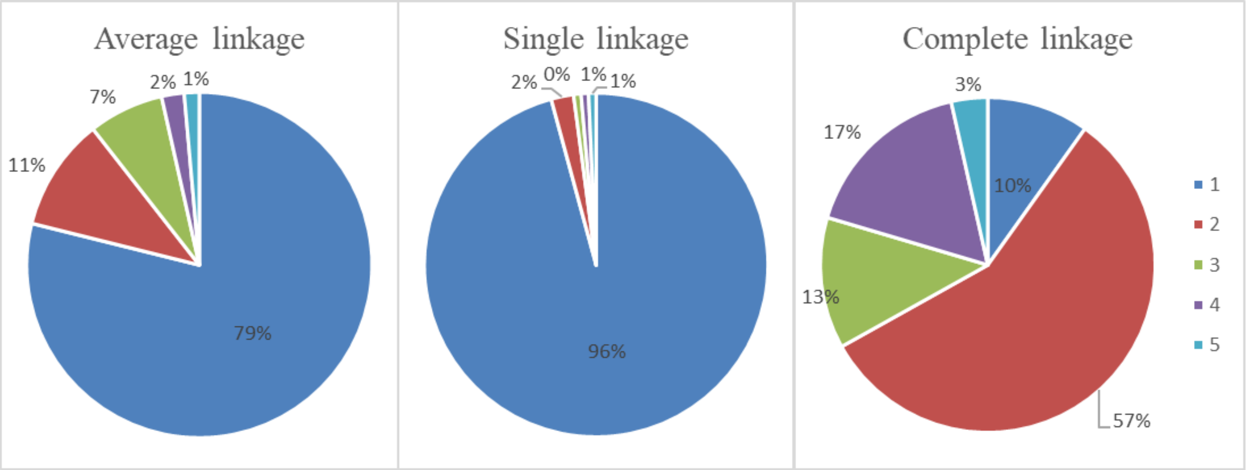


Figure 4.17 Population distribution from agglomerative clustering in first iteration

Single linkage results into large and thin clusters, where elements in the extremes of the cluster can be very far one from each other. In the case of study, the goal is to make categories of similar behaviour, so a more balanced distribution of the population among the clusters is desired. Hence, complete linkage is chosen for the agglomerative clustering.

4.2.5.3 Clustering algorithm

Once the parameters are set, the clustering is performed with the k-medoids and agglomerative methods. Interesting k-medoids outputs are the graphics of the medoids. They help identify the characteristics of each category, which is useful for the validation stage. The medoids for the three iterations are shown in Figure 4.18, Figure 4.19 and Figure 4.20. Each subplot in the following figures represents the medoid for each cluster. They are multivariate subsequences and the channels represented are the relevant channels rather used for candidate identification or clustering stages. The horizontal axe of each graph represents the time (s) and the vertical axe the scaled value of the variable that range from 0 to 1.

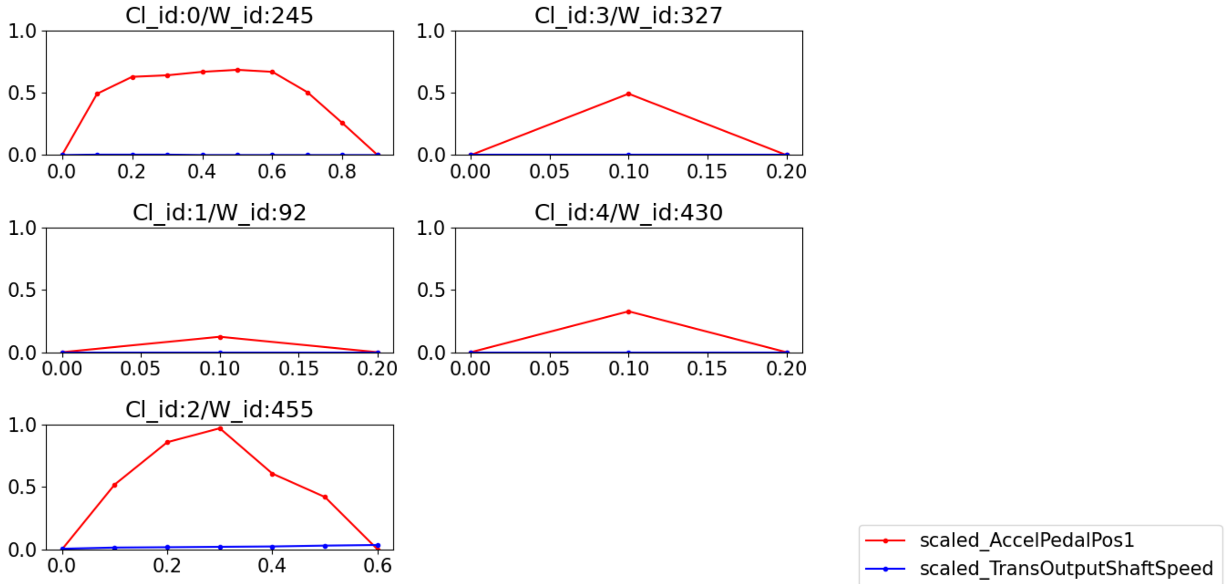


Figure 4.18 Cluster medoids for the first iteration, where BV is AccelPedalPos

In Figure 4.18 the subsequences assigned as medoids by the k-medoids algorithm in the first iteration are showed in the 5 subplots. Just two channels are displayed: the position of the acceleration pedal (red) and the speed transmitted to the wheels (blue). These two channels are those used for subsequence extraction and candidate identification during first iteration.

The medoid in the upper left part of the figure (cluster 0) is a subsequence where acceleration pedal is been pressed for almost one second, but no speed is observed. The medoids in the middle left (cluster 1) and in the right (clusters 3 and 4) correspond to spikes of acceleration of 0.2 s with amplitudes ranging from 0.2 to 0.5% with no observed speed. The length of these *words* seems too small to be caused by a driver pressing the pedal, hence it may be caused by an error in the lecture of the sensors. The medoid of cluster 2 at the bottom left side is about 0.6 seconds length, reaching the maximum acceleration where little speed is observed, attaining 10% of the truck's maximum speed.

Figure 4.19 represents the medoids for the second iteration. The channels in red and blue are the same as in the first iteration and EMPTO_STATUS (black) is represented because it was used to subdivide the time series. What is remarkable is the difference between EMPTO_STATUS *words* among the three sequences and the absence of vehicle speed in any of them.

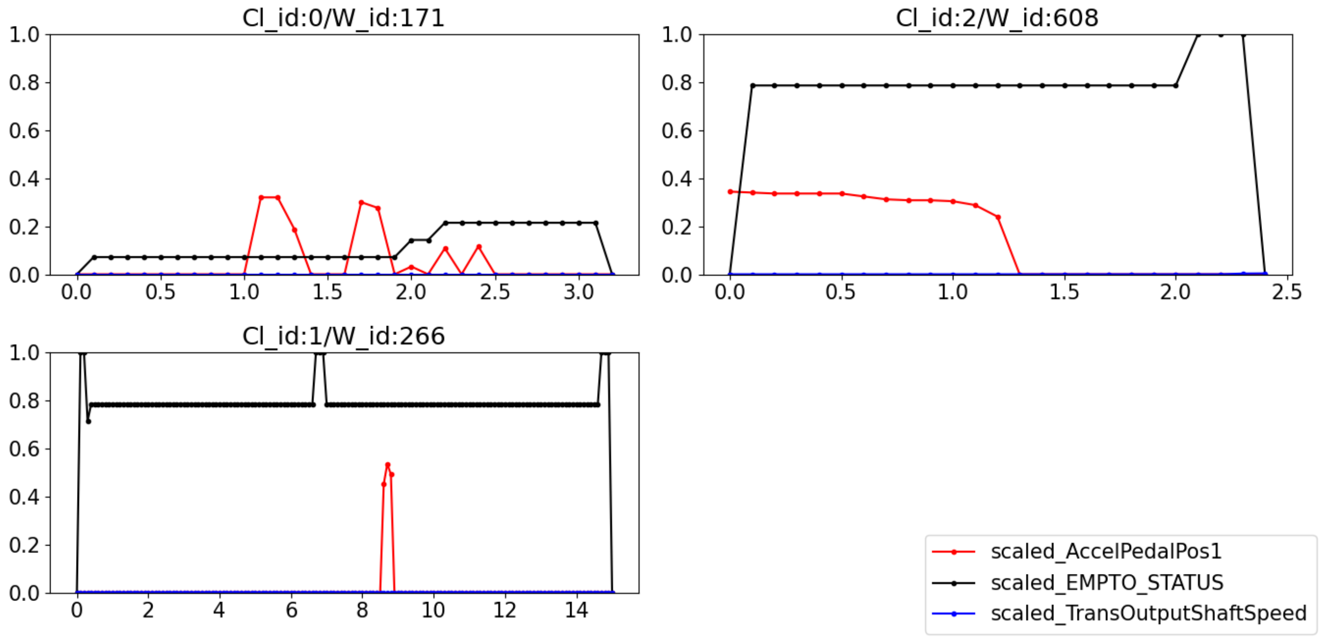


Figure 4.19 Cluster medoids for second iteration where BV is EMPTO_STATUS

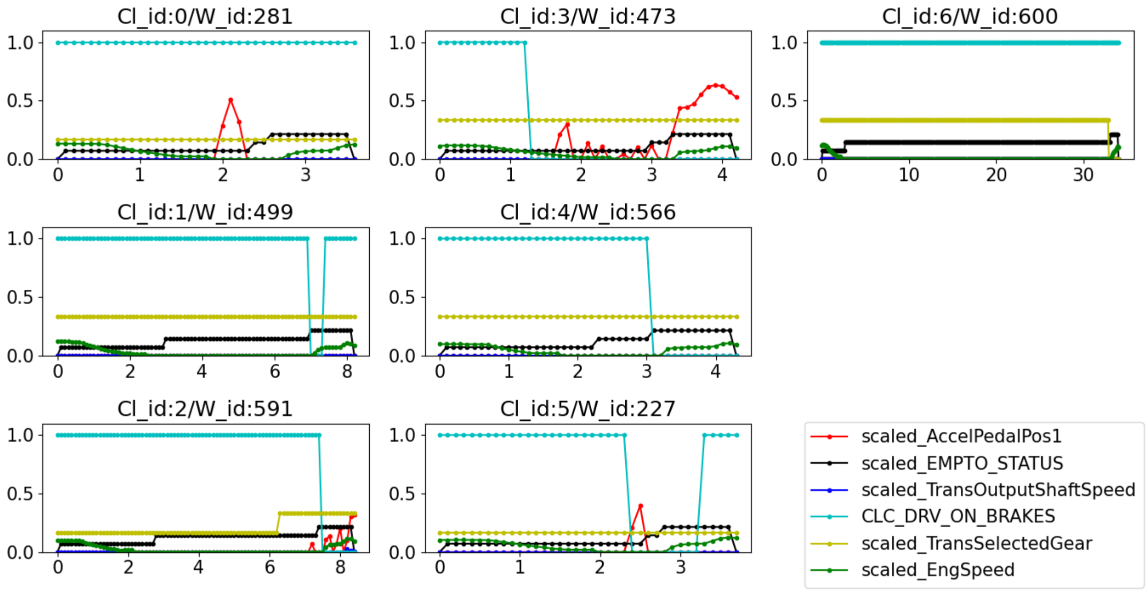


Figure 4.20 Cluster medoids for the third iteration where BV is EMTPO_STATUS

Figure 4.20 represents the medoids for the third iteration of the method. In addition to the channels previously mentioned, for the third iteration, the CLC_DRV_ON_BRAKES channel (light blue) is used to detect if the brakes are being pressed, TransSelectedGear (yellow) represents the gear selected by the driver and scaled_EngSpeed (green) represents the engine speed. It is worth noting the similar engine speed shape in all of the medoids, corresponding to a stop start operation where engine is turned off by the SSM and restarted when the systems detect the driver’s intention to continue driving. In subplots upper left (cluster 0), upper mid (cluster 3), bottom left (cluster 2) and bottom mid (cluster 5), engine restart is caused by the pressing of acceleration pedal. In the central subplot (cluster 4) and left mid subplot (cluster 1), the release of the brake pedal causes the restart. Restart in subplot upper right (cluster 6) is caused by the change of gear. An example of anomalous behavior would be the medoid in the upper left part of the figure (cluster 0), where driver presses the acceleration and brake pedals simultaneously. Another example is the delay in the restart in subplot upper mid (cluster 3), where even though acceleration pedal is pressed for more than 2s, no speed is observed.

Regarding agglomerative clustering, the dendrograms for each iteration are obtained. Branches of the same colour represent elements from the same cluster. They help understand the structure of the clustered data. Outputs for the three iterations are shown in Figure 4.21, Figure 4.22 and Figure 4.23.

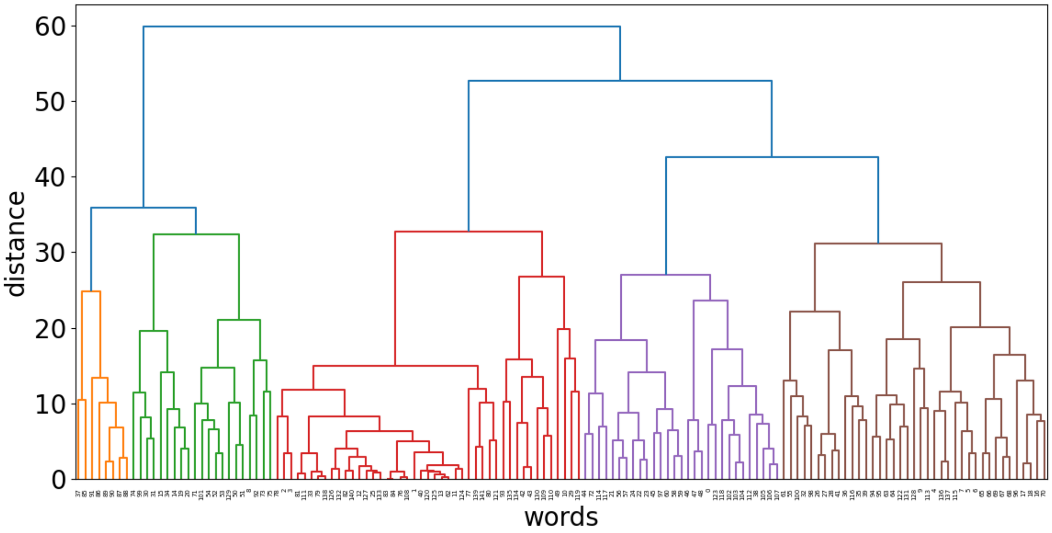


Figure 4.21 Dendrogram for the clusters of the first iteration

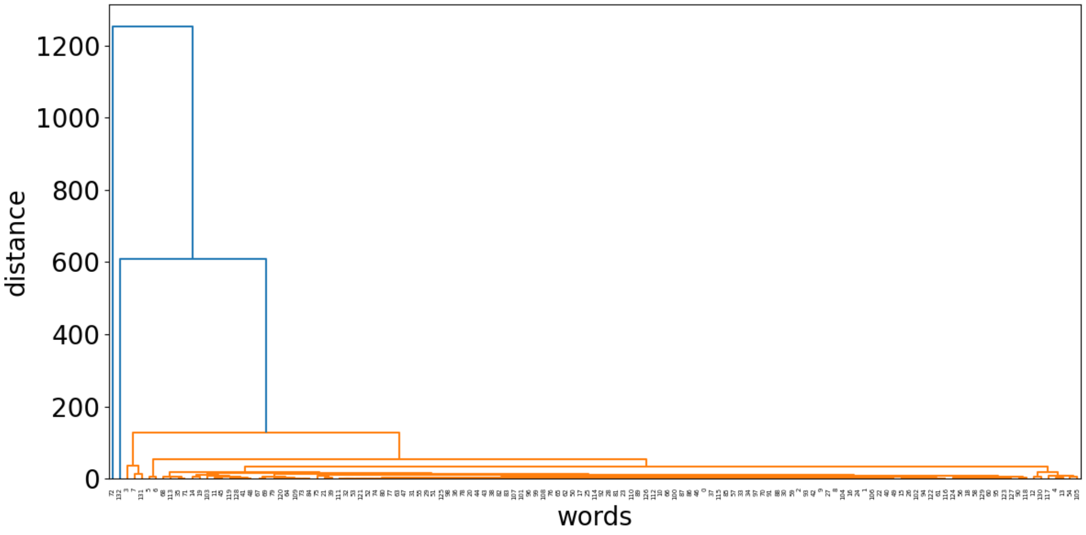


Figure 4.22 Dendrogram for the clusters of the second iteration

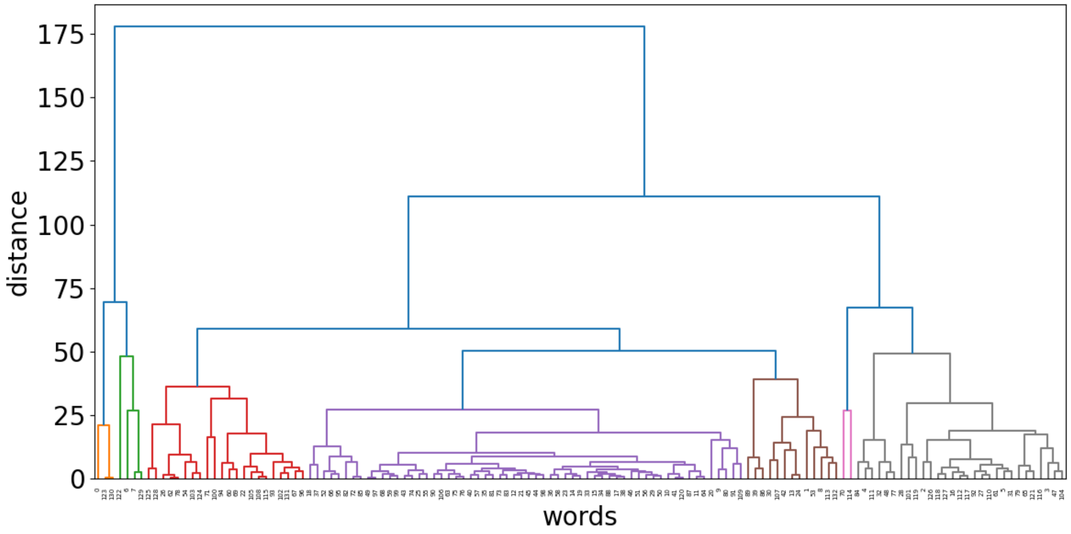


Figure 4.23 Dendrogram for the clusters of the third iteration

The results obtained from the clustering stage are summarized in Table 4.4. It is remarkable that some clusters aggregate the most part of the subsequences, while others remain small. This phenomenon is more frequent for the hierarchical clustering and is particularly noticed in the second iteration, where cluster 0 covers 98.5% of the element total duration. In fact, clustering in the second iteration leads to a general category to which all the individuals, except for two, belong. Two separate individuals are segregated by the clustering.

Table 4.5 Results of clustering for the three iterations of the method

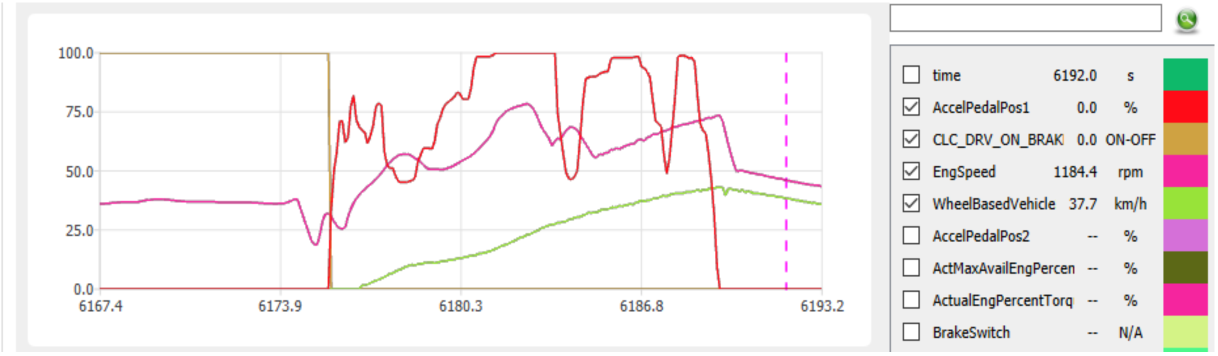
		Number of words		Duration (s)		Share of total time	
Iter	**Cluster**	**K-med**	**Agg**	**K-med**	**Agg**	**K-med**	**Agg**
1	Cl_id 0	14	14	17	22.3	9.9%	9.9%
	Cl_id 1	52	81	22.6	33.2	36.6%	57.0%
	Cl_id 2	17	18	25.2	21.7	12.0%	12.7%
	Cl_id 3	21	24	12.1	23.8	14.8%	16.9%
	Cl_id 4	38	5	25.9	1.8	26.8%	3.5%
2	Cl_id 0	131	131	2578.6	2578.6	98.5%	98.5%
	Cl_id 1	1	1	15.1	2.5	0.8%	0.8%
	Cl_id 2	1	1	2.5	15.1	0.8%	0.8%
3	Cl_id 0	17	43	151.7	361.2	13.2%	9.6%
	Cl_id 1	19	18	118.8	189.6	25.6%	2.7%
	Cl_id 2	49	12	368.5	60.8	14.0%	9.0%
	Cl_id 3	28	5	248.9	90.1	14.0%	13.8%
	Cl_id 4	18	4	950.6	49.1	20.9%	2.1%
	Cl_id 5	2	47	7.6	358.8	3.1%	55.3%
	Cl_id 6	10	14	53.2	789.7	9.3%	7.4%

4.2.6 User validation

After having grouped the candidates by similar dynamic characteristics, the groups are validated by the industrial experts. Two techniques have been used, visual inspection and rule-based validation.

4.2.6.1 Visual inspection

Visual inspection consists in examining one by one a set of candidates to determine whether they are considered as anomalies. To do so, the industrial experts use a visualization tool provided by their enterprise where the time series of all channels can be visualized.



☐ time	6192.0	s	🟩
☑ AccelPedalPos1	0.0	%	🟥
☑ CLC_DRV_ON_BRAK	0.0	ON-OFF	🟨
☑ EngSpeed	1184.4	rpm	🟪
☑ WheelBasedVehicle	37.7	km/h	🟩
☐ AccelPedalPos2	--	%	🟪
☐ ActMaxAvailEngPercen	--	%	🟫
☐ ActualEngPercentTorq	--	%	🟪
☐ BrakeSwitch	--	N/A	🟩

Figure 4.24 Visualization tool for user validation

Figure 4.24 shows the tool, where all the 257 original channels are ready to be displayed (right part of the figure). A list of *words* is provided to the experts as well as a validation sheet to be completed. In this sheet, attributes such as time of start, time of end, length, or anomaly score (AS), are provided. The expert is asked then to add a confirmation note (true/false) to each candidate and add a comment when necessary, explaining the decision.

4.2.6.2 Rule-based validation

Rule-based validation aims at increasing the automatization of the validation process while keeping it supervised. The basic idea is to construct a logic algorithm based on if and else rules to evaluate the candidates and automatically classify them into the known categories. Together with the industrial partner, the algorithm is designed based on the mechanical specifications of the SSM. This validation is implemented in the third iteration merging results from the AccelPedalPos1 *words* and EMPTO_STATUS *words*. An overview of the algorithm is presented in Figure 4.25 and Figure 4.26.

To ensure accurate results of the rule-based validation algorithm, two rounds of exhaustive visual validation are performed for the same file. After the first round, thresholds are adjusted, and additional conditions are defined. The second round is performed to certify that the changes meet the expert criteria.

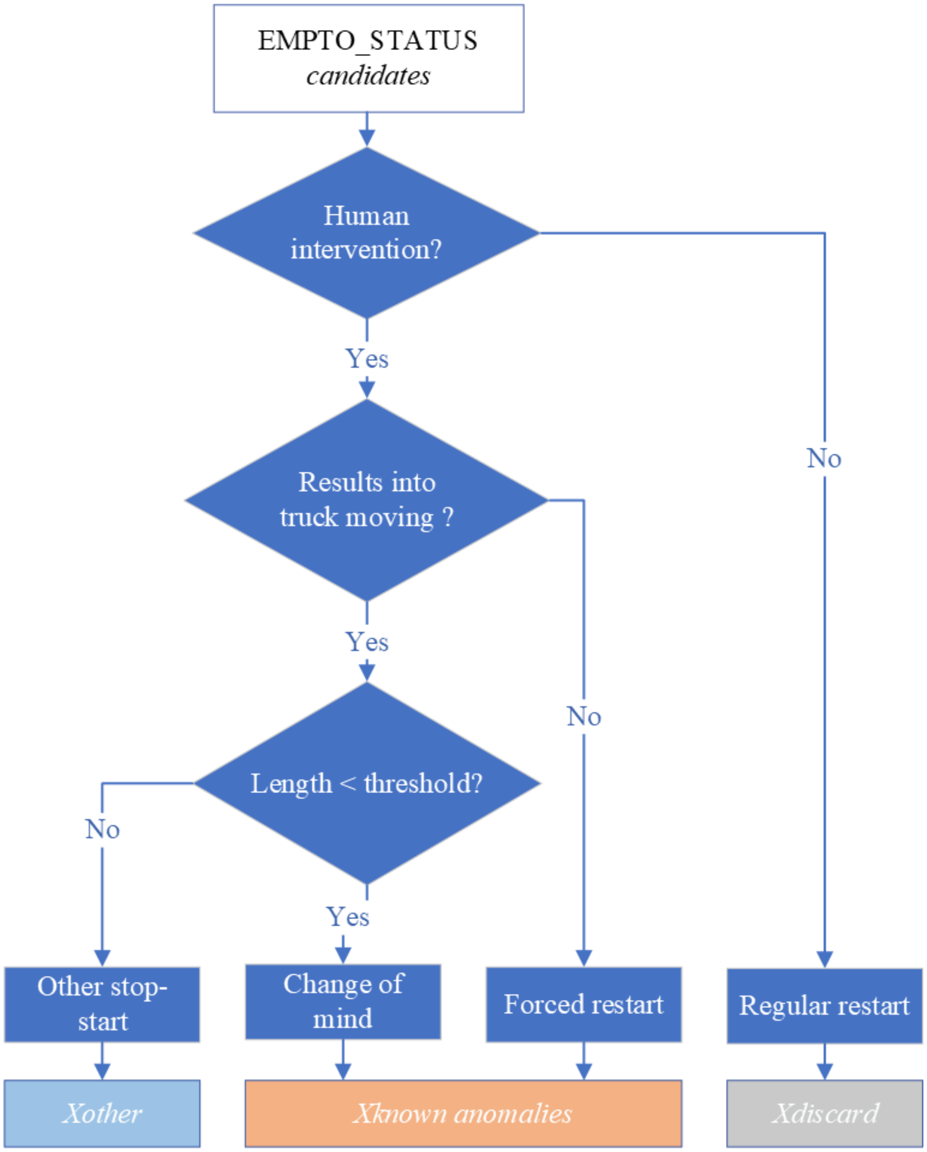


Figure 4.25 Validation algorithm for EMPTO_STAUS *words*

In the schema of Figure 4.25, EMPTO_STATUS candidates refer to the words made from the EMPTO_STATUS channel that meet the criteria for being a candidate to anomaly. The terms "Other stop-start", "Change of mind", "Forced restart" and "Regular restart" refer to behaviours detected by the industrial partner and already known. While the first one is not considered as an anomaly and the last one is discarded for being a regular behaviour, the other two are part of *Xknown* anomalies.

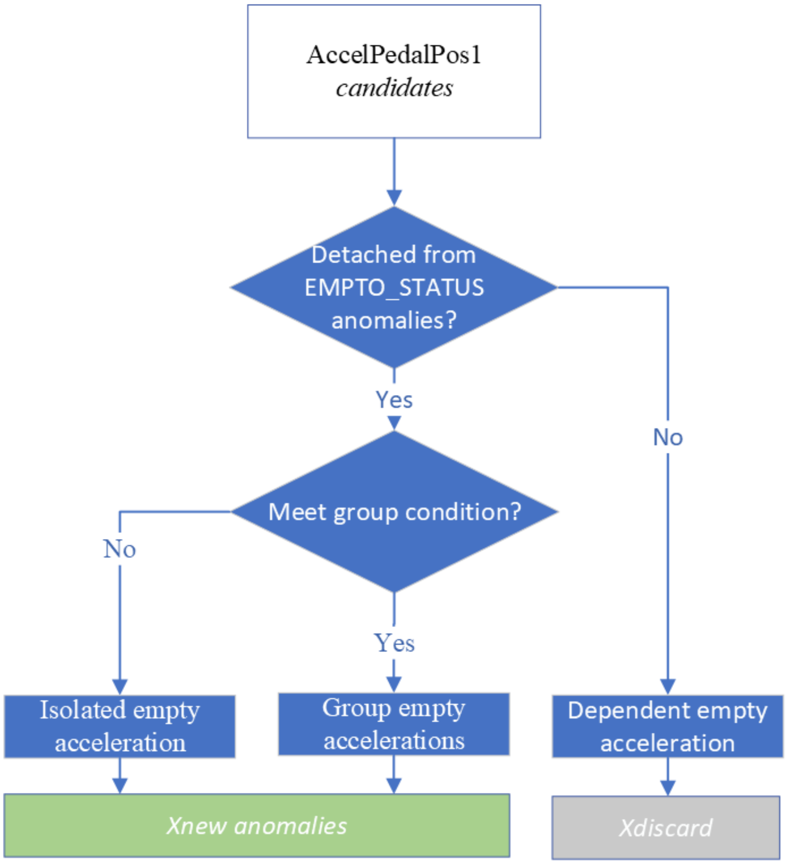

AccelPedalPos1
candidates
Detached from EMPTO_STATUS anomalies?
Yes
No
Meet group condition?
No
Yes
Isolated empty acceleration
Group empty accelerations
Dependent empty acceleration
Xnew anomalies
Xdiscard

Figure 4.26 Validation algorithm for AccelPedalPos1 *words*

The AccelPedalPos1 *words* detected in the first iteration are then incorporated to the results from the EMPTO_STATUS *words*. The visual analysis of the anomalies detected along the three iterations showed that some anomalies were been detected twice in the first and the third iterations. The validation schema shown in Figure 4.26 is used to make sure that anomalies are just detected once and there is no overlapping between anomalies detected in the third iteration and those in the first iteration. The results show a 13,4% of overlap between anomalies detected by both channels.

4.3 Results

The method has been implemented in a real study case as described in the last section. Three iterations of the method have been performed for a single file. For each of the iterations, results of subsequence extraction, identification of candidates, clustering and user validation are now exposed and discussed.

4.3.1 Results of the first iteration

Table 4.5 summarises the results obtained in the first iteration of the method. As explained in the previous sections, in the first iteration, the extraction of subsequences has been performed with the acceleration pedal position channel.

Table 4.6 Summary results of the first iteration

Stage	Label	Number of words		Duration (s)		Share of total time	
2. Subsequence extraction	Total words	876		2827.2		13.66%	
3. Identification of candidates	Xinit	142		102.8		0.50%	
4. Clustering (k-medoids/agglomerative)	C0	14	14	17	22.3	0.08%	0.11%
	C1	52	81	22.6	33.2	0.11%	0.16%
	C2	17	18	25.2	21.7	0.12%	0.10%
	C3	21	24	12.1	23.8	0.06%	0.12%
	C4	38	5	25.9	1.8	0.13%	0.01%
5. User validation	Xnew	19		16.4		0.08%	
	Xknown	0		0		0.00%	
	Xdiscard	123		172.8		0.42%	
	Xother	0		0		0.00%	

From Table 4.5, we observe that for the first iteration, a total of 876 words are extracted, of which 142 are initially considered as candidates to anomalies, representing a 16% from the complete set of words. Clustering leads to silhouette scores of 0.42 for k-medoids and 0.41 for agglomerative clustering. Agglomerative clustering results into clusters with more variance than k-medoids regarding the number of elements for each cluster. User validation leads to 19 new anomalies, meaning that 13.4% of the candidates are validated as anomalies in the first iteration, whereas 86.6% are not considered as anomalies and are then discarded.

4.3.2 Results of the second iteration

In Table 4.6, the results for the second iteration are presented. In this iteration EMPTO_STATUS channel is used to extract the subsequences.

Table 4.7 Summary results for the second iteration

Stage	Label	Number of words		Duration (s)		Share of total time	
2. Subsequence extraction	Total words	627		4578.4		22.13%	
3. Identification of candidates	Xinit	133		2596.2		12.55%	
4. Clustering (k-medoids/agglomerative)	C0	131	131	2578.6	2578.6	12.46%	12.46%
	C1	1	1	15.1	2.5	0.07%	0.01%
	C2	1	1	2.5	15.1	0.01%	0.07%
5. User validation	Xnew	0		0.0		0.0%	
	Xknown	82		435.4		2.1%	
	Xdiscard	36		1451.6		7.0%	
	Xother	15		709.2		3.4%	

The second iteration of the method starts with an extraction of 627 subsequences, of which 21% are considered as potential candidates. The clustering in the second iteration results in a silhouette score of 98% for both k-medoids and agglomerative clustering. Regarding the population of each cluster, it is clear that there is a main cluster C0, where population is concentrated and two outliers in C1 and C2. This distribution in the clusters leads one to think that clustering for the second iteration is not the most appropriate method for separation into categories of similar dynamic behavior. It is possible that the problem lies in the measurement of the distance between individuals. Figure 4.27 shows a sample of subsequences belonging to C0.

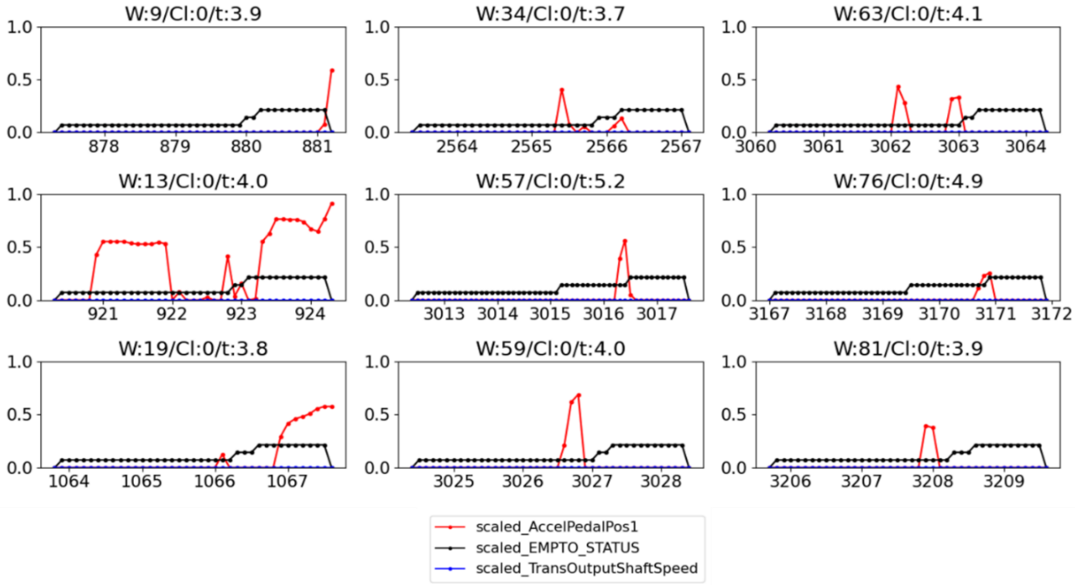


Figure 4.27 Sample of elements from cluster 0 in the second iteration

It is remarkable from this figure the similar shape of EMPTO_STATUS channel (black) in all the subsequences, as anticipated by the shape of the elbow curve. This shape is taken as an anomaly pattern and is employed in the third iteration to calculate the Anomaly Score (AS). The medoid of the cluster is used as the template pattern in the third iteration and the Levenshtein distance to the pattern is used as AS in the third iteration as it was explained previously.

4.3.3 Results of the third iteration

Table 4.7 presents the results for the third iteration of the method. In this iteration, as in the second iteration, the EMPTO_STATUS channel is used to extract the subsequences.

Table 4.8 Summary results for the third iteration

Stage	Label	Number of words		Duration (s)		Share of total time	
2. Subsequence extraction	Total words	627		4578.4		22.13%	
3. Identification of candidates	Xinit	143		1899.3		9.2%	
4. Clustering (k-medoids/agglomerative)	C0	17	43	342.4	1030.8	1.65%	4.98%
	C1	19	18	214.8	8.5	1.04%	0.04%
	C2	49	12	1031.2	63.6	4.98%	0.31%
	C3	28	5	60.8	386.3	0.29%	1.87%
	C4	18	4	151.7	195.0	0.73%	0.94%
	C5	2	47	60.2	208.5	0.29%	1.01%
	C6	10	14	38.2	6.6	0.18%	0.03%
5. User validation	Xnew	0		0.0		0.0%	
	Xknown	103		549.6		2.7%	
	Xdiscard	2		109.2		0.5%	
	Xother	38		1240.5		6.0%	

In the third iteration, 627 subsequences were extracted, and 143 of them were defined as potential candidates. Clustering in the third iteration results in seven clusters with a silhouette score of 0.65 for k-medoids. K-medoids keeps a more balanced population distribution compared to aggregation clustering. There is remarkable variability in terms of duration. Cluster 1 and 2 in k-medoids have almost the same number of anomalies but their length is five times larger on average for cluster 2 than for cluster 1.

For the third iteration, the clustering is also used to classify the anomalies. Following the labeled behaviours described in Figure 4.24, it is interesting to see if the clustering is capable of grouping together similar behaviours already known for the company. Four categories are already known by the company, namely “Regular stop-start”, “Change of mind”, “Forced restart” and “Other restart”. One of these categories is assigned to each cluster. For k-medoids, the category corresponds to the category of its medoid. For example, if the medoid of cluster 0 is labeled as a “Forced restart” following the schema of Figure 4.24, then all elements from cluster 0 will be predicted as “Forced restart”. In agglomerative clustering, as there are no medoids in the cluster, the category of each cluster is determined by the mode of the label of all its objects. The performance of clustering as a way of grouping anomalies with similar characteristics is evaluated through a confusion matrix and a classification report agglomerative clustering and for k-medoids.

Tables 4.8 and 4.9 show the confusion matrix and classification report obtained for agglomerative clustering during the third iteration.

Table 4.9 Confusion matrix for classification by agglomerative clustering in the third iteration

		Predicted category			
		Other stop start	**Change of mind**	**Forced restart**	**Regular restart**
Actual category	**Other stop start**	14	2	22	0
	Change of mind	3	10	12	0
	Forced restart	2	0	76	0
	Regular restart	0	0	2	0

Table 4.10 Classification report for the agglomerative clustering third iteration

	Change of mind	***Forced restart***	***Other Stop-start***	***Regular restart***	***Accuracy***	***Weighted average***
precision	0.83	0.68	0.74	0.00	0.70	0.71
recall	0.40	0.97	0.37	0.00	0.70	0.70
f1-score	0.54	0.80	0.49	0.00	0.70	0.66
support	25	78	38	2	0.7	143

The confusion matrix and classification report for agglomerative clustering give some important insights. The column “Regular restart” in Table 4.8 has all its elements equal to zero. It means that none of the clusters where classed as “Regular restart”. This can be due to the low support, since just two “Regular restart” where found in the validation process. On the other hand, “Forced restart” is well classified through agglomerative clustering, with an especially high recall score. 97% of the “Forced restart” are identified as such, even though its precision is not yet optimized.

“Change of mind” and “Other stop start” categories have significant number of elements off the diagonal of the confusion matrix, leading to low recall scores (0.40 and 0.37 respectively). Overall weighted precision, recall and f1-score are higher than 0.5 meaning the classification through clustering gives more information than just an random classification.

Tables 4.10 and 4.11 represent the confusion matrix and classification report obtained for k-medoids clustering during the third iteration.

Table 4.11 Confusion matrix for classification by k-medoids clustering in the third iteration.

		Predicted category			
		Other stop start	**Change of mind**	**Forced restart**	**Regular restart**
Actual category	**Other stop start**	23	2	13	0
	Change of mind	11	10	4	0
	Forced restart	1	0	77	0
	Regular restart	0	0	2	0

Table 4.12 Classification report k-medoids clustering third iteration

	Change of mind	***Forced restart***	***Other Stop-start***	***Regular restart***	***Accuracy***	***Weighted average***
precision	0.83	0.80	0.66	0.00	0.77	0.76
recall	0.40	0.99	0.61	0.00	0.77	0.77
f1-score	0.54	0.89	0.63	0.00	0.77	0.74
support	25	78	38	2	0.8	143

Better results are obtained with k-medoids. The performance for detecting “Forced restart” is still the best compared to other categories, increasing its precision by 12 points. Precision to detect “Other restart” decreases from 8 points but its recall and f-1 score significantly increase. On the other hand, it is still not possible detect any of the two “Regular restart” and “Change of mind” column in the confusion matrix keeps stays the same, meaning no improvements in the metrics of those categories. Overall weighted metrics are on average 6 points higher than agglomerative clustering.

Table 4.13 Final results after three iterations of the proposed method.

	Anomalous subsequences	**Duration (s)**	**Share of total time**
Xnew	19	16.4	0.08%
Xknown	103	549.6	2.66%
Xdiscard	125	282.0	0.95%
Xother	37	1240.5	5.99%
Total	284	2088.5	9.68%

In the above sections, the results of the three iterations of the proposed method have been presented. The results from the three iterations are then summed up and presented in Table 4.13.

It is important to notice that the final results are not the sum of the partial results from the three iterations because there are anomalies that have been detected in more than one iteration that have just been counted once. This overcounting especially affects to the second and third iteration where the subsequences have been extracted using the same basic variable.

To sum up the results, from the original data set of 206923 stamps a set of 284 (100%) subsequences are extracted as possible candidates for anomalies, 122 (43%) of which are validated as anomalies and 125 (44%) are discarded. The lasting 37 (13%) candidates need further analysis to determine whether they are anomalous subsequences or not.

In addition to the number of anomalies accurately detected, the output for the industrial partner is a visual tool that identifies and flags anomalous subsequences. The tool is integrated in its visualization program as an abnormal flag. Although the automatically detection of anomalies is still to be improved because it has just been performed for one file and the results show 43% of anomalies accurately detected, the addition of this flags results into a faster identification of anomalies in manual diagnosis phase.

CHAPTER 5 CONCLUSION AND RECOMMENDATIONS

The project, carried out in collaboration with an industrial partner from the transport sector, was aimed at developing a method for the detection of anomalies in time series with applications in the field of vehicle monitoring.

To carry it out, the current methods for time series anomaly detection have been studied, as well as the applications related to vehicle monitoring. Inspired by the studied methods and applications, an iterative method for the detection of known and new anomalies has been developed. This method has been applied in collaboration with an industrial partner in a real case of vehicle monitoring. As a result, the article "*Anomaly detection method applied to vehicle monitoring*" has been presented in the 10th International Federation of Automatic Control conference on Manufacturing Modelling, Management and Control in June 2022 (Garcia *et al.*, 2022).

A synthesis of the results obtained is presented below, followed by the limitations identified in the method, as well as the guidelines for the improvement of the method and possible lines of continuation of the project.

5.1 Synthesis

Three iterations of the method have been performed, using different channels for the subdivision of the time series, and employing two different clustering algorithms. Through the first iteration of the method, 19 new anomalies have been detected and validated by the industrial partner. These represent 0.08% of total time analyzed. Through the second iteration, a total of 82 already known anomalies have been detected for a total of 2.1% of the analyzed time. Furthermore, in this second iteration, an anomaly pattern was found which was then used in the third iteration. In this third iteration, 103 anomalies already known were automatically detected for a total of 2.7% of the total time analyzed. Taking into account the three iterations, a total of 19 new anomalies and 103 already known anomalies were automatically detected thanks to the proposed method.

In the use of clustering algorithms for automated anomaly classification, k-medoids performs slightly better than agglomerative clustering with an accuracy of 76%, recall of 77%, f1-score of 0.74% and a support of 143.

In addition to the detected anomalies and the classification performance of the clusters, another output from the present work is a tool that supports the diagnosis team of the industrial partner to identify and diagnose the abnormal behaviours. Although is still a work in progress, it already saves time to the enterprise by flagging the anomalous behaviours.

5.2 Limitations

The developed method has some limitations. First, it is a method that takes into account the opinions and expertise of industrial experts to adjust its parameters and choose the algorithms. This fact decreases its scalability and makes it highly dependent on the opinions, sometimes subjective, of the experts, thus losing robustness, especially for the validation stage.

On the other hand, some of the distance measurements, such as Dynamical Time Wrapping can lead to long computation times. Although there are ways to speed it up, if the method scales up, it could lead to much longer computation times.

Finally, there are still limitations linked to the validation stage and the measurement of method performance. Because there is no labeled anomaly data base, there is still no metric to define the performance of the anomaly detection.

5.3 Future work

Some lines of improvement for the method include adopting a way of optimizing the weights of the distance matrix for multivariate subsequence comparison. Currently these weights are the same for all variables and a comprehensive study on how these weights affect the clustering process has not yet been carried out.

In addition, it will be interesting to test the methodology with different data bases. This data can be from different drivers, different trucks, different weather conditions and geography. A deeper study including the variability of the initial data base will help to improve the methodology and reduce overfitting.

Another important aspect to consider for the future is the move to online time series analysis techniques. Since the information from the industrial partner could be processed in real time, the method should be adapted to an online procedure.

In relation to the subjectivity in the definition of anomalies by the industrial partner, another challenge that remains for the future is the integration of different points of view for the validation stage.

Finally, the incorporation of more complex methods such as Long Short-Term Memory (LSTM) and Hidden Markov Models (HMM), which employ trained neural networks could be very interesting lines of work. Having a base method with results easily interpretable by the industrial partner, the incorporation of less intuitive algorithms could help to find anomalies that cannot be detected by industrial experts. One of the requirements for using neural networks is a labeled data set to train the network. In the particular case of our industrial partner, there was no labeled data based, hence no neural networks were applied for in the project. However, the study case developed in this document can be a way of creating a data base where industrial expert's knowledge is used. This data base can then be employed to train complex models that might point out new anomalies that have not been studied yet.

REFERENCES

Aggarwal, C. C., & Yu, P. S. (2001). *Outlier detection for high dimensional data.* In Proceedings of the 2001 ACM SIGMOD international conference on Management of data, (pp. 37-46).

Aghabozorgi, S., Shirkhorshidi, A. S., & Wah, T. Y. (2015). Time-series clustering–a decade review. *Information Systems, 53*, 16-38

Alizadeh, M., Hamilton, M., Jones, P., Ma, J., & Jaradat, R. (2021). Vehicle operating state anomaly detection and results virtual reality interpretation. *Expert Systems with Applications, 177*, 114928

Aremu, O. O., Hyland-Wood, D., & McAree, P. R. (2019). A relative entropy weibull-sax framework for health indices construction and health stage division in degradation modeling of multivariate time series asset data. *Advanced Engineering Informatics, 40*, 121-134

Barnett, V., Lewis, T., & Abeles, F. (1979). Outliers in statistical data. *Physics Today, 32*(9), 73

Bawaneh, M., & Simon, V. (2019, 19-21 Sept. 2019). *Anomaly Detection in Smart City Traffic Based on Time Series Analysis.* In 2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), (pp. 1-6).

Benkabou, S.-E., Benabdeslem, K., Kraus, V., Bourhis, K., & Canitia, B. (2021). Local Anomaly Detection for Multivariate Time Series by Temporal Dependency Based on Poisson Model. *IEEE Transactions on Neural Networks and Learning Systems*, *1*, 1-11

Berndt, D. J., & Clifford, J. (1994). *Using dynamic time warping to find patterns in time series.* In KDD workshop, (pp. 359-370).

Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A review on outlier/anomaly detection in time series data. *ACM computing surveys (CSUR), 54*(3), 1-33

Boriah, S., Chandola, V., & Kumar, V. (2008). *Similarity measures for categorical data: A comparative evaluation.* In Proceedings of the 2008 SIAM international conference on data mining, (pp. 243-254).

Bortolami, D. (2020). CAN-Bus: Introduction and History. Retrieved from https://resources.altium.com/p/Controller-Area-Network-Bus-Introduction-and-History

Chen, J., Xu, X., & Zhang, X. (2020). Fault Detection for Turbine Engine Disk Based on Adaptive Weighted One-Class Support Vector Machine. *Journal of Electrical and Computer Engineering, 2020*, 9898546

Chen, Y., Liu, X., Li, X., Liu, X., Yao, Y., Hu, G., . . . Pei, F. (2017). Delineating urban functional areas with building-level social media data: A dynamic time warping (DTW) distance based k-medoids method. *Landscape and Urban Planning, 160*, 48-60

Cheng, H., Tan, P.-N., Potter, C., & Klooster, S. (2009). *Detection and characterization of anomalies in multivariate time series.* In Proceedings of the 2009 SIAM international conference on data mining, (pp. 413-424).

Chung, F. L. K., Fu, T.-C., Luk, W. P. R., & Ng, V. T. Y. (2001). *Flexible time series pattern matching based on perceptually important points.* In Workshop on Learning from Temporal and Spatial Data in International Joint Conference on Artificial Intelligence,

Cook, A. A., Mısırlı, G., & Fan, Z. (2019). Anomaly detection for IoT time-series data: A survey. *IEEE Internet of Things Journal, 7*(7), 6481-6494

Deloitte. (2020). Future of Automotive Sales and Aftersales. Impact of current industry trends on OEM revenues and profits until 2035. 104

Dong, H., Chen, F., Wang, Z., Jia, L., Qin, Y., & Man, J. (2021). An Adaptive Multisensor Fault Diagnosis Method for High-Speed Train Traction Converters. *IEEE Transactions on Power Electronics, 36*(6), 6288-6302

ElAtia, S., Ipperciel, D., & Zaïane, O. R. (2016). *Data mining and learning analytics: Applications in educational research*: John Wiley & Sons.

Eye, E. (2018). A Complete History Of Predictive Maintainence & Its Place In The World Today. Retrieved from https://www.easterneye.biz/a-complete-history-of-predictive-maintainence-its-place-in-the-world-today/

Fu-Lai, C., Tak-Chung, F., Ng, V., & Luk, R. W. P. (2004). An evolutionary approach to pattern-based time series segmentation. *IEEE Transactions on Evolutionary Computation, 8*(5), 471-489

Fu, T.-c. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence, 24*(1), 164-181

Fuse, T., & Kamiya, K. (2017). Statistical anomaly detection in human dynamics monitoring using a hierarchical dirichlet process hidden markov model. *IEEE Transactions on Intelligent Transportation Systems, 18*(11), 3083-3092

Garcia, P., Agard, B., & Saunier, N. (2022). *Anomaly detection method applied to vehicle monitoring*. Paper presented at the Manufacturing Modelling, Management and Control, Nantes, France,

Gomes, E. F., Jorge, A. M., & Azevedo, P. J. (2014). *Classifying heart sounds using sax motifs, random forests and text mining techniques.* In Proceedings of the 18th International Database Engineering & Applications Symposium, (pp. 334-337).

Grubbs, F. E. (1969). Procedures for detecting outlying observations in samples. *Technometrics, 11*(1), 1-21

Grzesiek, A., Zimroz, R., Śliwiński, P., Gomolla, N., & Wyłomańska, A. (2020). Long term belt conveyor gearbox temperature data analysis–Statistical tests for anomaly detection. *Measurement, 165*, 108124

Guo, C., Li, H., & Pan, D. (2010). *An improved piecewise aggregate approximation based on statistical features for time series mining.* In International conference on knowledge science, engineering and management, (pp. 234-244).

Gupta, M., Gao, J., Aggarwal, C. C., & Han, J. (2014). Outlier Detection for Temporal Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering, 26*(9), 2250-2267

Hamilton, J. D. (2020). *Time series analysis*: Princeton university press.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., & Bengio, Y. (2018). Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*

Hoang, D. H., & Nguyen, H. D. (2018). *A PCA-based method for IoT network traffic anomaly detection.* In 2018 20th International conference on advanced communication technology (ICACT), (pp. 381-386).

Hodge, V. J., & Austin, J. (2004). A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review, 22*(2), 85-126

Islam, H., & Ahmed, T. (2018). *Anomaly clustering based on correspondence analysis.* In 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA), (pp. 1019-1025).

Jiang, M.-F., Tseng, S.-S., & Su, C.-M. (2001). Two-phase clustering process for outliers detection. *Pattern recognition letters, 22*(6-7), 691-700

Jones, M., Nikovski, D., Imamura, M., & Hirata, T. (2014). *Anomaly detection in real-valued multidimensional time series.* In International Conference on Bigdata/Socialcom/Cybersecurity. Stanford University, ASE. Citeseer,

Kadri, F., Harrou, F., Chaabane, S., Sun, Y., & Tahon, C. (2016). Seasonal ARMA-based SPC charts for anomaly detection: Application to emergency department systems. *Neurocomputing, 173*, 2102-2114

Keogh, E., Chu, S., Hart, D., & Pazzani, M. (2001). *An online algorithm for segmenting time series.* In Proceedings 2001 IEEE international conference on data mining, (pp. 289-296).

Khan, S., Liew, C. F., Yairi, T., & McWilliam, R. (2019). Unsupervised anomaly detection in unmanned aerial vehicles. *Applied Soft Computing, 83*, 105650

Lai, K.-H., Zha, D., Xu, J., Zhao, Y., Wang, G., & Hu, X. (2021). *Revisiting time series outlier detection: Definitions and benchmarks.* In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1),

Li, J., Izakian, H., Pedrycz, W., & Jamal, I. (2021). Clustering-based anomaly detection in multivariate time series data. *Applied Soft Computing, 100*, 106919

Li, J., Pedrycz, W., & Jamal, I. (2017). Multivariate time series anomaly detection: A framework of Hidden Markov Models. *Applied Soft Computing, 60*, 229-240

Li, Y., Chattopadhyay, P., & Ray, A. (2015). Dynamic data-driven identification of battery state-of-charge via symbolic analysis of input–output pairs. *Applied Energy, 155*, 778-790

Linker, R. (2022). *Global Automotive Repair and Maintenance Services Industry*. Retrieved from https://www.reportlinker.com/p05817970/Global-Automotive-Repair-And-Maintenance-Services-Industry.html?utm_source=GNW

Liu, Y., Mu, Y., Chen, K., Li, Y., & Guo, J. (2020). Daily activity feature selection in smart homes based on pearson correlation coefficient. *Neural Processing Letters, 51*(2), 1771-1787

Long, K., Wang, G., Xu, Z., & Yang, X. (2020). Data-Driven Prediction Method for Truck Fuel Consumption Based on Car Networking. In *CICTP 2020* (pp. 638-650).

Lovrić, M., Milanović, M., & Stamenković, M. (2014). Algoritmic methods for segmentation of time series: An overview. *Journal of Contemporary Economic and Business Issues, 1*(1), 31-53

Lu, S., & Huang, S. (2020). Segmentation of Multivariate Industrial Time Series Data Based on Dynamic Latent Variable Predictability. *IEEE Access, 8*, 112092-112103

Marchetti, M., Stabili, D., Guido, A., & Colajanni, M. (2016). *Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms.* In 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), (pp. 1-6).

Martín Abadi, A. A., Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems.

McNally, S. (2022). *Benchmarking Report from TMC and Decisiv Details Higher Parts and Labor Expenses*. Retrieved from https://tmc.trucking.org/blog/fleet-maintenance-and-repair-costs-continue-increase-fourth-quarter

Mihailović, D. T., Bessafi, M., Marković, S., Arsenić, I., Malinović-Milićević, S., Jeanty, P., . . . Mihailović, A. (2018). Analysis of solar irradiation time series complexity and predictability by combining Kolmogorov measures and Hamming distance for La Reunion (France). *Entropy, 20*(8), 570

Mouret, F., Albughdadi, M., Duthoit, S., Kouamé, D., Rieu, G., & Tourneret, J.-Y. (2022). Reconstruction of Sentinel-2 derived time series using robust Gaussian mixture models—Application to the detection of anomalous crop development. *Computers and Electronics in Agriculture, 198*, 106983

Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2018). DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access, 7*, 1991-2005

Narayanan, S. N., Mittal, S., & Joshi, A. (2016). *OBD_SecureAlert: An anomaly detection system for vehicles.* In 2016 IEEE International Conference on Smart Computing (SMARTCOMP), (pp. 1-6).

Negi, N., Jelassi, O., Chaouchi, H., & Clemençon, S. (2020). *Distributed online data anomaly detection for connected vehicles.* In 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), (pp. 494-500).

Niennattrakul, V., & Ratanamahatana, C. A. (2007). *On clustering multimedia time series data using k-means and dynamic time warping.* In 2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07), (pp. 733-738).

Petitjean, F., Inglada, J., & Gançarski, P. (2012). Satellite image time series analysis under time warping. *IEEE transactions on geoscience and remote sensing, 50*(8), 3081-3095

Pimentel, M. A. F., Clifton, D. A., Clifton, L., & Tarassenko, L. (2014). A review of novelty detection. *Signal Processing, 99*, 215-249

Poteko, J., Eder, D., & Noack, P. O. (2021). Identifying operation modes of agricultural vehicles based on GNSS measurements. *Computers and Electronics in Agriculture, 185*, 106105

Qin, H., Yan, M., & Ji, H. (2021). Application of Controller Area Network (CAN) bus anomaly detection based on time series prediction. *Vehicular Communications, 27*, 100291

Rasheed, F., & Alhajj, R. (2013). A framework for periodic outlier pattern detection in time-series sequences. *IEEE transactions on cybernetics, 44*(5), 569-582

Roux, M. (2018). A comparative study of divisive and agglomerative hierarchical clustering algorithms. *Journal of Classification, 35*(2), 345-366

Ruff, L., Vandermeulen, R. A., Görnitz, N., Binder, A., Müller, E., Müller, K.-R., & Kloft, M. (2019). Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*

Safaei, M., Ismail, A. S., Chizari, H., Driss, M., Boulila, W., Asadi, S., & Safaei, M. (2020). Standalone noise and anomaly detection in wireless sensor networks: a novel time-series and adaptive Bayesian-network-based approach. *Software: Practice and Experience, 50*(4), 428-446

Saidi, R., Bouaguel, W., & Essoussi, N. (2019). Hybrid feature selection method based on the genetic algorithm and pearson correlation coefficient. In *Machine Learning Paradigms: Theory and Application* (pp. 3-24): Springer.

Salvador, S., & Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis, 11*(5), 561-580

Soleimani, G., & Abessi, M. (2020). DLCSS: A new similarity measure for time series data mining. *Engineering Applications of Artificial Intelligence, 92*, 103664

Spiegel, S., Gaebler, J., Lommatzsch, A., Luca, E. D., & Albayrak, S. (2011). *Pattern recognition and classification for multivariate time series*. Paper presented at the Proceedings of the Fifth International Workshop on Knowledge Discovery from Sensor Data, San Diego, California, (pp. 34–42).

Stein, D. W., Beaven, S. G., Hoff, L. E., Winter, E. M., Schaum, A. P., & Stocker, A. D. (2002). Anomaly detection from hyperspectral imagery. *IEEE signal processing magazine, 19*(1), 58-69

Sun, Y., & Genton, M. G. (2012). Adjusted functional boxplots for spatio-temporal data visualization and outlier detection. *Environmetrics, 23*(1), 54-64

Tamura, K., & Ichimura, T. (2017, 27 Nov.-1 Dec. 2017). *Clustering of time series using hybrid symbolic aggregate approximation.* In 2017 IEEE Symposium Series on Computational Intelligence (SSCI), (pp. 1-8).

Tamura, K., Sakai, T., & Ichimura, T. (2016). *Time series classification using macd-histogram-based sax and its performance evaluation.* In 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), (pp. 002419-002424).

Tavenard, R. (2021). An introduction to Dynamic Time Warping.

Theissler, A. (2014). Anomaly detection in recordings from in-vehicle networks. *Big data and applications, 23*, 26

Theissler, A. (2017). Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection. *Knowledge-Based Systems, 123,* 163-173

Thuy, H. T. T., Anh, D. T., & Chau, V. T. N. (2021). Efficient segmentation-based methods for anomaly detection in static and streaming time series under dynamic time warping. *Journal of Intelligent Information Systems, 56*(1), 121-146

Van Wyk, F., Wang, Y., Khojandi, A., & Masoud, N. (2019). Real-time sensor anomaly detection and identification in automated vehicles. *IEEE Transactions on Intelligent Transportation Systems, 21*(3), 1264-1276

Wang, C., Zhao, Z., Gong, L., Zhu, L., Liu, Z., & Cheng, X. (2018). A distributed anomaly detection system for in-vehicle network using HTM. *IEEE Access, 6*, 9091-9098

Yu, S.-S., Chu, S.-W., Wang, C.-M., Chan, Y.-K., & Chang, T.-C. (2018). Two improved k-means algorithms. *Applied Soft Computing, 68,* 747-755

Zhang, M., Chen, C., Wo, T., Xie, T., Bhuiyan, M. Z. A., & Lin, X. (2017). SafeDrive: Online driving anomaly detection from large-scale vehicle data. *IEEE Transactions on Industrial Informatics, 13*(4), 2087-2096

Zhu, K., Chen, Z., Peng, Y., & Zhang, L. (2019). Mobile edge assisted literal multi-dimensional anomaly detection of in-vehicle network using LSTM. *IEEE Transactions on Vehicular Technology, 68*(5), 4275-4284

APPENDIX A CORRELATION HEATMAP OF ALL NUMERICAL VARIABLES

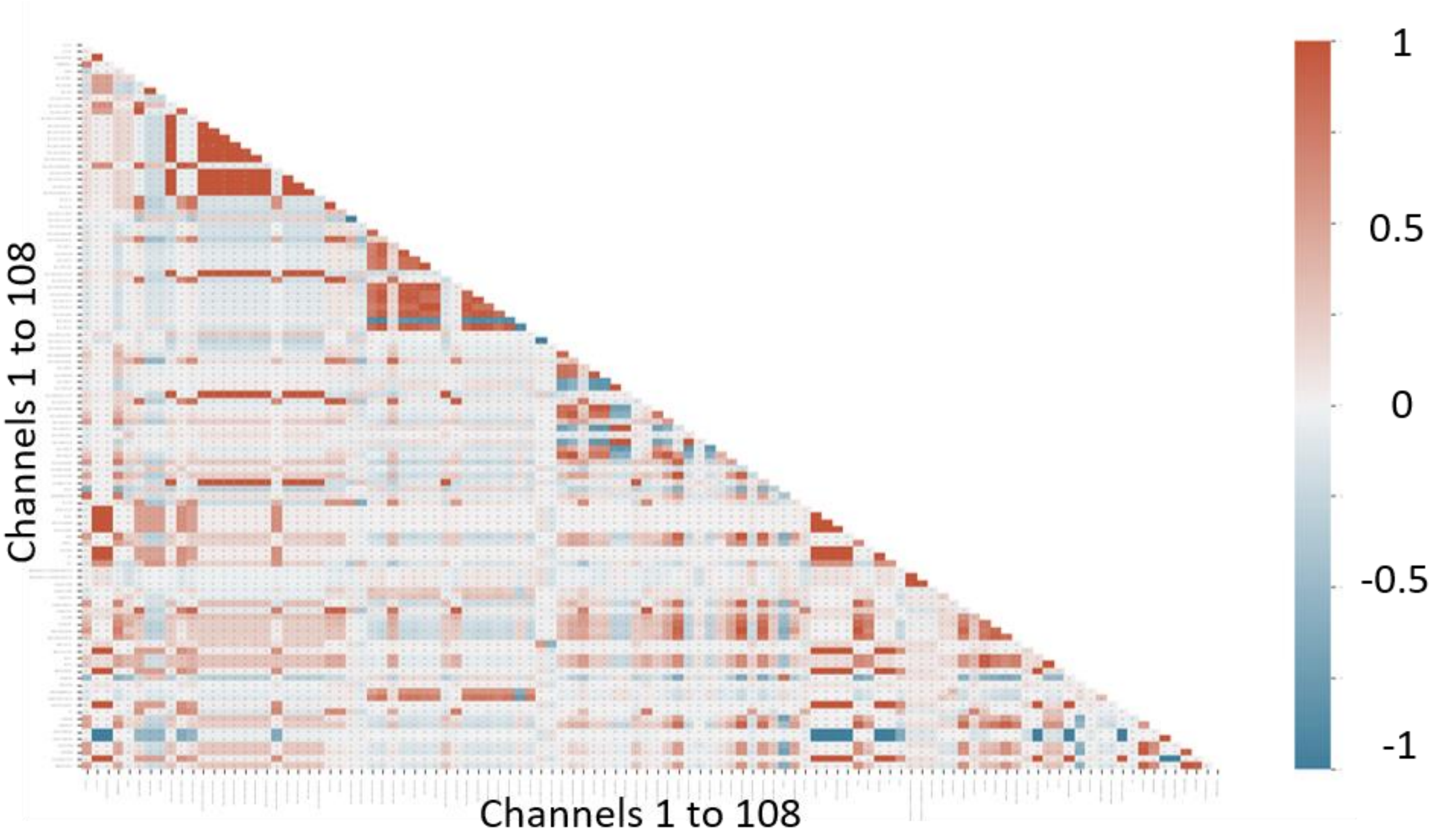


Figure A.1. Correlation heatmap of all numerical variables