



Degree Project in Technology

Second cycle, 30 credits

Motorway Traffic Incident Detection By Machine Learning Classification

A Simulation Study

SIYUE HUANG

Motorway Traffic Incident Detection By Machine Learning Classification

A Simulation Study

SIYUE HUANG

Master's Programme, Information and Network Engineering, 120 credits
Date: June 19, 2025

Supervisors: Xiaolang Ma, Magnus Nordenvaad
Examiner: Magnus Jansson

School of Electrical Engineering and Computer Science

Host company: Swedish Transport Administration

Swedish title: Anomaliavkänning för motorvägstrafik genom
mönsterklassificering

Swedish subtitle: En simuleringsstudie

Abstract

Standing vehicle incidents on motorways are a major concern, which may lead to significant traffic congestion and pose serious safety risks. Effective traffic incident detection is crucial in **Intelligent Transportation System (ITS)**, enabling rapid responses from traffic management agencies, thereby minimizing potential adverse impacts. This project focuses on investigating data-driven algorithms for detecting standing vehicle incidents using traffic measurement data, specifically for motorways in Stockholm. The approach emphasizes leveraging spatial and temporal correlations within traffic data, particularly between upstream and downstream road segments.

The problem is formulated as a multi-class pattern classification task, where traffic measurement data from multiple consecutive road segments is utilized to classify if an incident occurs and its specific location. To evaluate the proposed incident detection system, microsimulation traffic models are developed using the SUMO software. Three machine learning models, Logistic Regression, **eXtreme Gradient Boosting (XGBoost)**, and **3-Dimensional Convolutional Neural Network (3D CNN)**, are evaluated. The **3D CNN** model demonstrates the most efficient performance across various metrics, including accuracy, precision, AUC-ROC, and false alarm rate. Further evaluation using traffic models with on/off-ramps confirms the **3D CNN** model's ability to capture complex spatio-temporal patterns. **XGBoost** also shows robust performance, indicating that both the **3D CNN** and **XGBoost** models are feasible for real-world incident detection applications.

Keywords

Machine Learning, Auto Incident Detection, Traffic Model Simulation

Sammanfattning

Förekomsten av stillastående fordonsincidenter på motorvägar kan leda till trafikstockningar och medföra betydande risker för människors säkerhet. Effektiv trafikincidentdetektion spelar en avgörande roll i det Intelligent Transport Systemet, vilket gör det möjligt för trafikstyrningsmyndigheter att svara snabbt och därmed minimera potentiella förluster.

Detta projekt undersöker datadrivna algoritmer för att upptäcka stillastående fordonsincidenter genom trafikmätdata, med särskilt fokus på motorvägar i Stockholm. Fokus ligger på att utnyttja rumsliga och tidsmässiga korrelationer i trafikdata, särskilt mellan uppströms- och nedströmssegment. Problemet formuleras som en flerklassig mönsterklassificeringsuppgift för att upptäcka anomalier, där detektorer övervakar flera konsekutiva vägsegment för att klassificera om en incident inträffar vid en viss tidpunkt.

På grund av brist på verkliga data konstrueras simuleringsmodeller i mjukvaran SUMO för att utvärdera systemet. Tre maskininlärningsmodeller, logistisk regression, Extreme Gradient Boosting och 3D-faltande neurala nätverk, utvärderades. 3D CNN-modellen visade sig vara mest effektiv enligt prestationsmått för noggrannhet, precision, AUC-ROC och falskalarmfrekvens. Ytterligare simuleringar med på-/avfarter bekräftade modellens förmåga att fånga rumsliga och tidsmässiga mönster. XGBoost visade också robusta resultat, vilket gör att båda modellerna är genomförbara för verklig incidentdetektion.

Nyckelord

Maskininläring, Automatisk olycksdetektion, Trafikmodellsimulering

Acknowledgments

I would like to express my gratitude to my supervisors Xiaoliang Ma, Magnus Nordenvaad, and my examiner Magnus Jansson for their unwavering guidance, patience, and insightful feedback. I am truly grateful for their continuous support and valuable advice throughout my research that made this work possible.

I would like to thank Long Xuan Nguyen, Shinya Hanaoka, and Tomoya Kawasaki's article 'Traffic conflict assessment for non-lane-based movements of motorcycles under congested conditions' for granting permission to use material from their work. The figure, reprinted from IATSS Research, Vol. 37, Issue 2, Pages 6, Copyright 2014, has been crucial in illustrating key points in my research.

I am also deeply thankful to my family for their love, trust, and encouragement as I pursued my path to becoming an engineer.

Last, I would like to extend my thanks to all the people I have encountered during my master's journey. I am truly grateful for the connections I have made and the experiences that have enriched my life during this time.

Stockholm, June 2025

Siyue Huang

Contents

1	Introduction	1
1.1	Background	1
1.2	Research Questions	3
1.3	Purpose	3
1.4	Goals	4
1.5	Research Methodology	4
1.6	Scope of the Thesis Project	5
1.7	Structure of the thesis	5
2	Background	6
2.0.1	Intelligent Transportation Systems	6
2.1	Microsimulation of Traffic Model	7
2.2	Classification Methods	8
2.2.1	Logistic Regression	9
2.2.2	XGBoost	10
2.2.3	3D CNNs	12
3	Methods	14
3.1	Problem Formulation	14
3.2	Feature Engineering	16
3.3	Logistic Regression	19
3.4	XGBoost	21
3.5	3D CNN	22
4	Experiment	25
4.1	Simulation Experiment Design	25
4.1.1	Highway Network Configuration	25
4.1.2	Scenario Design	26
4.2	Data Pre-processing	28
4.2.1	Data Overview	28

4.2.2	Raw Data	28
4.2.3	Merging and Pre-processing Raw Data	30
4.2.4	Distribution Analysis	32
4.2.5	Dataset Structure and Dimensions	36
4.2.6	Dealing Imbalanced Dataset	38
4.3	Modelling Parameters	38
5	Results and Analysis	40
5.1	Performance Metrics	40
5.2	Result for Different Parameters	41
5.3	Robustness Test	46
6	Conclusions and Future work	51
	References	53

List of Figures

2.1	Flowchart of Traffic Simulation Model. Reprinted from IATSS Research, Vol. 37, Issue 2, Long Xuan Nguyen, Shinya Hanaoka, Tomoya Kawasaki, Traffic conflict assessment for non-lane-based movements of motorcycles under congested conditions, Pages 6, Copyright 2014, with permission from Elsevier [17]	8
2.2	Structure of the XGBoost model. Each tree attempts to correct the residuals of the previous ensemble.	10
3.1	Schematic diagram of the highway structure, where traffic detectors are placed at gantries between road segments.	14
3.2	Example of highway traffic measurements when no incident is present ($y_i = 0$).	15
3.3	Example of highway traffic measurements when an incident occurs on the third segment ($y_i = 3$).	15
3.4	Logistic Regression Structure	19
3.5	Architecture of the 3D CNN model.	23
4.1	Highway Network	26
4.2	Comparison of nVehContrib distributions for incidents at upstream and downstream.	33
4.3	Comparison of speed distributions for incidents at upstream and downstream.	34
4.4	Comparison of occupancy distributions for incidents at upstream and downstream.	36
5.1	Confusion Matrix of 3D CNN Model with ($S=2, t_c=5$)	44
5.2	Confusion Matrix of XGBoost Model with ($S=3, t_c=3$)	45
5.3	Shap Summary Plot of Class 0 for XGBoost Model with ($S=3, t_c=3$)	46

5.4	Shap Summary Plot of Class 1 for XGBoost Model with (S=3, $t_c=3$)	46
5.5	Shap Summary Plot of Class 2 for XGBoost Model with (S=3, $t_c=3$)	47
5.6	Shap Summary Plot of Class 3 for XGBoost Model with (S=3, $t_c=3$)	47
5.7	Highway with Ramps	48
5.8	Overall Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)	49
5.9	Class 0 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)	49
5.10	Class 1 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)	49
5.11	Class 2 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)	49

List of Tables

4.1	Highway Configuration Parameters	26
4.2	Traffic Demand Parameters under Different Scenarios.	27
4.3	Parameters defining a standing vehicle incident.	27
4.4	Speed Distribution Parameters for Different Vehicle Types.	28
4.5	Traffic Measurement Data Description	29
4.6	Incident Data Description	30
4.7	Traffic Detector Data Description	31
4.8	Number of Instances in the Dataset for Various Combinations of t_c (Time Cycles) and S (Road Segments)	37
4.9	Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments).	37
4.10	Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments) After Step 1.	38
4.11	Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments) After Step 2.	39
4.12	Classifier Parameters	39
5.1	Performance metrics for different models and datasets ($S=1$)	42
5.2	Performance metrics for different models and datasets ($S=2$)	42
5.3	Performance metrics for different models and datasets ($S=3$)	43
5.4	Performance Metrics for 3D CNN Model with ($S=2, t_c=5$)	43
5.5	Performance Metrics for XGBoost Model with ($S=3, t_c=3$)	44
5.6	Performance Metrics for 3D CNN Model with ($S=2, t_c=5$)	47
5.7	Performance Metrics for 3D CNN Model with ($S=3, t_c=5$)	48
5.8	Performance Metrics for Logistic Regression Model with ($S=3, t_c=5$)	49
5.9	Performance Metrics for XGBoost Model with ($S=3, t_c=5$)	50

List of acronyms and abbreviations

1D	1-Dimensional
3D CNN	3-Dimensional Convolutional Neural Network
AID	Auto Incident Detection
CCTV	Closed-Circuit TeleVision
CNN	Convolutional Neural Network
DR	Detection Rate
FAR	False Alarm Rate
ITS	Intelligent Transportation System
MCS	Motorway Control System
ML	Machine Learning
MLE	Maximum Likelihood Estimation
OvR	One-vs-Rest
ReLU	Rectified Linear Unit
SHAP	SHapley Additive exPlanations
SUMO	Simulation of Urban MObility
XGBoost	eXtreme Gradient Boosting

Chapter 1

Introduction

1.1 Background

The continuous expansion of urban areas and rapid population growth have significantly escalated the demand for mobility [1]. This heightened demand places immense pressure on the road infrastructure, leading to increased traffic density. Among the most common disruptions on highways, standing vehicle incidents have emerged as a critical concern. Such incidents lead to significant disruption in traffic flow, reduction in road capacity, and potentially secondary accidents. Accurate and quick detection of standing vehicles is therefore crucial to mitigate these outcomes effectively, ensuring efficient highway management and public safety.

Intelligent Transportation System (ITS) has been developed to address challenges in modern traffic management. **ITS** integrates advanced technologies and communication to enhance the efficiency and safety of transportation networks [2]. These systems monitor information such as **Closed-Circuit TeleVision (CCTV)** data and traffic measurement data by sensors in real-time and make intelligent decisions to help reduce congestion, improve public safety, and promote sustainable transportation solutions [3]. Within this framework, **Auto Incident Detection (AID)** system plays a vital role. It has been developed to detect incidents automatically by monitoring traffic conditions and setting alarms for incidents. Its performance is commonly evaluated by the metrics **Detection Rate (DR)** and **False Alarm Rate (FAR)**.

The core challenge in **AID** lies in anomaly detection [4], which involves identifying observations that significantly deviate from established, normal traffic patterns. Traditionally, model-driven approaches detect incidents

by constructing physical models and identifying anomalies based on these models. While often effective in controlled environments, these approaches may exhibit limited robustness when confronted with the complexity and inherent noise of real-world traffic data [5].

In contrast, data-driven approaches learn directly from the data itself. By treating labeled incident data as anomalies, it's possible to expand the understanding of these deviations and their potential impact within a system. This framework allows for the development of data-driven methods where classification algorithms can effectively differentiate between normal traffic conditions and incidents [6]. Furthermore, the integration of statistical or **Machine Learning (ML)** techniques enables data-driven models to detect anomalies within large datasets, fostering adaptability to the dynamic and complex conditions of highway traffic. Consequently, modern **AID** systems increasingly adopt a data-driven approach due to their enhanced flexibility and performance.

For data-driven classification, several well-established methods are commonly employed, including Logistic Regression [7], **eXtreme Gradient Boosting (XGBoost)** [8], and **Convolutional Neural Networks (CNNs)** [9]. Logistic Regression, for instance, maps binary predictions and their associated probabilities via a sigmoid function, relying on a set of independent input variables. While it offers valuable insights into the importance of input features, its strong assumptions about these variables can sometimes limit its applicability in diverse real-world scenarios.

XGBoost, on the other hand, addresses supervised classification tasks by ensembling weak learners, such as decision trees, in a sequential manner within a gradient boosting framework. Decision trees themselves function by iteratively branching based on decision rules, with their leaf nodes indicating the final classification results [10]. **XGBoost** is particularly effective at identifying efficient classification strategies and is also robustly capable of handling multi-class classification tasks.

Meanwhile, **CNNs** process input variables through multiple hidden layers, ultimately outputting probabilities for each class. The weights between these layers are typically optimized through backpropagation. A key characteristic of **CNNs** lies in their convolutional layers, where a small kernel filters across the input to extract features with the spatial relationships preserved [9].

Each of these mentioned methods offers unique and valuable insights into the classification task. Furthermore, significant improvements have been developed in recent years, enhancing their performance and broadening their applicability. Consequently, algorithms derived from these foundational

concepts will be explored and employed for incident detection within this project.

The Swedish Transport Administration aims to leverage traffic measurement data for the detection of standing vehicle incidents on motorways, to enable cost-effective and sustainable traffic management in Stockholm.

1.2 Research Questions

The primary objective of the project is to develop a data-driven multi-class classification model for the real-time detection and precise localization of standing vehicle incidents. The model will utilize traffic measurement data to monitor long stretches of highway, pinpointing the specific road segments where incidents occur with high accuracy.

To achieve this, we will investigate several data-driven machine learning methods, including Logistic Regression, **XGBoost**, and **3-Dimensional Convolutional Neural Networks (3D CNNs)**. The performance of these models will be evaluated using simulated traffic data generated from the microscopic simulation software SUMO [11]. Therefore, the research questions raised in the project are:

Which **ML** method provides the most accurate and reliable detection of standing vehicle incidents based on traffic measurements?

What is the performance and robustness of the best-performing model across different types of highway networks, especially those incorporating complex elements like on/off-ramps?

1.3 Purpose

The project aims to investigate the performance and robustness of data-driven methods for traffic incident detection. The benefits can be summarized as follows:

First, accurate real-time incident detection can lead to timely and efficient responses from the traffic management department, protecting public safety and reducing congestion [1].

Second, from an ethics and sustainability standpoint, this project explores a highly cost-effective solution for improving traffic management and incident response. It achieves this by leveraging aggregated traffic data derived from widely and intensively deployed, low-cost detectors. Furthermore, by utilizing

simulated data rather than real-world data, the project inherently minimizes potential privacy concerns associated with sensitive traffic information.

1.4 Goals

The overarching goal of this project is to investigate data-driven approaches for traffic incident detection using simulated traffic measurement data. From a system design perspective, this primary goal can be broken down into the following specific sub-goals:

1. Subgoal 1: Simulation-Based Dataset Generation

Develop and execute experiments within the microsimulation platform SUMO, modeling a representative Stockholm highway network. These simulation experiments will generate a robust dataset encompassing traffic measurement data for various incident types and diverse traffic scenarios, which is crucial for comprehensively evaluating different machine learning methods.

2. Subgoal 2: Machine Learning Algorithm Investigation

Investigate and apply different types of machine learning algorithms, including Logistic Regression, **XGBoost**, and **3D CNNs**, to address the multi-class classification task of incident detection and localization.

3. Subgoal 3: Evaluation Framework Design Design a structured framework to facilitate the assessment of different machine learning methods. This involves establishing procedures for off-line model training and subsequent testing, where their performance will be quantified using a defined set of metrics.

1.5 Research Methodology

The research problem is formulated as a multi-class classification task. Machine learning models: logistic Regression, **XGBoost**, and **3D CNNs** are implemented to provide accurate incident detection on specific road segments.

This project employs a microsimulation framework using SUMO [11] to generate the necessary traffic measurement data. This approach allows us to evaluate various incident detection models and facilitate a quantitative analysis. The simulation scenarios are controlled by varying traffic demand

levels and introducing single or multiple independent incidents. The known incident occurrences within these simulations serve as the ground truth for our study. The traffic measurement data, including parameters like velocity, density, and flow, are used as inputs to the incident detection models.

1.6 Scope of the Thesis Project

The experiments are designed and implemented based exclusively on simulation data. Real-world data was not considered for this project due to the significant imbalance between non-incident and incident data typically found in real-world datasets, which could lead to model overfitting [12].

Furthermore, this project focuses on detecting single traffic incidents rather than correlated incidents. Therefore, the project assumes that incidents are independent of each other and that at any given time, a maximum of one incident occurs.

1.7 Structure of the thesis

This thesis is organized into six chapters. Chapter 2 introduces essential background information about *ITS*, the principles of microsimulation in traffic modeling, and relevant classification theories pertinent to the study. Chapter 3 delves into the problem formulation, detailing the specific challenges and outlining the proposed structure for solving this problem using various classification methods. The experimental design and implementation are thoroughly presented in Chapter 4, covering everything from the setup of our evaluation framework in SUMO to the intricacies of data modeling. Subsequently, Chapter 5 is dedicated to the analysis and discussion of the performance metrics derived from the different machine learning models, complemented by additional robustness tests to validate their efficacy. Finally, Chapter 6 brings the project to a close by summarizing the key findings and delineating potential directions for future work.

Chapter 2

Background

2.0.1 Intelligent Transportation Systems

ITSs represent a modern approach to enhancing traffic management. These systems integrate traditional transportation infrastructure with advanced information technology, playing a crucial role in managing the increasing demand for mobility. Their deployment contributes to safer, more efficient, and more sustainable transportation networks [13].

A significant application of **ITS** on motorways is the **Motorway Control System (MCS)**, which serves as a foundational infrastructure for monitoring, collecting, and managing traffic data. By monitoring traffic state data and adjusting lane signals when needed, that system can manage traffic flow automatically [14]. For instance, in the event of traffic incidents, the **MCS** can display warning messages to inform drivers about lane closures or upcoming hazards and then guide the traffic flow.

Within **ITSs**, detectors are crucial for data collection. Widely distributed across road networks, these detectors gather real-time information on key traffic state parameters, enabling timely decision-making. Those data are foundations that drive the operations of **ITSs** and ensure accurate and timely traffic insights. Two primary types of data are commonly utilized:

- **CCTV Data:** Closed-circuit television cameras, strategically installed along the network, provide both video feeds and images of traffic. Detailed information about vehicle movements can be used for tasks like tracking vehicle trajectories or detecting incidents [15]. However, the deployment and maintenance of these cameras are often costly, and processing video data demands significant computational resources.

- **Traffic Measurement Data:** Detectors integrated into the **MCS** infrastructure collect numerical traffic state data, including vehicle counts, average speeds, and occupancy levels. These data are typically aggregated over short collection cycles, offering high-resolution insights into traffic flow dynamics. Compared to **CCTV**, the collection of traffic state data is more scalable and cost-efficient, making it a reliable choice for large-scale network monitoring.

By integrating **MCS** and other advanced technologies, **ITSs** proactively manages transportation systems. This capability helps mitigate traffic congestion, improve safety, and reduce environmental impact, making **ITSs** an essential tool for developing smart cities and fostering sustainable urban mobility.

2.1 Microsimulation of Traffic Model

Traffic modeling is an efficient way to tackle complex tasks in transport engineering. By modeling and simulating traffic networks in well-developed software [16], these traffic models can generate data that closely resembles real-world conditions, making it suitable for further investigation. Microsimulation is a specialized traffic modeling technique that offers a vehicle-level perspective. It focuses on the movement and interactions of individual vehicles over time, achieved by modeling distinct vehicle behaviors. The parameters involved typically include acceleration, lane-changing decisions, interactions with other vehicles, and drivers' responses to traffic signals. Some of these interactions are governed by predefined behavioral rules or algorithms, such as car-following and lane-changing models. Figure 2.1 illustrates a typical microsimulation process. It begins with the initialization of vehicles and the road network, then iterates vehicle movements based on dynamic environmental and traffic conditions until the simulation concludes.

Simulation of Urban MObility (SUMO) [18] is a commonly used traffic microsimulation platform. It stands out for being open-source and highly flexible, notably featuring an on-line interaction interface called *Traci*. SUMO provides various types of outputs, such as traffic measurements and vehicle trajectories, allowing researchers to analyze the simulated system under diverse conditions. Moreover, its flexible interaction capabilities make it a valuable tool for simulating traffic incidents [19], offering a robust framework for evaluating traffic incident detection algorithms.

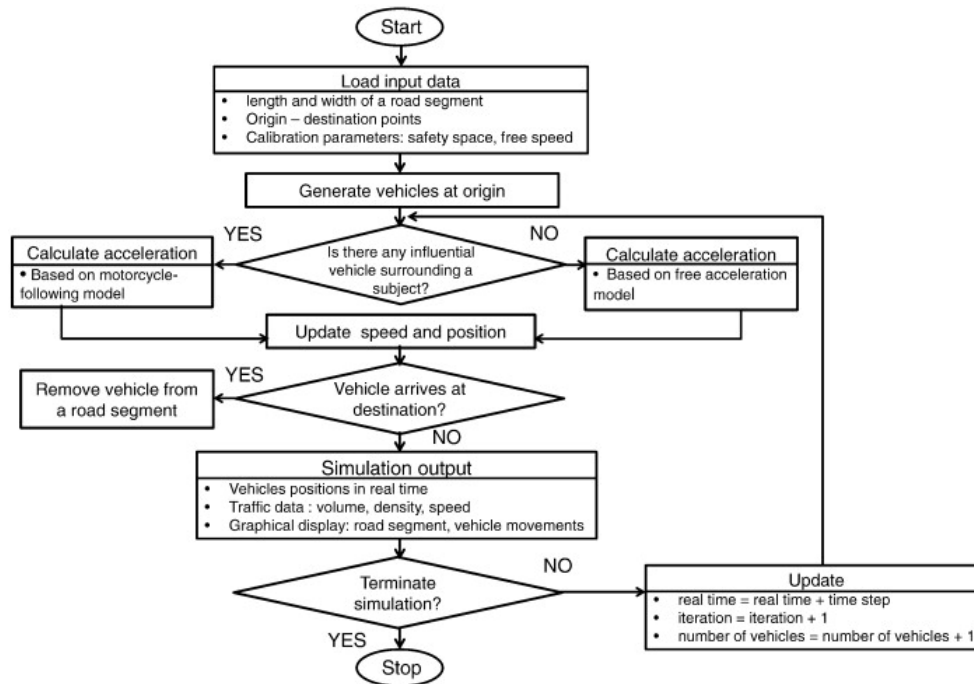


Figure 2.1: Flowchart of Traffic Simulation Model. Reprinted from IATSS Research, Vol. 37, Issue 2, Long Xuan Nguyen, Shinya Hanaoka, Tomoya Kawasaki, Traffic conflict assessment for non-lane-based movements of motorcycles under congested conditions, Pages 6, Copyright 2014, with permission from Elsevier [17]

2.2 Classification Methods

The real-time incident detection problem can be effectively formulated as a classification task [20]. In this context, the objective is to determine whether an incident is occurring based on observed traffic conditions under specific scenarios. Its simplest form is a binary classification task, where the model predicts one of two possible outcomes: incident or non-incident. More complex versions of this problem extend into multi-class classification, where the classes may vary according to the severity, reason, or specific location of the incidents, depending on the problem definition.

Incident detection methodologies have evolved significantly over the years, transitioning from traditional approaches based on fixed traffic thresholds to advanced machine learning models. Modern machine learning methods offer more robust and scalable solutions for the incident detection problem, encompassing both binary and multi-class classification frameworks [21].

These advanced methods possess the capability to model complex non-linear relationships and efficiently process large datasets with numerous features, leading to more accurate and adaptable detection systems.

2.2.1 Logistic Regression

Logistic regression is a widely used method for binary classification. Given an observation vector $\mathbf{x}_i \in \mathbb{R}^m$ and a corresponding binary target variable $y_i \in \{0, 1\}$, logistic regression models the conditional probability of $y_i = 1$ as a function of the input \mathbf{x}_i , parameterized by a weight vector \mathbf{w} and bias term w_0 .

The predicted probability that $y_i = 1$ given \mathbf{x}_i is defined as:

$$\hat{p}(\mathbf{x}_i) = P_{\mathbf{w}}(y_i = 1 \mid \mathbf{x}_i) = \frac{1}{1 + e^{-(w_0 + \mathbf{x}_i^T \mathbf{w})}}. \quad (2.1)$$

Accordingly, the probability that $y_i = 0$ is:

$$P_{\mathbf{w}}(y_i = 0 \mid \mathbf{x}_i) = 1 - \hat{p}(\mathbf{x}_i) = \frac{e^{-(w_0 + \mathbf{x}_i^T \mathbf{w})}}{1 + e^{-(w_0 + \mathbf{x}_i^T \mathbf{w})}}. \quad (2.2)$$

For a dataset with n observations, the model parameters are estimated by maximizing the log-likelihood function:

$$\log L(\mathbf{w}) = \sum_{i=1}^n [y_i \log \hat{p}(\mathbf{x}_i) + (1 - y_i) \log(1 - \hat{p}(\mathbf{x}_i))]. \quad (2.3)$$

The optimal parameters \mathbf{w}^* are obtained via maximum likelihood estimation (MLE):

$$\mathbf{w}^* = \arg \max_{\mathbf{w}} \log L(\mathbf{w}). \quad (2.4)$$

Given the learned parameters \mathbf{w}^* , the predicted class label \hat{y}_i is:

$$\hat{y}_i = \arg \max_{y_i \in \{0,1\}} P_{\mathbf{w}^*}(y_i \mid \mathbf{x}_i). \quad (2.5)$$

The binary logistic regression model can be extended to multi-class classification using the **One-vs-Rest (OvR)** strategy, in which multiple binary classifiers are trained independently, and the final prediction corresponds to the class with the highest estimated probability.

While logistic regression offers interpretability and insight into feature importance, its assumption of a linear relationship between the input features

and the log-odds of the outcome may limit its effectiveness in more complex, non-linear settings.

2.2.2 XGBoost

XGBoost is a highly efficient and scalable implementation of gradient-boosted decision trees. It leverages an ensemble learning framework to build a strong predictive model from a collection of weak learners [8]. As illustrated in Figure 2.2, XGBoost constructs a sequence of decision trees, where each tree is trained not only on the input features but also on the residual errors of the previous trees. This is achieved by setting the learning objective of each tree to minimize the residuals from the ensemble constructed thus far.

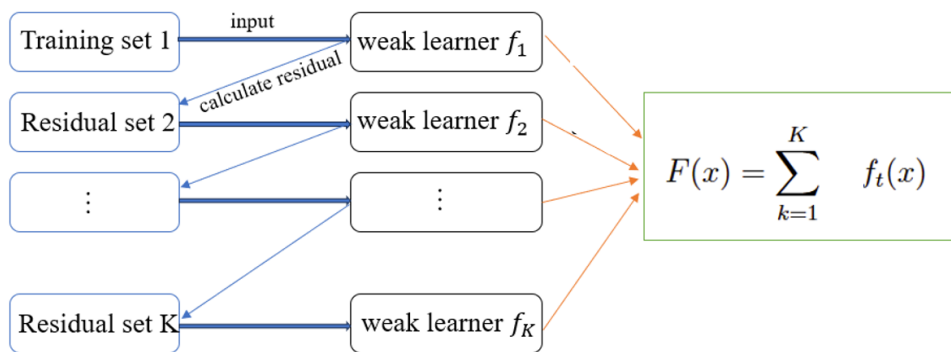


Figure 2.2: Structure of the XGBoost model. Each tree attempts to correct the residuals of the previous ensemble.

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where n is the number of samples, each input $\mathbf{x}_i \in \mathbb{R}^m$ consists of m features, and the corresponding binary target variable is $y_i \in \{0, 1\}$.

The prediction function of XGBoost is defined as:

$$\hat{y}_i = \sum_{k=1}^K f_k(\mathbf{x}_i), \quad f_k \in \mathcal{F}, \quad (2.6)$$

where K is the number of decision trees, each f_k is a regression tree (weak learner), and \mathcal{F} denotes the space of all possible trees.

The overall objective function is formulated as:

$$\text{obj}(\phi) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k), \quad (2.7)$$

where ϕ represents the model parameters, $l(\cdot, \cdot)$ is a differentiable loss function (e.g., logistic loss for binary classification), and $\Omega(f_k)$ is a regularization term that controls model complexity to avoid overfitting.

Each decision tree f_k is defined by a structure function $q(\mathbf{x})$, which maps an input to a leaf index, and a set of leaf weights ω . Formally, a tree can be represented as:

$$f_k(\mathbf{x}) = \omega_{q(\mathbf{x})}, \quad \omega \in \mathbb{R}^L, \quad (2.8)$$

where L is the number of leaves in the tree.

The regularization term $\Omega(f_k)$ is defined as:

$$\Omega(f_k) = \gamma L + \frac{1}{2} \lambda \sum_{j=1}^L \omega_j^2, \quad (2.9)$$

where γ and λ are regularization coefficients that penalize the number of leaves and the magnitude of leaf weights, respectively.

The prediction process begins with an initial prediction for all samples:

$$\hat{y}_i^{(0)} = 0. \quad (2.10)$$

At each boosting iteration k , a new tree is trained to minimize the loss based on the previous prediction, and the prediction is updated as:

$$\hat{y}_i^{(k)} = \hat{y}_i^{(k-1)} + f_k(\mathbf{x}_i). \quad (2.11)$$

After K iterations, the final prediction is given by:

$$\hat{y}_i = F(\mathbf{x}_i) = \sum_{k=1}^K f_k(\mathbf{x}_i). \quad (2.12)$$

XGBoost sequentially minimizes the residual errors of previous trees, making it highly effective for both binary and multi-class classification tasks when appropriately extended.

2.2.3 3D CNNs

CNNs are a class of deep learning architectures particularly effective for processing high-dimensional spatial data. **CNNs** consist of multiple types of layers, including convolutional, activation, pooling, and fully connected layers. These components collectively extract hierarchical features, reduce spatial dimensionality, and ultimately perform classification.

The core building block of a **CNN** is the **convolutional layer**, which applies a set of learnable filters, also called kernels, to the input tensor. Each kernel slides over the input tensor, performing element-wise multiplication followed by summation to produce a feature map. Unlike signal processing, where kernels represent fixed functions, in **CNNs**, kernels are trainable parameters optimized during the learning process.

3D CNNs extend this idea to three-dimensional convolutions, allowing the network to capture spatial and temporal dependencies in volumetric data. Mathematically, a 3D convolution operation is defined as:

$$y_{i,j,k} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \sum_{p=0}^{P-1} x_{i+m,j+n,k+p} \cdot w_{m,n,p} + b, \quad (2.13)$$

where x is the input tensor, w is the 3D kernel of size (M, N, P) , and b is a bias term.

When applying a set of n 3D kernels of shape (M, N, P) to an input tensor of shape (f, Q, W, E) without padding, the output tensor will have the shape:

$$(f \cdot n, Q - M + 1, W - N + 1, E - P + 1), \quad (2.14)$$

where f is the number of input channels and each kernel is applied across all input channels.

After convolution, a **non-linear activation function**, such as the **Rectified Linear Unit (ReLU)**, is typically applied to introduce non-linearity into the model:

$$\text{ReLU}(z) = \max(0, z), \quad (2.15)$$

where z is the input value. ReLU helps the model learn complex patterns and improves convergence by mitigating the vanishing gradient problem.

A **pooling layer** is often used after activation to reduce spatial dimensions while preserving the most salient features. Common pooling operations include max pooling and average pooling. For example, max pooling with

a window size (w_h, w_w, w_d) and stride (s_h, s_w, s_d) is defined as:

$$y_{i,j,k} = \max_{\substack{0 \leq p < w_h \\ 0 \leq q < w_w \\ 0 \leq r < w_d}} x_{i \cdot s_h + p, j \cdot s_w + q, k \cdot s_d + r}. \quad (2.16)$$

Given an input volume of dimensions (H, W, D) , the output dimensions after pooling (assuming no padding) are:

$$H_{\text{out}} = \left\lfloor \frac{H - w_h}{s_h} \right\rfloor + 1, \quad W_{\text{out}} = \left\lfloor \frac{W - w_w}{s_w} \right\rfloor + 1, \quad D_{\text{out}} = \left\lfloor \frac{D - w_d}{s_d} \right\rfloor + 1. \quad (2.17)$$

At the end of the **CNNs** architecture, one or more **fully connected layers** are used to map the learned feature representations to the desired output space. Each neuron in a fully connected layer is connected to all neurons in the preceding layer. The output of an fully connected layer is computed as:

$$y_i = \sigma \left(\sum_{j=1}^N w_{ij} x_j + b_i \right), \quad (2.18)$$

where x_j is the input, w_{ij} is the weight connecting neuron j to neuron i , b_i is the bias term, and σ is an activation function such as **ReLU** or softmax.

The overall **CNNs** architecture generally follows a forward hierarchical structure: multiple stacked convolutional and pooling layers perform feature extraction, followed by fully connected layers for classification.

For classification tasks, the model is typically trained using the **cross-entropy loss**, which measures the discrepancy between the predicted probabilities and the true labels:

$$L = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log(\hat{y}_{i,c}), \quad (2.19)$$

where $y_{i,c}$ is a one-hot encoded ground truth label, $\hat{y}_{i,c}$ is the predicted probability for class c , N is the number of samples, and C is the number of classes.

Model parameters are optimized via backpropagation and gradient descent. Gradients of the loss function with respect to the model parameters are computed using the chain rule and used to update parameters in the direction that minimizes the loss.

Chapter 3

Methods

3.1 Problem Formulation

The objective of this study is to develop a model capable of detecting standing vehicle incidents on a specific stretch of highway. This highway stretch is conceptually divided into multiple road segments. These segments are demarcated by the presence of gantries, each equipped with traffic detectors that provide continuous measurement data at their respective boundaries, as visually illustrated in Figure 3.1.

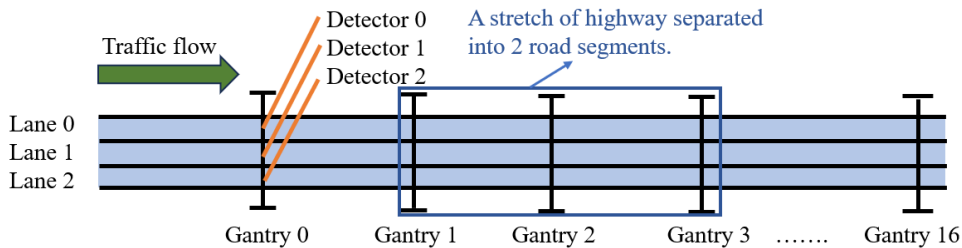


Figure 3.1: Schematic diagram of the highway structure, where traffic detectors are placed at gantries between road segments.

The dataset is denoted as $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where \mathbf{x}_i represents the i^{th} observation, and y_i is the corresponding ground truth label. Each \mathbf{x}_i is a sequence of traffic measurements collected from S road segments over t_c consecutive time steps. The label y_i indicates the presence and location of an incident during this observation window.

Under the assumption that at most one incident can occur within a single observation window, the classification task is formulated as a multi-class

problem with $S + 1$ possible outcomes:

$$y_i \in \{0, 1, \dots, S\},$$

where $y_i = 0$ denotes that no incident occurred, and $y_i = k$ indicates that an incident occurred in the k^{th} road segment.

For example, when $S = 3$, the input \mathbf{x}_i contains traffic measurements from 4 gantries (i.e., $S+1$). Figure 3.2 shows a scenario where no incident occurred, i.e., $y_i = 0$, while Figure 3.3 illustrates a case where an incident occurred on the third segment, i.e., $y_i = 3$.

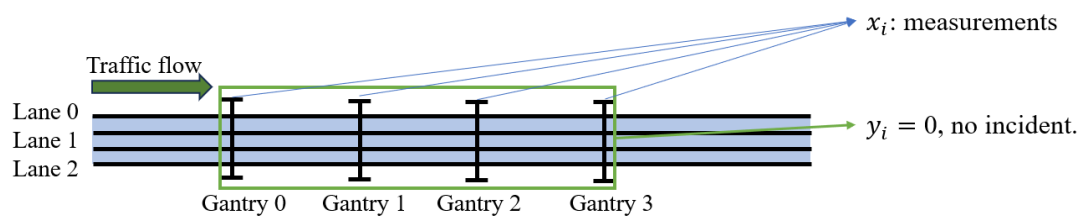


Figure 3.2: Example of highway traffic measurements when no incident is present ($y_i = 0$).

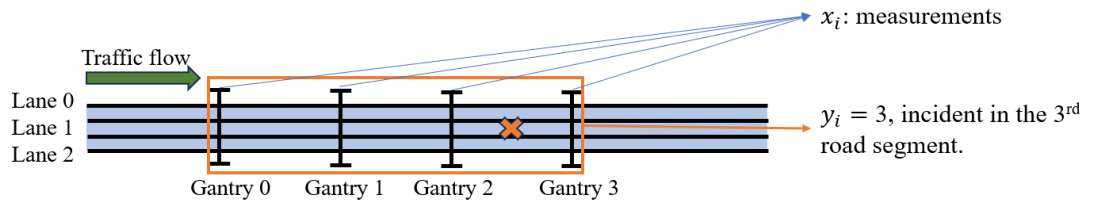


Figure 3.3: Example of highway traffic measurements when an incident occurs on the third segment ($y_i = 3$).

The detection problem is modeled as a supervised multi-class classification task. A classification model parameterized by Θ is defined as:

$$\hat{y}_i = F(\mathbf{x}_i | \Theta),$$

where \hat{y}_i is the model's prediction of the ground truth label y_i .

The learning objective is to minimize a regularized empirical risk:

$$L(\Theta) = \sum_{i=1}^N \ell(y_i, \hat{y}_i) + \lambda R(\Theta),$$

where $\ell(y_i, \hat{y}_i)$ is a loss function (e.g., cross-entropy) that measures the prediction error for sample i , $R(\Theta)$ is a regularization term penalizing model complexity to prevent overfitting, and $\lambda > 0$ is a hyperparameter controlling the strength of the regularization.

The goal is to find the optimal parameter set Θ^* that minimizes the total loss:

$$\Theta^* = \arg \min_{\Theta} L(\Theta).$$

Optimization of Θ is typically achieved using gradient-based methods, such as stochastic gradient descent or its variants. These algorithms iteratively adjust model parameters in the direction that reduces the loss function, thereby improving prediction accuracy on the classification task.

3.2 Feature Engineering

Classical machine learning models such as Logistic Regression and **XGBoost** typically require **1-Dimensional (1D)** input vectors. However, this constraint limits their ability to capture the inherent spatio-temporal dynamics present in traffic measurement data, which are crucial for effective incident detection. To address this limitation, the feature engineering strategy is designed to transform raw multi-dimensional measurements into a structured **1D** feature vector, enabling the models to utilize spatial and temporal variations more effectively.

Given the dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, each observation \mathbf{x}_i is a four-dimensional tensor of shape (F, G, L, T) , where:

- $F = 3$: number of measurement types (e.g., speed, flow, occupancy),
- $G = S + 1$: number of detectors (one at each end of each of the S road segments),
- $L = 3$: number of lanes per detector,
- $T = t_c$: number of time steps (observation window length).

To transform \mathbf{x}_i into a meaningful **1D** input vector for classification, four categories of engineered features are defined. Each category is specifically designed to capture distinct temporal or spatial characteristics and patterns within the data.

Feature Type 1: Segment-Based Spatial Difference Features

This feature type captures spatial differences between the upstream and downstream detectors of each road segment. Specifically, for each measurement type f , road segment index s , and lane index l , it computes the average value over the time window T for the upstream and downstream detectors, and takes their difference.

For each sample \mathbf{x}_i , define:

$$\phi_{f,s,l}^{(1)} = \frac{1}{T} \sum_{t=1}^T \left(\mathbf{x}_i^{(f,s,l,t)} - \mathbf{x}_i^{(f,s+1,l,t)} \right),$$

where $f = 1, \dots, F$ (feature type, e.g., speed, flow, occupancy), $s = 1, \dots, S$ (road segment index), and $l = 1, \dots, L$ (lane index).

Each $\phi_{f,s,l}^{(1)}$ is a scalar feature, and the complete feature vector of this type is:

$$\mathbf{x}_i^{(1)} = \left[\phi_{f,s,l}^{(1)} \right]_{f=1,\dots,F; s=1,\dots,S; l=1,\dots,L} \in \mathbb{R}^{F \times S \times L}.$$

This yields a total of $F \times S \times L$ features for Feature Type 1.

Feature Type 2: Segment-Based Temporal Change Features

This feature captures how the spatial difference between upstream and downstream detectors of a segment changes over time by comparing historical averages with the most recent measurement.

$$\phi_{f,s,l}^{(2)} = \left(\frac{1}{T} \sum_{t=1}^T \mathbf{x}_i^{(f,s,l,t)} - \mathbf{x}_i^{(f,s,l,T)} \right) - \left(\frac{1}{T} \sum_{t=1}^T \mathbf{x}_i^{(f,s+1,l,t)} - \mathbf{x}_i^{(f,s+1,l,T)} \right),$$

for $f = 1, \dots, F$; $s = 1, \dots, S$; $l = 1, \dots, L$.

This feature vector is:

$$\mathbf{x}_i^{(2)} = \left[\phi_{f,s,l}^{(2)} \right] \in \mathbb{R}^{F \times S \times L}.$$

This yields a total of $F \times S \times L$ features for Feature Type 2.

Feature Type 3: Detector-Based Spatial Difference Features

This feature quantifies the spatial variation between adjacent detectors (e.g., lanes) on the same gantry:

$$\phi_{f,g,l}^{(3)} = \frac{1}{T} \sum_{t=1}^T \left(\mathbf{x}_i^{(f,g,l,t)} - \mathbf{x}_i^{(f,g,l+1,t)} \right),$$

$$\text{for } f = 1, \dots, F; \quad g = 1, \dots, G; \quad l = 1, \dots, L - 1.$$

The resulting feature vector is:

$$\mathbf{x}_i^{(3)} = \left[\phi_{f,g,l}^{(3)} \right] \in \mathbb{R}^{F \times G \times (L-1)}.$$

This yields a total of $F \times S \times (L - 1)$ features for Feature Type 3.

Feature Type 4: Detector-Based Temporal Change Features

This feature captures the change over time in spatial differences within the same detector (gantry), similar in structure to Feature Type 2:

$$\phi_{f,g,l}^{(4)} = \left(\frac{1}{T} \sum_{t=1}^T \mathbf{x}_i^{(f,g,l,t)} - \mathbf{x}_i^{(f,g,l,T)} \right) - \left(\frac{1}{T} \sum_{t=1}^T \mathbf{x}_i^{(f,g,l+1,t)} - \mathbf{x}_i^{(f,g,l+1,T)} \right),$$

$$\text{for } f = 1, \dots, F; \quad g = 1, \dots, G; \quad l = 1, \dots, L - 1.$$

The resulting feature vector is:

$$\mathbf{x}_i^{(4)} = \left[\phi_{f,g,l}^{(4)} \right] \in \mathbb{R}^{F \times G \times (L-1)}.$$

This yields a total of $F \times S \times (L - 1)$ features for Feature Type 4.

Summary of Feature Dimensions

The total number of engineered features is:

$$N_F = 2 \times (F \times S \times L) + 2 \times (F \times G \times (L - 1)).$$

Thus, each input tensor $\mathbf{x}_i \in \mathbb{R}^{F \times G \times L \times T}$ is transformed into a 1D feature vector $\mathbf{x}_i^{(\text{featured})} \in \mathbb{R}^{N_F}$ suitable for input into conventional machine learning models.

3.3 Logistic Regression

The engineered input features are flattened into a one-dimensional vector, denoted as $\mathbf{x}_i \in \mathbb{R}^{N_F}$, where N_F is the total number of features as defined in Section 3.2, which is suitable for logistic regression.

The structure of the logistic regression framework used in the project is illustrated in Figure 3.4.

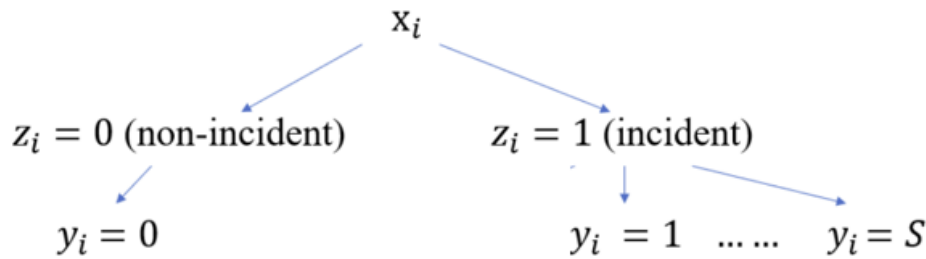


Figure 3.4: Logistic Regression Structure

Given an input feature vector \mathbf{x}_i , the model outputs two types of predictions:

- $z_i \in \{0, 1\}$: a binary indicator of whether an incident has occurred.
- $y_i \in \{0, 1, \dots, S\}$: a location label of the incident, where S is the number of road segments, and $y_i = 0$ denotes the absence of any incident.

The following dependency structure among variables is assumed:

$$\mathbf{x}_i \rightarrow z_i \rightarrow y_i,$$

which reflects a two-stage decision pipeline: first determine whether an incident occurred, then localize the incident if applicable.

Stage 1: Binary Logistic Regression for Incident Detection

In the first stage, a standard logistic regression model is used to estimate the probability of an incident:

$$P(z_i = 1 \mid \mathbf{x}_i) = \sigma(\mathbf{w}^\top \mathbf{x}_i) = \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}},$$

where $\mathbf{w} \in \mathbb{R}^{N_F}$ is the weight vector to be learned. The model is trained by maximizing the log-likelihood:

$$\mathcal{L}_z(\mathbf{w}) = \sum_{i=1}^N [z_i \log P(z_i = 1 | \mathbf{x}_i) + (1 - z_i) \log(1 - P(z_i = 1 | \mathbf{x}_i))].$$

Stage 2: Multiclass Logistic Regression for Incident Localization

If an incident is detected ($z_i = 1$), a **OvR** logistic regression scheme is applied to classify the incident location. For each segment class $j = 1, \dots, S$, a separate logistic regression model is trained:

$$P(y_i = j | z_i = 1, \mathbf{x}_i) = \frac{e^{\mathbf{v}_j^\top \mathbf{x}_i}}{\sum_{j'=1}^S e^{\mathbf{v}_{j'}^\top \mathbf{x}_i}},$$

where $\mathbf{v}_j \in \mathbb{R}^{N_F}$ is the parameter vector for class j . These parameters are also learned via **Maximum Likelihood Estimation (MLE)**.

Joint Inference via Marginalization

To compute the final predicted probability $P(y_i | \mathbf{x}_i)$, marginalize over the latent incident indicator z_i :

$$P(y_i | \mathbf{x}_i) = \sum_{z_i \in \{0,1\}} P(y_i | z_i, \mathbf{x}_i) \cdot P(z_i | \mathbf{x}_i). \quad (3.1)$$

We define:

$$P(y_i = 0 | z_i = 0, \mathbf{x}_i) = \begin{cases} 1 & \text{if } y_i = 0, \\ 0 & \text{otherwise,} \end{cases} \quad (3.2)$$

$$P(y_i = j | z_i = 1, \mathbf{x}_i) = \begin{cases} \frac{e^{\mathbf{x}_i^\top \mathbf{v}_j}}{\sum_{j'=1}^S e^{\mathbf{x}_i^\top \mathbf{v}_{j'}}} & \text{if } y_i = j, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } j = 1, \dots, S. \quad (3.3)$$

Final Prediction Rule

Given the optimized parameters \mathbf{w}^* and \mathbf{v}_j^* , the predicted incident location is:

$$\hat{y}_i = \arg \max_{y_i} P(y_i | \mathbf{x}_i), \quad (3.4)$$

with:

$$P(y_i | \mathbf{x}_i) = \begin{cases} P_{\mathbf{w}^*}(z_i = 0 | \mathbf{x}_i), & \text{if } y_i = 0, \\ P_{\mathbf{v}_j^*}(y_i = j | z_i = 1, \mathbf{x}_i) \cdot P_{\mathbf{w}^*}(z_i = 1 | \mathbf{x}_i), & \text{for } j = 1, \dots, S. \end{cases} \quad (3.5)$$

3.4 XGBoost

As before, the engineered input feature is a one-dimensional vector $\mathbf{x}_i \in \mathbb{R}^{N_F}$, where N_F is the number of input features. Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $y_i \in \{1, 2, \dots, S\}$ represents the class label over S classes, **XGBoost** is employed to solve the multi-class classification task using a **OvR** approach internally.

Unlike logistic regression, which models class probability directly using a linear function and softmax, **XGBoost** models the output for each class $j = 1, \dots, S$ as a sum of decision trees. The raw output scores are then normalized via the softmax function to form a valid probability distribution over classes, ensuring that all probabilities sum to 1 [22].

Each class j is associated with a function $F_{\Theta_j}^*(\mathbf{x}_i)$, modeled as an ensemble of K additive regression trees:

$$F_{\Theta_j}^*(\mathbf{x}_i) = \sum_{k=1}^K \alpha_{j,k} f_{j,k}(\mathbf{x}_i), \quad (3.6)$$

where $f_{j,k}(\cdot)$ is the k -th regression tree for class j , and $\alpha_{j,k}$ is its corresponding weight. The set $\Theta_j = \{\alpha_{j,k}, f_{j,k}\}_{k=1}^K$ denotes the model parameters for class j .

The final class prediction is given by:

$$\hat{y}_i = \arg \max_{j \in \{1, \dots, S\}} \text{Softmax}_j(\mathbf{z}) = \arg \max_j \frac{e^{F_{\Theta_j}^*(\mathbf{x}_i)}}{\sum_{m=1}^S e^{F_{\Theta_m}^*(\mathbf{x}_i)}}, \quad (3.7)$$

where the softmax function converts raw class scores into probabilities:

$$P(y_i = j \mid \mathbf{x}_i) = \frac{e^{F_{\Theta_j}^*(\mathbf{x}_i)}}{\sum_{m=1}^S e^{F_{\Theta_m}^*(\mathbf{x}_i)}}.$$

XGBoost optimizes the regularized loss function based on the second-order Taylor expansion of the objective, which balances model accuracy with complexity.

3.5 3D CNN

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where N is the number of samples, each input $\mathbf{x}_i \in \mathbb{R}^{F \times G \times L \times t_c}$ is a 4D tensor representing F traffic-related features across G gantries, L lanes, and t_c time steps. The corresponding label $y_i \in \{0, 1, \dots, S\}$ indicates one of $S+1$ possible classes, where 0 typically denotes a non-incident.

The goal is to perform multi-class classification to predict y_i based on spatio-temporal traffic patterns encoded in \mathbf{x}_i . To this end, a **3D CNN** is employed, leveraging its ability to capture local spatial and temporal dependencies.

Model Overview

The architecture of the 3D CNN model is depicted in Figure 3.5. The model is composed of three main stages:

1. Spatio-temporal feature extraction using 3D convolution and pooling layers.
2. Feature integration and dimensionality reduction via fully connected layers.
3. Classification through a final softmax output layer.

Network Architecture

The network processes each input tensor \mathbf{x}_i as follows:

- **3D Convolutional Layer 1:** Applies 16 filters of size $(2, 2, 2)$, capturing local patterns across gantry, lane, and temporal dimensions.

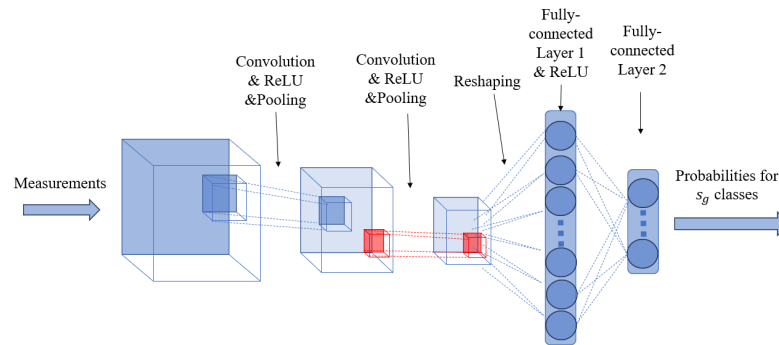


Figure 3.5: Architecture of the 3D CNN model.

- **ReLU Activation:** Introduces non-linearity to the feature maps.
- **Max Pooling Layer 1:** Reduces dimensionality along spatial and temporal axes to prevent overfitting and improve efficiency.
- **3D Convolutional Layer 2:** Applies 32 filters of size $(2, 2, 2)$, allowing the model to learn more abstract representations.
- **ReLU Activation.**
- **Max Pooling Layer 2.**
- **Fully Connected Layer 1:** The 3D feature maps are flattened and projected into a dense feature vector.
- **ReLU Activation.**
- **Fully Connected Layer 2:** Reduces the dimensionality to 64 units, forming a compact representation.
- **Output Layer:** A softmax layer produces a probability distribution over the $S + 1$ classes.

Training Procedure

The model is trained using the categorical cross-entropy loss function:

$$\mathcal{L} = - \sum_{j=0}^S \mathbf{1}\{y_i = j\} \log P(y_i = j \mid \mathbf{x}_i),$$

where $P(y_i = j \mid \mathbf{x}_i)$ is the predicted softmax probability for class j , and $\mathbf{1}\{\cdot\}$ is the indicator function.

To optimize the loss, the Adam optimizer [23] is used. Adam adaptively adjusts learning rates for each parameter based on estimates of first and second moments of the gradients, enabling faster convergence and better generalization.

Chapter 4

Experiment

4.1 Simulation Experiment Design

Microsimulation by software is designed in software SUMO, which enables microscopic simulation of urban mobility for traffic flow analysis. A highway network was built to gather traffic measurement data under various scenarios. The simulation is highly adaptable, allowing for easy adjustments to both the network layout and traffic flow parameters to mimic diverse conditions.

The simulation framework accounts for nine distinct scenarios, created by combining three different traffic demand levels with various incident configurations. For each scenario, 100 runs of simulation were performed to address random variability and ensure statistical significance. Each run lasted a total of 60 minutes. The initial 15 minutes served as a warm-up period, letting the traffic flow stabilize. Data was collected every 1 minute, only during the stable flow period after the warm-up phase, to minimize any transient effects from the simulation's start.

This structured approach ensures the simulation provides a robust dataset. This dataset is crucial for analyzing traffic flow, congestion, and the impacts of standing vehicle incidents under a wide range of conditions.

4.1.1 Highway Network Configuration

A digital version of the Stockholm highway network was constructed by the following configuration in Table 4.1. While not a precise replica of any specific real-world highway, the simulated network emulates characteristic structures found in Stockholm's highway system, particularly its dense distribution of gantries and high sampling frequency of traffic data. The simulated network,

incorporating an on-ramp and an off-ramp at its entry and exit points, is visually presented in Figure 4.1.

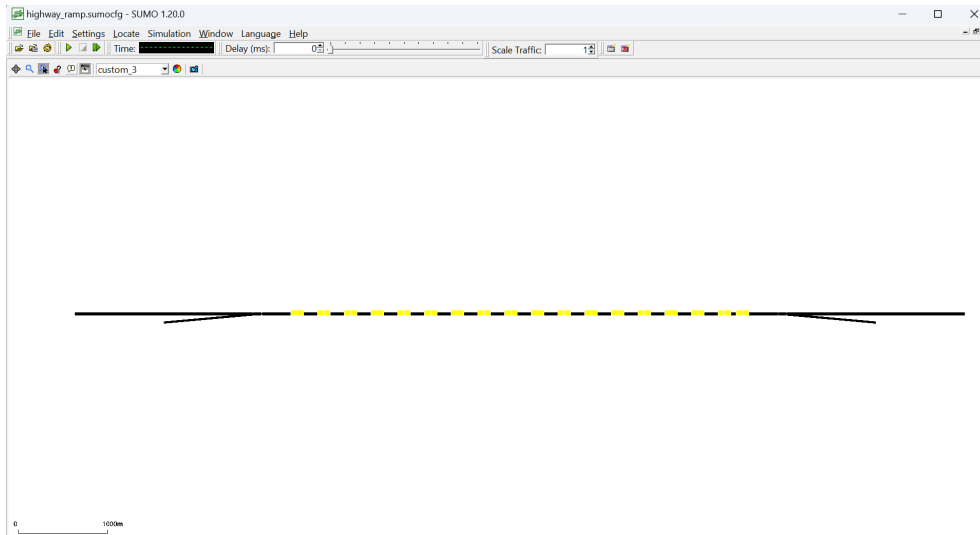


Figure 4.1: Highway Network

Name	Range
Highway overall distance	5km
Number of lanes	3
Number of gantries	12
Distance between gantries	300m
Detecting cycle time	60s
Lane allowance	[normal car, truck]
Lane max speed	130km/h

Table 4.1: Highway Configuration Parameters

4.1.2 Scenario Design

In this section, the design of various traffic scenarios used in the simulation is described. A total of nine distinct scenarios were developed by combining different traffic demand levels with varying numbers of incident occurrences.

Traffic flow within the simulation is primarily characterized by the demand parameter, expressed as the number of vehicles per lane per hour (#veh/lane/hour). This parameter defines the flow rate and reflects the overall intensity of traffic on the highway. To thoroughly examine a range of traffic

conditions, three distinct demand levels were utilized in the simulations: low, normal, and high demand. The specific demand values for each level are summarized in Table 4.2.

Traffic Demand Level	Low	Normal	High
Vehicles per Hour per Lane	800	1600	2100

Table 4.2: Traffic Demand Parameters under Different Scenarios.

To simulate real-world traffic disruptions, standing vehicle incidents were implemented in the simulation following a structured approach, with each incident designed to be independent. For each simulation run, the number of standing vehicle incidents was varied to represent different real-world conditions:

No Incident: In this configuration, the traffic flow follows the set demand level and no vehicle incident occurs throughout the simulation run.

One Incident: The traffic flow follows the set demand level with one standing vehicle incident happening.

Two Incident: The traffic flow follows the set demand level, with two independent standing vehicle incidents occurring. To prevent temporal clustering and adhere to the problem formulation's constraints, these incidents are spaced apart by a minimum time interval of 10 minutes.

Incidents are triggered by standing vehicles at random times and locations during the data collection period. A simulated vehicle remains stationary for a predefined duration before resuming its planned route. The specific parameters defining a standing vehicle incident are outlined in Table 4.3.

Parameter	Description
Stop Time (Start)	Random time between 10-15 minutes
Resume Time (End)	Predefined time after stopping
Location (Distance)	Random distance (in meters) from simulation start
Lane	Random lane in the simulation

Table 4.3: Parameters defining a standing vehicle incident.

These scenarios are designed to assess the effect of single and multiple vehicle incidents on overall traffic flow and performance, with the varying demand levels providing insight into how traffic congestion reacts to different levels of disruption.

These scenarios are designed to assess the impact of both single and multiple vehicle incidents on overall traffic flow and network performance.

The varying demand levels provide further insight into how traffic congestion reacts to different levels of disruption.

In addition to traffic demand, the simulation incorporates a distribution of vehicle types to more accurately represent real-world highway conditions. Specifically, the traffic flow consists of 70% normal passenger cars and 30% trucks. Each vehicle type is assigned a distinct speed distribution, summarized in Table 4.4, to reflect their characteristic behavior on the road.

Vehicle Type	Mean (km/h)	Std Dev (km/h)	Range (km/h)
Car	100	15	90 - 110
Truck	80	15	70 - 90

Table 4.4: Speed Distribution Parameters for Different Vehicle Types.

4.2 Data Pre-processing

In this section, a detailed analysis of the traffic measurement data is provided. The analysis encompasses raw data exploration and time series analysis to evaluate the behavior of traffic under various scenarios.

4.2.1 Data Overview

A total of 900 simulation runs were conducted, resulting in a comprehensive dataset. The simulation data are divided into two main categories: traffic measurement data from the detectors and ground truth data of incidents. The detector data captures real-time traffic flow information, such as vehicle count, speed, and density, while the incident data documents the occurrence of standing vehicle incidents, including time, location, and duration.

4.2.2 Raw Data

The raw traffic measurement data is extracted from detectors in every lane and gantries along the highway. As the data is generated from a microscopic controlled simulation, there are no instances of missing data, ensuring complete and consistent measurements across all variables. The structure of measurement data is shown in Table 4.5

Name	Type	Meaning
simulation_id	String	ID of the simulation.
time_begin	Timestamp	Time when detection begins.
time_end	Timestamp	Time when detection ends.
id_gantry	String	ID of gantry where the detector locates.
id_lane	String	ID of lane where the detector locates.
nVehContrib	Float	Number of vehicles completely pass the detector during the time interval.
occupancy	Float	Percentage (0-100%) of the time a vehicle is present at the detector during the time interval.
speed	Float	Arithmetic mean of the velocities during the time interval.
harmonicMeanSpeed	Float	Harmonic mean of the velocities during the time interval.
length	Float	Mean length of all completely collected vehicles during the time interval.
nVehEntered	Float	Number of all vehicles that touch the detector during the time interval.

Table 4.5: Traffic Measurement Data Description

The traffic measurements are collected at each cycle from detectors located on all lanes within each gantry, providing a comprehensive set of input variables for analysis.

The incident data is derived from the simulation vehicle data through SUMO's *Traci* interface, which allows access to real-time vehicle movement information during the simulation. By monitoring the movement of vehicles on the highway, the specific incidents where a vehicle stops as planned can be captured. The real-time data enables the collection of key incident-related variables such as start time, end time, lane, and proximity to gantries. The structure of the incident data is outlined in Table 4.6.

Name	Type	Meaning
<code>simulation_id</code>	String	ID of the simulation.
<code>vehicle_id</code>	String	ID of the vehicle which has an incident.
<code>incident_start_time</code>	Timestamp	Time when the incident starts.
<code>incident_end_time</code>	Timestamp	Time when the incident ends.
<code>incident_up_lane</code>	String	ID of the detector where the incident happens in the nearest downstream.

Table 4.6: Incident Data Description

4.2.3 Merging and Pre-processing Raw Data

To enable comprehensive analysis, the incident data are integrated into the detector data. For each detector, one binary column `incident_down` is introduced to indicate whether an incident is occurring within the nearest downstream road segment.

In addition to merging the data, several pre-processing steps are applied to prepare the dataset for further analysis: the reduction of features and normalization. In the first step, variables such as `harmonicMeanSpeed`, `length`, and `nVehEntered` are excluded. Both `harmonicMeanSpeed` and `nVehEntered` are removed because they offer limited additional insight compared to other features. Due to the short data collection cycle (1 minute), the difference between `harmonicMeanSpeed` and `speed`, as well as between `nVehEntered` and `nVehContrib`, becomes minimal, making these features redundant. Similarly, the feature `length`, which reflects the average vehicle length, is excluded as it is inherently tied to the predefined vehicle distribution in the simulation configuration and does not provide significant information for detecting incidents. Moreover, the features `harmonicMeanSpeed`, `nVehEntered`, and `length` rarely appear in real-world traffic measurements, further justifying their exclusion. The operation of feature reduction can help simplify the dataset while improving the ability to develop models aligned with real-world requirements.

In the second step, the features of traffic measurements `speed`, `occupancy`, and `nVehContrib` are normalized to a range of 0 to 1. The normalization is performed within each simulation run, where each simulation (i.e., all data points with the same `simulation_id`) is treated as an independent group. For each feature, it is scaled from its minimum and

maximum values within each simulation, according to the following formula:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}.$$

This step ensures that the variation of measurements within each simulation run is maintained while placing the values in a consistent range, helping further analysis and comparison across different simulation runs.

The structure of the integrated and normalized data is detailed in Table 4.7

Name	Type	Meaning
simulation_id	String	ID of the simulation.
time_begin	Timestamp	Time when detection begins.
time_end	Timestamp	Time when detection ends.
id_gantry	String	ID of gantry where the detector is located.
id_lane	String	ID of lane where the detector is located.
nVehContrib_norm	Float	Normalized number of vehicles completely pass the detector during the time interval.
occupancy_norm	Float	Normalized percentage of the time a vehicle is present at the detector during the time interval.
speed_norm	Float	Normalized arithmetic mean of the velocities during the time interval.
incident_down	Binary	1 if there is incident in the detector's nearest downstream road segment and 0 if not.

Table 4.7: Traffic Detector Data Description

Overall, the distributional differences between incident and non-incident cases are clear across all three features, especially for `nVehContrib`. This indicates strong potential for distinguishing incidents using these measurements. Additionally, the changes in `speed` and `occupancy` are more noticeable at upstream detectors, suggesting that upstream data may be more sensitive to incident impacts and valuable for classification.

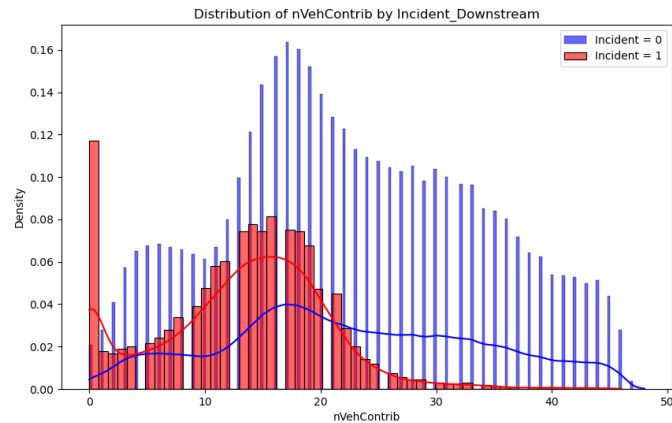
4.2.4 Distribution Analysis

For the key measurements `nVehContrib`, `speed`, and `occupancy`, analysis is conducted to identify distinct patterns indicative of an incident. The distribution of each measurement is characterized by its skewness, kurtosis, location, and the frequency of its peaks.

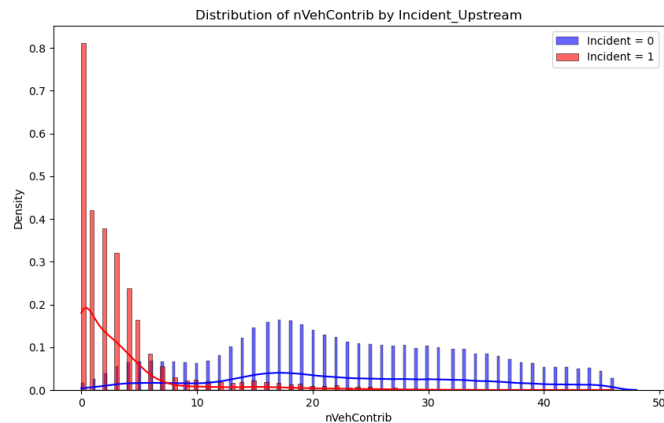
For each measurement, its distribution, including both the histogram and the kernel density estimation, is plotted. These plots differentiate between periods when an incident was present and when it was not. Furthermore, given that a standing vehicle incident exerts varying influences on upstream and downstream traffic flow, it is also crucial to investigate its impact on measurements from both upstream and downstream detectors separately.

The plots for `nVehContrib` are shown in Figure 4.2, revealing distinct distributional patterns when an incident is present versus absent, both downstream and upstream. When there is a downstream incident, the distribution without incidents exhibits a positive skew with a long right tail and a relatively sharp peak. This indicates high kurtosis and significant variability in vehicle contributions. Conversely, when an incident is present, the distribution becomes more symmetric and concentrated around its central value, reflecting lower skewness and kurtosis.

When the incident is present in the upstream, the distribution with no incident remains positively skewed and shows potential multimodality, suggesting heterogeneous traffic flow patterns. However, when an upstream incident occurs, the distribution becomes highly concentrated near zero, displaying an extremely sharp peak and pronounced right skewness. This change indicates minimal vehicle contribution and significantly restricted flow. Collectively, these observations strongly suggest that incident presence substantially alters the distributional characteristics of `nVehContrib`, affecting both its central tendency and dispersion in a location-dependent manner.



(a) Distributions of nVehContrib for when there is an incident at the downstream and not.

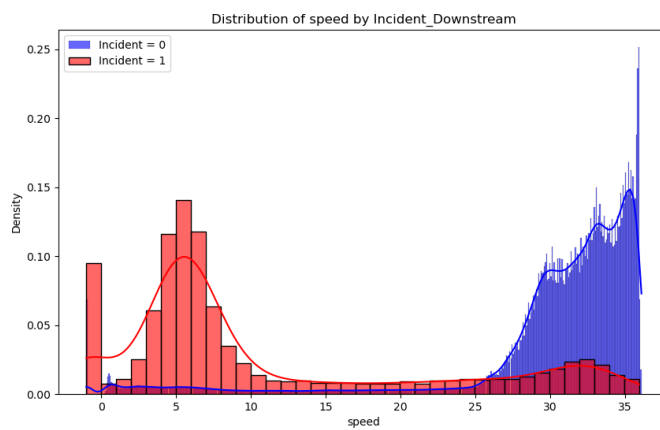


(b) Distributions of nVehContrib for when there is an incident at the upstream and not.

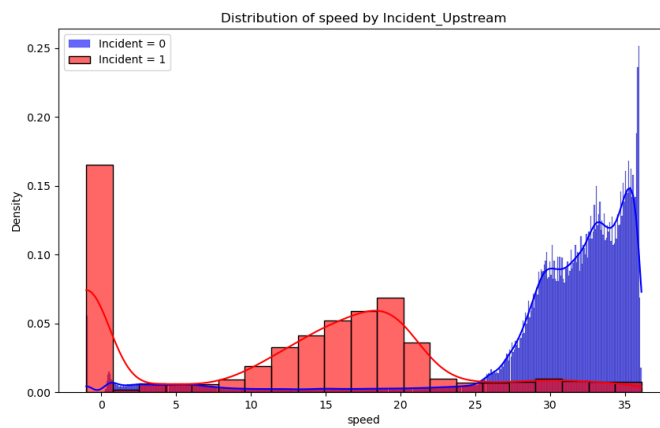
Figure 4.2: Comparison of nVehContrib distributions for incidents at upstream and downstream.

Figure 4.3 illustrates the distributions of vehicle speed under both incident and non-incident conditions, revealing significant disruptions to traffic flow. In the downstream incident scenario, the speed distribution when no incident is present shows a sharp peak around 35 units, indicating a highly concentrated pattern of vehicle speeds. Conversely, during an upstream incident, the distribution transforms into a bimodal and significantly left-skewed pattern. Most observations cluster around low speeds, clearly demonstrating substantial speed reductions caused by congestion or obstruction.

A similar pattern is observed when there is an incident upstream. Here, the non-incident case displays a similar strong concentration at high speeds. However, the presence of an upstream incident leads to a broader, left-skewed distribution with a moderate peak shifting to approximately 10–15 units. Collectively, these results suggest that incidents could lead to lower average speeds, increased variability, and the emergence of multimodal distribution patterns.



(a) Distributions of speed for when there is an incident downstream and not.

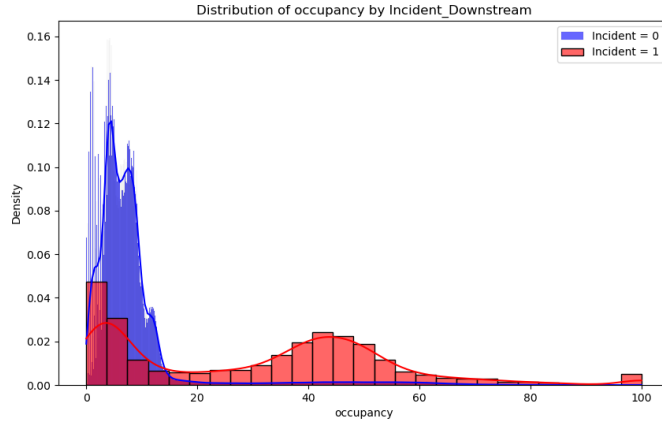


(b) Distributions of speed for when there is an incident upstream and not.

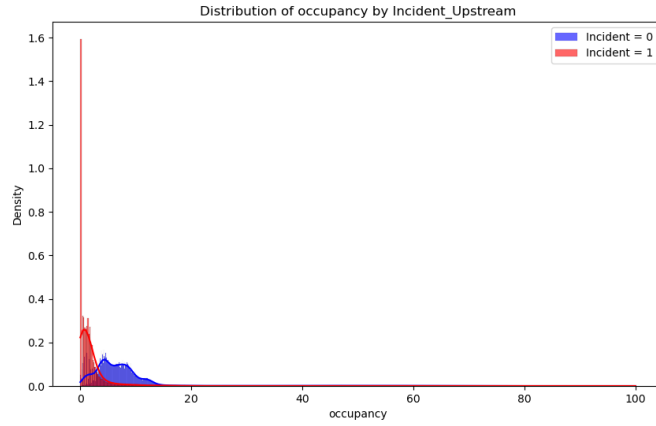
Figure 4.3: Comparison of speed distributions for incidents at upstream and downstream.

The plots for occupancy are shown in Figure 4.4. From the upstream gantry's detectors, the distribution without incidents is highly right-skewed, with most values concentrated below 20%. This indicates relatively free-flowing traffic conditions and low road utilization. In contrast, the distribution with downstream incidents displays a broader and flatter shape, with occupancy values spreading towards higher levels and peaking around 40–50%. This shift strongly suggests increased congestion and reduced traffic throughput directly influenced by the incident.

For the distribution from downstream detectors, both distributions are concentrated at low occupancy levels, and the overall shift in occupancy due to an upstream incident is less pronounced than what's observed in the downstream scenario. Those patterns suggest that incidents at the downstream have a more substantial and widespread impact on occupancy, likely due to the immediate formation of queues and the propagation of congestion, and upstream incidents appear to result in more localized effects on occupancy levels.



(a) Distributions of occupancy when there is an incident downstream and not.



(b) Distributions of occupancy when there is an incident upstream and not.

Figure 4.4: Comparison of occupancy distributions for incidents at upstream and downstream.

4.2.5 Dataset Structure and Dimensions

The dataset used for analysis is constructed from the integrated and normalized simulation data, and is denoted as $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where \mathbf{x}_i represents the i^{th} observation and y_i is the target variable of \mathbf{x}_i .

The dimension of \mathbf{x}_i is defined as $(F = 3, G = S + 1, L = 3, t_c)$, where:

- L denotes the number of lanes ($L = 3$),

- G represents the number of gantries for G road segments, where $G = S + 1$,
- $F = 3$ refers to the number of features, specifically speed, occupancy, and nVehContrib,
- t_c denotes the number of time cycles.

The target variable y_i belongs to $S + 1$ classes, i.e., $y_i \in \{0, 1, \dots, S\}$, where 0 represents no incident and k represents an incident occurring at the k^{th} road segment.

In this project, various parameter combinations are explored to determine which configuration can yield the best outcomes. Specifically, the parameters chosen for experimentation are $S = 1, 2, 3$ and $t_c = 2, 3, 4, 5$. Notice that to ensure a balance between incident samples, consecutive road segments, and time windows are allowed to overlap. In that way, one incident will be presented in $S \times t_c$ samples with each sample capturing the incident at different relative times and locations. Table 4.8 below presents the number of instances for each combination of parameters.

Road Segments	Number of Instances			
	$t_c = 2$	$t_c = 3$	$t_c = 4$	$t_c = 5$
$S = 1$	600730	398900	288990	239732
$S = 2$	298450	199150	149180	119344
$S = 3$	197604	132650	99252	79552

Table 4.8: Number of Instances in the Dataset for Various Combinations of t_c (Time Cycles) and S (Road Segments)

Road Segments	Share of Incident Instances			
	$t_c = 2$	$t_c = 3$	$t_c = 4$	$t_c = 5$
$S = 1$	1.60%	2.38%	3.14%	3.82%
$S = 2$	3.18%	4.70%	6.23%	7.79%
$S = 3$	4.69%	7.00%	9.31%	11.65%

Table 4.9: Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments).

Instances are divided into training, validation, and test sets using a 7:1:2 ratio. To prevent data leakage and ensure the independence of observations, instances from a single simulation run are not split across different sets.

Therefore, for each scenario (defined by traffic demand and the number of incidents), data from the first 70 simulation runs form the training set, the subsequent 10 runs constitute the validation set, and the final 30 runs are used for the test set.

4.2.6 Dealing Imbalanced Dataset

Given that non-incident instances represent the majority of the dataset in every simulation run, the undersampling methods (as discussed in Chapter 3) are used to balance the dataset.

The first step involved reducing the sampling rate for non-incident periods. While the original data detection period was 1 minute, the sampling rate for non-incident cases was reduced to 10 minutes.

The second step employed the Neighborhood Cleaning Rule (NCR) method [24]. This technique further decreases the number of samples in the majority class, which remains non-incident instances even after the first step, by removing those farthest from the center, determined by a nearest neighbor algorithm.

Since incident instances are balanced within their distribution, the ratio between the numbers of non-incident and incident instances becomes critical. Tables 4.10 and 4.11 illustrate the proportion of incident instances after each of these balancing steps.

Road Segments	Share of Incident Instances			
	$t_c = 2$	$t_c = 3$	$t_c = 4$	$t_c = 5$
$S = 1$	13.99%	19.60%	24.45%	28.45%
$S = 2$	24.80%	34.03%	39.99%	45.92%
$S = 3$	32.98%	42.94%	51.24%	57.01%

Table 4.10: Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments) After Step 1.

4.3 Modelling Parameters

The methods used for multi-class classification are based on logistic Regression, XGBoost, and 3D CNN, as illustrated in Chapter 3. The implementation of the three methods is conducted in Python.

Road Segments	Share of Incident Instances			
	$t_c = 2$	$t_c = 3$	$t_c = 4$	$t_c = 5$
$S = 1$	18.02%	22.13%	26.54%	30.45%
$S = 2$	28.75%	36.16%	42.98%	47.69%
$S = 3$	37.01%	45.07%	54.72%	60.90%

Table 4.11: Composition of the Dataset for Various Combinations of t_c (Time Cycles) and s (Road Segments) After Step 2.

Classifier	Library	Parameters	Description
Binary Logistic Regression	scikit	C=1 max_iters=1000 solver='lbfgs'	Regularization parameter. Maximum number of iterations. Algorithm used for optimization.
Multiclass Logistic Regression	scikit	C=1 multi_class=ovr max_iters=1000 solver='lbfgs'	Regularization parameter. Training algorithm. Maximum number of iterations. Algorithm used for optimization.
XGBoost	xgboost	max_depth=6 eta=0.1 n_estimators=100 objective='multi:softmax'	Maximum depth of a tree. Learning rate. Number of trees. Objective function.
3D CNN	pytorch	n_epoch=50 optimizer=Adam lr=0.001 cross entropy function batch size = 32 early stopping = True weight decay = 1e-5 dropout = 0.2	Number of epochs. Optimizer. Learning rate. Loss function. Mini-batch size for training. Stop training if validation loss does not improve. L2 regularization coefficient. Dropout rate for regularization.

Table 4.12: Classifier Parameters

Chapter 5

Results and Analysis

5.1 Performance Metrics

This section introduces the performance metrics used to evaluate the incident detection model. For the classification model, accuracy, precision, and AUC-ROC are key. Specifically for incident detection applications, the false alarm rate is also critical.

Since the model performs multi-class incident detection, performance metrics are calculated separately for each class to provide a comprehensive evaluation. For Class 0 (no incident detected), the accuracy, precision, and AUC-ROC are assessed. For Classes 1 through S (incidents detected), the relevant performance metrics include accuracy (also known as detection rate), precision, AUC-ROC, and false alarm rate [21].

The definitions of these metrics are provided below:

- **Accuracy:** A fundamental metric measuring the proportion of correct predictions out of the total predictions, also known as **detection rate**. It is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

- **Precision:** The proportion of correctly predicted incidents out of all instances predicted as incidents. It is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \quad (5.1)$$

- **AUC-ROC**: The Area Under the Receiver Operating Characteristic (ROC) curve. The ROC curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at different threshold levels. The AUC (Area Under the Curve) is a single value summarizing the curve:

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR}) d(\text{FPR}).$$

- The False Alarm Rate (FAR) measures the proportion of non-incidents that are incorrectly predicted as incidents. It is defined as:

$$\text{False Alarm Rate} = \frac{\text{FP}}{\text{FP} + \text{TN}}.$$

Also, the macro-average of the metric for every class can help evaluate the model in the big picture. It is calculated by the average metric for each class. [25]

5.2 Result for Different Parameters

Experiments were run based on datasets with different parameters S, t_c for all three models. The performance metrics in the macro-average way are shown in the 5.1, 5.2, and 5.3, grouped by the number of S . Quantitative analysis will be conducted to evaluate the models and find the best-performing model, along with its corresponding dataset. For comparison, in each dataset, the highest value for every metric across the three models is marked in bold to help identify the best-performing model for the given dataset. Also, in each table, the highest value for the best-performing model along with its corresponding dataset for each performance metric is highlighted in red.

For 5.1, where the three classifiers are predicting binary outputs, which is a special case in the problem, the **XGBoost** and **3D CNN** models show their superior performance, much better than logistic regression. At the same time, no model in this list presents a unique advantage.

For 5.2, within each dataset, the **3D CNN** model shows its advantage over the other two models, especially when $t_c \geq 4$. Especially when adopting the dataset with $S = 2, t_c = 5$, the **3D CNN** model turns out to be the best performer in all metrics. Also, a trend can be observed quantitatively in this table that as the t_c gets larger, almost every model's performance gets better.

Dataset	Model	Macro A	Macro AUC	Macro P	Macro FAR
$S=1, t_c=2$	LR	0.9224	0.9051	0.9002	0.0212
	XGBoost	0.9752	0.9904	0.9681	0.0086
	3D CNN	0.9782	0.9825	0.9681	0.0098
$S=1, t_c=3$	LR	0.9233	0.9097	0.9050	0.0206
	XGBoost	0.9699	0.9886	0.9638	0.0093
	3D CNN	0.9748	0.9766	0.9750	0.0050
$S=1, t_c=4$	LR	0.9210	0.9137	0.9061	0.0204
	XGBoost	0.9735	0.9873	0.9669	0.0096
	3D CNN	0.9757	0.9735	0.9709	0.0080
$S=1, t_c=5$	LR	0.9174	0.9081	0.9031	0.0217
	XGBoost	0.9695	0.9857	0.9665	0.0086
	3D CNN	0.9705	0.9757	0.9667	0.0089

Table 5.1: Performance metrics for different models and datasets ($S=1$)

Dataset	Model	Macro A	Macro AUC	Macro P	Macro FAR
$S=2, t_c=2$	LR	0.9421	0.9505	0.9042	0.0168
	XGBoost	0.9802	0.9945	0.9625	0.0080
	3D CNN	0.9833	0.9938	0.9648	0.0082
$S=2, t_c=3$	LR	0.9433	0.9590	0.9123	0.0149
	XGBoost	0.9803	0.9952	0.9652	0.0047
	3D CNN	0.9764	0.9896	0.9596	0.0085
$S=2, t_c=4$	LR	0.9439	0.9613	0.9164	0.0145
	XGBoost	0.9778	0.9940	0.9598	0.0091
	3D CNN	0.9802	0.9913	0.9624	0.0088
$S=2, t_c=5$	LR	0.9447	0.9618	0.9165	0.0154
	XGBoost	0.9749	0.9936	0.9550	0.0104
	3D CNN	0.9851	0.9957	0.9697	0.0081

Table 5.2: Performance metrics for different models and datasets ($S=2$)

For the last 5.3, the XGBoost and 3D CNN models appear to perform similarly. Besides, the highlighted red values occur in datasets with a shorter t_c , as not observed in the previous two tables.

By comparing the models' performances across all parameter settings summarized in the three tables, the 3D CNN model trained with $S = 2$ and $t_c = 5$ consistently outperforms the others, demonstrating superior accuracy and robustness in incident detection and localization. Among traditional machine learning approaches, the XGBoost model, particularly with $S = 3$ and $t_c = 3$, exhibits competitive performance, closely approaching the deep

Dataset	Model	Macro A	Macro AUC	Macro P	Macro FAR
$S=3, t_c=2$	LR	0.9473	0.9624	0.9013	0.0138
	XGBoost	0.9814	0.9931	0.9611	0.0066
	3D CNN	0.9839	0.9948	0.9697	0.0046
$S=3, t_c=3$	LR	0.9469	0.9666	0.9063	0.0127
	XGBoost	0.9829	0.9953	0.9653	0.0058
	3D CNN	0.9822	0.9949	0.9605	0.0069
$S=3, t_c=4$	LR	0.9501	0.9688	0.9119	0.0126
	XGBoost	0.9809	0.9951	0.9596	0.0072
	3D CNN	0.9784	0.9923	0.9492	0.0098
$S=3, t_c=5$	LR	0.9494	0.9701	0.9136	0.0122
	XGBoost	0.9777	0.9949	0.9545	0.0081
	3D CNN	0.9835	0.9957	0.9629	0.0073

Table 5.3: Performance metrics for different models and datasets ($S=3$)

learning model in certain configurations. In contrast, the logistic regression model lags significantly behind both 3D CNN and XGBoost, indicating limited capability in capturing the complex spatio-temporal patterns inherent in the dataset.

It is reasonable to observe that the 3D CNN model performs better when the time window t_c is larger. This likely stems from CNNs' strong capability to detect temporal changes and patterns; a larger time window provides richer information, along with a potentially more balanced training dataset. Concurrently, as the number of road segments S increases, the CNN's performance does not show a significant change. This could be because while more spatial features become available for extraction, the corresponding increase in the number of classes to be predicted effectively offsets any potential gains.

The detailed metrics for every class of 3D CNN model trained with $S = 2, t_c = 5$ are shown Figure 5.1 and Table 5.4.

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.983956	0.995026	0.988833	
1	0.989224	0.995344	0.944286	0.001726
2	0.982040	0.996652	0.964286	0.014384

Table 5.4: Performance Metrics for 3D CNN Model with ($S=2, t_c=5$)

The 5.4 of performance metrics shows that the model can detect all classes with high accuracy and precision. The high AUC-ROC values for all three

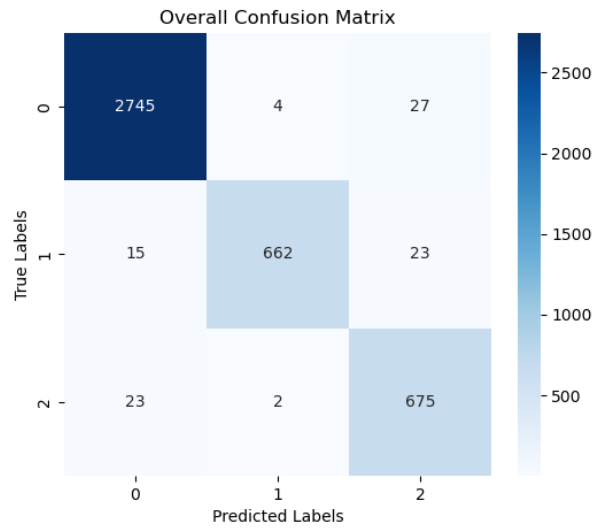


Figure 5.1: Confusion Matrix of 3D CNN Model with ($S=2, t_c=5$)

classes indicate the model's strong ability to separate incidents from non-incidents at various thresholds. Also, the low FAR values ensure the model's prediction is both accurate and actionable. The confusion matrix 5.1 shows that although the majority of test cases are non-incident, the 3D CNN model could still distinguish between incident and non-incident scenarios, with a balanced performance when classifying the specific incident location.

The detailed metrics for every class of the XGBoost model trained with $S = 3, t_c = 3$ are shown in Figure 5.2 and Table 5.5. When the number of classes increased to four, the XGBoost model still maintains a relatively balanced performance. For incident classes 1, 2, and 3, its classification results are quite similar. However, the model shows slightly weaker performance in detecting incidents occurring on relatively downstream road segments. In Section 4.2.4, the differences in feature distributions from downstream monitors are also smaller, which could explain this observation.

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.974977	0.994755	0.969600	0.041735
1	0.989242	0.993975	0.981132	0.002998
2	0.986202	0.996964	0.969178	0.004906
3	0.981291	0.995584	0.941374	0.009539

Table 5.5: Performance Metrics for XGBoost Model with ($S=3, t_c=3$)

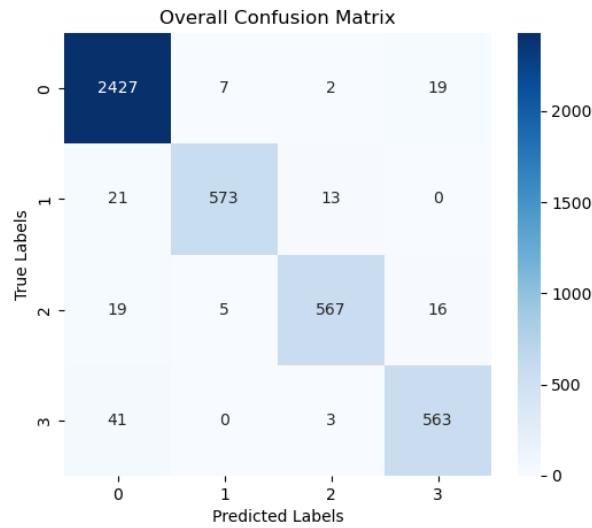


Figure 5.2: Confusion Matrix of XGBoost Model with ($S=3$, $t_c=3$)

The **SHapley Additive exPlanations (SHAP)** value summary plot could visualize feature importance for one class [26]. Features are ranked from most to least important, top to bottom. The X-axis, which is the SHAP value, shows a feature's impact. Positive SHAP values push towards predicting that specific class, while negative values push away. The color of each point indicates the feature's original value for the sample, where red means high, blue means low.

Figure 5.3, Figure 5.4, Figure 5.5, and Figure 5.6 are SHAP value summary plots of the XGBoost model, for each class. For the non-incident class 0, features 58, 25, 43, and 22 are the most influential features in the model's predictions. Specifically, both high and low values of Feature 58 can correspond to high SHAP values, signifying a notable positive impact on the model's output. Also, for features 25, 43, and 22 the low values correspond to low SHAP values, and high values correspond to the opposite, suggesting a consistent, nearly linear relationship. Crucially, for incident classes 1, 2, and 3, the majority of the most important features fall within Features 1-27. This range corresponds to Feature Type 1: Segment-Based Spatial Difference Features. This prominence suggests that features derived from the relationships between upstream and downstream road segments are more effective for incident detection than those based on individual lane-level characteristics.

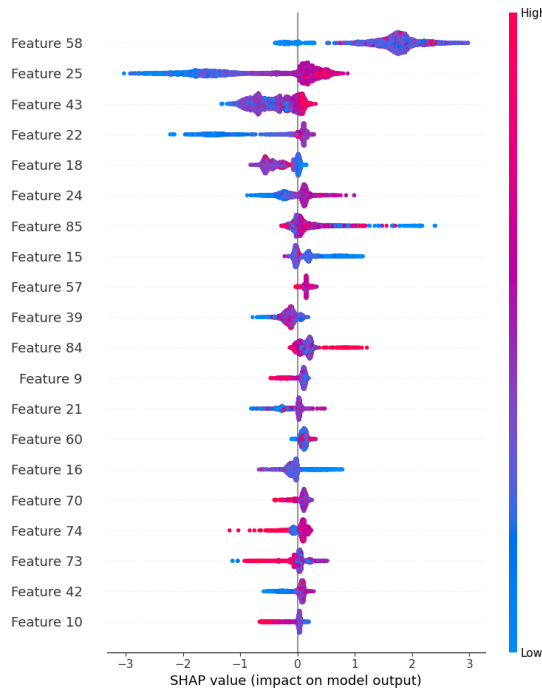


Figure 5.3: Shap Summary Plot of Class 0 for XGBoost Model with (S=3, $t_c=3$)

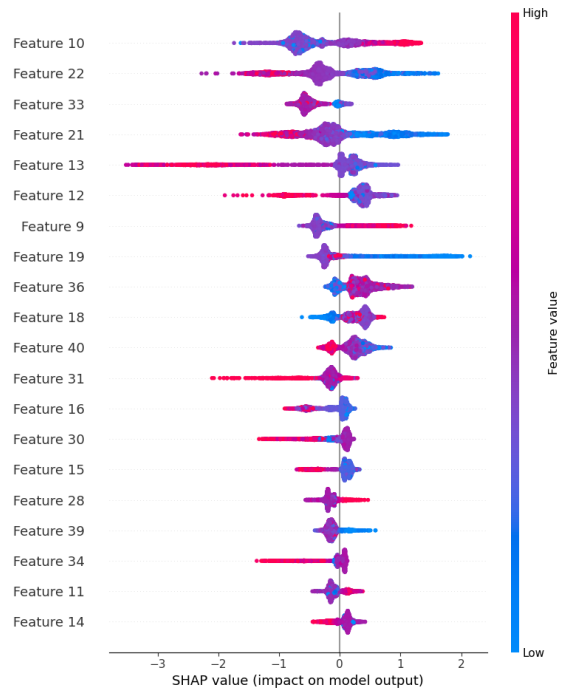


Figure 5.4: Shap Summary Plot of Class 1 for XGBoost Model with (S=3, $t_c=3$)

5.3 Robustness Test

Besides the test set from the original highway network, traffic measurement data from highways with more ramps is also used for the evaluation of the model. The network structure is shown in Figure 5.7. With several on and off-ramps on the highway, this set of test data brings a more complex pattern of traffic flow and can test the models in a more challenging way.

The 3D CNN model with $S = 2, t_c = 5$ maintains relatively strong performance metrics, as shown in Table 5.6. The detection rates for incident *Class1* and *Class2* are both above 92%. The false alarm rates for the two incidents are below 0.04%. Although the performance in false alarm rate is almost 6 times worse than that in the previous section, it is considered as a relatively low level. The overall confusion matrix and confusion matrix for each class are shown in Figure 5.8, Figure 5.9, Figure 5.10, and Figure 5.11, where we can see in 135 of non-incident data are wrongly predicted as incident. However, since the data in robustness is not pre-processed with any imbalanced techniques, it naturally has a higher ratio of the non-incident data in dataset and will leads to a higher false alarm rate.

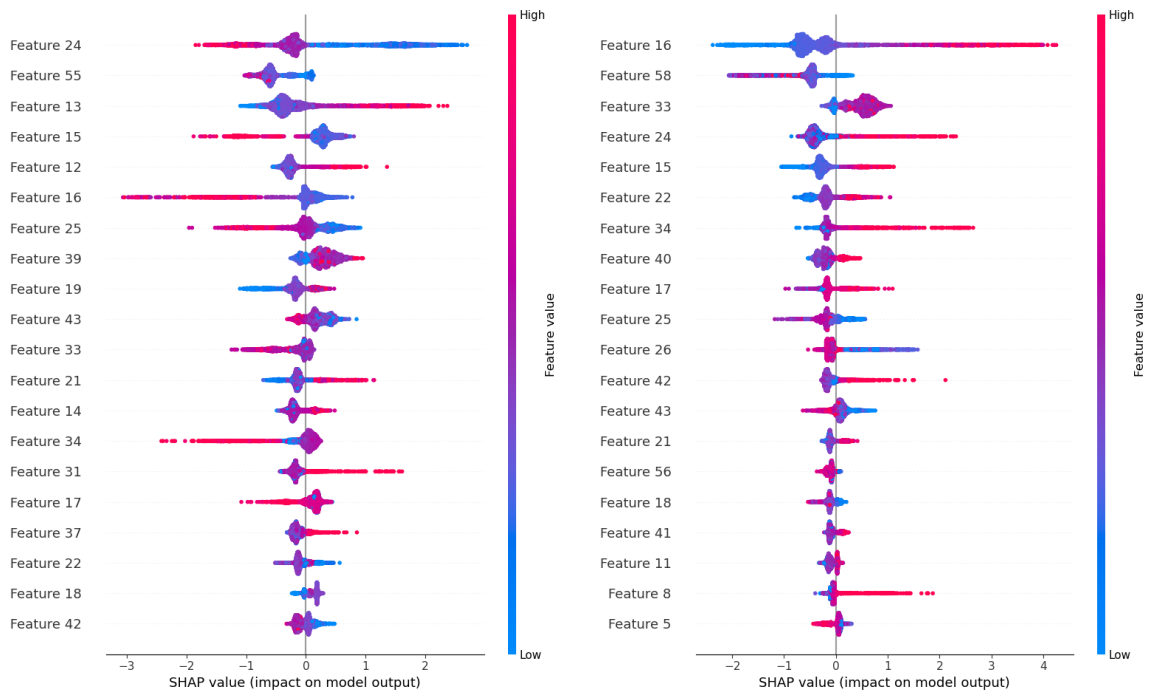


Figure 5.5: Shap Summary Plot of Class 2 for XGBoost Model with (S=3, $t_c=3$)
 Figure 5.6: Shap Summary Plot of Class 3 for XGBoost Model with (S=3, $t_c=3$)

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.899821	0.947645	0.925771	
1	0.948718	0.970322	0.816000	0.032100
2	0.927874	0.941379	0.775785	0.034892

Table 5.6: Performance Metrics for 3D CNN Model with (S=2, $t_c=5$)

The test is also conducted with $S = 3, t_c = 5$ on all the three models to analyze which method is the most robust when the number of classes increases.

The metrics of 3D CNN in Table 5.7 show an imbalanced performance, as a significant decrease regarding the incident detection on the center road segment. A possible reason is that the 3D CNN model extracts features through spatially continuous variables, which may lead to confusion between classes that have adjacent spatial relations.

However, the logistic Regression and XGBoost models do not face such challenges. The metrics in Table 5.8 and Table 5.9 show their balanced and better performance of detection across the road segments, with the XGBoost model taking the lead. A possible reason is that in Logistic

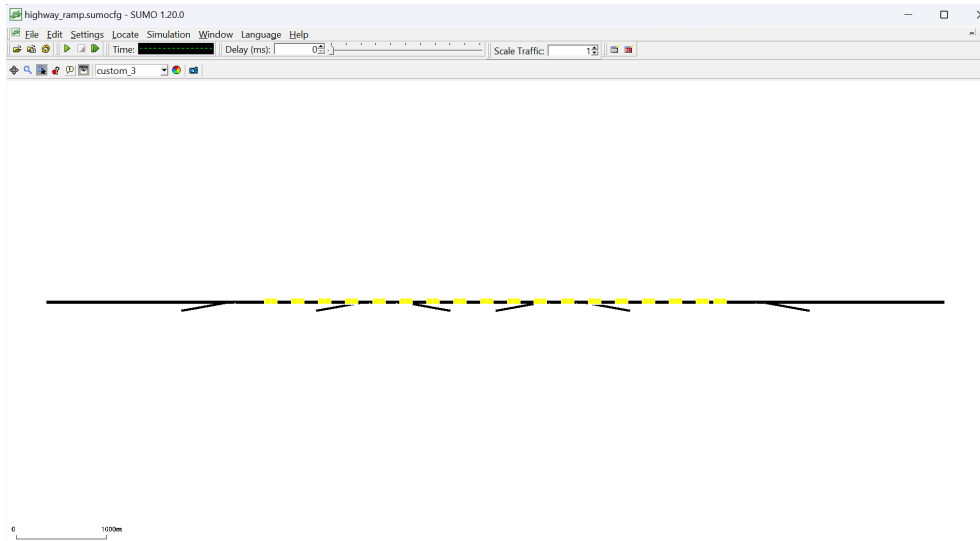


Figure 5.7: Highway with Ramps

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.860307	0.961197	0.959504	0.051174
1	0.964604	0.964788	0.888790	0.016042
2	0.916073	0.955633	0.634027	0.076745
3	0.953584	0.970414	0.819983	0.027977

Table 5.7: Performance Metrics for 3D CNN Model with ($S=3$, $t_c=5$)

Regression and XGBoost, the position of input parameters has no longer holds significant spatial meaning since they have been converted in an encoded way. Alternatively, after feature engineering, the spatial relationships of traffic measurement may be effectively embedded within the features. Additionally, the ensemble learning way and the decision trees, as weak learners in the XGBoost model, may also contribute to its superior performance in robustness while logistic Regression may struggle to handle complex correlated input features.

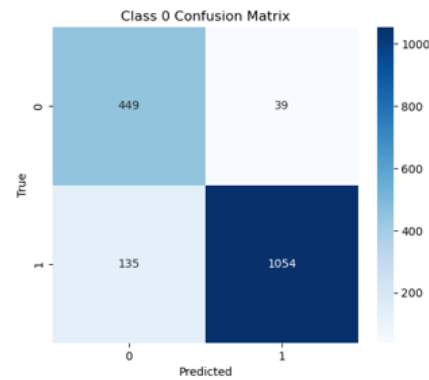
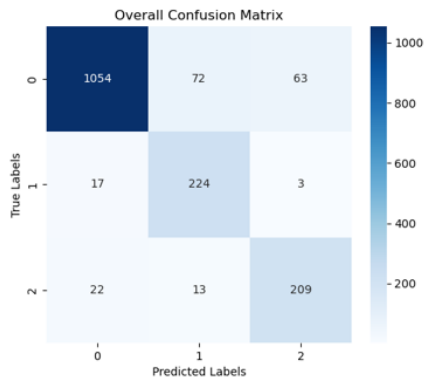


Figure 5.8: Overall Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$) Figure 5.9: Class 0 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)

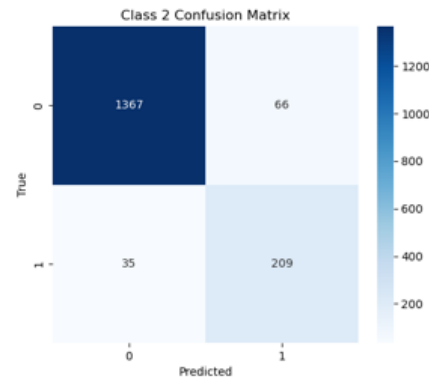
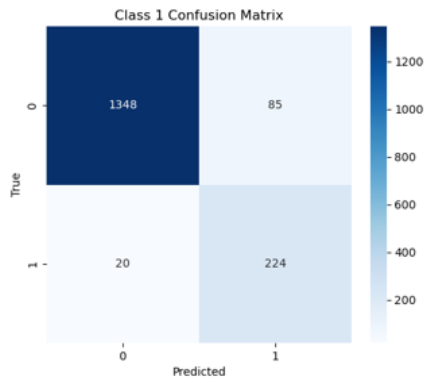


Figure 5.10: Class 1 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$) Figure 5.11: Class 2 Confusion Matrix for 3D CNN Model with (S=2, $t_c=5$)

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.847729	0.886164	0.819823	0.318233
1	0.926759	0.917209	0.937705	0.004877
2	0.942119	0.934223	0.910758	0.009369
3	0.909951	0.898440	0.665514	0.049666

Table 5.8: Performance Metrics for Logistic Regression Model with (S=3, $t_c=5$)

Class	Accuracy	AUC ROC	Precision	False Alarm Rate
0	0.896037	0.954105	0.886489	0.183725
1	0.958593	0.973147	0.919223	0.010139
2	0.957814	0.974347	0.906094	0.012064
3	0.948019	0.965925	0.839738	0.021946

Table 5.9: Performance Metrics for XGBoost Model with ($S=3, t_c=5$)

Chapter 6

Conclusions and Future work

In this simulation study, three types of data-driven methods, Logistic Regression, XGBoost, and 3D CNN are thoroughly investigated in their capability for traffic incident detection. The robust simulation framework is designed to generate data under various realistic scenarios, enabling to examine the response patterns of traffic flow to disruptions at different demand levels. All three adapted algorithms, configured for multi-class classification, consistently exhibit satisfactory performance across multiple metrics, each demonstrating unique strengths.

The in-depth analysis of key traffic measurements, including `nVehContrib`, `speed`, and `occupancy`, reveals distinct distributional patterns indicative of standing vehicle incident presence. For instance, downstream incidents could significantly alter the distributions of all three metrics, leading to decreased speeds, increased occupancy, and overall more congested patterns, often characterized by bimodal or highly skewed shapes. Upstream impacts, while present, are generally less pronounced, particularly for occupancy. These observed spatio-temporal shifts in traffic characteristics form the empirical basis for the data-driven models learned to detect.

Regarding model performance, for both Logistic Regression and XGBoost, expanding the temporal window for incident detection yielded only minimal performance improvements. In contrast, for the 3D CNN model, increasing the time range could enhance the model's performance. This is attributable to CNNs' inherent capability to effectively capture complex temporal changes and patterns, benefiting from richer time-series information. Interestingly, increasing the spatial scope of road segments S , while expanding the dataset, could not lead to substantial performance gains across the models. This suggests that the benefits of more spatial features are largely offset by the

increased complexity introduced by a greater number of classes requiring prediction.

In the robustness test, which involved a more intricate highway configuration with a higher number of road segments for incident detection, the XGBoost model proves to be exceptionally robust, outperforming others in terms of balance and overall stability. It delivers more consistent and balanced results compared to the 3D CNN and achieves superior performance over both Logistic Regression and 3D CNN in these challenging, expanded spatial scenarios. Besides, further insights from SHAP value analysis corroborated these findings, highlighting the critical role of Segment-Based Spatial Difference Features (Features 1-27) for incident classes 1, 2, and 3 in the XGBoost model. This strongly suggests that features derived from upstream-downstream traffic relationships are more effective for incident detection than isolated lane-level observations, explaining XGBoost's adaptability to larger spatial scopes.

From an economic perspective, these data-driven incident detection methods hold significant potential for cost-effective solutions. By leveraging widely available traffic state data, they could reduce the reliance on expensive specialized monitoring infrastructure. Environmentally, accurate and timely incident detection is poised to mitigate traffic congestion and enhance road safety, leading to reduced fuel consumption and lower emissions, thereby contributing to sustainability goals. Furthermore, by exclusively using simulated data, this project inherently minimized potential privacy risks associated with processing real-world traffic data.

In conclusion, the 3D CNN model shows superior performance in detecting traffic incidents due to its ability to effectively capture spatio-temporal patterns. However, XGBoost proves to be more robust, particularly when the spatial scope increases, offering a balanced and stable performance across various scenarios. The choice between these models should be determined by the specific application context—if detailed spatial-temporal analysis is needed, the 3D CNN is more appropriate, whereas XGBoost may be preferred for its adaptability and robustness in complex, larger-scale scenarios. Overall, this research explores a possible approach to improving traffic efficiency while considering sustainability and privacy aspects, offering insights that may contribute to future developments in **ITS**.

References

- [1] O. ElSahly and A. Abdelfatah, “A systematic review of traffic incident detection algorithms,” *Sustainability*, vol. 14, no. 22, 2022. doi: 10.3390/su142214859. [Online]. Available: <https://www.mdpi.com/2071-1050/14/22/14859> [Pages 1 and 3.]
- [2] J. Andersen and S. Sutcliffe, “Intelligent transport systems (its) - an overview,” *IFAC Proceedings Volumes*, vol. 33, no. 18, pp. 99–106, 2000. doi: [https://doi.org/10.1016/S1474-6670\(17\)37129-X](https://doi.org/10.1016/S1474-6670(17)37129-X) IFAC Conference on Technology Transfer in Developing Countries: Automation in Infrastructure Creation (DECOM-TT 2000), Pretoria, South Africa, 5-7 July 2000. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S147466701737129X> [Page 1.]
- [3] H. T. Muttaqin, E. Rachmawati, and E. Ferdian, “Toward end-to-end detection for traffic accidents from cctv footage,” in *Proceedings of the 2024 10th International Conference on Computing and Artificial Intelligence*, 2024, pp. 74–83. [Page 1.]
- [4] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009. doi: 10.1145/1541880.1541882. [Online]. Available: <https://doi.org/10.1145/1541880.1541882> [Page 1.]
- [5] D. A. Freedman, “On the so-called “huber sandwich estimator” and “robust standard errors”,” *The American Statistician*, vol. 60, no. 4, pp. 299–302, 2006. doi: 10.1198/000313006X152207. [Online]. Available: <https://doi.org/10.1198/000313006X152207> [Page 2.]
- [6] K. Kalair and C. Connaughton, “Anomaly detection and classification in traffic flow data from fluctuations in the flow–density relationship,” *Transportation Research Part C: Emerging Technologies*, vol. 127, p. 103178, 2021. doi: <https://doi.org/10.1016/j.trc.2021.103178>. [Online].

Available: <https://www.sciencedirect.com/science/article/pii/S0968090X21001947> [Page 2.]

- [7] D. R. Cox, “The regression analysis of binary sequences,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958. [Online]. Available: <http://www.jstor.org/stable/2983890> [Page 2.]
- [8] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’16, vol. 11. ACM, Aug. 2016. doi: 10.1145/2939672.2939785 p. 785–794. [Online]. Available: <http://dx.doi.org/10.1145/2939672.2939785> [Pages 2 and 10.]
- [9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [Page 2.]
- [10] Y.-Y. Song and L. Ying, “Decision tree methods: applications for classification and prediction,” *Shanghai archives of psychiatry*, vol. 27, no. 2, p. 130, 2015. [Page 2.]
- [11] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner, “Sumo (simulation of urban mobility)-an open-source traffic simulation,” in *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, 2002, pp. 183–187. [Pages 3 and 4.]
- [12] M. Mujahid, E. Kına, F. Rustam, M. G. Villar, E. S. Alvarado, I. De La Torre Diez, and I. Ashraf, “Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering,” *Journal of Big Data*, vol. 11, no. 1, p. 87, 2024. [Page 5.]
- [13] M. Sreelekha and Midhunchakkaravarthy, “Intelligent transportation system for sustainable and efficient urban mobility: Machine learning approach for traffic flow prediction,” in *International Conference on Multi-Strategy Learning Environment*. Springer, 2024, pp. 399–412. [Page 6.]
- [14] Trafikverket, “Its på väg,” <https://urn.kb.se/resolve?urn=urn:nbn:se:trafikverket:diva-1873>, 2011, retrieved from Trafikverket’s publication archive. [Page 6.]

- [15] M. Chowdhury, K. Dey, and A. Apon, *Data analytics for intelligent transportation systems*. Elsevier, 2024. [Page 6.]
- [16] V. Krivda, J. Petru, D. Macha, and J. Novak, “Use of microsimulation traffic models as means for ensuring public transport sustainability and accessibility,” *Sustainability*, vol. 13, no. 5, 2021. doi: 10.3390/su13052709. [Online]. Available: <https://www.mdpi.com/2071-1050/13/5/2709> [Page 7.]
- [17] L. Nguyen, S. Hanaoka, and T. Kawasaki, “Traffic conflict assessment for non-lane-based movements of motorcycles under congested conditions,” *IATSS Research*, vol. 37, 01 2013. doi: 10.1016/j.iatssr.2013.10.002 Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License. Available at: <https://doi.org/10.1016/j.iatssr.2013.10.002>. [Pages ix and 8.]
- [18] P. Alvarez 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 *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, November 2018, pp. 2575–2582. [Online]. Available: <https://elib.dlr.de/127994/> [Page 7.]
- [19] M. Colley, J. Czymmeck, M. Kücükocak, P. Jansen, and E. Rukzio, “Pedsumo: Simulacra of automated vehicle-pedestrian interaction using sumo to study large-scale effects,” in *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024, pp. 890–895. [Page 7.]
- [20] M. Ulu, E. Kilic, and Y. S. Türkan, “Prediction of traffic incident locations with a geohash-based model using machine learning algorithms,” *Applied Sciences*, vol. 14, no. 2, p. 725, 2024. [Page 8.]
- [21] O. ElSahly and A. Abdelfatah, “A systematic review of traffic incident detection algorithms,” *Sustainability*, vol. 14, no. 22, p. 14859, 2022. doi: 10.3390/su142214859. [Online]. Available: <https://doi.org/10.3390/su142214859> [Pages 8 and 40.]
- [22] J. S. Bridle, “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition,” in *Neurocomputing*, F. F. Soulié and J. Héroult, Eds. Berlin, Heidelberg:

- Springer Berlin Heidelberg, 1990. ISBN 978-3-642-76153-9 pp. 227–236. [Page 21.]
- [23] Z. Zhang, “Improved adam optimizer for deep neural networks,” in *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*. Ieee, 2018, pp. 1–2. [Page 24.]
- [24] J. Laurikkala, “Improving identification of difficult small classes by balancing class distribution,” in *Artificial Intelligence in Medicine: 8th Conference on Artificial Intelligence in Medicine in Europe, AIME 2001 Cascais, Portugal, July 1–4, 2001, Proceedings 8*. Springer, 2001, pp. 63–66. [Page 38.]
- [25] M. Grandini, E. Bagli, and G. Visani, “Metrics for multi-class classification: an overview,” 2020. [Online]. Available: <https://arxiv.org/abs/2008.05756> [Page 41.]
- [26] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *Advances in neural information processing systems*, vol. 30, 2017. [Page 45.]

TRITA-EECS-EX-2025:948
Stockholm, Sverige 2024

www.kth.se