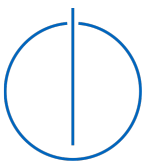
SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Robotics, Cognition, Intelligence

Unsupervised Learning for Detection of Rare Driving Scenarios

Dat Le



SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

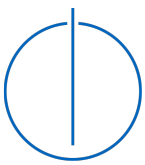
TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Robotics, Cognition, Intelligence

Unsupervised Learning for Detection of Rare Driving Scenarios

Unüberwachtes Lernen zur Erkennung von seltenen Fahrszenarien

Author:	Dat Le
Examiner:	Prof. Dr. Johannes Betz
Supervisor:	M.Sc. Thomas Manhardt, Dr. Moritz Venator
Submission Date:	January 30, 2025



I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, January 30, 2025 Dat Le

Acknowledgments

First, I would like to thank my thesis advisor, M.Sc. Thomas Manhardt, for his invaluable guidance and constructive feedback, especially during moments of difficulty and uncertainty.

Next, I would like to thank Dr. Moritz Venator for his mentorship. His guidance, motivation, and feedback has helped me become a better researcher.

Additionally, I would like to thank my supervisor, Prof. Dr. Johannes Betz, who provided me with the opportunity to collaborate on a research project between the Chair of Autonomous Vehicle Systems at the Technical University of Munich (TUM) and CARIAD SE. He has encouraged and supported me in writing a conference paper and apply for IEEE IV 2025.

Furthermore, I extend my appreciation to my colleagues at CARIAD SE, who directly and indirectly contribute to this project.

Lastly, I would like to express my deepest gratitude to my family and friends for their unwavering support throughout my academic journey.

Abstract

The detection of rare and hazardous driving scenarios is a critical challenge for ensuring the safety and reliability of autonomous systems. This research explores an unsupervised learning framework for detecting rare and extreme driving scenarios using naturalistic driving data (NDD). We leverage Deep Isolation Forest (DIF), a novel anomaly detection algorithm that combines neural network-based feature representations with Isolation Forests (IFs), to identify non-linear and complex anomalies that traditional methods may struggle to detect.

Data from vehicle bus signals and perception modules outputs, capturing vehicle dynamics and environmental conditions, is preprocessed into structured statistical features extracted from sliding windows. The framework incorporates t-distributed stochastic neighbor embedding (t-SNE) for dimensionality reduction and visualization, enabling better interpretability of detected anomalies. Evaluation is conducted using a proxy ground truth, combining quantitative metrics with qualitative video frame inspection to validate the relevance of detected anomalies in real-world scenarios.

Our results demonstrate that the proposed approach effectively identifies rare and hazardous driving scenarios, providing a scalable and interpretable solution for anomaly detection in autonomous driving systems. Given the study's methodology, it was unavoidable to depend on proxy ground truth and manually defined feature combinations, which do not encompass the full range of real-world driving anomalies or their nuanced contextual dependencies.

Contents

1 Introduction

1.1 Problem Statement

Advanced Driver Assistance Systems (ADAS) and *Automated Driving* (AD) have greatly improved road safety and driver comfort by handling challenging situations and avoiding potential hazards. A key challenge for these systems is determining when to intervene. Although certain situations, such as lane departure, clearly require warning or feedback, it can be challenging to identify scenarios that deviate from normal driving conditions - such as ones shown in Figure 1.1.

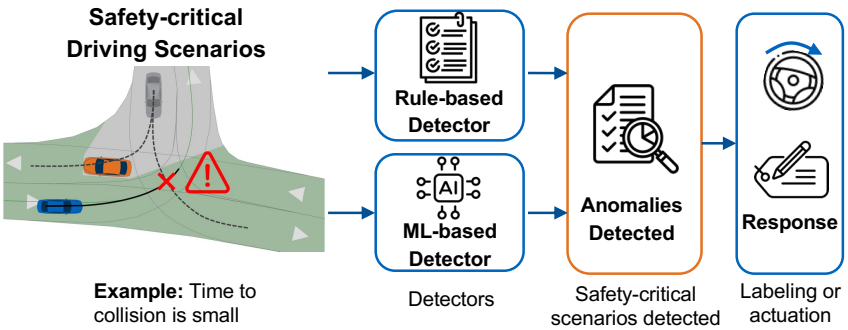


Figure 1.1: Illustration of the pipeline for detecting safety-critical driving scenarios, showcasing the integration of ML-based and rule-based detectors to identify anomalies and respond effectively. Illustration by Prof. Dr. Johannes Betz.

Traditionally, detecting driving anomalies has relied heavily on expert domain knowledge and manual analysis of driving data. Numerous rule-based approaches have been proposed for driving anomaly detection, including abnormal driving behaviors [1–4], risky driving scenarios [5–7], and monitoring road conditions [8–10]. This process is labor-intensive, time-consuming, and challenging to scale as data volumes grow. It is nearly impossible to exhaustively tabulate all possible actions or situations that lead to hazardous scenarios. Moreover, manual analysis frequently overlooks subtle or complex patterns, limiting its ability to capture potential anomalies. In contrast, data-driven methods using machine learning excel at detecting non-linear and complex

anomalies and can effectively handle diverse data features, making them suitable for dynamic driving environments. However, these models often require extensive labeled datasets for training and evaluation, which is impractical in real-world scenarios due to the scale of the dataset and the rarity of certain anomalies.

1.2 Objectives

To address these challenges, we propose an automated framework leveraging unsupervised learning to detect driving anomalies without relying on labeled data. These anomalies include a variety of types, ranging from point anomalies (e.g., sensor errors) to contextual anomalies and collective patterns in driving scenarios [11]. Our study primarily focuses on rare, extreme, and abnormal events, such as sensor malfunctions and risky maneuvers.

Unlike traditional rule-based approaches, which rely on pre-defined conditions [3, 12], our method identifies non-intuitive patterns that cannot easily be categorized. A key challenge in this work is evaluating unsupervised learning models when labeled data is unavailable. As mentioned earlier, creating labeled data is both time-consuming and impractical, especially for extreme driving scenarios. To address this, we propose the use of proxy ground truth set, consisting of approximate annotations derived from domain knowledge. This proxy set allows us to evaluate our unsupervised learning framework effectively without labeled data. Additionaly, we utilize *t-distributed Stochastic Neighbor Embedding* (t-SNE) [13], a dimensionality reduction technique, to visualize high-dimensional data in 2D, making it easier to interpret detected anomalies.

Contributions of the Thesis The main contributions of our study are:

1. We provide a novel unsupervised learning framework using Deep Isolation Forest [14] for detecting anomalies in driving and street scenarios.
2. We provide the implementation of a robust data pre-processing pipeline, including the selection and engineering of features from multimodal driving data (e.g., vehicle bus signals, object detection, lane detection).
3. We provide a robust evaluation and visualization of our anomaly detection framework by providing driving anomalies using proxy ground truth and t-SNE.

1.3 Research Questions

The following research questions will be addressed throughout the thesis:

RQ1: How can a dataset be constructed to facilitate the detection or rare events and safety-critical driving scenarios using unsupervised learning?

We aim to create a dataset that is designed to support the detection of rare events and safety-critical driving scenarios. The objective is to collect large-scale, real-world sensor data that capture diverse aspects such as vehicle kinematics, environmental conditions, and driver behaviors. For this, we will carefully select relevant features that represent critical aspects of driving scenarios while avoiding including excessive or irrelevant features that could introduce noise.

Pre-processing and feature engineering are essential steps to prepare the dataset for anomaly detection. The data needs to be appropriately transformed to meet the requirements of different unsupervised anomaly detection methods. For example, clustering-based approaches often require standardized data, whereas tree-based models can work directly with raw numerical features. Additionally, we will clean the data to ensure quality and reliability, including handling missing values, noise removal, and reduction of inconsistencies.

RQ2: Which unsupervised learning methods are the most effective for detecting anomalies in driving scenarios?

There are several types of unsupervised anomaly detection methods, such as statistical, distance-based, density-based, isolation-based, and reconstruction-based approaches. Choosing the right method is important, as each has its strengths depending on the nature of the anomalies and the characteristics of the dataset. Among these approaches, isolation-based methods are strong candidates for our work. They are effective at detecting rare events by isolating instances that are "few and different." These methods are also very computationally efficient, scalable, and can handle high-dimensional data without making assumptions about the data distribution. This makes them particularly suitable for large-scale, real-world datasets like ours, where rare anomalies need to be identified quickly and reliably.

RQ3: What is the capability of unsupervised learning methods to reliably detect rare events and complex anomalies in real-world driving scenarios?

Unsupervised learning methods offer a promising approach to anomaly detection by identifying deviations from normal patterns without the need for labeled data. However, their effectiveness in real-world scenarios depends on several factors, such as data quality, feature selection, and model parameters selection. Through experimentation, we aim to assess whether unsupervised learning models can effectively identify rare events or complex anomalies in real-world driving scenarios. This research question

focuses on determining the ability of these methods to derive such scenarios from large-scale, unlabeled data despite challenges like feature noises, high dimensionality, and diverse anomaly types.

RQ4: How can unsupervised anomaly detection methods be effectively evaluated without labeled data?

Evaluating unsupervised anomaly detection methods is challenging due to the lack of labeled data. Unlike supervised learning, which uses metrics like precision and recall derived from ground truth, unsupervised methods have no direct labels for evaluation. This becomes even more challenging with large, imbalanced real-world datasets, where anomalies may or may not exist, and their characteristics are often unknown. The imbalance between normal and anomalous instances can make it difficult to distinguish meaningful patterns. At the same time, the lack of prior knowledge about the type and severity of anomalies further complicates evaluation. Additionally, the high dimensionality and noise of real-world data add to the complexity, making the evaluation even more difficult.

In this work, we will adopt indirect evaluation strategies based on domain knowledge. Our approach combines both quantitative and qualitative methods. A proxy ground truth will be created using domain knowledge as an alternative to traditional labels. Data visualization will also be used to interpret model outputs, and image frames corresponding to detected anomalies will be exported for manual inspection.

1.4 Thesis Outline

This thesis is structured with the current chapter, followed by chapters listed below, concluded with a list of references and an appendix.

- Chapter 2: **Background** This chapter introduces the fundamental background that underpin this thesis.
- Chapter 3: **Related Work** This chapter reviews related research across various components essential for unsupervised driving anomaly detection.
- Chapter 4: **Methodology** This chapter explains the chosen approaches and methodologies for addressing the research questions.
- Chapter 5: **Results and Analysis** This chapter presents our results from different experiments and provides several quantitative and qualitative analyses to assess the performance of our methods.

- Chapter 6: **Discussion** This chapter interprets the results and discuss the answers for research questions.
- Chapter 7: **Conclusion** This chapter provides a final summary of our contributions, findings for future improvement, and suggests directions for future research.

2 Background

2.1 Perception system

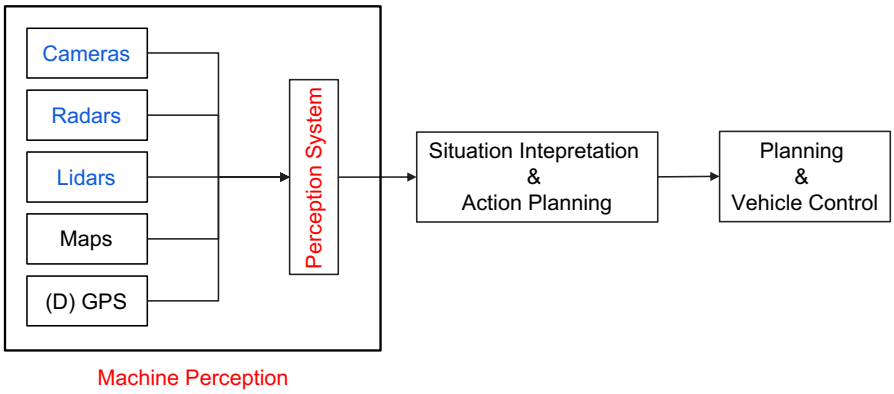


Figure 2.1: General information processing structure of automated vehicle driving. Illustration based on Dietmayer et al. [15].

Machine Perception Machine perception is the process by which an autonomous vehicle senses, interprets, and understands its surrounding environment using a combination of sensors and data processing techniques. As shown in Figure 2.1, machine perception relies on multiple sensor modalities, including *cameras*, *radars*, *lidars*, *GPS*, and *high-definition maps*, to capture environmental data and build a structured environmental model, preparing for next steps such as situation comprehension and action planning [15]. The vehicle's environmental information is collected by various sensors. Each has its own function and characteristics, described as follows:

1. *Cameras*: capture visual information from the environment and provides image data for visual perception tasks such as object detection or traffic sign recognition.
2. *Radars*: use electromagnetic waves to detect objects. It is primarily used for measuring distance, velocity, and relative motion of surrounding vehicles and obstacles.

3. *Lidars*: use laser pulses (electromagnetic pulses) to create a high-resolution 3D map of the surroundings. It determines the position, shape, and depth of objects, which is essential for precise obstacle detection and free-space estimation.

Perception Modules The perception system is a fundamental part of machine perception, responsible for processing raw sensor data and converting it into structured environmental information [16]. This information is organized into perception modules, each generating specific output signals essential for autonomous operation. The key perception modules include:

- **Visual Perception**:
  - *Object Detection*: outputs signals describing the position, size, and classification of detected vehicles, cyclists, and obstacles.
  - *Pedestrian Detection*: provides information on pedestrian presence, movement, and trajectory.
  - *Traffic Sign Recognition*: identifies and classifies traffic signs for regulatory compliance.
- **Lane & Road Perception**:
  - *Lane Detection*: generates signals on lane boundaries, curvature, and road markings for lane-keeping and navigation.
  - *Environmental Condition Recognition*: provides road surface conditions (e.g. dry, wet, snow-covered).
- **Visibility & Safety Perception**:
  - *Blindness Detection*: generate fail-save impairments signals from environmental factors.
  - *Obstacle & Free-Space Detection*: determines drivable space by identifying obstacles and open areas around the vehicle.

2.2 Anomaly Detection

Anomaly detection is the process of identifying instances or events that deviate from the norm in a dataset. These abnormal instances are often referred to as *anomalies*, *outliers*, *corner cases*, *novelty*, *aberrations*, etc. Among these terms, anomalies and outliers are most commonly used in the context of anomaly detection. Goldstein et al. [17] summarizes that anomalies generally have two key characteristics:

1. Anomalies are different from the norm with respect to their features and
2. They are rare in the dataset compared to normal instances.

2.2.1 Key Concepts

Anomaly detection is a broad field encompassing various techniques. Each technique has its own advantage and is designed to address specific problems. Figure 2.2 illustrates key components associated with an anomaly detection technique at a high level. The choice of an anomaly detection technique depends on the application domains, problem characteristics, and research areas.

This section presents an overview of anomaly detection to provide a fundamental background for the methodology in this thesis. Figure 2.2 serves as a reference for understanding how the terms are positioned within the high-level context of the anomaly detection problem.

Data Nature The nature of input data is an important factor in defining an anomaly detection problem. The input is typically a collection of data instances, where each instance can be described by a set of features or attributes. These features can have different types of values, such as *continuous*, *categorical*, or *binary*. Each data instance can be categorized into two types:

- *Univariate*: Contains one single features.
- *Multivariate*: Contains multiple features (these features can be of the same type or a mixture of different types).

In general, these data instances can be related to each other and form the whole dataset as the following type:

- *Sequence data*: Data instances are linearly ordered, often representing temporal relationships (e.g., time series, protein sequence).
- *Tabular data*: Data is structured into rows and columns, where each row represents an instance, and columns represent features (e.g., customer purchase history).
- *Graph data*: Data instances represent nodes in a graph, connected by edges (e.g., social networks, biological interaction).
- *Spatial data*: Data instances are related to their neighboring instances (e.g., geographical data, medical scans).

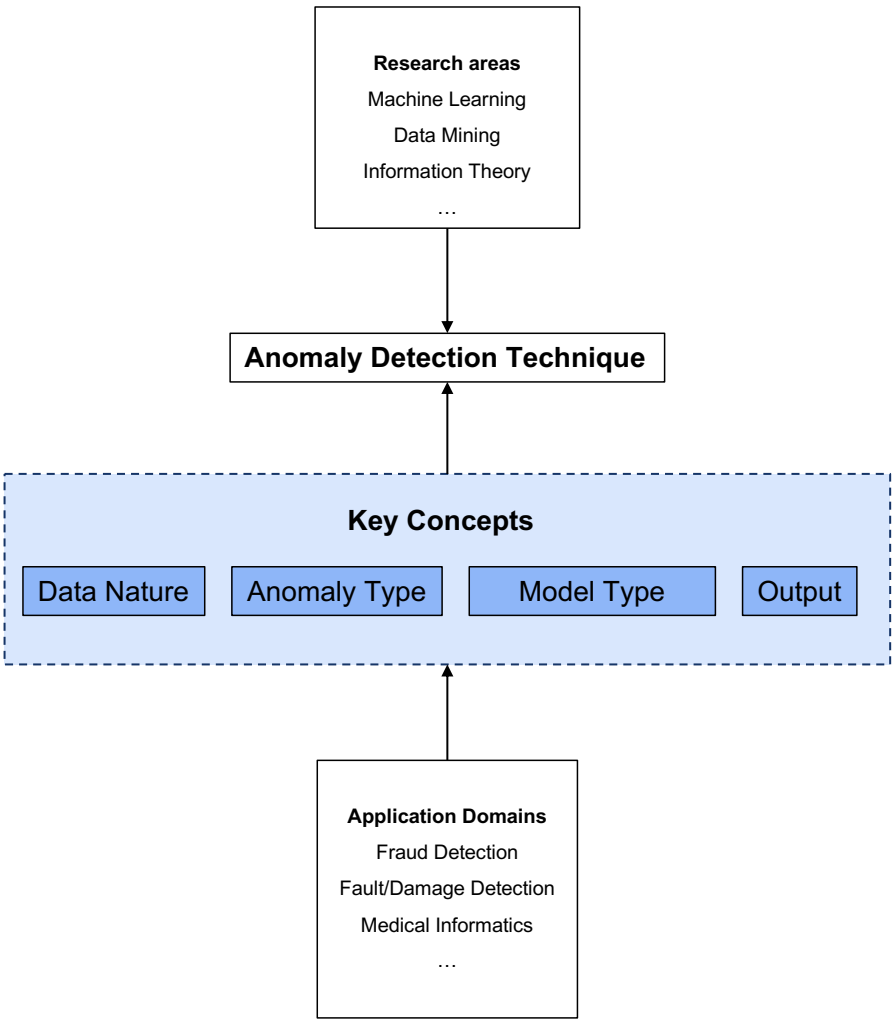


Figure 2.2: Key components associated with an anomaly detection technique. Illustration based on Chandola et al. [18].

Anomaly Type Anomalies are highly dynamic in nature, varying across types and applications. Determining what type of anomaly is an important aspect of anomaly detection, as it defines the goal of the problem. For generalization, Chandola et al. [18] categorized anomalies into three main types:

1. *Point anomalies*: A single data point that deviates significantly from the norm in the dataset. An example of autonomous driving is a sudden spike in vehicle speed to 150 km/h in a residential area where the speed limit is 30 km/h. This

is the simplest and most widely studied type of anomaly. Most unsupervised anomaly detection algorithms are designed for this type.

2. *Contextual anomalies*: A data point that appears normal under usual conditions but is anomalous in a specific context. For instance, a vehicle traveling at 100 km/h during a heavy rainstorm with poor visibility is a contextual anomaly. The speed may be normal in ideal conditions but are anomalous in adverse weather.

3. *Collective anomalies*: A group of data points that collectively deviate from the normal, even though individual points in the group may not be anomalous on their own. For example, a group of protesters blocking the main road represents a collective anomaly. A single pedestrian crossing the road may not be anomalous, but a group of people are.

While point anomalies can occur in any dataset, collective anomalies require datasets where data instances are interrelated. Contextual anomalies, on the other hand, depend on the availability of context features in the data. Point and collective anomalies can also become contextual anomalies when analyzed in a specific context. Thus, a point or collective anomaly detection problem can be turned into a contextual anomaly detection problem by incorporating contextual information.

Model Type Besides the described characteristics, one significant factor in selecting the anomaly detection algorithms is the label availability in the data. Labeling is often done manually by human experts and is thus time-consuming and expensive. Moreover, getting a labeled dataset covering all types of anomalous cases is even more challenging because anomalies are dynamic in nature. Therefore, anomaly detection problems usually come with unlabeled data. Based on the availability of labels in data, we can distinguish anomaly detection problems into three main types, as illustrated in Figure 2.3:

1. *Supervised Anomaly Detection*: The model is trained on a fully labeled dataset and makes predictions on unseen test data. This is similar to traditional pattern recognition, except the classes are usually highly imbalanced. However, in many application domains, anomalies are dynamic and not known in advance, making this approach impractical and not widely used.

2. *Semi-Supervised Anomaly Detection*: Similar to supervised anomaly detection, but in this case, the training set contains only normal data without anomalies. The model is trained to learn the normal pattern of the data and detect anomalies by identifying deviations from the norm.

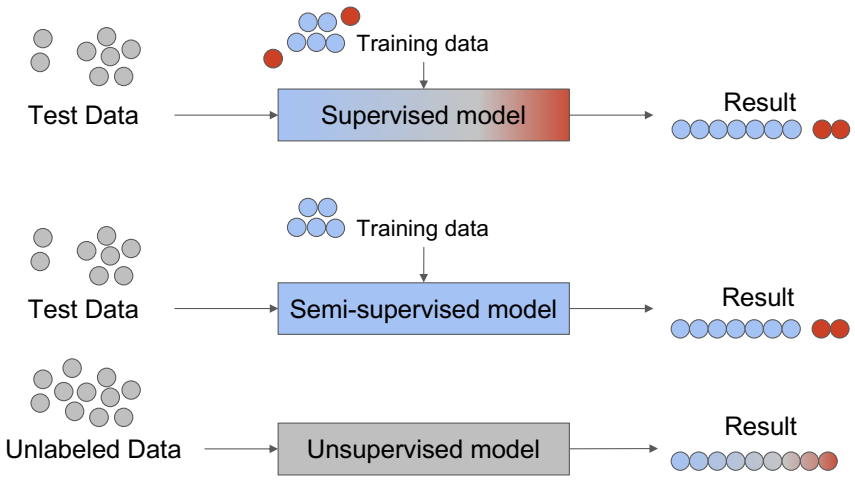


Figure 2.3: Anomaly detection categorization depending on the availability of labels in the dataset. Illustration based on Goldstein et al. [17].

3. *Unsupervised Anomaly Detection*: The model takes only a single dataset (no distinction between a training and test set) and does not require any labels. Thus, this approach is practical and widely used in many applications. The algorithm scores the data solely based on intrinsic properties. Typically, distances, densities, or isolation tree path lengths are used to determine these anomaly scores.

Output of Anomaly Detection The outputs produced by anomaly detection algorithms directly affect the decision of whether an instance is a anomaly or not. Typically, these outputs fall into one of the following types:

1. *Labels*: Anomaly detection models assign a label (*normal*/*anomalous*) to each instance, indicating whether it is anomalous. This type of output is commonly used in supervised anomaly detection.

2. *Scores*: Anomaly detection models assign an anomaly score to each instance, representing the degree of abnormality. This output is often used in semi-supervised and unsupervised anomaly detection algorithms. The algorithms rank anomalies based on these scores and report only the top anomalies. These top anomalies can then be further analyzed, or a domain-specific threshold can be applied to identify the most relevant anomalies.

2.2.2 Unsupervised Anomaly Detection Techniques

There are numerous anomaly detection techniques, each designed to address specific aspects (as described Subsection 2.2.1) and tailored for particular application domains. Several studies have attempted to review and group these techniques systematically. For instance, Chandola et al. [18] categorized anomaly detection algorithms into different groups, including *classification-based*, *nearest-neighbor-based*, *clustering-based*, *statistical*, and *spectral* methods. Schmidl et al. [19] organized the algorithms into different method families, such as *forecasting*, *reconstruction*, *encoding*, *distance*, *distribution*, and *isolation tree*, and then further categorized them into subfamilies and research areas.

In this thesis, we focus solely on algorithms that belong to the category of unsupervised learning methods, as they align with the objectives of our work. According to Goldstein et al. [17], unsupervised anomaly detection algorithms can be divided into the main groups as follow:

1. **Nearest-neighbor-based methods**.
2. **Clustering-based methods**.
3. **Statistical algorithms**.
4. **Isolation-based methods**.

Nearest-neighbor-based Methods Nearest-neighbor-based methods operate on the key assumption that normal data points locate in dense neighborhoods, whereas anomalies occur far from their nearest neighbors. These methods typically measure the similarity between data points using a distance metric. The choice of distance metric depends on the type of features in the dataset.

Euclidean distance is a common choice for numeric features but requires data to be on the same scale and is sensitive to outliers. *Manhattan distance* is often preferred for high-dimensional data where absolute differences between features are more relevant. For datasets where the directionality of features is more important than their magnitude, such as text data, *cosine similarity* is suitable. *Hamming distance* is effective for binary or categorical features. In contrast, *Mahalanobis distance* is more robust for multivariate datasets, as it considers feature correlations and scales distances based on the data distribution.

Typical methods in this category are *k-nearest neighbors* (k-NN) [20], *Local Outlier Factor* (LOF) [21], or *Connectivity-Based Outlier Factor* (COF) [22]. k-NN focuses on identifying global anomalies, whereas LOF and COF focus on local anomalies.

Clustering-based Methods Clustering-based methods identify anomalies by grouping similar data points into clusters. According to Chandola et al [18]. They detect anomalies based on three main assumptions:

- Normal data points belong to a cluster in the data, while anomalies do not belong to any cluster.
- Normal data points lie close to their closest cluster centroid, while anomalies are far away from their closest cluster centroid.
- Normal data points belong to large and dense clusters, while anomalies belong to small or sparse clusters.

Clustering-based methods are highly versatile and can be applied to different types of datasets, including numeric, categorical, or mixed-type data. However, these methods may struggle with high-dimensional data or detecting contextual anomalies without additional preprocessing or modifications. Some typical methods are *Density-based spatial clustering of applications with noise* (DBSCAN) [23], *k-means clustering* [24], *Cluster-Based Local Outlier Factor* (CBLOF) [25], and *Local Density Cluster-based Outlier Factor* (LDCOF) [26].

Statistical Methods Statistical anomaly detection methods identify anomalies based on the idea that normal data points are located in high-probability regions of a stochastic model, while anomalies occur in low-probability regions. These models fit a statistical model to the training data and use inference tests to determine whether an unseen data point belongs to the model. Data points that have a low probability of being generated from the learned model are considered anomalies. Common algorithms in this category include *Gaussian Mixture Models* (GMMs) [27], *ARIMA* [28] for time-series data, and *Histogram-Based Outlier Score* (HBOS) [29].

Isolation-based Methods Isolation-based anomaly detection methods identify anomalies by isolating data points directly rather than modeling normal behavior. The key idea is that anomalies are "few and different" from the majority of data points, making them easier to isolate. The most well-known algorithm in this category is *Isolation Forest* (IF) [30], which recursively partition the data using random splits. Isolation-based methods are computationally efficient, work well with high-dimensional data, and do not rely on assumptions about data distribution. This makes them ideal for large-scale data. These methods are particularly effective for detecting point anomalies but may require tuning for contextual or collective anomalies. Besides, there are extensions of IF, such as *Extended Isolation Forest* (EIF) [31], *SCiForest* (SCIF) [32], and *Deep Isolation Forest*

(DIF) [14], developed to further improve detection performance and adapt to specific scenarios.

2.3 Time Series Analysis

Time series Data Time series data is a sequence of data points that monitors a particular process, indexed in chronological order. It is typically dynamic, as feature values change over time, meaning each data point represents one or more observations recorded sequentially [33]. A *time series* is an ordered set:

$$T = \{T_1, T_2, \ldots, T_m\}$$

consisting of m real-valued data points, where each data point $T_i \in \mathbb{R}^n$ can be one-dimensional $(n = 1)$ or multi-dimensional $(n > 0)$. A *subsequence* is a continuous segment of T:

$$T_{i,j} = \{T_i, \ldots, T_j\} \subseteq T$$

with a specific length $|T_{i,j}| = j - i + 1 \quad \text{and} \quad |T_{i,j}| \geq 1$.

Time series data is widely used in real-world applications to analyze trends, patterns, and relationships over time. Examples include stock prices, weather data, heart rate monitoring (ECG), energy usage, and driving data. It is also a key focus in research areas such as signal processing, pattern recognition, forecasting, statistics, and any domain involving temporal measurements. Time series analysis is used to extract insights from time series data, such as trends, seasonality, cycles, and anomalies. Common techniques include segmentation, classification, forecasting, and feature engineering.

Time Series Segmentation Time series segmentation is a method in time series analysis that divides a time series into a discrete segments to reveal its underlying properties. This technique is widely used in fields such as driving data analysis, audio signal processing, medicine, and finance. The primary goal is to detect change points in the time series. According to Keogh et al. [34], most time series segmentation algorithms fall into one of the following three categories:

1. *Sliding Windows*: A segment is extended until it goes beyond an error bound, then the process starts again with the next point.
2. *Top-Down*: The time series is recursively partitioned into smaller segments until a predefined stopping criteria is met.
3. *Bottle-Up*: Starting with the smallest possible segment, these are gradually merged until a specific stopping criteria is met.

Exploratory Data Analysis Exploratory Data Analysis (EDA) is the process of examining and analyzing data to understand its structure, characteristics, and patterns before applying to the models. It is an essential step in any data analysis or machine learning workflow, helping to uncover valuable insights to guide feature engineering and model selection. Some common steps include visualizing the time series to detect trends, seasonality, and irregularities; computing descriptive statistics to understand the data's distribution and variability; and performing correlation analysis to identify relationships between features in multivariate time series.

2.4 Feature Engineering

Feature engineering is the process of transforming raw data into a machine-readable format in order to optimize machine learning models' performances. In other words, it is the process of selecting and extracting predictive features. A *feature* (or *attribute*) is a measurable property or characteristic of a dataset that might not be directly observable but can be inferred from related observations [35].

2.4.1 Feature Extraction and Selection

In many studies, *feature engineering* and *feature extraction* are often used interchangeably to describe the process of creating model variables [36]. However, *feature extraction* is sometimes used to refer to the process of transforming the input feature space onto a lower-dimensional subspace while retaining most of the relevant information [37]. In contrast, *feature selection* is a form of dimensionality reduction that involves selecting a subset of variables to reduce multicollinearity, thereby improving model generalizability and performance [38].

2.4.2 Feature Transformation

Feature transformation is the process of converting features into a format that is more suitable for a specific model.

Label Encoding Label encoding is a technique that converts descriptive categorical labels into numeric labels. Inverse label encoding is the reverse process, which transforms numeric labels back into their descriptive categorical labels. In this work, the categorical features in our dataset are stored as *enumeration* type, where each named feature is associated with an integer value. Therefore, inverse label encoding is frequently implemented in this work. A simple example of label encoding is illustrated in Figure 2.4.

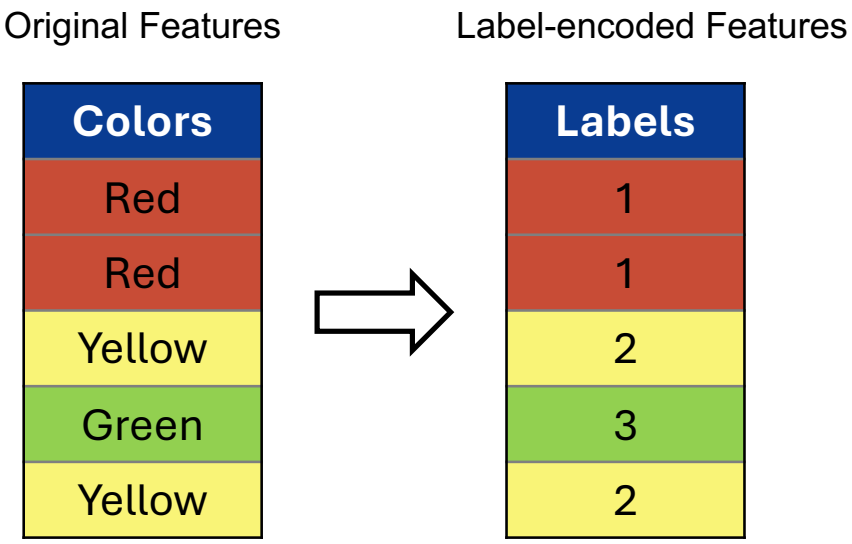


Figure 2.4: Example of label encoding.

One-hot Encoding One-hot encoding transforms categorical features into numerical values by mapping each category to a unique *binary representation*. These binary representations are organized in a matrix or vector space. Because one-hot encoding ignores the order, it is best suited for nominal categories data. Figure 2.5 provides a simple example of one-hot encoding.

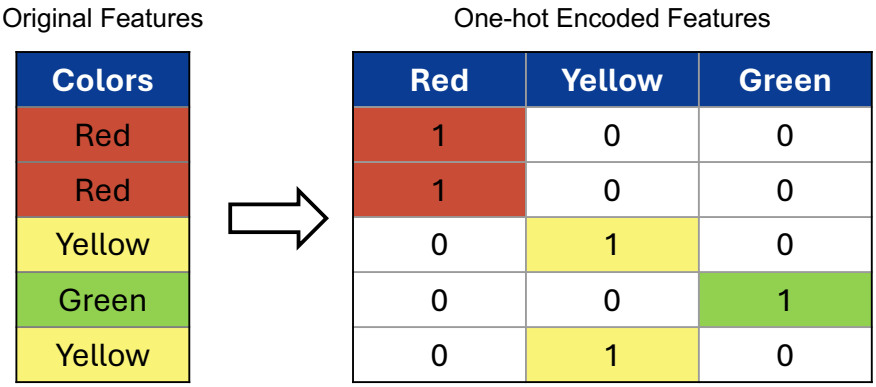


Figure 2.5: Example of one-hot encoding.

2.4.3 Feature Normalization

In a dataset, different features can have different ranges of values. For example, in a multivariate time series driving dataset, speed may range from 0 to 200 km/h, while longitudinal acceleration may range from -10 to 6m/s2. Such large differences in feature scales can negatively affect certain models' decisions. That is why we need *feature normalization* (or *feature scaling*), a standardization technique that rescales feature values and limits the impact of large differences in scale on models.

Min-max Normalization Min-max normalization rescales feature values to a fixed range, typically between 0 and 1. For each feature value x, the rescaled value $\tilde{x}$ is calculated as:

$$\tilde{x} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

where $min(x)$ and $max(x)$ are the decided minimum and maximum feature values, respectively. This ensures that all the features in the dataset lie within the same value range, reducing their influence on model decisions. However, since min-max normalization scales all data to a fixed range, it may compress the outliers.

Z-score Normalization Z-score normalization (also referred to as standardization or variance scaling) rescales features by centering them around 0 (i.e., mean of 0) and scaling them to a standard deviation of 1. This means there is no fixed range for feature values; these values depend on the mean and standard deviation. This technique is unbounded and, thus, retains outliers (unlike min-max). The formula to rescaled value $\tilde{x}$ is:

$$\tilde{x} = \frac{x - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation of the feature x.

2.4.4 Dimensionality Reduction

Dimensionality reduction is a technique to represent data with fewer features (i.e., dimension) while preserving its meaningful properties. This often involves removing irrelevant, redundant, or noisy variables. By transforming high-dimensional data into a lower-dimensional space, dimensionality reduction facilitates better analysis and visualization.

Dimensionality reduction is crucial in machine learning to address the challenges of the *Curse of Dimensionality* [39], where increasing the number of features leads to data sparsity, reduced model accuracy, and overfitting. High-dimensional datasets can result

in poor generalizability and computational inefficiency due to redundant or highly correlated variables. By reducing the feature space, dimensionality reduction methods improve model performance, enhance interpretability, and reduce computational requirements.

Principal component analysis (PCA) PCA is a widely used dimensionality reduction technique, sometimes referred to as a feature extraction method. PCA combines and transforms original features into new variables called principal components. These components are selected to capture the majority of the variance in the original dataset. PCA then projects the data onto a new space defined by this subset of variables.

t-Distributed Stochastic Neighbor Embedding (t-SNE) t-SNE is a non-linear dimensionality reduction technique primarily used for visualizing high-dimensional data. It maps data points to a lower-dimensional space, typically 2D or 3D while preserving local relationships between points. Unlike PCA, t-SNE focuses on capturing the neighborhood structure rather than maximizing variance, making it effective for exploring clusters and patterns in complex datasets.

3 Related Work

This chapter reviews the relevant literature on driving anomaly detection and isolation-based anomaly detection methods.

Driving Anomaly Detection

Anomaly detection is the process of identifying unexpected events that deviate from the norm. The types of anomalies range from point anomalies and collective anomalies to more complex contextual anomalies. In automated driving, Breitenstein et al. [11] provided a systematization of anomalies for visual perception, with the categories structured by detection complexity. Heidecker et al. [40] even extends the categorization of anomalies for perception concerning both cameras, lidars, and radars sensor modalities. Studies have proposed anomaly detection approaches in particular problems by setting thresholds.

Malta et al. [41] proposed an anomaly detection method based on vehicle speed and brake pedal thresholds. The scenarios are considered dangerous when the mean velocity is above a certain threshold and the brake pedal pressure is high. Zhao et al. [42] used acceleration and steering wheel information to identify aggressive driving behavior. They set the thresholds based on the angles of the steering wheel. These established rule-based models are only effective for simple cases. When the driving environment is complicated, they may underestimate risk when the thresholds are not met or overstate risk even when the drivers are controlling the car properly.

An alternative approach is to detect anomalies using machine learning algorithms. Hofmockel et al. [43] and Matousek et al. [44] implement Isolation Forest to detect anomalous driving behaviors such as aggressive maneuvers, drowsiness, and tailgating, using raw vehicle sensor data via the *Controller Area Network* (CAN-bus). The model shows its effectiveness across different benchmarks and strong scalability but fails to detect hard anomalies that are difficult to isolate in non-linear separable data space. Chen et al. [3] introduced an SVM-based method to classify the types of abnormal driving behaviors. The study utilized orientation data from a smartphone. However, due to the complexity and variability of driving anomalies, creating a comprehensive

classification-based solution that accurately identifies all anomalous scenarios is challenging and time-intensive. Clustering-based methods have also been employed for driving anomaly detection. Zheng et al. [45] proposed an unsupervised clustering technique utilizing smartphone accelerometer sensor data. The outliers on the clustering map were considered anomalies. Still, identifying meaningful clusters becomes problematic as the dimensionality of the feature space grows.

Isolation-based Anomaly Detection Methods

Isolation Forest (IF) [30] is widely used for anomaly detection due to its simplicity, efficiency, and scalability. It performs well across various benchmarks and is particularly suited for large datasets [46]. The core assumption of IF is that anomalies are easier to isolate from the rest of the data than normal instances. The algorithm constructs an ensemble of binary decision trees, which are known as isolation trees, built by recursively applying random splits on the data until all instances are isolated. Anomalies typically result in shorter average path lengths within these trees. One major limitation of IF is that it employs a linear, axis-parallel isolation approach, where each split considers only one feature at a time, making it hard to detect complex anomalies effectively. Several extensions of IF have been proposed to address this. SCIF [32] introduced a non-axis-parallel branching criterion by employing optimal slicing hyperplanes rather than single-feature splits. EIF [31] also used hyperplanes but with random slopes and intercepts, enabling more diverse partitions. To mitigate the empty branching issue in EIF, Lesouple et al. [47] selected splitting thresholds from the range of projected values along the hyperplane direction. A probability-based method from Tokovarov et al. [48] also refined split selection by finding more effective splitting values than random choices. These extensions generally improve performance by using non-axis-parallel and heuristic partitions. However, the main issue is that they still rely on linear isolation techniques, making it hard to handle complex data that requires non-linear partitions.

4 Approach and Methodology

This chapter presents the methodologies employed for our unsupervised anomaly detection framework. The overall workflow of our framework is illustrated in Figure 4.1. We utilize naturalistic driving data from vehicle bus signals and perception outputs from perception modules such as object detection, blindness detection, and lane detection, which capture both vehicle dynamics and contextual information. This data, stored as a multivariate time series, is processed and feature-engineered into a multivariate tabular format, referred to as *Driving Anomaly Data* (DAD), and used as input for unsupervised anomaly detection. Our study primarily employs *Deep Isolation Forest* (DIF) [14], an advanced anomaly detection algorithm that integrates neural networks, and *Isolation Forest* [30] to capture complex feature interactions and detect non-linear patterns. For unsupervised evaluation, we created a rule-based proxy dataset, considered as ground truth anomalies, and employed a three-pronged evaluation strategy: **(i)** analyzing the anomaly score distributions of the proxy and normal sets, comparing the top 100 highest-score anomaly segments with randomly selected normal segments, **(ii)** visualizing data and reducing dimensionality using t-SNE, and **(iii)** exporting video frames of detected anomalies for perceptual evaluation.

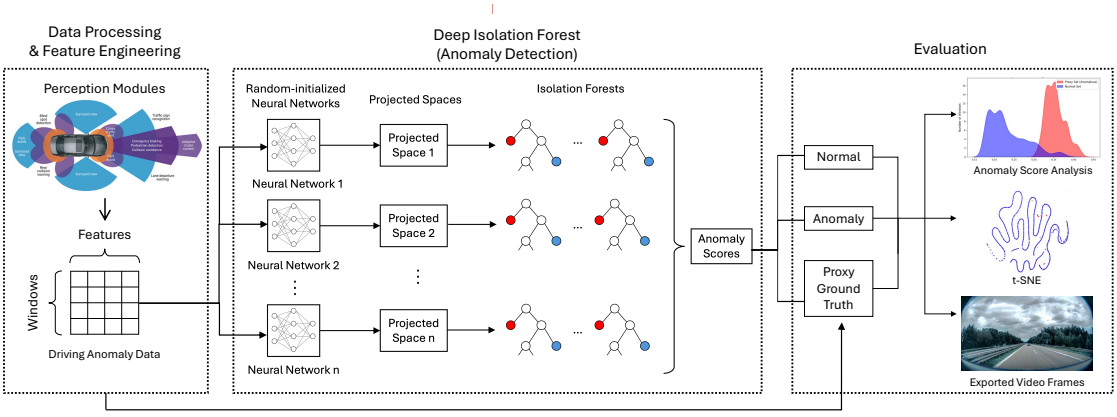


Figure 4.1: Overview workflow of our unsupervised anomaly detection.

This chapter consists of three main sections, each section represents a main components as shown in the pipeline, as follow:

- Section 4.1: **Data Processing**
- Section 4.2: **Unsupervised Anomaly Detection**
- Section 4.3: **Evaluation Frameworks**

4.1 Data Processing

The foundation of any machine learning model is the availability of high-quality data. For this study, we leverage a significant advantage: access to a vast dataset provided by CARIAD SE. This dataset comprises extensive real-world, naturalistic driving data collected from multiple measurements recorded during test drives on public roads in Europe. These real-world driving scenarios provide a rich, diverse dataset that is invaluable for understanding and analyzing complex driving conditions.

4.1.1 Data Source

The primary data sources for this study are vehicle bus signals and perception outputs obtained from the perception systems of autonomous vehicles. A perception system serves as the core component for acquiring and interpreting environmental data, enabling the vehicle to "understand" its surroundings and driving context. These systems integrate multiple sensors, including cameras, lidars, and radars, which work together to process raw data into high-level signals that represent the environment, vehicle dynamics, and driving conditions.

A typical perception system consists of various modules, each designed to handle specific tasks. These modules generate the high-level signals that are crucial for the study. Among the many available modules in the perception system, this study focuses on the following key modules:

- *Vehicle Bus Signals*: Signals data extracted from the *Controller Area Network* (CAN) [49] describing internal vehicle dynamics, such as speed, acceleration, and steering angle, etc.
- *Object Detection*: This module provides detected objects information such as positions, velocities, etc.
- *Blindness Detection*: This module identifies sensor occlusions or degradations caused by environmental factors, such as rain, fog, or sunlight glare, etc. It helps estimating the severity of these conditions and their impact on sensor visibility.

- *Lane Detection*: This module identifies lane boundary markings, estimating vehicle's position information relative to the lanes.
- *Road Condition Detection*: This module provides surface conditions, such as dry, wet, or snow-covered. These signals are essential for understanding the effects of road conditions on driving behavior.

4.1.2 Feature Engineering

Time Series Segmentation Time series data are typically recorded over extended periods, often encompassing hours, days, or even longer durations. For anomaly detection tasks, anomalous events are generally localized and occur over short intervals within the sequence. Consequently, it becomes essential to segment the time series data into smaller, manageable segments to enable more focused analysis and efficient anomaly detection. A common method for this is the *sliding-window* approach, which partitions the data into overlapping segments.

The approach is illustrated in Figure 4.2. First, a window of fixed size **w** is defined, representing the length of each segment. The window slides across the time series with a defined step size **s**, often set to 50% of the window size to ensure sufficient overlap. This overlap is crucial to avoid missing critical patterns or anomalies at segment boundaries. By segmenting the data in this manner, the sliding-window approach enables localized analysis while preserving the temporal structure of the original sequence. The selection of **w** and **s** is critical and depends on the duration and characteristics of the events being studied. Larger windows may capture more context, while smaller windows focus on finer details.

In this work, vehicle bus signals and perception outputs are stored as a multivariate time series, where each row represents a frame (sampled at approximate 10 fps), and each column corresponds to a relevant signals selected from the described perception modules in Subsection 4.1.1. We segment the data using with a window size of 6 seconds and a step size of 3 seconds. The 6-second window provides an appropriate temporal resolution for driving scenarios, as it is sufficiently long to capture meaningful patterns and interactions within the data, while short enough to preserve localized anomalies without dilution.

Feature Extracting Feature extraction is the process of transforming raw data into a reduced and informative representation that highlights the most relevant characteristics.

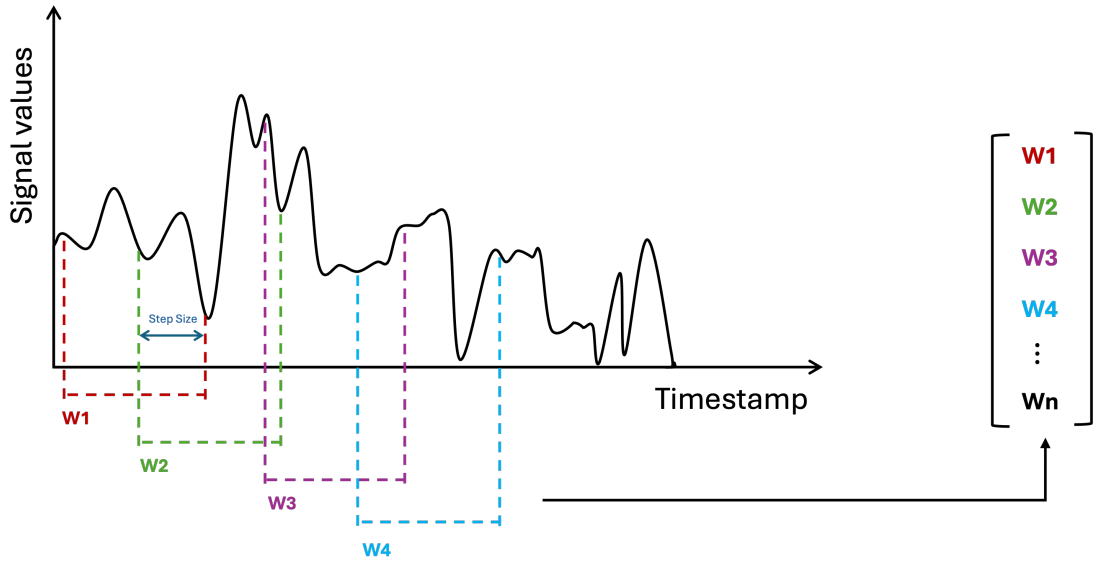


Figure 4.2: Sliding window approach (Windowing): Applying a sliding window technique to a speed signal sample, where **W** represents the window-size, and an overlapping step-size is used.

Time series data often contains redundant noise and a large amount of irrelevant information, which can obscure meaningful patterns and hinder analysis. By extracting only the most relevant features, this process not only highlights important information but also simplifies the data, improving model interpretability and performance. For unsupervised anomaly detection in time series data, feature extraction is particularly critical, as it helps reduce the dimensionality of the data and mitigate the risk of overfitting, especially in cases with limited anomalous samples.

One of the most commonly extracted feature types for time series data are *statistical features*, which are widely used due to their simplicity, effectiveness, and interpretability. These features summarize key characteristics of the data within a given time window, capturing essential patterns and trends. Common statistical features include the *mean*, *minimum*, *maximum*, *standard deviation*, *skewness*, etc. These features provide a compact representation of the data, highlighting its central tendency, variability, and distributional properties. Beyond statistical features, advanced transformations like the *Fourier Transform* or *Wavelet Transform* can extract frequency-domain information or localized frequency variations, respectively. However, these techniques often require higher computational resources and domain expertise. For this reason, the focus

of this thesis is on statistical features. These features effectively summarize the key characteristics of the time series data while remaining computationally efficient and easy to interpret.

Signals and Features Selection In this work, statistical features are extracted separately for each sensor channel within a segmented time series window. The features from all sensor channels are then concatenated to form a multi-dimensional feature vector for each window. As a result, each row in the transformed dataset represents a single window and contains a feature vector, where the number of dimensions corresponds to the total number of extracted features. This structured representation enables downstream analysis while preserving the temporal and contextual information necessary for anomaly detection.

Aspects	Features	Values	Types
Vehicle Kinematics	Relative Speed Range	$\frac{max(speed)-min(speed)}{max(speed)}$	Continuous
Environmental Factor	Rain Severe	Severe, Normal	Categorical
	Sunray Severe	Severe, Normal	Categorical
	Camera Image Blurriness Severe	Severe, Normal	Categorical
	Road-Type Conditions	Dry, Wet, Snow-covered	Categorical
Driving Behavior	Lane Keeping Quality	Good, Bad, Worst	Categorical
	Time-to-Collision Riskiness	$\frac{1}{TTC} \cdot \max(0, \frac{2.2-\lvert LateralPosition\rvert}{2.2})$	Continuous

Table 4.1: Key signal features characteristics.

The extracted features in this work are detailed in Table 4.1. These features were carefully selected to capture three key aspects:

1. **Vehicle Driving Kinematics.**
2. **Environmental Factors.**
3. **Driving Behavior.**

For vehicle driving kinematics, the *relative speed range* is used as it reflects variations in speed over a window, which can indicate sudden accelerations or decelerations. For environmental factors, the *statistical mode* is used, derived from categorical signals such as *weather severity* and *road type*. These categorical features are one-hot encoded to integrate seamlessly with continuous features. Driving behavior is assessed using a feature that represents *lane-keeping quality*. This is aggregated from three lane boundary safety signals (left, middle, and right of the ego lane) and categorized into three levels:

1. *Good* (all lane boundaries are safe)
2. *Bad* (one lane boundary is unsafe)
3. *Worst* (all lane boundaries are unsafe)

Additionally, *collision risk* is estimated using a derived feature based on *time-to-collision* (TTC) with detected vehicles and their *lateral position* relative to the ego vehicle. A scenario is considered risky if the TTC is less than 2 seconds and the lateral position is less than 2.2 meters. The resulting collision riskiness feature ranges from 0 (not risky) to 1 (high risk), providing a continuous measure of risk level.

4.1.3 Data Pre-processing

Data pre-processing involves operations that clean and prepare data to reduce flaws caused by noise, inconsistencies, or missing values. They can be duplicate eliminations, handling missing data, converting data types, or normalizing data. These operations are essential for ensuring that the data used for modeling is accurate, consistent, and suitable for analysis. Data denoising is an essential procedure for anomaly detection. Noise in data can closely resemble anomalies, but the key difference lies in the information they contain. Anomalies provide critical information about specific events, while noise or outlier represents redundant or misleading information that can obscure meaningful patterns and hinder the learning process of models. Removing noise ensures that the data reflects relevant and meaningful information for anomaly detection.

In this work, after time series segmentation and feature extraction, the data undergoes several pre-processing steps to prepare it for analysis. Missing values are handled by either removing affected windows or filling in appropriate values. Column names are standardized, and irrelevant windows containing nonsensical feature values are removed. Categorical signals are transformed into binary values through one-hot encoding, enabling seamless integration with continuous features. Continuous features are normalized using *Min-Max Normalization*, scaling values between 0 and 1. This normalization ensures consistency across features, aligning continuous and binary signals in the same range. Random sampling is applied to the segmented windows to avoid biases that might affect anomaly detection results. This step prevents the data from favoring specific patterns or sequences that could skew the results, ensuring a more robust and representative dataset. By randomly sampling the windows, the multivariate time series data is transformed into a multivariate tabular dataset. Each row in this dataset represents a window, and each column corresponds to an extracted

feature, as described in Table 4.1.

The resulting multivariate tabular dataset is the final input for the anomaly detection models. Rows represent randomly sampled windows, and columns are the extracted features. For convenience, we refer to this final dataset as *Driving Anomaly Data* (DAD) throughout this thesis.

4.2 Unsupervised Anomaly Detection

Unsupervised anomaly detection aims to identify rare and unusual driving scenarios without relying on labeled data. Given the scale and unlabeled nature of the dataset, the chosen methods must be efficient, scalable, and capable of detecting deviations from normal patterns. Isolation-based methods are particularly suited for this task due to their computational efficiency, ability to handle large-scale datasets, and mechanism for isolating anomalies based on rarity and deviation.

This section explores two approaches:

1. **Isolation Forest (IF)** [30]: a widely used tree-based method as baseline for our unsupervised anomaly detection framework.
2. **Deep Isolation Forest (DIF)** [14]: a novel state-of-the-art extension that integrates Isolation Forest with neural networks for improved anomaly detection.

4.2.1 Isolation Forest (IF)

Isolation Forest (IF) is a widely used unsupervised anomaly detection algorithm based on decision trees and ensemble learning methods. It has been arguably one of the the most popular anomaly detectors due to its effectiveness across various benchmarks and its strong scalability [46]. Unlike many existing methods, such as distance-based or density-based approaches, IF directly captures the essence of anomalies, which is "few and different." It makes no additional assumptions about data characteristics, enabling effective performance across diverse datasets. Furthermore, IF has linear time complexity, making it highly suitable for industrial applications with large datasets.

Isolation Tree (iTree) The core assumption of IF is that anomalies are rare and different, making them easier to isolate than normal instances. Unlike many anomaly detection algorithms that first define what is "normal" and then flag deviations as anomalies, IF isolates anomalous instances directly. As shown in Figure 4.3, the

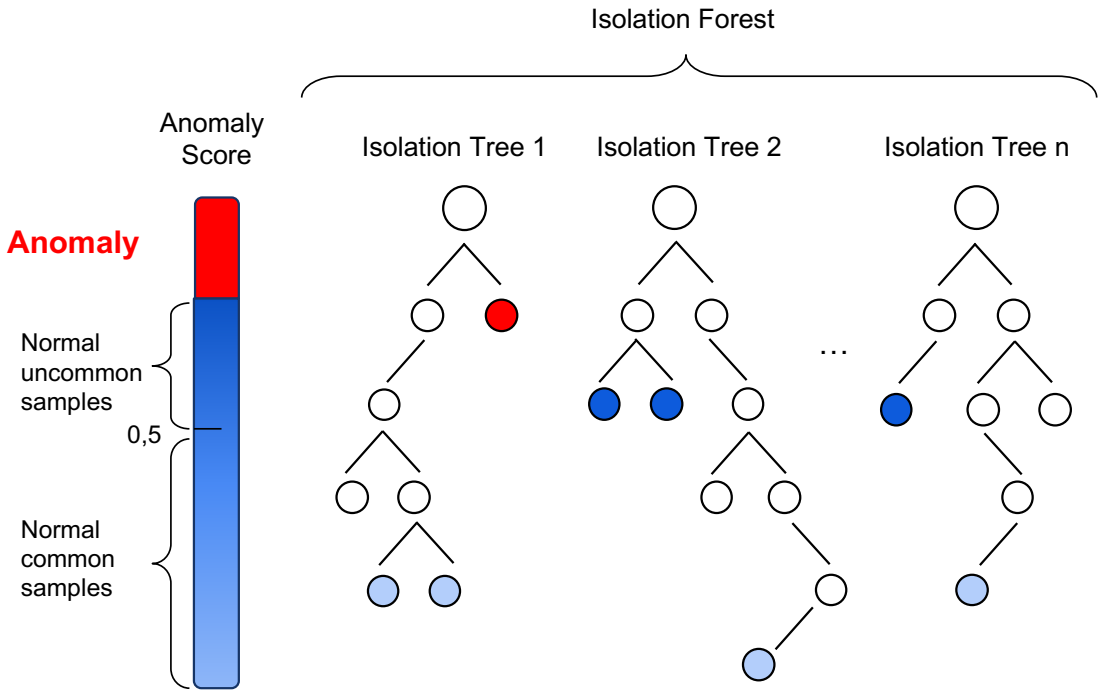


Figure 4.3: Anomaly detection with iForest. Illustration based on Chen et al. [50].

algorithm constructs a forest of binary decision trees, known as Isolation Trees (iTrees), which gives the method its name, Isolation Forest. Each iTree is created using a randomly selected subset X' from the input dataset X. The construction is set by two parameters:

- N': The size of sub-sample.
- t: The number of trees.

Since the sup-sample size N' is much smaller than the total dataset size N, the algorithm can efficiently create t iTrees in the training forest. The iTrees recursively apply random splits to the data until all instances are isolated. The average path length across all trees is then used to compute an anomaly score for each instance. Anomalous instances tend to have shorter average path lengths due to their sparse and distinct nature. In short, anomaly detection using IF is performed as follows:

1. Construct multiple iTrees using sup-samples of the dataset.
2. For each instance:

- Pass the instance through all iTrees, counting the path length in each tree.
- Calculate an anomaly score for the instance.
- Label the instance as an anomaly if its score exceeds a predefined threshold, which is determined based on the specific domain requirements.

Isolation Process The isolation process in iTrees is described in Figure 4.4. A binary search tree is randomly constructed to isolate the anomalous point *(13, 11)* (red circle) in a single split, while the normal point *(7, 7)* (yellow circle) requires four splits. In the tree, the anomalous point has a depth of 1 from the root node, whereas the normal point has a depth of 4 (Figure 4.4).

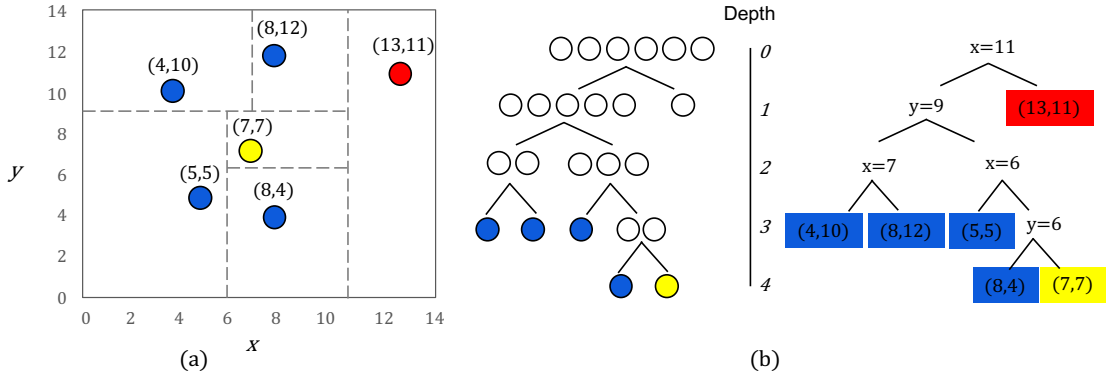


Figure 4.4: A randomly constructed binary search tree to isolate a two-dimensional data set: (a) Data points and isolation operations; (b) Binary tree and isolation process. Illustration based on Chen et al. [50].

The number of splits needed to isolate a point, referred to as the path length, is a key indicator of abnormality. Anomalies are characterized by shorter path lengths because the random partitioning process isolates them more quickly. When an instance consistently has short path lengths across a forest of randomly constructed binary trees, it is likely to be an anomaly. The anomaly score of an instance is derived from the average path length across all trees in the forest. The steps to calculate the anomaly score involve the following key terms:

Path Length The number of edges traversed from the root of an iTree to the leaf node of instance x is called path length $h(x)$.

Average Path Length The average path length of x across a collection of iTrees is $E(h(x))$.

Normalization Value Simply averaging with $E(h(x))$ can cause issues because larger trees with more instances inherently result in longer path lengths, even for anomalies. To address this, the expected path length for an instance is normalized using a value derived from the theory of *Binary Search Trees* (BST) [51]. iTrees share a similar structure with BST, where the average path length $c(m)$ for an unsuccessful search can be approximated as:

$$c(m) = \begin{cases} 2H(m-1) - \frac{2(m-1)}{m}, & \text{if } m > 2, \\ 1, & \text{if } m = 2, \\ 0, & \text{otherwise.} \end{cases} \tag{4.1}$$

Here, m is the sample set size, $H(i)$ is the i-th harmonic number, approximated as $H(i) = ln(i) + \gamma$, where $\gamma \approx 0.5772$ is the *Euler-Mascheroni Constant* [52].

Anomaly Score Finally, the anomaly score is computed as: $s(x, m) = 2^{\frac{-E(h(x))}{c(m)}}$. The score $s(x, m)$ ranges between 0 and 1 and can be interpreted as follows:

- If $E(h(x)) < c(m)$ (shorter path length than expected), $s(x, m) \to 1$, then x is very likely an anomaly.
- If $E(h(x)) > c(m)$ (longer path length than expected), $s(x, m) \to 0$, then x is likely normal.
- If $E(h(x)) = c(m)$, then $s(x, m) = 0.5$, indicating no clear evidence of anomaly or normality.

Extensions and Limitations One major limitation of IF is its linear, axis-parallel isolation approach, where each split considers only one feature at a time. This makes it less effective at detecting hard anomalies that depend on interactions between multiple features in higher-dimensional spaces. Figure 4.5 provides an example of such anomalies, which cannot be separated by either x-axis or y-axis splits.

Several extensions of IF have been proposed to address this. SCIF [32] introduces a non-axis-parallel branching criterion by employing optimal slicing hyperplanes rather than single-feature splits. EIF [31] also uses hyperplanes but with random slopes

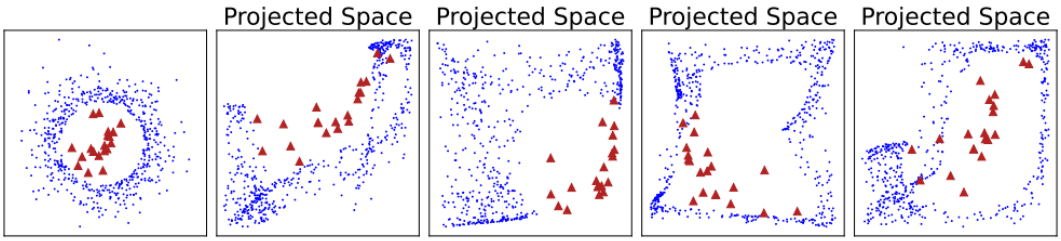


Figure 4.5: An illustration of hard anomalies in the original space and their projected space by Deep Isolation Forest. Red triangles represent anomalies, while blue points denote normal samples. Source: [14].

and intercepts, enabling more diverse partitions. To mitigate the empty branching issue in EIF, [47] selects splitting thresholds from the range of projected values along the hyperplane direction. A probability-based method [48] also refines split selection by finding more effective splitting values than random choices. These extensions generally improve performance by using non-axis-parallel and heuristic partitions. However, the main issue is that they still rely on linear isolation techniques, which makes it challenging to handle hard data requiring non-linear partitions, as illustrated in Figure 4.5.

4.2.2 Deep Isolation Forest (DIF)

Deep Isolation Forest (DIF) [14] is a novel extension of Isolation Forest (IF) designed to address the linear isolation limitations described in paragraph 4.2.1. It integrates randomly initialized (i.e., non-optimized) *Deep Neural Networks* (DNNs) with IF, projecting the original data into random representation spaces. These projections enable IF to detect complex, nonlinear anomalies that are challenging to isolate in the original data space.

Workflow The overview workflow of DIF is illustrated in Figure 4.6. The randomly-initialized DNNs project the original data into random representation spaces, enabling the algorithm to capture more complex patterns. IF then applies simple axis-parallel cuts in these new representation spaces to identify anomalies [53].

The main goal of the randomly-initialized DNNs is to better detect complex anomalies that are not easily isolated in the original non-linear data space. By liberating the algorithm from linear constraints, DIF significantly improves its capacity to detect

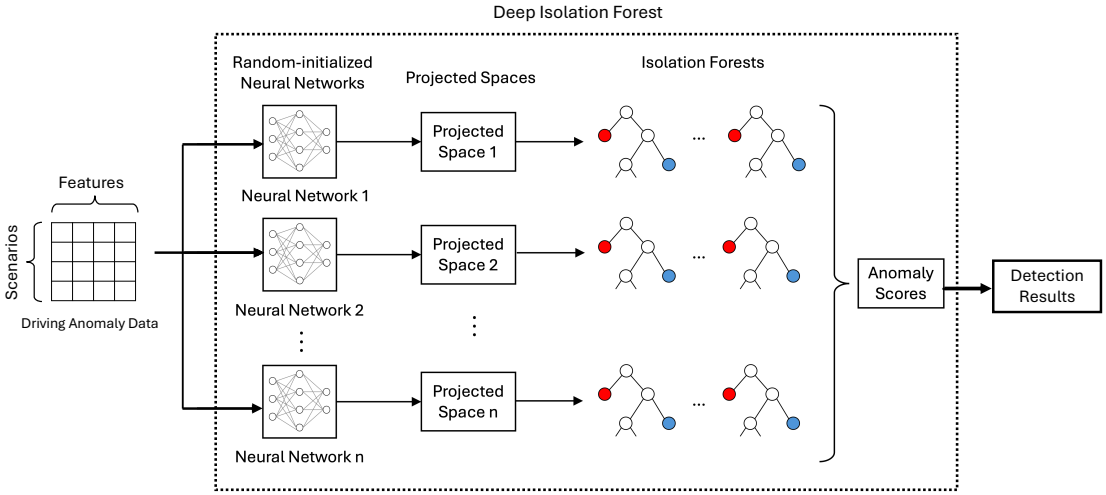


Figure 4.6: The simplified architecture of Deep Isolation Forest.

subtle, complex anomalies that are otherwise hidden in the original space while maintaining the scalability and efficiency of the IF framework.

Random representation ensemble DIF starts by generating random representations using random initialized neural networks (i.e., non-optimized). The *random representation ensemble* is defined as:

$$R(D) = \{\chi_u \subset \mathbb{R}^d \mid \chi_u = \phi_u(D;\theta_u)\}_{u=1}^{r} \tag{4.2}$$

where:

- χ_u is the u-th random representation.
- ϕ_u is the u-th neural network that maps the original input data D into a new d-dimensional space.
- r is the ensemble size (number of networks).
- θ_u are randomly initialized network weights.

Isolation in iTrees Each representation χ_u is assigned with t iTrees, and a forest $F = \{\tau_i\}_{i=1}^{T}$ containing $(T = r \times t)$ iTrees is constructed. Each iTree τ_i of χ is initialised by a root node with a set of projected data $P_1 \subset \chi$ and splits recursively:

1. Select a dimension j_k randomly from the d-dimensional space.
2. Choose a random split value η_k within the range of values in that dimension.
3. Divide the data at the k-th node with the data pool P_k into two subsets:

$$P_{2k} = \{x \mid x^{(j_k)} \leq \eta_k, x \in P_k\},$$

$$P_{2k+1} = \{x \mid x^{(j_k)} > \eta_k, x \in P_k\}.$$

where $x^{(j_k)}$ is the j_k-th dimension of the projected data object

The process repeats until all points are isolated.

Anomaly scoring After constructing the forest F, the anomaly score for a data object o is calculated based on how difficult it is to isolate o in the trees. The scoring function is defined as:

$$S(o \mid F) = \Omega_{\tau_i \sim F} I(o \mid \tau_i) \tag{4.3}$$

where:

- $I(o \mid \tau_i)$ is a function to measure how difficult to isolate o in iTree τ_i.
- Ω is an integration function. The final score S aggregates isolation diffuculty across the forest F.

Implementation DIF formulation comprises two main functions: *random representation ensemble* $R(D)$ and *anomaly scoring* $S(o \mid F)$, as described in 4.2 and 4.3, respectively. To further complement these functions, DIF proposed two novel methods, each has its own purpose as follows:

1. *Computation-Efficient deep Representation Ensemble* (CERE): used to improve the time efficiency of the representation function, by allowing computing all the ensemble members simultaneously in a given mini-batch.
2. *Deviation-Enhanced Anomaly Scoring* (DEAS): used to enhance the accuracy of anomaly detection by leveraging both hidden quantitative information embedded in the projected dense representation and qualitative comparisons.

4.3 Evaluation Frameworks

When labeled data is available, standard methods for evaluating anomaly detection models are *PR* (Precision-Recall) [54] and *ROC* (Receiver Operating Characteristic) [55] curves. Both are calculated based on *TP* (True Positive), *TN* (True Negative), *FP* (False Positive), and *FN* (False Negative) derived from the *confusion matrix*. However, in many cases, datasets lack labels. Particularly in industrial settings, companies often have to deal with massive amounts of real-world data, which is costly and time-consuming to label. This challenge also applies to our work. The dataset used in this study, provided by CARIAD SE, is measured from real-world driving scenarios and is entirely unlabeled. Evaluating unsupervised models in such cases is challenging, as the process is inherently subjective and cannot match the precision of supervised approaches.

In this work, we propose an evaluation framework comprising three main components:

1. **Evaluation using Proxy Ground Truth**: Provides quantitative insights.
2. **Perceptual Evaluation**: Offers qualitative validation through human inspection.
3. **Visualization using t-SNE [13]**: Adds qualitative interpretation.

4.3.1 Evaluation with Proxy Ground Truth

There are various strategies for creating labels to evaluate unsupervised anomaly detection in driving scenarios. Hofmockel et al. [43] introduce enforced anomalies during test rides, such as accidents, hard full braking, and overspeeding, as ground truth labels. Matousek et al. [44] generate labeled data by simulating normal and anomalous driving behavior using the SUMO traffic simulator [56] and modifying vehicle parameters to represent aggressive driving. Qiu et al. [57] manually categorize driving scenes into "candidate" (e.g., traffic violations, hazardous scenarios), "maneuver" and "normal" sets, using synchronized video, CAN bus data, and physiological signals. In this work, we propose an alternative approach by defining proxy ground truth labels for anomalies.

Proxy Ground Truth The proxy ground truth is a subset of the data derived from a set of heuristic rules based on domain knowledge. These rules define specific thresholds or conditions to capture potential anomalies. For example, time-to-collision values below a critical threshold, extreme lateral acceleration, or abrupt changes in speed during bad weather can all indicate anomalous events. By systematically applying these rules to

the data, we identify and extract all events that meet the specified criteria to form the proxy ground truth. This derived set of anomalous events is treated as our "hidden" ground truth, enabling the evaluation of unsupervised anomaly detection models.

This approach is particularly well-suited for large dataset. However, the rules used to define anomalies depend on the specific signals and features in the dataset. As the number of features increases, the complexity of creating and managing these rules grows significantly, making it harder to maintain interpretability and consistency. To address this, we focus on selecting only the most critical features in our dataset, avoiding the inclusion of less relevant features that could complicate the rule definition process and dilute the quality of the proxy ground truth.

Anomaly Scores Distribution An anomaly score is a value generated by the model that quantifies how much an event deviates from the normal pattern. Typically, a higher score indicates a more likely anomaly. The distribution of anomaly scores shows how these scores are spread across the dataset. For evaluation, we compare the distribution of anomaly scores for the proxy ground truth set against a normal set of equal size. Specifically, we compare the anomaly scores distribution of two groups: events in the proxy ground truth (expected to be anomalous) and events randomly sampled from the remaining dataset (expected to be normal). For example, suppose we identified 1000 anomalous events using heuristic rules to define the proxy ground truth, representing approximately 1% of the dataset. We then randomly sampled another 1000 events from the presumed normal set to compare their anomaly scores distribution.

Figure 4.7 illustrates an example of the anomaly score distributions for both sets. It shows that events in the proxy ground truth set tend to have significantly higher anomaly scores than those in the normal set. A significant difference in distributions validates the model's ability to assign higher anomaly scores to proxy anomalies. This validates that the proposed anomaly detection model effectively learns patterns from the data to identify anomalies.

Top Anomaly Scores Analysis Given the anomaly scores, the performance of unsupervised models can be evaluated by validating events with the highest anomaly scores. As previously described, we treat the proxy ground truth as our "hidden" labels or an alternative to ground truth labels. Therefore, we aim to check whether these high anomaly score events correspond to the proxy ground truth. This approach is similar to supervised learning, where the goal is to verify whether the model's predictions align with the annotated labels.

For instance, the top 100 events with the highest anomaly scores generated by the

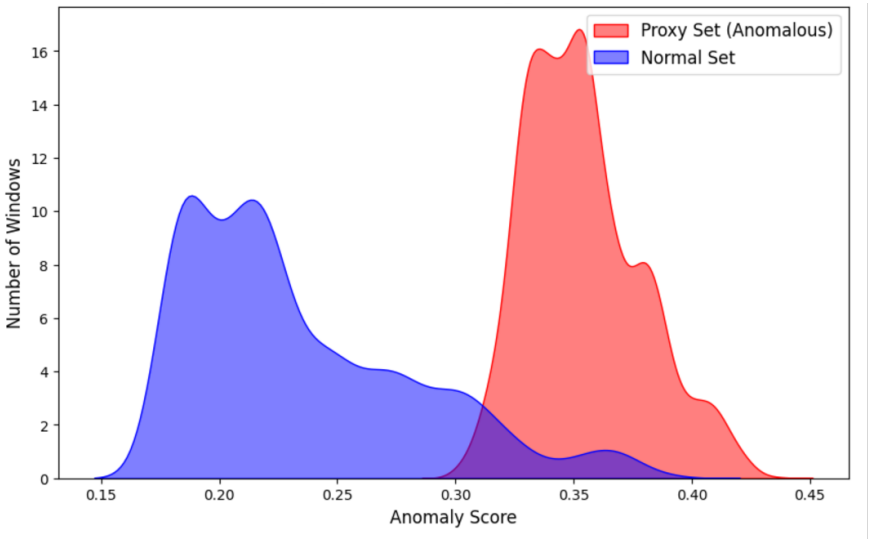


Figure 4.7: An example of anomaly scores distribution for the proxy ground truth set (anomalous) compared to the random sample set (normal).

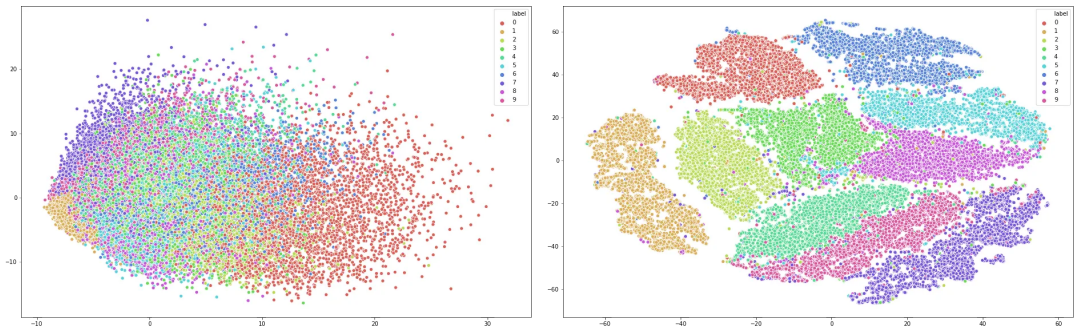
model are selected to determine whether they belong to the proxy ground truth or the normal set. If the majority of these events are found in the proxy ground truth, it indicates that the model effectively detects anomalies. Otherwise, its performance is considered poor. For comparison, another 100 random events from the dataset are also selected. Since these events are chosen randomly, they should be more likely to belong to the normal set.

4.3.2 Perceptual Evaluation

To qualitatively validate the model's performance, frames corresponding to extreme anomalies were exported for manual inspection. These frames were carefully reviewed to determine whether they depicted true anomalies, risky situations, or dangerous driving scenarios. This process allowed domain experts to assess whether the detected anomalies represented genuinely rare or interesting events, such as unusual maneuvers, hazardous conditions, or near-miss incidents. Given the time and resource-intensive nature of perceptual evaluations, we focus solely on the most anomalous events, which have the highest anomaly scores detected by the model.

4.3.3 t-SNE Visualization

t-SNE (t-distributed Stochastic Neighbor Embedding) [13] is a powerful tool for interpretation and qualitative evaluation of unsupervised anomaly detection models. It is a non-linear dimensionality reduction technique that visualizes high-dimensional data by mapping it to a low-dimensional space, typically 2D or 3D, while retaining a lot of the original information. Non-linear dimensionality reduction means that the algorithm allows us to separate data that cannot be separated by a straight line.



2D scatter plot of MNIST by PCA. 2D scatter plot of MNIST by t-SNE.

Figure 4.8: Visualization of the MNIST dataset using PCA and t-SNE. Source: [58].

Unlike PCA (Principal Component Analysis) [59], which is a linear dimensionality reduction technique suited for data with a linear structure, t-SNE is a non-linear method designed to preserve pairwise similarities between data points in a lower-dimensional space. While PCA aims to maximize variance by preserving large pairwise distances, t-SNE focuses on maintaining small pairwise distances to better capture the local structure of the data.

In short, PCA performs better on linear data, whereas t-SNE has no such restriction and works well on both linear and non-linear data. Figure 4.8 illustrates the visualization of the MNIST dataset [60] using PCA and t-SNE. It shows that t-SNE achieves superior partitioning of the digit clusters compared to PCA. This improvement is due to t-SNE's ability to capture the non-linear characteristics of the MNIST data, which PCA struggles to handle.

t-SNE determines similarity measures between pairs of instances in both high-dimensional and low-dimensional spaces and optimizes these measures though the following steps:

1. Compute *pairwise similarities in high-dimensional space*.

2. Compute *pairwise similarities in low-dimensional space*.
3. Optimization using *cost function Kullback-Leibler divergence [61]*.

Pairwise Similarities in High-Dimensional Space For high-dimensional data points x_i, t-SNE computes pairwise similarities p_{ij} based on Gaussian probabilities as follows:

$$p_{ij} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)} \tag{4.4}$$

Here, the variance σ_i for point x_i is adjusted to match a user-defined perplexity, ensuring adaptive scaling based on local density.

Pairwise Similarities in Low-Dimensional Space For the low-dimensional embeddings $\{y_i\}$, similarities are computed using a Student-t distribution:

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}}, \tag{4.5}$$

where q_{ij} represents the probability of similarity between the low-dimensional points y_i and y_j, and the heavy-tailed nature of the Student-t distribution allows better representation of pairwise relationships in the reduced space.

Cost Function Kullback-Leibler Divergence The algorithm then maps the high-dimensional data points to a lower-dimensional space, aiming to preserve these pairwise similarities. This is done by minimizing the Kullback-Leibler (KL) divergence [61] between the high-dimensional probabilities p_{ij} and low-dimensional probabilities q_{ij}:

$$\mathrm{KL}(P\|Q) = \sum_{i \neq j} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right), \tag{4.6}$$

where $P = \{p_{ij}\}$ and $Q = \{q_{ij}\}$ are the high-dimensional and low-dimensional probability distributions, respectively. The cost function is minimized using gradient descent, refining the embedding to align the two distributions. The cost is minimized using gradient descent, which iteratively updates the low-dimensional embeddings $\{y_i\}$. The gradient of the KL divergence with respect to y_i is given by:

$$\frac{\partial \mathrm{KL}(P\|Q)}{\partial y_i} = 4 \sum_{j \neq i} \left(p_{ij} - q_{ij}\right)\left(y_i - y_j\right)\left(1 + \|y_i - y_j\|^2\right)^{-1}. \tag{4.7}$$

Here, p_{ij} and q_{ij} are the high-dimensional and low-dimensional similarities, respectively, and the term $\left(1 + \|y_i - y_j\|^2\right)^{-1}$ accounts for the heavy-tailed nature of the Student-t distribution. This gradient is used in each iteration of gradient descent to refine the embeddings. This optimization process gradually refines the lower-dimensional embedding until it stabilizes, creating clusters and sub-clusters of similar data points in the lower-dimensional space. These clusters can then be visualized to uncover the structure and relationships within the original high-dimensional data.

5 Results and Analysis

This chapter discusses the experiments conducted and analyzes their results. The outcomes of each experiment contribute to answering the research questions outlined in Section 1.3.

Experimental strategy Our experimental strategy is to start with a simple toy example and gradually increase the complexity. This approach is common in research across various fields. It begins with introducing multiple "constraints" to make the problem more manageable, conducting experiments within those constraints, and then gradually removing each constraint step by step to explore more complex scenarios. This strategy ensures a structured and manageable progression rather than starting with overly complex setups. The implementation of this work consists of three main experiments, as follows:

- **Experiment 1: Categorical Features Only**

  This represents the simplest case, where the dataset contains only categorical features. The categorical features are one-hot encoded into binary representations, making the results straightforward to interpret. The goal of this experiment is to address **RQ2** and **RQ3**.

- **Experiment 2: Categorical Features + One Continuous Feature**

  Building on the experiment 1, this experiment introduces a single continuous feature alongside the categorical features. This step relaxes one "constraint" by gradually adding a continuous feature to the dataset. The goal remains to address **RQ2** and **RQ3**.

- **Experiment 3: Full Features (Categorical and Complex Continuous Features)**

  This is the most complex case, where we use the full range of features, including both categorical and continuous variables. The continuous features include a derived complex metric, such as time-to-collision riskiness, calculated from time-to-collision and the lateral position of detected vehicles. This experiment allows us to address **RQ4** by applying our evaluation strategy to the most complex scenario, in addition to addressing **RQ2** and **RQ3**.

Because **RQ1** focuses on understanding how the data is processed, all experiments inherently contribute to answering this question, as data processing is a fundamental step required for conducting each experiment.

Dataset description The whole data processing pipeline was introduced in Section 4.1, where we fetch the data containing vehicle bus signals and perception system outputs into a multivariate time series dataset. Through feature engineering and pre-processing steps, we obtain the final dataset used as input to the model, which we refer to as *Driving Anomaly Data* (DAD) for convenience in this thesis.

DAD is a multivariate tabular dataset. An example format of a multivariate dataset with m features and n scenarios is shown in Table 5.1. Each row represents a six-second driving scenario (also referred to as a window, segment, or scene), while each column corresponds to a specific feature. DAD covers a total of approximate 100 hours of driving, meaning the total number of scenarios is approximate $\frac{100\times60\times60}{6} = 60000$. These scenarios have been randomly sampled from different measurements to reduce bias.

Scenario ID	**Feature 1**	**Feature 2**	**...**	**Feature m**
Scenario 1				
Scenario 2				
...				
Scenario n				

Table 5.1: Example format of the multivariate tabular dataset, with rows represent scenarios and columns represent features.

Environmental setting The experiment was conducted in *Jupyter Notebook* on *Google Colab*, utilizing an *NVIDIA A100 GPU* with *CUDA* for training Deep Isolation Forest, which involves deep neural networks. The implementation was based on Python, using *PyTorch*, *scikit-learn*, and other data science frameworks.

5.1 Experiment 1: Categorical Features Only

5.1.1 Experimental Settings

Features Selection In this experiment, we focus solely on categorical features, as they are one-hot encoded and represented in binary form, making anomaly detection

Aspects	Features	Values	Types
Vehicle Kinematics	Relative Speed Range	$\frac{max(speed)-min(speed)}{max(speed)}$	Continuous
Environmental Factor	Rain Severe	Severe, Normal	Categorical
	Sunray Severe	Severe, Normal	Categorical
	Camera Image Blurriness Severe	Severe, Normal	Categorical
	Road-Type Conditions	Dry, Wet, Snow-covered	Categorical
Driving Behavior	Lane Keeping Quality	Good, Bad, Worst	Categorical
	Time-to-Collision Riskiness	$\frac{1}{TTC} \cdot \max(0, \frac{2.2-\lvert LateralPosition \rvert}{2.2})$	Continuous

Table 5.2: Selected features (highlighted) used for the experiment 1

more straightforward and the results easier to evaluate and interpret. Table 5.2 lists the selected features along with their corresponding aspects categories, values, and types. Specifically, the selected features include:

- *Rain Severity*: Extracted from the blindness detection module, this feature represents fail-safe impairments caused by rain, corresponding to its severity. The raw signal includes multiple values such as normal, medium, severe, not recognized, and init. For simplicity, we retain only normal and severe.
- *Sunray Severity*: Similar to the rain severity signal but measuring sunray intensity.
- *Camera Blurriness*: Indicates how blurry the camera image is, reflecting potential weather conditions such as fog, moisture, or snow.
- *Road-Type Condition*: Represents road conditions as detected by sensors, categorized as dry, wet, or snow-covered.
- *Lane Keeping Quality*: Derived from the lane boundary safety signals of the left, right, and middle lane boundaries of the ego vehicle. These signals are aggregated into a single feature:
  - *Good*: All lane boundaries are safe.
  - *Bad*: One or two lane boundaries are unsafe.
  - *Worst*: All lane boundaries are unsafe.

One-hot Encoding of Features All categorical features in our dataset are one-hot encoded to make them compatible with anomaly detection models. One-hot encoding transforms categorical variables into a binary format, allowing models to process

them efficiently without assuming any ordinal relationships between categories. This representation also helps in reducing bias that might arise if numerical values were assigned arbitrarily. Table 5.3 illustrates an example of one-hot encoding, where the *rain intensity* feature, categorized as *light*, *medium*, and *severe*, is converted into separate binary columns with highlights.

Original	**One-hot Encoded**		
Rain Intensity	**Light Rain**	**Medium Rain**	**Severe Rain**
Light	1	0	0
Medium	0	1	0
Severe	0	0	1
Severe	0	0	1
Light	1	0	0

Table 5.3: Example of one-hot encoding for rain intensity. Highlighted values correspond to the respective categories.

Models The models used in this experiment are:

1. *Isolation Forest* (IF)
2. *One-Class Support Vector Machine* (OC-SVM)
3. *Deep Isolation Forest* (DIF)

As discussed in Subsection 2.2.2 and Section 4.2, isolation-based anomaly detection models are well-suited for our dataset due to their efficiency, scalability, and ability to handle high-dimensional data. For all experiments in this thesis, Isolation Forest (IF) is chosen as the baseline due to its computational efficiency, scalability, and ability to run without a GPU. The other two models serve as benchmarks: OC-SVM employs a boundary-based classification approach, while Deep Isolation Forest (DIF) extends IF by incorporating deep learning techniques. To ensure a fair performance comparison, all models are trained using their default hyperparameters.

Contamination Level One key parameter is the contamination level, which defines the percentage of anomalies we aim to identify in the dataset. Since our data is unlabeled,

the actual proportion of anomalies is unknown. To estimate this percentage, we analyze the distribution of anomaly scores results. Figure 5.1 and Figure 5.2 visualize the anomaly score distribution generated by IF using a histogram and scatter plot, respectively. Based on these visualizations, we can subjectively determine a reasonable contamination level. For example, if a 2% contamination level results in anomalies that do not appear sparse or rare in the plots, we can lower it to 1% or less. For this experiment, we set the contamination level to 1% for all models.

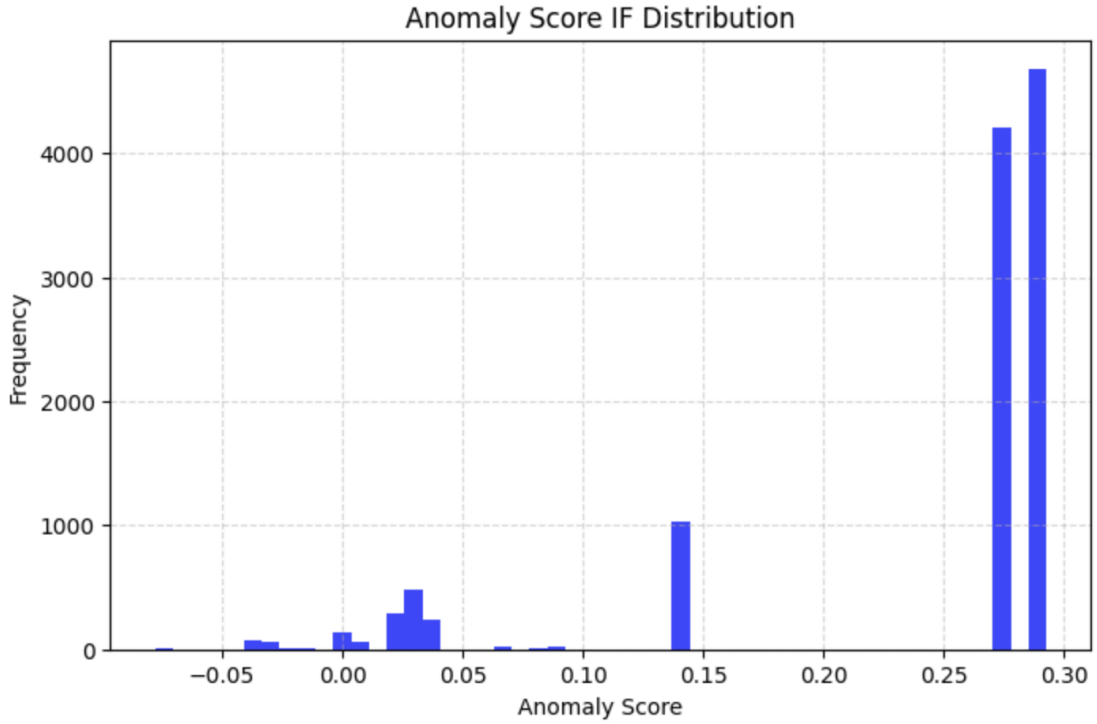


Figure 5.1: Histogram of anomaly score distribution by Isolation Forest. Lower scores indicate higher anomaly levels.

5.1.2 Results Analysis

Cross Product Table of Feature Combinations A cross product table is a comprehensive summary of all possible combinations of categorical feature values, represented as one-hot encoded binary vectors. Additionally, it includes the count of each combination, allowing us to assess model predictions based on their frequency. An example of cross product table is illustrated as Figure 5.3, with the following structure:

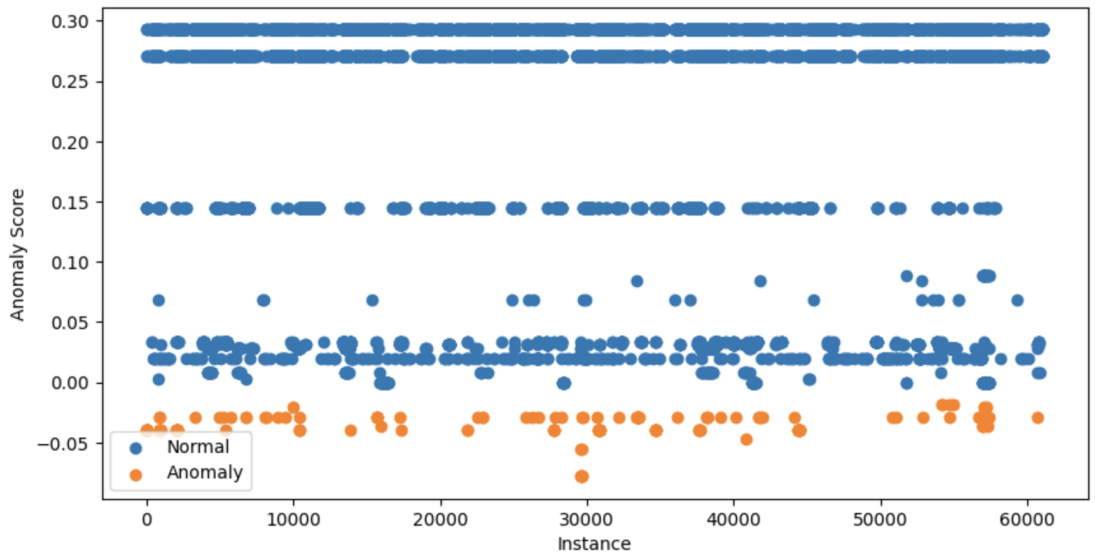


Figure 5.2: Scatter plot of anomaly score distribution by Isolation Forest. Lower scores indicate higher anomaly levels.

1. **Rows**:
   - Each row represents a unique combination of one-hot encoded values across all categorical features.
   - These combinations are generated from the cross-product of the categorical feature spaces.
2. **Columns**:
   - *One-hot encoded features*: Binary values (0 or 1) indicating each feature's specific category.
   - *Count*: The frequency of each unique feature combination in the dataset.
     - *Low count*: Indicates rare combinations (potential anomalies).
     - *High count*: Indicates common combinations.

This table is essential for analyzing and validating anomaly detection models on categorical features. The count column helps identify rare feature combinations that may indicate anomalies. Additionally, it allows for systematic cross-referencing of model predictions with the rarity of feature combinations, helping to evaluate the model's performance and enhancing interpretability by linking anomalies to specific, uncommon feature combinations.

	Feature 1	Feature 2	…	Feature n	Count
Combination 1	0 / 1	0 / 1	…	0 / 1	Count 1
Combination 2	0 / 1	0 / 1	…	0 / 1	Count 2
…	…	…	…	…	…
Combination m	0 / 1	0 / 1	…	0 / 1	Count n

Figure 5.3: Example format of a cross product table of feature combinations.

Isolation Forest Results Isolation Forest (IF) is used as our baseline model. We trained it with default parameters. The contamination level is set to 1%, meaning the model identifies the top 1% most anomalous points. The model outputs anomaly scores, which are distributed as in Figure 5.1 and Figure 5.2. For IF, lower scores indicate higher anomaly levels. Since all features are categorical, each data point corresponds to a unique feature combination, which can be validated using the Cross Product Table. To assess IF's performance, we examine whether the 1% highest anomaly score points fall into the "low count" category of the Cross Product Table. If detected anomalies are in the low-count region, they are considered rare, indicating a good detection. Conversely, if they appear in the high-count region, the model's prediction is likely incorrect.

Figure 5.4 presents a histogram of feature combination counts, where the x-axis represents unique feature combinations and the y-axis shows their frequency. Due to the high imbalance in counts, rare combinations are difficult to see in Figure 5.4. To address this, Figure 5.5 provides a zoomed-in version for better visibility of low-count combinations. From both figures, we observe that IF successfully detects some of the rarest feature combinations, capturing 10 out of the 15 lowest-count combinations.

OC-SVM results Similar to the IF evaluation setup, Figure 5.6 and Figure 5.7 present the histogram of feature combination counts and detected anomalies for OC-SVM. Among the 15 lowest-count combinations, OC-SVM can only detect 5.

Deep Isolation Forest Results Finally, the results of Deep Isolation Forest (DIF) are presented in Figure 5.8 and Figure 5.9. Among the 15 rarest feature combinations, DIF successfully detects 14, capturing nearly all the lowest-count anomalies.

Compared to the other two models, DIF outperforms both IF and OC-SVM in

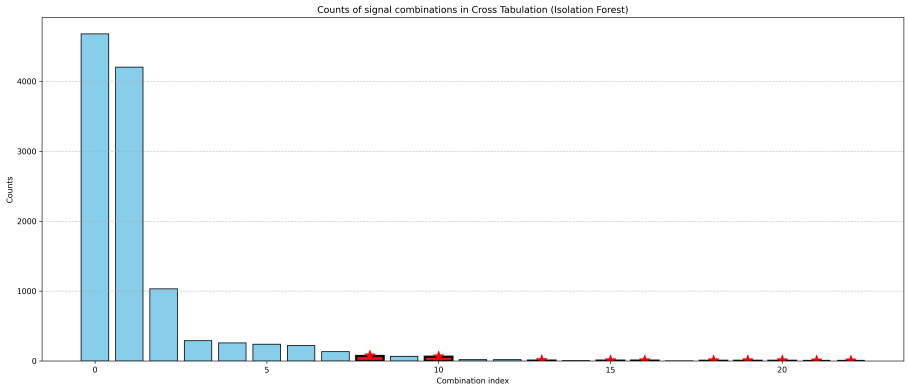


Figure 5.4: Histogram of feature combination counts, with the x-axis representing unique combinations and the y-axis showing their frequency. Red marks indicate anomalies detected by Isolation Forest. Isolation Forest captures 10 out of the 15 lowest-count combinations.

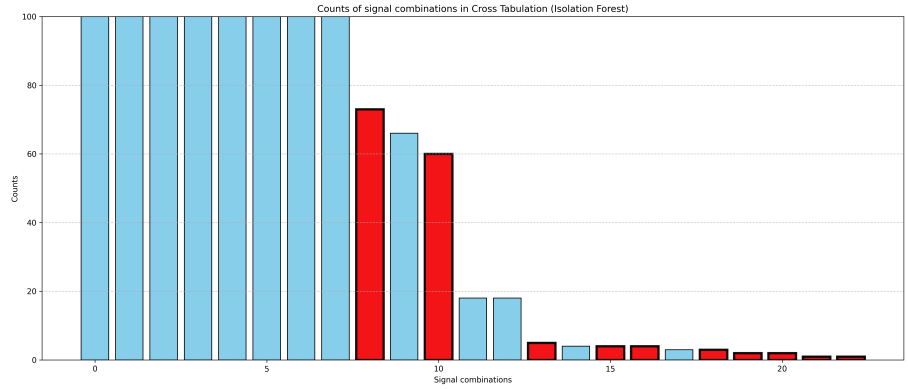


Figure 5.5: Zoomed-in view of low-count feature combinations for better visibility. Red marks highlight anomalies detected by Isolation Forest.

scenarios where all features are categorical. This is due to its ability to combine Deep Neural Networks with Isolation Forest, enhancing anomaly detection through non-linear isolation via data space projection. While DIF retains the strengths of IF, it further improves detection by capturing complex, rare patterns that IF might miss.

In contrast, OC-SVM performs poorly because it relies on a boundary-based approach that assumes a clear separation between normal and anomalous data. However, in

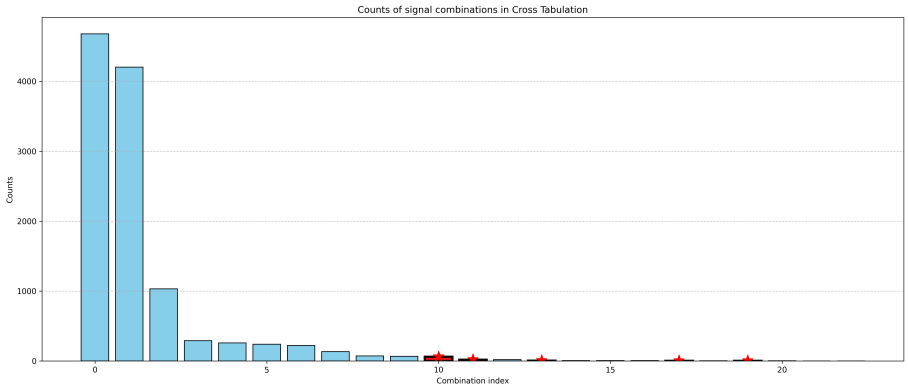


Figure 5.6: Histogram of feature combination counts, with the x-axis representing unique combinations and the y-axis showing their frequency. Red marks indicate anomalies detected by OC-SVM. OC-SVM captures only 5 out of the 15 lowest-count combinations.

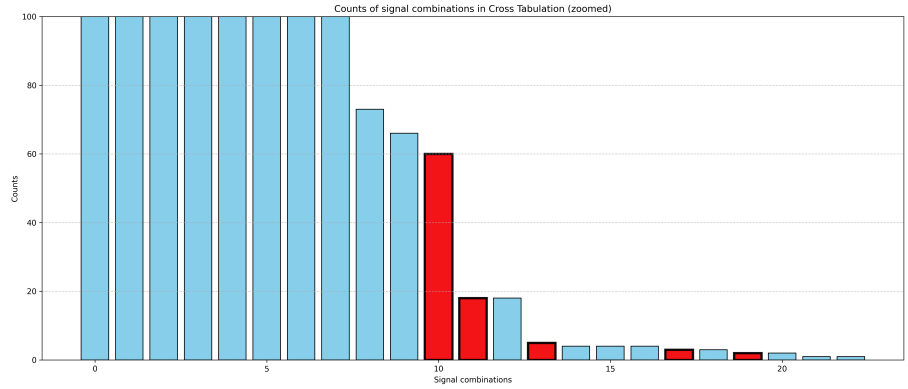


Figure 5.7: Zoomed-in view of low-count feature combinations for better visibility. Red marks highlight anomalies detected by OC-SVM.

high-dimensional categorical data with imbalanced feature distributions, defining such boundaries becomes difficult. As a result, OC-SVM struggles to generalize, leading to frequent misclassifications and failure to detect rare feature combinations effectively.

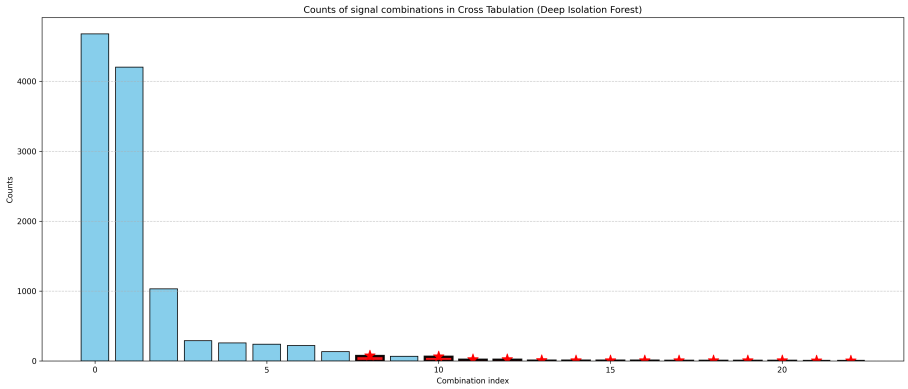


Figure 5.8: Histogram of feature combination counts, with the x-axis representing unique combinations and the y-axis showing their frequency. Red marks indicate anomalies detected by Deep Isolation Forest. Deep Isolation Forest captures 14 out of the 15 lowest-count combinations.

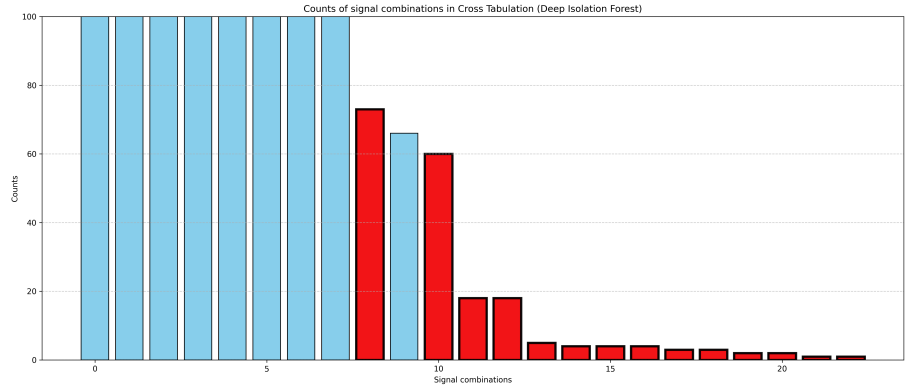


Figure 5.9: Zoomed-in view of low-count feature combinations for better visibility. Red marks highlight anomalies detected by Deep Isolation Forest.

Aspects	Features	Values	Types
Vehicle Kinematics	Relative Speed Range	$\frac{max(speed)-min(speed)}{max(speed)}$	Continuous
Environmental Factor	Rain Severe	Severe, Normal	Categorical
	Sunray Severe	Severe, Normal	Categorical
	Camera Image Blurriness Severe	Severe, Normal	Categorical
	Road-Type Conditions	Dry, Wet, Snow-covered	Categorical
Driving Behavior	Lane Keeping Quality	Good, Bad, Worst	Categorical
	Time-to-Collision Riskiness	$\frac{1}{TTC} \cdot \max(0, \frac{2.2-\lvert LateralPosition \rvert}{2.2})$	Continuous

Table 5.4: Selected features used in the experiment 2. The newly introduced feature is highlighted in red.

5.2 Experiment 2: Categorical Features + One Continuous Feature

5.2.1 Experimental Settings

Features Selection The features used in this experiment are the same as in Experiment 1, with the addition of one continuous feature: "relative speed range". Table 5.4 shows the selected features for this experiment, as follows:

- *Categorical Features*: Rain severity, sunray severity, camera image blurriness severity, and road-type conditions.
- *Continuous Feature*: Relative speed range.

Relative speed range is defined as the difference between the maximum and minimum speed within a six-second scenario, normalized by the maximum speed:

$$\frac{max(speed) - min(speed)}{max(speed)} \tag{5.1}$$

This feature is chosen because it effectively captures variations in speed over a time window while accounting for proportional changes, making it more robust than absolute speed range. For example, braking from 200 km/h to 180 km/h is more common and less likely to be considered anomalous than braking from 30 km/h to 10 km/h. With *speed range*, both cases have a value of 20 km/h, making it impossible to distinguish anomalies. However, with *relative speed range*, the values are normalized to $\frac{200-180}{200} = 0.1$ and $\frac{30-10}{30} = 0.67$, respectively, thus making it easier to quantify how anomalous each case is.

By incorporating relative speed range, the model gains an additional dynamic feature, improving its ability to detect speed fluctuations that may be indicative of anomalous driving scenarios.

Models The choice of models remains the same as in the experiment 1, using Isolation Forest (IF), One-Class SVM (OC-SVM), and Deep Isolation Forest (DIF). This is because the experiment 2 shares the same objective as the experiment 1, aiming to answer **RQ2** and **RQ3**, but extends the analysis to a more complex feature set by incorporating an additional continuous feature.

5.2.2 Results Analysis

For the experiment 1, evaluation is solely based on the *Cross Product Table* of categorical feature combinations. This table provides the count of each unique combination, allowing us to assess how rare (i.e., anomalous) a data point is. However, this method is only applicable when all features are categorical because categorical feature spaces are discrete and finite, making it possible to enumerate all possible combinations. When a dataset includes continuous features, the number of unique feature combinations becomes too large or even infinite, making direct frequency-based anomaly detection impractical.

In the experiment 2, we adopt an alternative evaluation strategy using t-SNE, as described in Subsection 4.3.3. Although t-SNE provides a qualitative assessment, it helps visualize how anomalies are distributed in a lower-dimensional space. While this experiment primarily focuses on comparative model evaluation, we aim to incorporate both qualitative and quantitative evaluation methods in the final and most complex experiment 3.

t-SNE results As described in Subsection 4.3.3, t-SNE is a technique for projecting high-dimensional feature spaces into 2D or 3D representations. Figure 5.10 illustrates the t-SNE visualizations for the three evaluated models:

- OC-SVM
- Isolation Forest
- Deep Isolation Forest

In these plots, red points represent detected anomalies, while blue points indicate normal scenarios. The results show that OC-SVM performs poorly, failing to distinguish

anomalies effectively. Isolation Forest performs better but struggles with challenging local outliers. In contrast, Deep Isolation Forest outperforms both models, successfully capturing anomalies that the other methods miss.

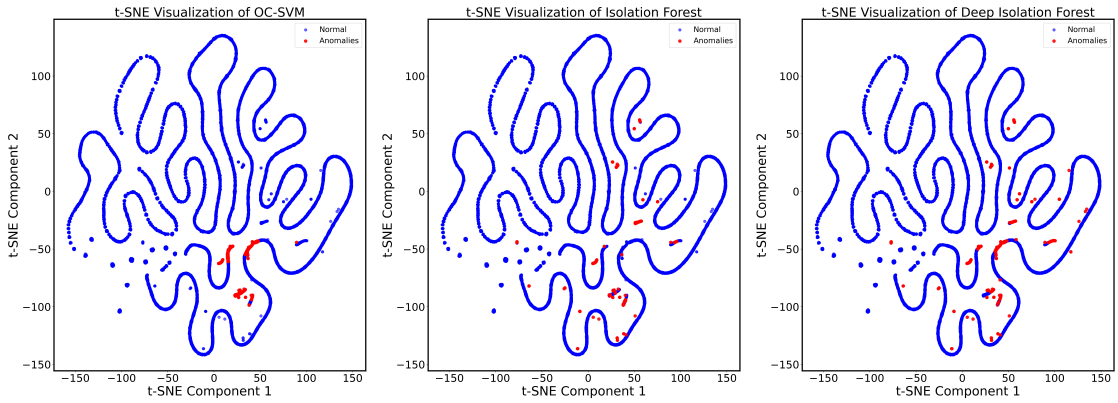


Figure 5.10: t-SNE visualization of OC-SVM (left), Isolation Forest (middle), and Deep Isolation Forest (right). Red points represent detected anomalies, while blue points indicate normal data.

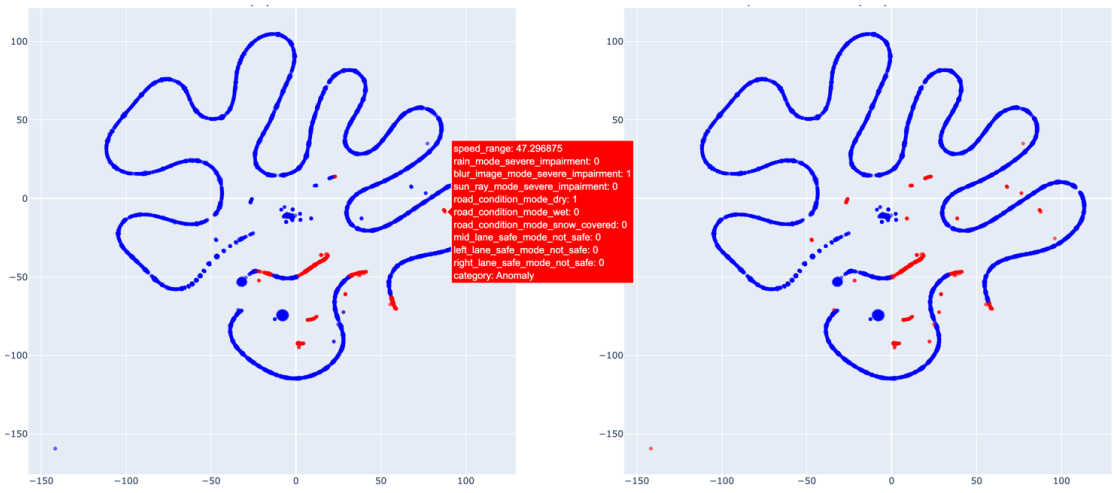


Figure 5.11: Interactive t-SNE visualizations showing detection results from OC-SVM (left) and Deep Isolation Forest (right).

Interactive t-SNE Interactive t-SNE is an extension of traditional t-SNE visualization that allows for dynamic exploration of high-dimensional data. Unlike static t-SNE

plots, interactive t-SNE allows us to zoom, pan, and inspect specific data points in the visualization. Figure 5.11 shows the interactive t-SNE visualizations with detection results from OC-SVM and Deep Isolation Forest. We can hover over a data point (i.e., an event or scenario) to view its corresponding feature values. This provides a quick and intuitive way to interpret anomalous scenarios.

5.3 Experiment 3: Full Features

Aspects	Features	Values	Types
Vehicle Kinematics	Relative Speed Range	$\frac{max(speed)-min(speed)}{max(speed)}$	Continuous
Environmental Factor	Rain Severe	Severe, Normal	Categorical
	Sunray Severe	Severe, Normal	Categorical
	Camera Image Blurriness Severe	Severe, Normal	Categorical
	Road-Type Conditions	Dry, Wet, Snow-covered	Categorical
Driving Behavior	Lane Keeping Quality	Good, Bad, Worst	Categorical
	Time-to-Collision Riskiness	$\frac{1}{TTC} \cdot \max(0, \frac{2.2-\lvert LateralPosition\rvert}{2.2})$	Continuous

Table 5.5: Selected features used in the experiment 3. The newly introduced feature is highlighted in red.

5.3.1 Experimental Settings

Features selection This experiment is the most complex case, which contains the full list of features, as shown in Table 5.5. The new feature introduced in this experiment is *Time to Collision (TTC) Riskiness*, which is derived from *Time to Collision* (TTC) and the *lateral position* of detected vehicles relative to the ego vehicle. It is calculated using the following formula:

$$\text{TTC Riskiness} = \frac{1}{TTC} \cdot \max(0, \frac{2.2 - |LateralPosition|}{2.2}) \tag{5.2}$$

A scenario is considered risky if TTC is less than 2 seconds and the lateral position is within 2.2 meters of the ego vehicle. The TTC Riskiness score ranges from 0 to 1, where higher values indicate greater risk. This formula captures two key factors affecting collision risk:

- *Inverse TTC*: The term $\frac{1}{TTC}$ ensures that shorter TTC result in higher risk scores, as a lower TTC suggests a potential collision. When TTC is very small, indicating that

a detected vehicle is approaching quickly, the riskiness score increases. Conversely, for large TTC values, the score remains low, reflecting a safer situation.

- *Lateral Position*: The term $\max(0, \frac{2.2-|LateralPosition|}{2.2})$ adjusts the risk score based on how close the detected vehicle is to the centerline of the ego vehicle. If the lateral position is close to 0 (i.e., the detected vehicle is directly in front or behind), this term is 1, meaning full risk contribution. If the detected vehicle is more than 2.2 meters away laterally, the term becomes 0, effectively eliminating lateral influence on collision risk.

By combining these two factors, TTC Riskiness provides a balanced risk assessment, prioritizing scenarios where a vehicle is both close in time to collision and close in lateral position. This feature helps identify potentially hazardous interactions, such as sudden braking or lane intrusions.

Models After the experiment 1 and experiment 2, we have answered **RQ2 (Which unsupervised learning methods are the most effective for detecting anomalies in driving scenarios?)**. Deep Isolation Forest (DIF) produced the best results compared to the other two models. Therefore, in this experiment, we focus solely on DIF, with IF included for benchmark comparison, as the objective is no longer to answer **RQ2** but to address the following research questions:

- **RQ3**: Can we derive rare events or complex anomalies in real-world driving scenarios using unsupervised learning methodologies?
- **RQ4**: How can we evaluate unsupervised anomaly detection without labeled data?

Regarding DIF parameters, the model uses parameters as follow:

- 50 representations (numer of random-initialized neural networks) ($r = 5$).
- Six isolation trees per representation ($t = 6$).
- Subsampling size of 256 ($n = 256$) for each isolation tree.

DIF processes tabular data using a fully connected *multi-layer perceptron* (MLP) network with:

- Two hidden layers: 500 and 100 units, respectively.
- Activation function: *tanh*.

- Output representation: 20 dimensions.
- Optional *skip connections* and *dropout*, depending on configuration.

Weights in the network are initialized with a normal distribution (mean = 0, standard deviation = 1), and the network processes data in batches of 64 samples.

5.3.2 Results Analysis

Proxy Ground Truth Proxy ground truth is a subset of the main dataset, extracted using heuristic rules that define specific conditions or thresholds to capture anomalies. Simply put, it serves as a "hidden" ground truth derived from the dataset itself (as described in Subsection 4.3.1). For this experiment, we define a set of rules to identify anomalous cases based on the available features.

Table 5.6 presents the full list of rules used in this experiment. These rules are manually defined based on the extracted features and thus do not capture all possible anomalies in the dataset. They rely on domain knowledge and are inherently subjective. However, the proxy ground truth offers a quick and resource-efficient approach to annotating and filtering data, compared to costly manual annotation, synthetic data injection, or simulations.

The size of the proxy ground truth set can be adjusted by modifying the thresholds defined for each signal in the heuristic rules. These thresholds determine the criteria for selecting proxy anomalies. By modifying these signal-specific thresholds in the heuristic rules, we can adjust the number of proxy anomalies, with stricter thresholds resulting in fewer anomalies and relaxed thresholds capturing more.

Evaluation approaches Given the proxy ground truth set, we follow the evaluation approaches outlined in Section 4.3. The complete evaluation framework consists of:

1. **Evaluation using proxy ground truth**:
   - *Anomaly Scores Distribution*: Analyzing the distribtuion of the anomaly scores accross the dataset.
   - *Top Anomalies Analysis*: Examining the most anomalous cases identified by the model.
2. **t-SNE Visualizations**: Projecting high-dimensional data into 2D for qualitative evaluation.
3. **Perceptual Evaluation**: Human inspection of exported frames from extreme detected anomalies.

Anomalous Scenarios	Heuristic Rules
Rare impairments with road conditions	Rain severe impairment on dry roads / sunray severe impairment on wet roads.
Uncommon impairment combinations	Any two or more of: rain severe impairment, blur image severe impairment, sunray severe impairment.
Multiple lane boundaries unsafe	Bad lane keeping quality (at least two unsafe lane boundaries: mid, left, or right).
Lane boundaries unsafe in adverse road conditions	At least one unsafe lane boundary with road conditions marked as wet or snow-covered.
Lane boundaries unsafe in severe weather	At least one unsafe lane boundary with rain, blur image, or sunray severe impairment.
Lane boundaries unsafe with high relative speed range	At least one unsafe lane boundary with relative speed range > 0.5.
High relative speed range in adverse road conditions	Relative speed range > 0.5 with road conditions marked as wet or snow-covered.
High relative speed range in severe weather	Relative speed range > 0.5 with rain, blur image, or sunray severe impairment.
Extreme high-risk TTC	TTC riskiness score > 0.5.
High-risk TTC in adverse road conditions	TTC riskiness score > 0.3 with wet or snow-covered road conditions.
High-risk TTC in severe weather impairments	TTC riskiness score > 0.3 with rain, blur image, or sunray severe impairment.
High-risk TTC with unsafe lane boundaries	TTC riskiness score > 3 with at least one unsafe lane boundary.
Extreme inverse TTC with high relative speed range	TTC riskiness score > 0.3 with relative speed range > 0.5.

Table 5.6: Heuristic rules used to define proxy ground truth anomalies.

Anomaly Scores Distribution The distribution of anomaly scores for the proxy set is compared against a normal set (randomly sampled) of equal size. In particular, we

compare the anomaly score distributions between two sets:

- Events in the proxy ground truth (expected to be anomalous).
- Events randomly sampled from the remaining dataset (expected to be normal).

In our experiment, we identified 550 events using heuristic rules to define the proxy ground truth, representing approximately 1% of the dataset. We then randomly sampled 550 additional events from the presumed normal set to compare their anomaly score distributions. Figure 5.12 illustrates the anomaly score distributions for both sets. It shows that events in the proxy ground truth set tend to have higher anomaly scores than those in the normal set. This higher anomaly score distribution for the proxy set indicates that both DIF and IF effectively distinguish anomalous events from normal ones.

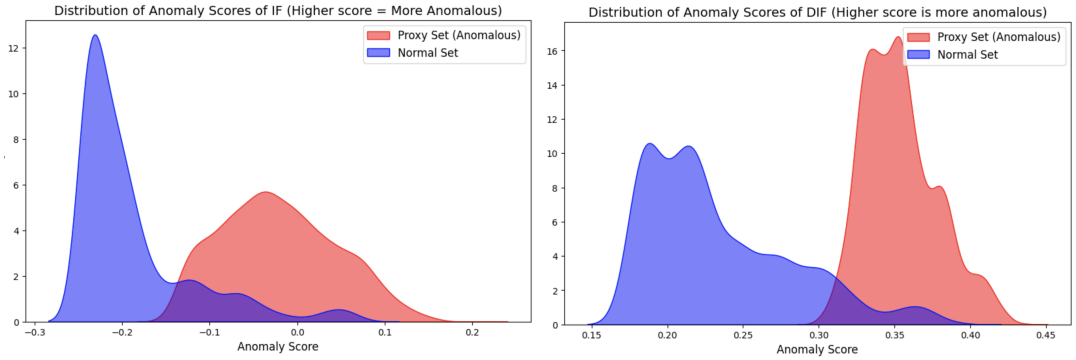
Top Anomalies Analysis The top 100 scenarios (i.e., events) with the highest anomaly scores are extracted. These scenarios are classified into two sets:

1. Those overlapping with the proxy ground truth (anomalous).
2. Those outside it (normal).

A summary table, as in Table 5.7, shows the proportion of these high-scoring anomalies aligning with the proxy labels. The majority of the scenarios in the Top 100 group overlap with the proxy ground truth set, while only a small percentage are classified as normal. In contrast, the Random 100 group contains a much higher percentage of scenarios from the normal set. This shows that our model effectively identifies most of the anomalies in the proxy set.

Events	Normal set	Proxy set
Top 100 of DIF	16	84
Top 100 of IF	35	65
Random 100	88	12

Table 5.7: Assignment of the top 100 highest-scoring anomaly events detected by DIF and IF into the normal and proxy sets.



Anomaly scores distributions by IF.

Anomaly scores distributions by DIF.

Figure 5.12: Anomaly score distributions for the proxy ground truth set (anomalous) compared to the randomly sampled set (normal) using IF (left) and DIF (right).

t-SNE Visualizations To further evaluate the separation between anomalous and normal scenarios, we applied t-SNE to project the high-dimensional feature space into a two-dimensional visualization.

Figure 5.13 presents the t-SNE visualizations of the dataset for IF (left) and DIF (right). Red points represent detected anomalous scenarios, while blue points indicate normal scenarios. The results show that IF struggles with challenging local outliers, whereas DIF effectively detects these anomalies. Additionally, compared to Figure 5.10, the visualization appears noisier due to the inclusion of full feature sets. As the number of features increases, t-SNE plots tend to become noisier.

Perceptual Evaluation Lastly, frames corresponding to extreme anomalies are exported for human inspection. This step provides a qualitative assessment of the model's performance, allowing domain experts to verify whether the identified anomalies represent genuinely interesting or rare scenarios. Given the time and resource-intensive nature of perceptual evaluations, we concentrated only on the most anomalous segments identified by the model. These segments were exported as image frames from video data and manually reviewed to determine whether they represent real anomalies, risky situations, or dangerous scenarios. We selected and exported three interesting scenarios, as follows:

- Figure 5.14: depicts *high speed* on *slippery roads* with *unsafe lane boundaries*.
- Figure 5.15: shows a scenario with *low time-to-collision* under *blurry image conditions*.

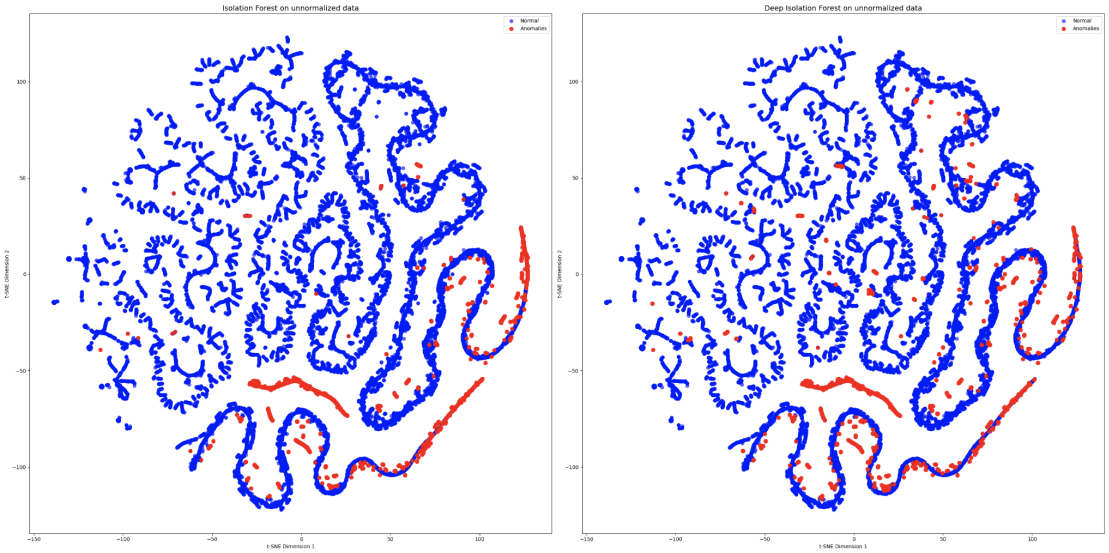


Figure 5.13: t-SNE visualization of IF (left) and DIF (right)

- Figure 5.16: captures *sudden braking* triggered by a front vehicle's *abrupt*, *unsignaled lane change*, illustrating the hazards of unpredictable driving behavior.

This perceptual evaluation confirms that the proposed unsupervised approach effectively identifies anomalous events. The high anomaly scores correspond to genuine hazards or rare scenarios, reinforcing the method's utility in detecting meaningful anomalies within the data.

Figure 5.14: High sped on slippery roads with unsafe lane boundaries.

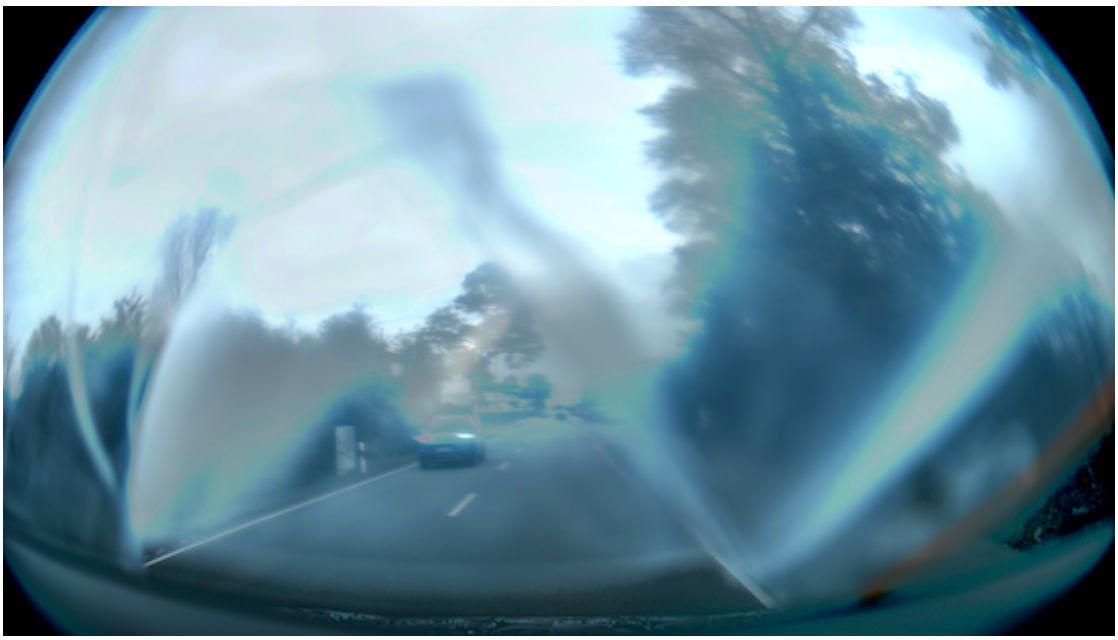


Figure 5.15: Low time-to-collision under blurry image conditions.

Figure 5.16: Sudden braking triggered by the front vehicle's abrupt lane change without a turn signal.

6 Discussion

This chapter presents an analysis to the research questions defined in this study, based on insights gained from our experimental results and analysis. Each research question is addressed by interpreting the findings and discussing their implications in the context of anomaly detection for autonomous driving.

RQ1: How can a dataset be constructed to facilitate the detection or rare events and safety-critical driving scenarios using unsupervised learning?

In this work, the dataset was constructed by leveraging perception signals from the vehicle's environmental perception system and applying statistical feature extraction over segmented driving scenarios. The primary goal was to create a dataset that effectively represents both normal and rare driving situations while being suitable for unsupervised anomaly detection. To achieve this, we followed these key steps:

- *Sliding Window Segmentation*: The raw perception signals were segmented into 6-second windows with a step size of 50% overlap to capture local driving behavior while maintaining a reasonable dataset size.
- *Feature Extraction*: Instead of using raw signals, statistical features were computed for each window. This transformation reduced noise and provided a compact, informative representation of driving scenarios.
- *Handling Categorical and Continuous Data*: Categorical signals (e.g., traffic sign recognition, road condition type) were one-hot encoded, while continuous signals (e.g., speed, time-to-collision) were preserved as numerical features.
- *Random Sampling Across Measurements*: To ensure diversity and avoid overfitting to specific test drives, windows were randomly sampled from different driving sessions, creating a dataset that represents various road conditions, weather, and traffic scenarios.

Through this approach, we constructed a dataset that is well-suited for anomaly detection in autonomous driving, ensuring that the data contains both frequent driving scenarios and rare, safety-critical events that models can learn to identify.

RQ2: Which unsupervised learning methods are the most effective for detecting anomalies in driving scenarios?

Isolation Forest (IF) has been shown to perform well across various benchmarks and is particularly suited for large datasets [46]. In our case, it served as a strong baseline, effectively detecting anomalies. However, Deep Isolation Forest (DIF), an extension of IF that integrates deep neural networks, has been proved to outperformed IF and its extensions [14]. It is the same case for our work; we compared IF and DIF across three experiments, and DIF consistently achieved better performance.

RQ3: What is the capability of unsupervised learning methods to reliably detect rare events and complex anomalies in real-world driving scenarios?

From our experiments, when the dataset consists only of categorical features, DIF is able to capture almost all of the lowest-count feature combinations, which represent the rarest events in the dataset. This reduces false negative errors, ensuring that fewer true anomalies are undetected.

In the case of both categorical and continuous features, DIF demonstrated its effectiveness in detecting safety-critical scenarios, as confirmed by our evaluation methods. By examining exported video image frames of events with the highest anomaly scores, we observed that DIF successfully identified risky anomalous situations, such as high relative speed range (sudden acceleration or deceleration) on slippery roads with unsafe lane boundaries and sudden braking due to abrupt lane changes by the front vehicle without a turn signal.

RQ4: How can unsupervised anomaly detection methods be effectively evaluated without labeled data?

Evaluating unsupervised anomaly detection without labeled data requires alternative validation strategies, as traditional supervised metrics such as accuracy and recall cannot be directly applied. In this work, we employed a combination of heuristic-based evaluation (*proxy ground truth*), feature space visualization (*t-SNE*), and qualitative inspection (*perceptual evaluation*) to assess model performance.

7 Conclusion

7.1 Summary of Contributions

This thesis presents a novel unsupervised anomaly detection framework for identifying rare and safety-critical driving scenarios using Deep Isolation Forest. The key contributions of this work include:

- **Unsupervised Anomaly Detection Framework**: We propose an unsupervised learning approach based on Deep Isolation Forest to detect anomalies in driving scenarios.
- **Robust Data Processing Pipeline**: We develop a systematic data preprocessing framework, including time series segmentation and feature engineering from multimodal driving data sources comprising vehicle bus signals and environmental perception outputs.
- **Comprehensive Evaluation & Visualization**: We establish a robust evaluation framework for unsupervised anomaly detection, incorporating proxy ground truth anomalies, t-SNE visualizations, and perceptual validation, ensuring meaningful interpretation of detected anomalies.

7.2 Future Work

This section outlines suggestions for future research in several directions.

Feature Engineering In unsupervised anomaly detection for driving scenarios, the detected results may not always correspond to meaningful real-world scenarios, such as risky or dangerous situations. This issue arises from the inclusion of irrelevant feature combinations or the effects of high dimensionality, which introduce noise and dilute the model's focus on critical patterns. Addressing this limitation will require a more advanced process for feature engineering. A promising direction is the use of *Deep Feature Synthesis* (DFS) [62], an automated approach that generates and evaluates complex features by combining raw data columns through mathematical and logical

operations. This technique can help reduce noise and prioritize features that are more relevant to identifying meaningful anomalies. Additionally, complex features from object detection interface, such as object lists, object tracking over time, and object interaction dynamics, should be incorporated and processed (even though complex and challenging) to achieve a more comprehensive representation of driving scenarios.

Anomaly Detection Model Parameters Tuning Another limitation lies in the sensitivity of the density-based Isolation Forest algorithm to its parameters, such as the contamination level, which defines the proportion of anomalies in the data. Since the dataset is unlabeled, the true contamination level remains unknown, and the highly imbalanced nature of the data further complicates parameter tuning. Future work could involve applying optimization techniques, such as *Grid Search* [63] or *Bayesian Optimization* [64], to systematically identify optimal parameter values.

Proxy Ground Truth Refinement Additionally, refining the proxy ground truth with more sophisticated rules that can capture complex anomalies is essential for improving the evaluation process. Advanced approaches like rule-based systems augmented with machine learning methods for *driver fingerprinting* [65] could enhance the accuracy and utility of the proxy ground truth.

Visualization and Interpretation Lastly, the t-SNE visualization method, while valuable for understanding the model's outputs, is highly dependent on hyperparameters such as perplexity and learning rate. Determining the optimal parameters for t-SNE is complex and dataset-specific, requiring significant experimentation. To streamline this process, future work could explore *automated hyperparameter optimization techniques* [66, 67], or consider alternative dimensionality reduction methods like *Uniform Manifold Approximation and Projection* (UMAP) [68]. These methods can provide robust and interpretable visualizations with reduced manual effort, enabling more effective analysis of anomaly clusters and outliers.

List of Figures

List of Tables

Bibliography

[1] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan. "Mobile phone based drunk driving detection." In: *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*. 2010, pp. 1–8. DOI: 10.4108/ICST.PERVASIVEHEALTH2010.8901.

[2] H. Eren, S. Makinist, E. Akin, and A. Yilmaz. "Estimating driving behavior by a smartphone." In: *2012 IEEE Intelligent Vehicles Symposium*. 2012, pp. 234–239. DOI: 10.1109/IVS.2012.6232298.

[3] Z. Chen, J. Yu, Y. Zhu, Y. Chen, and M. Li. "D3: Abnormal driving behaviors detection and identification using smartphone sensors." In: *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 2015, pp. 524–532. DOI: 10.1109/SAHCN.2015.7338354.

[4] J. Yu, Z. Chen, Y. Zhu, Y. Chen, L. Kong, and M. Li. "Fine-Grained Abnormal Driving Behaviors Detection and Identification with Smartphones." In: *IEEE Transactions on Mobile Computing* 16.8 (2017), pp. 2198–2212. DOI: 10.1109/TMC.2016.2618873.

[5] J. Wahlström, I. Skog, and P. Händel. "Detection of Dangerous Cornering in GNSS-Data-Driven Insurance Telematics." In: *IEEE Transactions on Intelligent Transportation Systems* 16.6 (2015), pp. 3073–3083. DOI: 10.1109/TITS.2015.2431293.

[6] F. Li, H. Zhang, H. Che, and X. Qiu. "Dangerous driving behavior detection using smartphone sensors." In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 2016, pp. 1902–1907. DOI: 10.1109/ITSC.2016.7795864.

[7] C. Ryan, F. Murphy, and M. Mullins. "End-to-End Autonomous Driving Risk Analysis: A Behavioural Anomaly Detection Approach." In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2021), pp. 1650–1662. DOI: 10.1109/TITS.2020.2975043.

[8] P. Mohan, V. Padmanabhan, and R. Ramjee. "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones." In: Nov. 2008, pp. 323–336. DOI: 10.1145/1460412.1460444.

[9] C. Yang, A. Renzaglia, A. Paigwar, C. Laugier, and D. Wang. "Driving Behavior Assessment and Anomaly Detection for Intelligent Vehicles." In: *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. 2019, pp. 524–529. DOI: 10.1109/CIS-RAM47153.2019.9095790.

[10] Z. Liu, M. Wu, K. Zhu, and L. Zhang. "SenSafe: A Smartphone-Based Traffic Safety Framework by Sensing Vehicle and Pedestrian Behaviors." In: *Mobile Information Systems* 2016 (Oct. 2016), pp. 1–13. DOI: 10.1155/2016/7967249.

[11] J. Breitenstein, J.-A. Termöhlen, D. Lipinski, and T. Fingscheidt. "Systematization of Corner Cases for Visual Perception in Automated Driving." In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. 2020, pp. 1257–1264. DOI: 10.1109/IV47402.2020.9304789.

[12] T. Chakravarty, A. Ghose, C. Bhaumik, and A. Chowdhury. "MobiDriveScore — A system for mobile sensor based driving analysis: A risk assessment model for improving one's driving." In: Dec. 2013, pp. 338–344. ISBN: 978-1-4673-5222-2. DOI: 10.1109/ICSensT.2013.6727671.

[13] L. van der Maaten and G. Hinton. "Visualizing Data using t-SNE." In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605.

[14] H. Xu, G. Pang, Y. Wang, and Y. Wang. "Deep Isolation Forest for Anomaly Detection." In: *IEEE Transactions on Knowledge and Data Engineering* 35.12 (Dec. 2023), pp. 12591–12604. ISSN: 2326-3865. DOI: 10.1109/tkde.2023.3270293.

[15] K. Dietmayer. "Predicting of Machine Perception for Automated Driving." In: *Autonomous Driving: Technical, Legal and Social Aspects*. Ed. by M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 407–424. ISBN: 978-3-662-48847-8. DOI: 10.1007/978-3-662-48847-8_20.

[16] H. Zhu, K.-V. Yuen, L. Mihaylova, and H. Leung. "Overview of Environment Perception for Intelligent Vehicles." In: *IEEE Transactions on Intelligent Transportation Systems* 18.10 (2017), pp. 2584–2601. DOI: 10.1109/TITS.2017.2658662.

[17] M. Goldstein and S. Uchida. "A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data." In: *PLoS ONE* 11 (2016).

[18] V. Chandola, A. Banerjee, and V. Kumar. "Anomaly detection: A survey." In: *ACM Comput. Surv.* 41.3 (July 2009). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882.

[19] S. Schmidl, P. Wenig, and T. Papenbrock. "Anomaly detection in time series: a comprehensive evaluation." In: *Proc. VLDB Endow.* 15.9 (May 2022), pp. 1779–1797. ISSN: 2150-8097. DOI: 10.14778/3538598.3538602.

[20] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer. "KNN Model-Based Approach in Classification." In: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. Ed. by R. Meersman, Z. Tari, and D. C. Schmidt. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 986–996. ISBN: 978-3-540-39964-3.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. "LOF: identifying density-based local outliers." In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. SIGMOD '00. Dallas, Texas, USA: Association for Computing Machinery, 2000, pp. 93–104. ISBN: 1581132174. DOI: 10.1145/342009.335388.

[22] J. Tang, Z. Chen, A. W.-c. Fu, and D. W. Cheung. "Enhancing Effectiveness of Outlier Detections for Low Density Patterns." In: *Advances in Knowledge Discovery and Data Mining*. Ed. by M.-S. Chen, P. S. Yu, and B. Liu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 535–548. ISBN: 978-3-540-47887-4.

[23] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise." In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[24] X. Jin and J. Han. "K-Means Clustering." In: *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, 2010, pp. 563–564. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_425.

[25] Z. He, X. Xu, and S. Deng. "Discovering Cluster Based Local Outliers." In: *Pattern Recognition Letters* 24 (June 2003), pp. 1641–1650. DOI: 10.1016/S0167-8655(03)00003-5.

[26] M. Amer and M. Goldstein. "Nearest-Neighbor and Clustering based Anomaly Detection Algorithms for RapidMiner." In: 2012.

[27] D. Reynolds. "Gaussian Mixture Models." In: *Encyclopedia of Biometrics*. Ed. by S. Z. Li and A. Jain. Boston, MA: Springer US, 2009, pp. 659–663. ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_196.

[28] A. C. Harvey. "ARIMA Models." In: *Time Series and Statistics*. Ed. by J. Eatwell, M. Milgate, and P. Newman. London: Palgrave Macmillan UK, 1990, pp. 22–24. ISBN: 978-1-349-20865-4. DOI: 10.1007/978-1-349-20865-4_2.

[29] M. Goldstein and A. Dengel. "Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm." In: Sept. 2012.

[30] F. T. Liu, K. M. Ting, and Z.-H. Zhou. "Isolation Forest." In: *2008 Eighth IEEE International Conference on Data Mining*. 2008, pp. 413–422. DOI: 10.1109/ICDM.2008.17.

[31] S. Hariri, M. C. Kind, and R. J. Brunner. "Extended Isolation Forest." In: *IEEE Transactions on Knowledge and Data Engineering* 33.4 (Apr. 2021), pp. 1479–1489. ISSN: 2326-3865. DOI: 10.1109/tkde.2019.2947676.

[32] F. T. Liu, K. M. Ting, and Z.-H. Zhou. "On Detecting Clustered Anomalies Using SCiForest." In: *ECML/PKDD*. 2010.

[33] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah. "Time-series clustering – A decade review." In: *Information Systems* 53 (2015), pp. 16–38. ISSN: 0306-4379. DOI: https://doi.org/10.1016/j.is.2015.04.007.

[34] E. Keogh, S. Chu, D. Hart, and M. Pazzani. "SEGMENTING TIME SERIES: A SURVEY AND NOVEL APPROACH." In: *Data Mining in Time Series Databases*, pp. 1–21. DOI: 10.1142/9789812565402_0001. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789812565402_0001.

[35] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[36] A. Zheng and A. Casari. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. 1st. O'Reilly Media, Inc., 2018. ISBN: 1491953241.

[37] S. Khalid, T. Khalil, and S. Nasreen. "A survey of feature selection and feature extraction techniques in machine learning." In: *2014 Science and Information Conference*. 2014, pp. 372–378. DOI: 10.1109/SAI.2014.6918213.

[38] S. Wang, J. Tang, and H. Liu. "Feature Selection." In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, 2017, pp. 503–511. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_101.

[39] R. Bellman. "Dynamic programming." In: *Science* 153.3731 (1966), pp. 34–37.

[40] F. Heidecker, J. Breitenstein, K. Rösch, J. Löhdefink, M. Bieshaar, C. Stiller, T. Fingscheidt, and B. Sick. "An Application-Driven Conceptualization of Corner Cases for Perception in Highly Automated Driving." In: *CoRR* abs/2103.03678 (2021). arXiv: 2103.03678.

[41] L. Malta, C. Miyajima, and K. Takeda. "A Study of Driver Behavior Under Potential Threats in Vehicle Traffic." In: *IEEE Transactions on Intelligent Transportation Systems* 10.2 (2009), pp. 201–210. DOI: 10.1109/TITS.2009.2018321.

[42] H. Zhao, H. Zhou, C. Chen, and J. Chen. "Join driving: A smart phone-based driving behavior evaluation system." In: *2013 IEEE Global Communications Conference (GLOBECOM)*. 2013, pp. 48–53. DOI: 10.1109/GLOCOM.2013.6831046.

[43] J. Hofmockel and E. Sax. "Isolation Forest for Anomaly Detection in Raw Vehicle Sensor Data." In: *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS,* INSTICC. SciTePress, 2018, pp. 411–416. ISBN: 978-989-758-293-6. DOI: 10.5220/0006758004110416.

[44] M. Matousek, M. Yassin, A. Al-Momani, R. van der Heijden, and F. Kargl. "Robust Detection of Anomalous Driving Behavior." In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. 2018, pp. 1–5. DOI: 10.1109/VTCSpring.2018.8417777.

[45] Y. Zheng and J. H. Hansen. "Unsupervised driving performance assessment using free-positioned smartphones in vehicles." In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 2016, pp. 1598–1603. DOI: 10.1109/ITSC.2016.7795771.

[46] R. Bouman, Z. Bukhsh, and T. Heskes. *Unsupervised anomaly detection algorithms on real-world data: how many do we need?* 2023. arXiv: 2305.00735 [cs.LG].

[47] J. Lesouple, C. Baudoin, M. Spigai, and J.-Y. Tourneret. "Generalized Isolation Forest for Anomaly Detection." In: *Pattern Recognition Letters* 149 (June 2021). DOI: 10.1016/j.patrec.2021.05.022.

[48] M. Tokovarov and P. Karczmarek. "A probabilistic generalization of isolation forest." In: *Inf. Sci.* 584.C (Jan. 2022), pp. 433–449. ISSN: 0020-0255. DOI: 10.1016/j.ins.2021.10.075.

[49] K. H. Johansson, M. Törngren, and L. Nielsen. "Vehicle Applications of Controller Area Network." In: *Handbook of Networked and Embedded Control Systems*. Ed. by D. Hristu-Varsakelis and W. S. Levine. Boston, MA: Birkhäuser Boston, 2005, pp. 741–765. ISBN: 978-0-8176-4404-8. DOI: 10.1007/0-8176-4404-0_32.

[50] H. Chen, H. Ma, X. Chu, and D. Xue. "Anomaly detection and critical attributes identification for products with multiple operating conditions based on isolation forest." In: *Advanced Engineering Informatics* 46 (2020), p. 101139. ISSN: 1474-0346. DOI: https://doi.org/10.1016/j.aei.2020.101139.

[51] E. Karimov. "Binary Search Tree." In: Mar. 2020, pp. 87–100. ISBN: 978-1-4842-5768-5. DOI: 10.1007/978-1-4842-5769-2_11.

[52] E. W. Weisstein. *Euler-Mascheroni Constant*. From MathWorld–A Wolfram Web Resource. Accessed: January 19, 2025. n.d.

[53] Y. Cao, H. Xiang, H. Zhang, Y. Zhu, and K. M. Ting. *Anomaly Detection Based on Isolation Mechanisms: A Survey*. 2024. arXiv: 2403.10802 [cs.LG].

[54] K. Boyd, K. H. Eng, and C. D. Page. "Area under the Precision-Recall Curve: Point Estimates and Confidence Intervals." In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by H. Blockeel, K. Kersting, S. Nijssen, and F. Železný. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 451–466. ISBN: 978-3-642-40994-3.

[55] A. P. Bradley. "The use of the area under the ROC curve in the evaluation of machine learning algorithms." In: *Pattern Recognition* 30.7 (1997), pp. 1145–1159. ISSN: 0031-3203. DOI: https://doi.org/10.1016/S0031-3203(96)00142-2.

[56] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. "Microscopic Traffic Simulation using SUMO." In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[57] Y. Qiu, T. Misu, and C. Busso. "Driving Anomaly Detection with Conditional Generative Adversarial Network using Physiological and CAN-Bus Data." In: *2019 International Conference on Multimodal Interaction*. ICMI '19. Suzhou, China: Association for Computing Machinery, 2019, pp. 164–173. ISBN: 9781450368605. DOI: 10.1145/3340555.3353749.

[58] D. Zhang. *Dimensionality Reduction using t-Distributed Stochastic Neighbor Embedding (t-SNE) on the MNIST Dataset*. https://towardsdatascience.com/dimensionality-reduction-using-t-distributed-stochastic-neighbor-embedding-t-sne-on-the-mnist-9d36a3dd4521. Accessed: 2025-01-23.

[59] A. Maćkiewicz and W. Ratajczak. "Principal components analysis (PCA)." In: *Computers Geosciences* 19.3 (1993), pp. 303–342. ISSN: 0098-3004. DOI: https://doi.org/10.1016/0098-3004(93)90090-R.

[60] L. Deng. "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]." In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142. DOI: 10.1109/MSP.2012.2211477.

[61] F. Perez-Cruz. "Kullback-Leibler divergence estimation of continuous distributions." In: *2008 IEEE International Symposium on Information Theory*. 2008, pp. 1666–1670. DOI: 10.1109/ISIT.2008.4595271.

[62] J. M. Kanter and K. Veeramachaneni. "Deep feature synthesis: Towards automating data science endeavors." In: *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 2015, pp. 1–10. DOI: 10.1109/DSAA.2015.7344858.

[63] S. M. LaValle, M. S. Branicky, and S. R. Lindemann. "On the relationship between classical grid search and probabilistic roadmaps." In: *The International Journal of Robotics Research* 23.7-8 (2004), pp. 673–692.

[64] P. I. Frazier. *A Tutorial on Bayesian Optimization*. 2018. arXiv: 1807.02811 [stat.ML].

[65] Y. Xun, J. Liu, N. Kato, Y. Fang, and Y. Zhang. "Automobile Driver Fingerprinting: A New Machine Learning Based Authentication Scheme." In: *IEEE Transactions on Industrial Informatics* 16.2 (2020), pp. 1417–1426. DOI: 10.1109/TII.2019.2946626.

[66] Y. Cao and L. Wang. *Automatic Selection of t-SNE Perplexity*. 2017. arXiv: 1708.03229 [cs.AI].

[67] N. Sharma and S. Sharma. "Optimization of t-SNE by Tuning Perplexity for Dimensionality Reduction in NLP." In: Sept. 2023, pp. 519–528. ISBN: 978-981-99-3484-3. DOI: 10.1007/978-981-99-3485-0_41.

[68] L. McInnes, J. Healy, and J. Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML].