



Multivariate Time-Series Anomaly Detection with Temporal Self-supervision and Graphs: Application to Vehicle Failure Prediction

Hadi Hojjati^{1,2}(✉), Mohammadreza Sadeghi^{1,2}, and Narges Armanfard^{1,2}

¹ Department of Electrical and Computer Engineering, McGill University, Montreal, Canada

{hadi.hojjati,mohammadreza.sadeghi,narges.armanfard}@mcgill.ca

² MILA - Quebec AI Institute, Montreal, Canada

Abstract. Failure prediction is key to ensuring the reliable operation of vehicles, especially for organizations that depend on a fleet of vehicles. However, traditional approaches often rely on rule-based or heuristic methods that may not be effective in detecting subtle anomalies, rare events, or in more modern vehicles containing a complex sensory network. This paper presents a novel approach to vehicle failure prediction, called mVSG-VFP, which employs self-supervised learning and graph-based techniques. The proposed method realizes the failure prediction task by exploring information hidden in the time-series data recorded through the sensors embedded in the vehicle. mVSG-VFP includes two main components: a graph-based autoencoder that learns representations of normal data while considering the relationship between different sensors and a self-supervised component that maps temporally-adjacent data to similar representations. We propose a novel approach to define the notion of adjacency in vehicle temporal data.

To evaluate mVSG-VFP, we apply it to a dataset comprised of vehicle sensor recordings to identify the abnormal data samples that signal a potential future failure. We performed a flurry of experiments to verify the accuracy of our model and demonstrate it outperforms state-of-the-art models in this task. Overall, the method is robust and intuitive, making it a useful tool for real-world applications.

Keywords: Self-Supervised Learning · Failure Prediction · Time-Series Anomaly Detection · Graph Neural Networks · Predictive Maintenance

1 Introduction

Failure prediction is a crucial aspect of modern predictive maintenance systems in transportation, as it helps to prevent unexpected breakdowns and costly repairs [1, 36]. Modern vehicles are equipped with complex sensory networks,

which are helpful for the early detection of malfunctioning subsystems before they harm other parts of the vehicle. However, analyzing the sheer volume of sensor data is not feasible for a technician. Therefore, an automated framework is needed to extract relevant information from the data efficiently and objectively. One promising approach to achieving this goal is through deep anomaly detection, a family of machine learning algorithms tailored to detect irregular data patterns [30].

1.1 Problem Statement

Vehicles' sensors commonly output the data as multivariate time series, i.e. a set of sequentially recorded points from different sources. Detecting anomalies in such data is highly challenging for several reasons: First, time-series data can be complex, non-stationary and high-dimensional, making it difficult to identify meaningful patterns [9]. Additionally, distinguishing anomalies from variations in driving patterns or weather and road conditions is an arduous task in time series. Furthermore, capturing the temporal relationship in the data is a complex job, particularly since the length of the recordings is variable, and that they are not continuously recorded. Machine learning models, particularly deep learning algorithms, have helped design efficient algorithms that can address part of these challenges. However, most existing methods suffer from four crucial drawbacks [10]: I) they do not consider the inter-relationship between different sensors; therefore, their application is limited to cases in which anomalies can be easily identified by analyzing the data of individual sensors, which is not the case in complex systems such as a vehicle; II) since the time series recording are commonly very long, the models need to split the data into smaller partitions hence losing the long-term temporal relationships; III) most of the existing methods are not capable of identifying the abnormal sensor and they can only perform system-level anomaly detection; IV) most state-of-the-art algorithms define their anomaly score based on a single data point in the time series. However, time series are commonly contaminated with noise, making the sample-based anomaly scores unreliable, causing performance underestimation. To handle this matter, rather than defining a proper anomaly score, they use a biased evaluation protocol – more specifically, they label the whole data points within a segment of time series as abnormal even if only one data point is abnormal. Such relabeling causes a performance overestimation [23].

1.2 Addressing the Challenges: Our Methodology

Very recently, several studies have revisited the existing deep time-series anomaly detection models more closely and concluded that most of them cannot perform better than simple baselines [11, 23]. According to these papers, confirmed by our own experiments, a combination of simple benchmark datasets and biased evaluation metrics has led to the illusion of progress in this field, and the majority of existing algorithms fail in more challenging datasets if an unbiased evaluation protocol is used [11]. One of the very few exceptions was graph-based methods,

which could outperform other baselines in these scenarios [10,23]. Inspired by these findings, in this paper, we develop a graph-based model for tackling the problem of vehicle failure prediction. By representing the data as a graph, we can capture the complex relationships between components, allowing us to detect subtle changes that existing anomaly detection methods may miss [8,40].

To develop a system that can better identify abnormal sensors, we utilize a generative approach wherein we mask one sensor at a time and encourage the model to reconstruct it. In a model trained in this fashion, if it faces a normal sample during inference, we expect it to still have a small reconstruction error. Thus, we can leverage the reconstruction error to detect abnormal sensors [16].

Similar to other machine learning algorithms, graph-based models cannot process long sequences due to memory and computational constraints [10]. Therefore, it is a common practice in the literature to split the data into smaller windows and treat each partition as a single sample when feeding them to the network [28]. As a result of such partitioning, the model ignores the long-term dependencies between adjacent partitions as the partitions are treated as independent and identically distributed (IID) samples during training. This is an undesirable property which limits the performance of models. To overcome this barrier, we leverage the recent progress in self-supervised learning and the unique properties of our dataset to propose a novel and ground-breaking solution. Our idea is based on the notion of contrastive learning – more specifically the proposed maps the adjacent partitions into similar representations. By doing so, the model learns to extract meaningful and discriminative features that can be used for a wide range of downstream tasks [6,15]. Contrastive learning has gained popularity in recent years, particularly in the field of computer vision, where it has achieved state-of-the-art results in tasks such as image recognition and object detection [6]. However, its application in time-series analysis is underexplored mainly because of the fact that it is extremely difficult to define positive and negative pairs in time series [20]. Our method leverages a unique characteristic of vehicle data to overcome this challenge; the data that are recorded during one trip¹ of the vehicle are more similar compared to those that are produced during another trip; e.g. the driver, weather, road, and load conditions are more probably the same during one vehicle trip. Our model uses this unique vehicle data property to learn a better representation of the data. In this paper, we refer to the data recorded in one vehicle trip as *Block*. The Block size varies with the trip duration. We partition each Block into fixed-length time series; these sub-Blocks are referred to as *Segment* throughout this paper. In general, a Block is usually comprised of several data Segments.

Contribution: We can summarize the main contributions of our work as follows:

1. This is the first study that proposes an effective deep learning framework designed based on the unique properties of the vehicle failure prediction task.

¹ We define a trip as a continuous recording of sensors in which the engine is not turned off for more than 5 min.

2. This is the first study on multivariate time-series analysis that can effectively capture the long-term dependencies within and across time series.
3. We realized the proposed idea through the concepts of graphs, contrastive, and generative learnings.
4. In addition to the vehicle-level failure prediction, the proposed method is capable of identifying faulty sensors. This makes the model more interpretable and helps technicians fix the vehicle.
5. As opposed to the other time-series anomaly detection methods that provide an anomaly score based on individual timestamps and a biased protocol, our proposed method directly outputs the segment anomaly score, and we use them to determine if the whole block is anomalous.
6. We demonstrate the effectiveness of the proposed method on a real-world dataset for early engine failure prediction. The proposed method significantly outperforms all existing and SOTA time-series anomaly detection methods in Precision, Recall, and F1 score on average by 11.2%, 0.8% and 7.4%.

2 Related Works

2.1 Vehicle Predictive Maintenance with Machine Learning

Machine learning (ML) has been applied to predictive vehicle maintenance to improve the accuracy of detecting potential vehicle failures and reduce maintenance costs [36]. However, there are still challenges to overcome, such as data quality and availability, the interpretability of the machine learning models, and the need for collaboration between domain experts and data scientists. Machine learning algorithms can analyze sensor data from vehicles to detect anomalies and diagnose faults. In an early attempt, Wang et al. [37] leveraged the vibration signals and a neural network to detect engine and fuel injection system failure. A similar study by Wong et al. [39] proposed a supervised method based on the ensemble of Bayesian extreme learning machines (BELMs) [32] to detect engine faults. In another recent work, Wolf et al. [38] used the data of electric control units (ECU) as the input to a customized deep learning model comprised of Convolutional Neural Networks (CNNs) [25] and Long short-term memory (LSTM) [17] to detect faults in turbocharged engines. Besides engine fault detection, machine learning has been extensively used for failure prediction and diagnosis in other vehicle subsystems. For instance, Rengasamy et al. [29] used CNNs to detect faults in the air pressure system of heavy trucks. Recent progress in the design of autonomous vehicles has also led to the development of innovative failure prediction frameworks. Jeong et al. [21] leveraged the IoT infrastructure of autonomous vehicles to design a predictive maintenance model using neural networks. In another work, van Wyk et al. [41] leveraged CNN and anomaly detection methods to identify anomalies in automated vehicles.

2.2 Time-Series Anomaly Detection

Time-series anomaly detection is a longstanding problem in various fields, such as finance [2], healthcare [14], cybersecurity [3], and industrial maintenance [4].

Before the introduction of machine learning algorithms, linear models such as the autoregressive integrated moving average (ARIMA) [42] were used for modelling and anomaly detection in time series. These models could identify linear patterns in the stationary data but failed to generalize to more complex data.

Following the unprecedented success of machine learning and deep learning algorithms, researchers have started to leverage them for designing effective anomaly detection frameworks for time series. In early attempts, generic anomaly detection algorithms such as Autoencoders (AEs) were applied to time series, ignoring their temporal and spatial relationships [13, 18]. With the introduction of RNNs and their variants for handling temporal data, methods such as LSTM-AE [33] and OmniAnomaly [34] were developed. Recently, transformers became state-of-the-art in analyzing sequential data, and successful methods such as Anomaly Transformers were designed based on them [28]. All mentioned models have shown promising results on common time-series anomaly detection benchmarks. However, recent studies have cast a shadow of doubt over their generalizability power. They have shown that the performance of deep time-series models is not necessarily better than traditional or, in some cases, random baselines, and the superior performance that some papers have reported can be attributed to other factors, such as incorrect evaluation procedures and inappropriate benchmark datasets. Thus, most of the recent progress in deep time-series anomaly detection is not practical for many real-life applications [11, 23, 26].

Recently, graph-based models became the new research trend for anomaly detection. Graph-based models can effectively handle these relationships and dependencies by modelling the data as a graph, where the nodes represent the different sensors and the edges represent the relationships between them. Based on this idea, several graph-based time series anomaly detection methods have been proposed [10, 43, 44]. MTAD-GAT, proposed by Zhao et al. [44], was one of the early methods that used two Graph Attention Networks (GAT) [?] to model the relationship between sensors as well as the temporal data. They trained the model to reconstruct and forecast the normal data simultaneously and defined the anomaly score based on reconstruction and prediction errors. In another similar work, Deng and Hooi [10] proposed GDN, which learns the relationship between different sensors and employs GAT for fitting a forecasting model on normal data. In a recent work, DVGCRN [7], authors used Variational AEs [24] to improve the performance of their model. They jointly modelled the stochastic relationship between different sensors and the multi-level temporal dependencies in each sensor. Generative adversarial networks (GANs) [12] have also shown remarkable improvements when combined with graph-based algorithms. For instance, HAD-MDGAT [45] combines GAN with graph attention networks to simultaneously learn the temporal and spatial relationship between sensors. Another popular direction in the field is modelling the density of normal data using normalizing flows and graph neural networks. Graph-augmented Normalizing Flow (GANF) [8] is proposed based on this idea and learns the density of normal data via factorizing its density and a graph encoder. Anomalies can then be detected as points that lie in low-probability regions.

Parallel to the graph-based models, research in developing contrastive learning methods for anomaly detection also gained momentum [20]. Lately, researchers such as Tack et al. [35], and Li et al. [27] have shown that self-supervision can significantly boost anomaly detection performance on images. Since then, most research has been focused on visual anomaly detection [20]. Recently, a few studies attempted to apply these methods to other data types, such as audio [19] and brain signals [16]. Ho and Armanfard [16] proposed a method for anomaly detection in brain signals using graphs and self-supervision. However, the method is based on the spatial distance between the sensors, which is not applicable to vehicle data since the spatial configuration of the vehicle sensor does not bear meaningful information about their relationship.

Both Graph-based and self-supervised anomaly detection is a relatively young research field, and there is no surprise that the aforementioned algorithms are not thoroughly investigated in the context of real-world applications. In this paper, we aim to take a step toward filling this research gap by proposing a self-supervised graph-based method which is suitable for vehicle failure prediction applications.

3 Proposed Model

Our model aims to detect anomalies in a multivariate time series at entity and sensor levels after training on normal data. The data is commonly a very long time series. To be able to process it, we use a sliding window of size L and stride length m to generate fixed-size Segments. After windowing, the input to our model is a Segmented multivariate time series dataset denoted by $X \in \mathbb{R}^{N \times L \times (K+1)}$, where N is the number of Segments in a mini-batch, L is the window size, and K is the number of sensors. One additional column also denotes the Block ID of the data points. The algorithm produces a vector $y \in \mathbb{R}^N$ where $y_i \in \{0, 1\}$ denotes the true anomaly score of the i^{th} sample, in which 0 denotes normal condition, and 1 is an abnormality which can signal the failure.

Figure 1 shows an overview of our method. In summary, our method is comprised of the following components:

1. First, we use a sliding window to generate fixed-length Segments from the time series and preprocess them. We then construct a feature vector from each Segment. We build mini-batches from the data and augment them.
2. Several GAT layers then process the data. Each sensor represents a node in the graph and is associated with its feature vector. The GAT layers GNN_e map the features into a lower-dimensional representation, and another stack of GAT layers GNN_d tries to reconstruct them.
3. A contrastive loss is applied to the latent space to pull the Segments with the same Block ID closer and push them away from the rest of the batch.
4. Parallel to the above steps, we stochastically mask the data of one sensor and try to generate the data of the affected sensor. We call this self-supervised task the *generative* task.

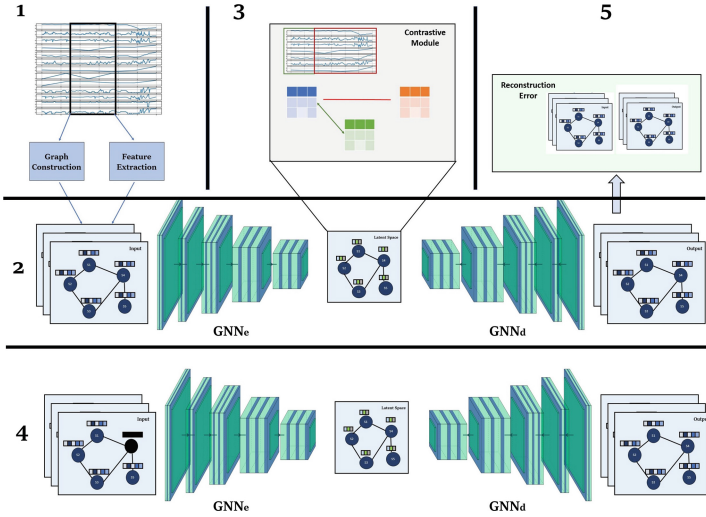


Fig. 1. Overview of the proposed method and its modules.

5. The anomaly score is defined as a combination of the reconstruction error and a score to quantify how accurately the model solves the generative task.

3.1 Data Preprocessing and Feature Construction

As discussed earlier, we Segment the Blocks using a sliding window of size L and stride length of m . To improve the robustness and follow the state-of-the-art literature [10, 28], we normalize each Segment with the maximum and minimum values of the training set as follows:

$$\tilde{x}_i = \frac{x_i - \min(X_{\text{train}})}{\max(X_{\text{train}}) - \min(X_{\text{train}})} \tag{1}$$

where x_i is the i -th sample, X_{train} is the set of all training samples and $\min(\cdot)$ and $\max(\cdot)$ represent the minimum and maximum functions, respectively. Then, from each sensor’s normalized data, we extract a fixed-length feature vector v_i using a feature extractor $v_i = R(\tilde{x}_i)$ ².

3.2 Graph Autoencoder

We represent the data as a graph structure to consider the relationship between sensors. In this graph, the nodes represent sensors, and the edges denote their relationship. If an edge exists between two sensors, it indicates that they are useful for modelling the behaviour of each other. We represent the edges using

² Here $R(\cdot)$ can be any feature extraction function, including predefined feature extraction functions as well as trainable neural networks.

an adjacency matrix \mathcal{A} , where $\mathcal{A}_{ij} = 1$ if there is an edge between the i^{th} and j^{th} sensors. This adjacency matrix can be built by incorporating prior knowledge about sensor relationships or by using a dependency measure between the sensors. In this work, we use a data-driven approach and build the adjacency matrix based on mutual information (MI). MI is a measure of the statistical dependence between two random variables \mathcal{X} and \mathcal{Y} and can be calculated as:

$$I(\mathcal{X}; \mathcal{Y}) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P(x, y) \log\left(\frac{P(x, y)}{P(x)P(y)}\right) \quad (2)$$

where $P(x, y)$ is the joint probability between x and y and $P(x)$ and $P(y)$ are marginal probabilities of x and y , respectively. In the context of our work, the MI is used to quantify the amount of information that is shared between two sensors. We digitize the raw time series Segments into bins to calculate the MI. We build the adjacency matrix as follows:

$$\mathcal{A}_{ij} = \begin{cases} 1 & \text{if } \text{MI}(x_i, x_j) > T \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where T is the connectivity threshold, and $\text{MI}(\cdot)$ is the mutual information.

The feature vector of each sensor is used as the node embedding. We then use GAT layers to process the data. GATs are a type of layer used in graph neural networks to perform message passing and feature aggregation on graph-structured data. GATs enhance the ability of GNNs to capture relationships and interactions between nodes in a graph by using attention mechanisms to weight the neighbouring nodes during message passing. This allows the model to focus on the most relevant nodes for a given task, improving its overall performance.

Mathematically, given a set of node features $V = \{v_1; v_2; \dots; v_K\}$, where $v_i \in \mathbb{R}^F$, we calculate the attention score α_{ij} as follows:

$$e_{ij} = \text{LeakyReLU}(a^T \cdot (\mathbf{W}v_i \oplus \mathbf{W}v_j)) \quad (4)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{q \in \text{adj}\{i\}} \exp(e_{iq})} \quad (5)$$

where ‘ \cdot ’ is the standard vector inner product, $\mathbf{W} \in \mathbb{R}^{F \times F'}$ is the weight matrix of a linear transformation that maps feature space F to F' , $a \in \mathbb{R}^{2F'}$ is a learnable attention vector, \oplus is the concatenation operation, and $\text{adj}\{i\}$ is the set of adjacent nodes of node i . We define a node j to be adjacent to i if $\mathcal{A}_{ij} = 1$. The aggregated representation of node i can then be calculated as:

$$h_i = \text{ReLU}(\alpha_{i,i} \mathbf{W}v_i + \sum_{j \in \text{adj}\{i\}} \alpha_{ij} \mathbf{W}v_j) \quad (6)$$

We stack the GAT layers to encode the node embedding as a low-dimensional representation \hat{h}_i , and then use another set of GAT layers to decode them and reconstruct the features:

$$\hat{v}_i = G(v_i, \mathcal{A}) = \text{GNN}_d(\text{GNN}_e(v_i, \mathcal{A}), \mathcal{A}) \quad (7)$$

The loss function of the network is defined as the reconstruction error of the feature embeddings:

$$\mathcal{L}_{rec} = \frac{1}{N} \sum_{n=1}^N \frac{1}{K} \sum_{i=1}^K \|\hat{v}_i^{(n)} - v_i^{(n)}\| \quad (8)$$

3.3 Graph Generative Learning

We propose a generative task, a self-supervised learning module, to help the model better learn the contextual information and use them in the inference phase for detecting abnormal sensors. It also improves the model generalization.

In this task, we stochastically mask the feature vector of one sensor and use the model and other sensors' data to reconstruct the masked node's embedding. We perform this pretext task during the training on the ζ portion of the mini-batch samples at each epoch. By performing the generative task, the model learns to reconstruct data even in the absence of one sensor embedding. This task operates under the assumption that missing sensor embeddings can be reconstructed by other data points when the data is normal. Consequently, the generative task facilitates anomaly detection by indicating that the sensor data is abnormal or that the other sensors are malfunctioning so that their data is not viable for reconstructing the missing node if the model cannot reconstruct a sensor's embedding. Additionally, the generative task aids in identifying anomalous sensors by concealing their embeddings and evaluating whether the model can successfully reconstruct them.

3.4 Temporal Contrastive Learning

The other notion of self-supervision is Contrastive learning which aims to pull together the Segments with the same Block ID in the latent representation space. Let $X = \{x_1, x_2, \dots, x_N\}$ represents all Segments of our training dataset, and $\mathcal{B} = \{x_1, x_2, \dots, x_{\hat{N}}\}$ be a mini-batch of size \hat{N} that we use for training. \mathcal{B} is random subset of X . For every Segment x_i with Block ID b_i in \mathcal{B} , we stochastically sample another Segment \bar{x}_i with the same Block ID b_i from X , i.e. $\bar{x}_i \sim \{x_j \in X | b_j = b_i, j \in \{1, \dots, N\} \setminus i\}$ where ' \setminus ' denotes that j is any number between 1 to N except for i . This makes sure that for every sample in the mini-batch \mathcal{B} , we have at least one other sample with the same Block ID.

When we Segment the time series using a sliding window, the information about the temporal dependency between different Segments is lost. To encourage our model to map the Segments from the same Block of operation closer together, we employ the idea of contrastive learning. To this end, we first concatenate the latent embedding of sensors \hat{h}_i and pass them through a projection head $f(\cdot)$:

$$z = f(\hat{h}_1 \oplus \hat{h}_2 \oplus \dots \oplus \hat{h}_K) \quad (9)$$

As we might have more than one positive sample for some data points of the batch, we use the SupCON loss [22] instead of the normal contrastive loss. If the representation vectors, i.e. the z vectors associated to the mini-batch segments, are normalized, the SupCON loss can be calculated as:

$$\mathcal{L}_{con} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{a \in A(i)} \exp(z_i \cdot z_a / \tau)} \quad (10)$$

where $I = \{1, 2, \dots, 2N\}$ is the set of indices of the augmented batch (each point has an augmentation, so we have $2N$ points in the augmented batch), $A(i) = I \setminus \{i\}$, $P(i) = \{A(i) : b_p = b_i\}$ is the set of indices of Segments that share the same Block IDs, and τ is called the temperature hyperparameter, and determines the strength of repulsion or attraction between representation vectors.

The final loss function of the network is defined as:

$$\mathcal{L}_{total} = \lambda \mathcal{L}_{rec} + (1 - \lambda) \mathcal{L}_{con} \quad (11)$$

3.5 Anomaly Scoring

To find the anomaly score for a sample x_i during the test phase, we first calculate the reconstruction loss as follows:

$$S_{rec}(x_i) = \|G(x_i, \mathcal{A}) - x_i\| \quad (12)$$

Then, we mask the data of one sensor at a time and re-calculate the average reconstruction error over all masked sensors:

$$S_{gen}(x_i) = \frac{1}{K} \sum_k \|G(M_k(x_i), \mathcal{A}) - x_i\| \quad (13)$$

where $M_k(\cdot)$ is the masking operator which masks out the k -th sensor.

The underlying assumption behind adding the generative loss is that other sensors' information can generate the missing sensor normal data since the model has learned how to leverage the sensors' relationship for data reconstruction. However, for abnormal data, the reconstructed pattern should have a large reconstruction error mainly because the other sensors' data is also abnormal and misses some important structural relationships as well.

4 Experiments

4.1 Dataset

To evaluate the performance of our method, we used a real-world dataset consisting of five trucks. All the trucks are from the same model and same manufacturer and thus have the same set of sensors. The data on the vehicle were recorded

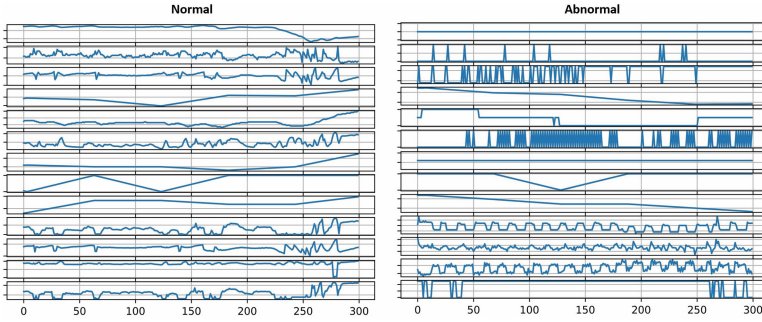


Fig. 2. Visualization of a normal and an abnormal Segment from the dataset. Data are normalized to the range of 0 and 1.

in six months between August 2021 to February 2022. The data also includes information about the operation Blocks of the vehicles, as well as the vehicle's state at each time, both annotated by the company that recorded the data. The vehicle's state is determined by a team of technicians who analyze the data Blocks. The vehicle state identifies if the vehicle is operating in normal condition or if there is a potential defect that may lead to a future failure. Figure 2 shows a visualization of one normal and one abnormal data Segment.

Our work focuses on predicting engine-related failures, and to this end, we picked the data from the same set of sensors that technicians use for engine failure detection and diagnosis. In total, thirteen sensors are used, each monitoring a particular parameter in the engine. The data were recorded with a one-second sampling period. We used a window with length $L = 300$ to partition the data down into 5-minute Segments.

4.2 Evaluation Protocol

We used the labels provided by the company as the ground truth. To evaluate the performance of our algorithm, we used precision (P), recall (R), and F1 score as our metrics. Calculating the F1 score on single timestamps has shown to underestimate the anomaly detector performance [11], mainly because of two reasons: First, even if the data is abnormal, some points might represent normal patterns. Furthermore, for several reasons, one or more normal points might have unexpected values. To alleviate this issue, our proposed algorithm directly outputs the anomaly score for every Segment of the data. Since there are multiple Segments in each Block, we modify the score of different Segments by assigning all of them to abnormal if one anomalous Segment is found in their Block.

More details about the dataset and implementation of the methods can be found in the appendix. We compared our model against several state-of-the-art baselines: One-Class Support Vector Machine (OCSVM) which is a popular traditional anomaly detection method [31], Autoencoder (AE) [13] which is one of the most common tools for anomaly detection [30], LSTM-Autoencoder

Table 1. Precision, Recall, and F1 of Vehicles. Training and test are done on the same vehicle model. V_i is short for the i -th vehicle. The best-performing model is denoted in boldface, and the second best is marked by *.

Algorithm	Metric	V_1	V_2	V_3	V_4	V_5	Average
OCSVM	Prec	0.56	0.48	0.49	0.63	0.46	0.52
	Recall	0.81	0.74	0.71	0.83	0.77	0.77
	F1	0.66	0.58	0.58	0.71	0.57	0.62
AE	Prec	0.58	0.51	0.53	0.67	0.58	0.57
	Recall	0.88	0.83	0.85	0.91	0.81	0.85
	F1	0.69	0.83*	0.65	0.71	0.676	0.68
LSTM-AE	Prec	0.63	0.59	0.63	0.67	0.62	0.62
	Recall	0.93	0.88	0.86	0.97	0.91	0.91
	F1	0.75	0.70	0.72	0.79	0.73	0.74
TCN-AE	Prec	0.61	0.52	0.57	0.63	0.59	0.58
	Recall	0.96	0.93	0.91	0.95	0.93*	0.93
	F1	0.74	0.66	0.70	0.75	0.72	0.71
USAD	Prec	0.68	0.57	0.64*	0.70	0.59	0.63
	Recall	0.92	0.90	0.94*	0.95	0.88	0.91
	F1	0.78	0.69	0.76*	0.80	0.706	0.75
GDN	Prec	0.72*	0.63*	0.61	0.72*	0.68*	0.67*
	Recall	0.94*	0.92*	0.92	0.98	0.93*	0.93*
	F1	0.81	0.74	0.73	0.83*	0.78*	0.78*
mVSG-VFP	Prec	0.85	0.74	0.71	0.84	0.78	0.78
	Recall	0.94*	0.92*	0.95	0.96*	0.96	0.94
	F1	0.89	0.82	0.81	0.89	0.86	0.85

(LSTM-AE), another autoencoder-based method [33] that uses LSTM layers instead of fully-connected ones to capture the temporal relationship, Temporal Convolutional Network Autoencoder (TCNAE), which uses TCN instead of fully-connected layers, USAD [5] which uses two adversarial-trained autoencoders to detect anomalies, and Graph Deviation Network (GDN) [10], a graph-based method that detects anomalies based on the prediction error.

4.3 Experimental Results

Vehicle-Specific Training: We compared the performance of our proposed model against several common and state-of-the-art algorithms in Table 1. In this experiment, we train the model on the same vehicle for training and testing. We did the experiments ten times and reported the average.

These results demonstrate the superiority of our method over other baselines in detecting abnormal events which lead to a potential vehicle failure. In terms

of F1 and Precision, we can confirm that our model outperforms all the methods in all vehicles and has superior Recall compared to the baselines on average. A remarkable observation is that graph-based methods, such as our algorithm and GDN, outperform other methods. This highlights the key role of capturing sensor relationships in detecting multivariate time-series anomalies. Furthermore, the results show that our proposed model can outperform GDN on all vehicles. This can be attributed to the two main differences between our model and GDN:

1. Our model is reconstruction-based, while GDN is predictive-based and defines the anomaly score based on the deviation from expected future behaviour. Although it showed promising results on benchmarks such as SWAT, the predictive-based nature of GDN limits its performance on data that does not possess predictable temporal patterns. As a result, the GDN model will have a large prediction error even on normal samples of our dataset, which is also suggested by its low precision score in the table.
2. An essential component of our model is the self-supervised module which encourages the network to build a more representative feature space. This module helps capture the long-term dependencies of Segments within a Block, while this relationship is discarded in GDN and other baselines.

Cross-Vehicle Anomaly Detection: A possible vehicle failure prediction system scenario is to train the model using data from a subset of vehicles and deploy it on a new one. To assess the reliability of our framework in this situation, we held one vehicle out for the test and trained the model using the normal samples of the other four. During the test phase, we used the test dataset of the held-out vehicle to measure the performance. The results of this experimental protocol are shown in Table 2.

Table 2. Results of the Cross-vehicle training.

	Metric	V ₁	V ₂	V ₃	V ₄	V ₅	<i>Average</i>
Cross Vehicle	Prec	0.65	0.53	0.51	0.65	0.68	0.60
	Recall	0.74	0.80	0.77	0.77	0.89	0.79
	F1	0.69	0.63	0.61	0.70	0.67	0.66

Comparing these results with Table 1, we can see that the cross-vehicle training has a lower performance than the vehicle-specific training protocol. This can be attributed to the fact that the vehicle operates in different weather and load conditions and has different depreciation levels. Therefore, having access to the recordings of the same vehicle or other vehicles that operate under similar conditions during training can help the model to perform better.

Anomalous Sensor Detection: An interesting aspect of a failure prediction method is its interpretability and the ability to localize the defect location. Therefore, we explore the model’s performance for detecting anomalous sensors in this experiment. Since our original dataset did not have the annotation for the abnormal sensors, we devised synthetic data by replacing the embedding of one of the sensors of the normal data with noise. Then we mask one node at a time and reconstruct it. We threshold the reconstruction errors to get the anomaly labels. This experiment yielded a 97.58% F1 score, highlighting the proposed model’s application for localizing abnormal channels.

Effect of Hyperparameter λ : Figure 3 shows the average F1 score of our model for different values of λ . The two extreme cases of $\lambda = 0$ and $\lambda = 1$ represent our model if we remove the reconstruction and contrastive losses, respectively. We can confirm that including both losses with $\lambda = 0.8$ yields the best performance. This shows that both loss terms complement each other and can improve the model’s efficiency in detecting abnormal patterns. We can still achieve good performance for the case of $\lambda = 1$, which effectively means keeping the reconstruction error alone. However, setting $\lambda = 0$ and removing the construction loss on the other side of the spectrum can significantly degrade the F1 score. This can be attributed to two main causes: I) If we remove the reconstruction loss, the model can trivially minimize the contrastive loss by concentrating the samples of the same Block in one single point [18]. Therefore, the latent space will not represent all data characteristics, II) We defined the anomaly score based on the reconstruction error of the input. If we exclude the reconstruction loss, the model will not be guided to reconstruct the normal samples, and thus, it will have a large reconstruction error on both normal and abnormal samples.

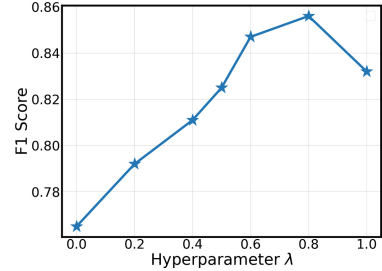


Fig. 3. Effect of Hyperparameter λ

5 Conclusion

Overall, our work presents a significant step forward in predictive vehicle maintenance. By combining the power of graph-based anomaly detection with the unique characteristics of vehicle sensor data, our approach has the potential to improve the reliability and efficiency of transportation systems significantly. Our method identifies defects in the sensor data by modelling the sensor network as a graph and using leveraging self-supervised learning to capture temporal dependency between the features. We evaluated our model on a dataset which includes annotated ground truth and showed that our model achieved promising results for detecting the anomalies that lead to the failure. The results of this work demonstrate the potential of graph-based and contrastive learning in multivariate time-series anomaly detection for solving real-world problems.

Acknowledgement. We would like to express our sincere gratitude to Ken Sills, CTO and Co-Founder of Preteckt Inc. company and his team for their invaluable support in providing us with the data used in this research paper. Their contribution was crucial in enabling us to analyze and draw meaningful conclusions from the dataset. We would also acknowledge Fonds de Recherche du Quebec Nature et technologies (FRQNT), Natural Sciences and Engineering Research Council of Canada (NSERC), and Scale AI for funding this research project.

Ethical Statement. We acknowledge that our research involves collecting and processing potentially sensitive data. We have taken measures to protect the privacy and confidentiality of the organizations represented in the data. We have obtained all necessary permissions and approvals. We recognize that our work has implications for the automotive industry, and we acknowledge our responsibility to consider these implications carefully. We have taken care to report our findings accurately and transparently, and we have made efforts to minimize any potential negative impacts of our research.

References

1. Achouch, M., et al.: On predictive maintenance in industry 4.0: overview, models, and challenges. *Appl. Sci.* **12**(16) (2022). <https://doi.org/10.3390/app12168081>. <https://www.mdpi.com/2076-3417/12/16/8081>
2. Ahmed, M., Mahmood, A.N., Islam, M.R.: A survey of anomaly detection techniques in financial domain. *Future Gener. Comput. Syst.* **55**, 278–288 (2016). <https://doi.org/10.1016/j.future.2015.01.001>. <https://www.sciencedirect.com/science/article/pii/S0167739X15000023>
3. Ahmed, M., Naser Mahmood, A., Hu, J.: A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **60**, 19–31 (2016). <https://doi.org/10.1016/j.jnca.2015.11.016>. <https://www.sciencedirect.com/science/article/pii/S1084804515002891>
4. Atha, D.J., Jahanshahi, M.R.: Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection. *Struct. Health Monit.* **17**(5), 1110–1128 (2018). <https://doi.org/10.1177/1475921717737051>
5. Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A.: USAD: unsupervised anomaly detection on multivariate time series. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2020*, pp. 3395–3404. Association for Computing Machinery, New York (2020). <https://doi.org/10.1145/3394486.3403392>
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: *International Conference on Machine Learning*, pp. 1597–1607. PMLR (2020)
7. Chen, W., Tian, L., Chen, B., Dai, L., Duan, Z., Zhou, M.: Deep variational graph convolutional recurrent network for multivariate time series anomaly detection. In: *International Conference on Machine Learning*, pp. 3621–3633. PMLR (2022)
8. Dai, E., Chen, J.: Graph-augmented normalizing flows for anomaly detection of multiple time series. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022* (2022)
9. Darban, Z.Z., Webb, G.I., Pan, S., Aggarwal, C.C., Salehi, M.: Deep learning for time series anomaly detection: a survey (2022)

10. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. In: Proceedings of the AAAI Conference on Artificial Intelligence, AAAI 2021 (2021)
11. Garg, A., Zhang, W., Samarani, J., Savitha, R., Foo, C.S.: An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Trans. Neural Netw. Learning Syst.* **33**(6), 2508–2517 (2022). <https://doi.org/10.1109/TNNLS.2021.3105827>
12. Goodfellow, I., et al.: Generative adversarial nets. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K. (eds.) *Advances in Neural Information Processing Systems*, vol. 27. Curran Associates, Inc. (2014)
13. Goodfellow, I.J., Bengio, Y., Courville, A.: *Deep Learning*. MIT Press, Cambridge (2016). <http://www.deeplearningbook.org>
14. Gugulothu, N., Malhotra, P., Vig, L., Shroff, G.M.: Sparse neural networks for anomaly detection in high-dimensional time series (2018)
15. Hjelm, R.D., et al.: Learning deep representations by mutual information estimation and maximization. In: *International Conference on Learning Representations* (2019). <https://openreview.net/forum?id=Bklr3j0cKX>
16. Ho, T.K.K., Armanfard, N.: Self-supervised learning for anomalous channel detection in EEG graphs: application to seizure analysis. In: Proceedings of the AAAI Conference on Artificial Intelligence, AAAI 2023 (2023)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
18. Hojjati, H., Armanfard, N.: DASVDD: deep autoencoding support vector data descriptor for anomaly detection. In: arXiv (2021)
19. Hojjati, H., Armanfard, N.: Self-supervised acoustic anomaly detection via contrastive learning. In: *ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3253–3257 (2022). <https://doi.org/10.1109/ICASSP43922.2022.9746207>
20. Hojjati, H., Ho, T.K.K., Armanfard, N.: Self-supervised anomaly detection: a survey and outlook (2022). <https://doi.org/10.48550/ARXIV.2205.05173>. <https://arxiv.org/abs/2205.05173>
21. Jeong, K., Choi, S.B., Choi, H.: Sensor fault detection and isolation using a support vector machine for vehicle suspension systems. *IEEE Trans. Veh. Technol.* **69**(4), 3852–3863 (2020). <https://doi.org/10.1109/TVT.2020.2977353>
22. Khosla, P., et al.: Supervised contrastive learning. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) *Advances in Neural Information Processing Systems*, vol. 33, pp. 18661–18673. Curran Associates, Inc. (2020)
23. Kim, S., Choi, K., Choi, H.S., Lee, B., Yoon, S.: Towards a rigorous evaluation of time-series anomaly detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 7, pp. 7194–7201, June 2022. <https://doi.org/10.1609/aaai.v36i7.20680>. <https://ojs.aaai.org/index.php/AAAI/article/view/20680>
24. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. *CoRR* abs/1312.6114 (2013)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) *Advances in Neural Information Processing Systems*, vol. 25. Curran Associates, Inc. (2012)
26. Lai, K.H., Zha, D., Xu, J., Zhao, Y., Wang, G., Hu, X.: Revisiting time series outlier detection: definitions and benchmarks. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)* (2021). <https://openreview.net/forum?id=r8IvOsnHchr>

27. Li, C.L., Sohn, K., Yoon, J., Pfister, T.: CutPaste: self-supervised learning for anomaly detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9664–9674 (2021)
28. Li, D., Chen, D., Jin, B., Shi, L., Goh, J., Ng, S.-K.: MAD-GAN: multivariate anomaly detection for time series data with generative adversarial networks. In: Tetko, I.V., Kurková, V., Karpov, P., Theis, F. (eds.) ICANN 2019. LNCS, vol. 11730, pp. 703–716. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30490-4_56
29. Rengasamy, D., Jafari, M., Rothwell, B., Chen, X., Figueredo, G.P.: Deep learning with dynamically weighted loss function for sensor-based prognostics and health management. *Sensors* **20**(3) (2020). <https://doi.org/10.3390/s20030723>. <https://www.mdpi.com/1424-8220/20/3/723>
30. Ruff, L., et al.: A unifying review of deep and shallow anomaly detection. *Proc. IEEE* **109**(5), 756–795 (2021). <https://doi.org/10.1109/JPROC.2021.3052449>
31. Schölkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J., Platt, J.: Support vector method for novelty detection. In: Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS 1999, pp. 582–588. MIT Press, Cambridge (1999)
32. Soria-Olivas, E., et al.: BELM: Bayesian extreme learning machine. *IEEE Trans. Neural Netw.* **22**(3), 505–509 (2011). <https://doi.org/10.1109/TNN.2010.2103956>
33. Srivastava, N., Mansimov, E., Salakhutdinov, R.: Unsupervised learning of video representations using LSTMs. In: ICML (2015)
34. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, pp. 2828–2837. Association for Computing Machinery, New York (2019). <https://doi.org/10.1145/3292500.3330672>. <https://doi.org/10.1145/3292500.3330672>
35. Tack, J., Mo, S., Jeong, J., Shin, J.: CSI: Novelty detection via contrastive learning on distributionally shifted instances. *Adv. Neural. Inf. Process. Syst.* **33**, 11839–11852 (2020)
36. Theissler, A., Pérez-Velázquez, J., Kettelgerdes, M., Elger, G.: Predictive maintenance enabled by machine learning: use cases and challenges in the automotive industry. *Reliab. Eng. Syst. Safety* **215**, 107864 (2021). <https://doi.org/10.1016/j.res.2021.107864>. <https://www.sciencedirect.com/science/article/pii/S0951832021003835>
37. Wang, M.H., Chao, K.H., Sung, W.T., Huang, G.J.: Using ENN-1 for fault recognition of automotive engine. *Expert Syst. Appl.* **37**(4), 2943–2947 (2010). <https://doi.org/10.1016/j.eswa.2009.09.041>. <https://www.sciencedirect.com/science/article/pii/S0957417409008227>
38. Wolf, P., Mrowca, A., Nguyen, T.T., Bäker, B., Günemann, S.: Pre-ignition detection using deep neural networks: a step towards data-driven automotive diagnostics. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 176–183 (2018). <https://doi.org/10.1109/ITSC.2018.8569908>
39. Wong, P.K., Zhong, J., Yang, Z., Vong, C.M.: Sparse Bayesian extreme learning committee machine for engine simultaneous fault diagnosis. *Neurocomputing* **174**, 331–343 (2016). <https://doi.org/10.1016/j.neucom.2015.02.097>. <https://www.sciencedirect.com/science/article/pii/S09525231215011765>
40. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(1), 4–24 (2020)

41. van Wyk, F., Wang, Y., Khojandi, A., Masoud, N.: Real-time sensor anomaly detection and identification in automated vehicles. *IEEE Trans. Intell. Transp. Syst.* **21**(3), 1264–1276 (2020). <https://doi.org/10.1109/TITS.2019.2906038>
42. Zare Moayedi, H., Masnadi-Shirazi, M.: Arima model for network traffic prediction and anomaly detection. In: 2008 International Symposium on Information Technology, vol. 4, pp. 1–6 (2008). <https://doi.org/10.1109/ITSIM.2008.4631947>
43. Zhang, W., Zhang, C., Tsung, F.: GRELEN: multivariate time series anomaly detection from the perspective of graph relational learning. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, pp. 2390–2397 (2022)
44. Zhao, H., et al.: Multivariate time-series anomaly detection via graph attention network. In: 2020 IEEE International Conference on Data Mining (ICDM), pp. 841–850. IEEE (2020)
45. Zhou, L., Zeng, Q., Li, B.: Hybrid anomaly detection via multihead dynamic graph attention networks for multivariate time series. *IEEE Access* **10**, 40967–40978 (2022). <https://doi.org/10.1109/ACCESS.2022.3167640>