UNIVERSITÄT BONN

Traffic Model based on Heterogeneous Data Fusion to support Intelligent Transportation Systems

Master Thesis

Philipp Zißner

Matriculation Number

2960472

This work was submitted to the

Institute of Computer Science IV

University of Bonn, Germany

Adviser(s):

Dr. Paulo Henrique Lopes Rettore
Dr. Bruno P. Santos

Examiners:

Prof. Dr. Michael Meier and Dr. Paulo Henrique Lopes Rettore

Registration date: 13-05-2022
Submission date: 11-11-2022

In collaboration with the Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE), Bonn, Germany

Rheinische
Friedrich-Wilhelms-
Universität Bonn

Prüfungsausschuss Informatik

universität **bonn** • Institut für Informatik • 53012 Bonn

Vorsitzender des Prüfungsausschusses Prof. Dr. Thomas Kesselheim

Prüfungsamt:
Judith König
Tel.: 0228/73-4418
Fax : 0228/73-4788
pa@informatik.uni-bonn.de
Postanschrift: 53115 Bonn
Endenicher Allee 19a

www.informatik.uni-bonn.de

Erklärung über das selbständige Verfassen einer Abschlussarbeit/ Declaration of Authorship

Titel der Arbeit/Title:

Traffic Model based on Heterogeneous Data Fusion to support Intelligent Transportation Systems.

Hiermit versichere ich, Zißner (Name / name), Philipp (Vorname / first name),

dass ich die Arbeit – bei einer Gruppenarbeit meinen entsprechend gekennzeichneten Anteil der Arbeit – selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

P. Zißner
Unterschrift/signature

Bonn, den 10.11.2022
date

Abstract

In this thesis, we propose a traffic model that collects and fuses real traffic data from multiple sources, characterizes it and feeds two data applications related to Intelligent Transportation Systems (ITS): Traffic estimation and incident classification. Our investigation starts with the hypothesis that fusing information from heterogeneous data sources can increase data coverage and support the development of a robust traffic model. More specifically, the fused data provide enriched information about the traffic in an urban area and support a more robust and reliable traffic estimation and incident classification. Therefore, we introduce a traffic model that covers two main parts: i) Data Fusion on Intelligent Transportation System (DataFITS), a framework to collect data from numerous data sources and fuse them in a spatiotemporal domain, and ii) two data applications to estimate traffic and classify incidents based on the fused data. The traffic estimation is implemented within two different models, using naive statistics and polynomial regression. It provides an estimated traffic feature value for an arbitrary time interval in a certain area. The incident classification is based on a k-nearest neighbors (k-NN) algorithm that uses Dynamic Time Warping (DTW) and the Wasserstein metric for distance measuring, capable of comparing time series and distributions. As a result, DataFITS achieves a significant enrichment of information, increasing the road coverage by 137% and fusing spatiotemporal data from multiple sources on 40% of all roads. Furthermore, we create an extensive data characterization, conducting a variety of analysis methods, which is possible due to the fused information. The traffic estimation achieves average results using a naive statistical approach, with an R^2 score of -0.11 and error metrics above 1. Using the polynomial regression model, the results are significantly better, reaching a high R^2 score of up to 0.91 and low error metrics of 0.05, providing a robust and accurate traffic estimation application. Finally, the robustness of our incident classification provides a good result. The model achieves an accuracy of 90% in a binary classification, detecting the occurrence of an incident. For the more complex task of classifying a certain type of incident, it scores an accuracy of more than 80%. In conclusion, the thesis shows that using heterogeneous data fusion improves traffic-related information in an urban area. Furthermore, it is useful to provide high-quality traffic estimation and incident classification applications, if appropriate data engineering is applied filtering and correcting the data for a given application.

Acknowledgments

My personal thanks go to my examiners and advisors, namely Prof. Dr. Michael Meier, Dr. Paulo Henrique Lopes Rettore and Dr. Bruno P. Santos who made this thesis possible in the first place. Their long-term support, motivation and patience, especially from Dr. Rettore, accompanied me during the process of the entire thesis.

Moreover, I am very grateful to my respected colleagues at Fraunhofer FKIE for their cooperation. I also wish to thank my family and friends that supported me in each and every respect during the process of the thesis.

Contents

1 Introduction **1**

1.1 Problem 2

1.2 Hypothesis 3

1.3 Proposed Solutions 3

1.3.1 Methodology 4

1.4 Thesis structure 4

2 Background **7**

2.1 Civilian Context 7

2.2 Data Fusion 8

2.3 Military Context / Urban Warfare 8

2.4 Understanding / Modeling of Traffic 9

3 Related Work **11**

3.1 Data Collection and Fusion 11

3.2 Traffic Estimation 12

3.3 Incident Classification 14

4 Problem Statement **17**

4.1 Data Fusion 17

4.2 Traffic Estimation 19

4.3 Incident Classification 19

5 Design **21**

5.1 Data Fusion Framework 21

5.1.1 Data Acquisition 21

5.1.2 Data Preparation 22

5.1.3 Data Processing 24
5.1.3.1 Spatial Fusion 24
5.1.3.2 Temporal Fusion 28
5.1.4 Data Usage 29
5.2 Traffic Estimation 29
5.2.1 Preprocessing 30
5.2.2 Model Creation 32
5.2.2.1 Naive Statistical Model 32
5.2.2.2 Machine Learning (ML) Model 33
5.3 Incident Classification 35
5.3.1 Preprocessing 36
5.3.1.1 Input Strategy 37
5.3.1.2 Incident Validation 38
5.3.1.3 Further Preprocessing 40
5.3.2 Model Creation 40

6 Evaluation **45**
6.1 DataFITS 45
6.1.1 Experimental Setup and Performance 45
6.1.2 The Data 46
6.1.3 Data Fusion 49
6.1.4 Traffic Data Characterization 50
6.1.4.1 Traffic Overview 50
6.1.4.2 Spatiotemporal Visualization 53
6.1.5 Incident Data Characterization 54
6.1.5.1 Incident Overview 55
6.1.5.2 Spatiotemporal Visualization 57
6.2 Traffic Estimation 59
6.2.1 Naive Statistical Model 60
6.2.2 Polynomial Regression Model 62
6.2.3 Comparison 65
6.3 Incident Classification 67
6.3.1 Input Strategy 68
6.3.2 Incident Validation 69
6.3.3 Model Performance 70

7 Conclusion **77**

7.1 Publications 78

7.2 Future Work 78

Bibliography **81**

List of Figures **87**

List of Tables **89**

1

Introduction

Mobility plays a key role in modern human life. Our daily routines, travels and goods transportation are dramatically influenced by the transportation ecosystem and the way that we use it. For instance, the number of registered passenger cars in Germany reached an all-time high of 48.5 Million in 2022, an increase of more than 50% over the last 30 years [46]. Similarly, the amount of passengers in public transportation services, like buses and trains, has increased over the last years with a total of 12.6 million users in 2019 and 8.7 million in 2020 [16]. Besides the decline in the year 2020, as a result of the COVID-19 pandemic, the expressive growth requires efficient and environmental-friendly transportation systems to support the new demands.

A major drawback, related to the global transportation system, is given through a seemingly unavoidable emergence of traffic congestion. This is especially problematic in urban areas, due to the urbanization that leads to a higher vehicle population. According to the UN, the amount of people resided in urban areas is going to increase from 55% of the world's population (2018) and is expected to reach 68% by 2050 [49]. This describes an alarming state, leading to an increase of environmental damage due to higher emissions, more fuel consumption, and furthermore, causes serious time delays in the transportation system [51]. Those delays are especially problematic, because our modern society highly relies on a timely transportation of people and goods, so minor slow downs can have a big impact on the supply chain.

Moreover, traffic congestion has a negative impact on the aspect of road safety. This is a consequence of the higher speed variance among the vehicles and irregular driving behavior in the presence of congestion, which leads to an increase of road incidents [52]. Incident statistics show an increase of traffic accidents, with a total of 1.35 million deaths and 50 million incidents per year, making traffic accidents the eight leading cause of death globally [59]. Considering these negative effects of traffic congestion clearly shows a reduction of people's quality of life and motivates to improve the mobility and safety of the transportation system.

Academia and industry have already started efforts to develop the ITS, the new age for transportation systems. Some of the intents of the ITS are to be environmental-friendly, cost-efficient and use different techniques in the context of communication, data analysis, and information technology [40]. ITS is especially useful within smart cities, which offer a well distributed communication infrastructure. This enables real-time monitoring and the access to historical data from urban activities, opening opportunities for the design of intelligent systems [8]. As a result, this thesis is going to study certain ITS applications that can be used to provide solutions to the problem of traffic congestion. In general, the usage of ITS helps to reduce risks, high accident rates, traffic congestion, emissions and more, while increasing safety, reliability and the general flow of traffic [40].

Most applications in the context of ITS require a significant amount of high quality data, related to the transportation context. Together with a smart city, offering access to historical data, this thesis uses heterogeneous data fusion to combine data from many diverse data sources. We overcome the issue of data availability by using this concept and enrich the information which can be used in ITS applications. For this thesis, we are considering a spatiotemporal fusion of data, combining information based on the same location and point in time. In the next section, we are introducing a set of challenges related to the mentioned techniques and explain the main problems discussed by the thesis.

1.1 Problem

Given the introduced topic of heterogeneous data fusion, we define the first major research question of this study:

How to fuse free, heterogeneous information from different data sources in a spatiotemporal way, to provide a robust dataset of traffic-related information?

To answer this question, we investigate the availability of transportation data in Germany, that is openly accessible or covered by a free subscription of a commercial data provider. Moreover, we are discussing heterogeneous data fusion, providing a technique to combine different types of data in a spatiotemporal way. The concept can be used to partially overcome existing data problems, like spatiotemporal gaps, sensor errors or low accuracy, for instance. However, using the data fusion concept adds new challenges, like the conversion of data types or comparing data in a spatial and temporal aspect, increasing the complexity to create a robust dataset of information. Furthermore, we need to overcome data issues, such as different data structures, errors in the acquisition and acquired data (e.g., wrong measurements, missing values), outliers, conflict, incompleteness and vagueness.

Designing a solution for the first research question provides access to an enriched set of heterogeneous transportation data that can be used within applications to support the ITS. This leads to the second major question discussed in this thesis:

How to provide robust and accurate data applications for traffic estimation and incident classification using the heterogeneous fused data

The second research question is related to the design of ITS applications, based on the enriched dataset. First, we discuss the problems related to the estimation of

future traffic values, based on a certain set of input parameters. This task has a high complexity, due to the general heterogeneity of the traffic state, influenced by aspects like the street type, weather or time of the day. Furthermore, the status of traffic depends on multiple data features, such as traffic level and speed. Subsequently, providing a model to create a robust incident classification introduces further problems to this research, including the hard distinctness between several types of incidents. Precisely, the classification of an accident, congestion or normal traffic situation, based on a traffic pattern, has a very high complexity. Moreover, this discipline requires major data preprocessing tasks, including combining several types of information and removing biased data from the input.

1.2 Hypothesis

Based on the previously defined research questions, our hypothesis is that a traffic model, using heterogeneous fused data, is more likely to be robust and reliable in the estimation of traffic and classification of incidents. To prove the given hypothesis, we propose a complete traffic model, containing: i) DataFITS, a data fusion framework to collect and combine heterogeneous data from various sources to enrich the quality of the available information and ii) two ITS applications that utilize the enriched information from the heterogeneous fused dataset. The first data application is given by a traffic estimation model, to precisely estimate the status of traffic, depending on a set of different parameters. Subsequently, the second application is an incident classification model to detect and categorize different types of incidents with a high accuracy. Our given hypothesis is verified, by providing a detailed evaluation on all proposed parts of the traffic model. First, we evaluate the DataFITS by analyzing and characterizing the data, to indicate the benefits of heterogeneous data fusion. Subsequently, the performance regarding both data applications is measured, conducting a set of tests that use different setups. Therefore, we can show the benefits of using heterogeneous fused data within those data applications.

1.3 Proposed Solutions

Following the given problems and stated hypothesis, the proposed solution of this thesis is split into multiple parts compiling a traffic model:

First, we introduce the DataFITS, a data fusion framework to collect information from an arbitrary number of data sources. It can be used to create a suitable collection of data within the context of transportation. We implemented the acquisition from seven different data sources (commercial and open access), which cover four unique categories of transportation data (traffic, incident, vehicular and weather). The provided spatiotemporal data fusion increases the data quantity (higher information coverage) and the data quality (combining spatiotemporally overlapping information).

Furthermore, our solution contains two ITS applications, using our collection of heterogeneous fused data. Therefore, we are introducing a traffic estimation application that benefits through the usage of fused information. The model is capable

of estimating future traffic, based on a set of input parameters and historical data. Furthermore, the second application is given by an incident classification model to detect and classify incidents based on traffic patterns. The models are developed using naive statistics and ML techniques, trained on a large amount of heterogeneous fused data and further utilize data dependencies, including various correlating features.

1.3.1 Methodology

The proposed solutions, discussed in this thesis, provide an answer to the research question on how to use available transportation data to support and improve ITS applications. We define multiple steps that need to be completed to achieve this goal:

- Data Collection: Acquire multiple types of data from different transportation-related data sources, to ensure a reasonable amount of information. The underlying methodology should offer possibilities for expandability of data types and sources.
- Data Fusion: Fuse the heterogeneous data in a spatiotemporal way. To combine data in a spatial way, we are using a map matching approach and the temporal fusion is realized by grouping the data using temporal aspects.
- Data Processing: Process the data to be applicable as an input for the traffic applications. This task includes data analysis, detection and removal of biased data and further pre-processing.
- Data Applications: Use naive statistics and ML to create applications in the context of ITS, capable of i) estimating future traffic levels and ii) classifying multiple types of incidents. The respective models require to achieve a high accuracy, to create a robust macroscopic traffic estimation and incident classification. Furthermore, the quality of the proposed models is an essential measure for the benefits of data fusion in the context of traffic modeling.

1.4 Thesis structure

The structure of the thesis text is as follows. Chapter 2 discusses the problems of traffic congestion, explains the civilian and military context of this research and finally introduces the concepts of data fusion and applications in the context of ITS. In Chapter 3, we provide a detailed review about recent solutions focusing on data fusion, traffic estimation and the classification of incidents, comparing them to our proposed solution. Chapter 4 defines the concrete problems that are faced when developing methods for each of the discussed topics, to provide a solution for our final traffic model. The design of the proposed solution is explained in Chapter 5, introducing the framework DataFITS to collect and fuse heterogeneous data and explaining the methodology of both data applications to estimate traffic and classify incidents. The evaluation conducted in Chapter 6 shows the benefits of data fusion

and effectiveness of our proposed data applications, providing results that confirm our hypothesis. Finally the observations are summarized and the potential future work is discussed in Chapter 7.

2

Background

The following chapter introduces the background of this study, providing an overview of the problems related to traffic congestion and furthermore, explains the concepts of ITS and smart cities. Subsequently, we introduce spatiotemporal data fusion, combining heterogeneous types of data from multiple sources that describe the same real-world object or situation. Moreover, we introduce the idea of combining ITS and data fusion to support military tasks, like emergency rescue operations or information superiority in urban warfare. Finally, we motivate the use case of an extensive data analysis and the creation of data applications, that can be used to improve the current state of transportation.

2.1 Civilian Context

As stated in the preceding introduction chapter, transportation is a highly relevant aspect of our daily life, with traffic congestion introducing problems that require smart solutions. Our research is covering ideas to utilize various modern concepts and technologies, trying to find solutions for better traffic management. Supporting the existing transportation systems, by using available data and advanced communication technologies, is a main goal of ITS. An advantage of this concept is the low cost requirement to improve the available infrastructure by a smart use of the information through applications and services.

The initial idea of ITS was originally developed in the 1980s, recognizing the combination of modern computation and communication technologies together with transportation [56]. Following this, the development of various ITS applications continued, now taking a vital part in the global transportation and being of great interest, due to an extensive research and development of new solutions [40].

In contrast to other approaches that increase the efficiency of traffic, e.g., expanding of existing roads, ITS can be used to provide applications to the transportation system, which require less effort and cost to be realized. The main goal of these

applications is to increase traffic efficiency and reduce congestion, by utilizing a combination of various technologies. An exemplary ITS application is to improve the traffic flow using variable signs (e.g., showing the speed limit), that change their value according to the current traffic condition and other factors. This technique has been used for a long time and is widely distributed in Germany, covering 1,500 km of the 13,000 km total highway road [19].

In general, the data used by ITS applications is collected from sensors, like cameras or inductive loops, which are distributed along the road infrastructure. Additionally, those applications can be supported through the data that is available within a smart city, offering a well distributed communication infrastructure. This includes intelligent systems, real-time monitoring and historical data from urban activities. A smart city is defined by the smart and efficient use of digital technologies to achieve goals, such as better resource management and less emissions. Some examples are the improvement of urban transport networks, smarter water supply and more efficiency when providing energy to buildings [12].

2.2 Data Fusion

We can improve the amount of data which is used for the development of ITS applications, by combining various data types from different sources. This method is called heterogeneous data fusion and defined as the process of fusing multiple data entries, representing the same real-world object or situation, into one single, consistent representation [5]. Regarding the discussed topic of transportation, this is translated into the combination of information, acquired by multiple sources, to provide an improved description about a traffic situation.

In general, data fusion is applied in several fields of civilian and military applications, with an increasing interest of using the technique in the context of ITS and smart cities [11, 28]. Data fusion describes a set of techniques, varying in complexity, that can be used to combine the information acquired through different types of sensors. In context of ITS, the use of statistical methods, like arithmetic mean, a probabilistic bayesian approach or artificial intelligence methods, provide suitable data fusion solutions. Some examples for ITS applications, that emerged throughout the last years, include Advanced Traveler Information Systems (ATIS), automatic incident detection, traffic forecasting and traffic monitoring [11]. Furthermore, the amount of smart city applications utilizing data fusion increases. More precisely, most domains of smart city applications, like smart human mobility, smart living or smart urban area management, commonly apply data fusion techniques to ensure a high data availability [28].

2.3 Military Context / Urban Warfare

Alongside the civilian context of this research area, there is a further use-case for the discussed techniques in the military area. A possible scenario, related to urban areas, are emergency rescue operations, which can be supported through the usage of civilian transportation data. When considering a threat (e.g., terrorism, incident

or disaster) in an urban area, involving civilians that are possibly injured, the fusion of information, acquired through various sensors within the city, can support military operations. The combination of data from different sources provides a more accurate view on the operation area, improving the decision making and availability of information. Furthermore, the transportation system has a significant impact within urban area dynamics, such as the transportation of people and goods. As a consequence, military operations, such as rescue, counter-terrorism and disasters in crowd and traffic areas, can suffer delays trying to reach the emergency area.

Moreover, a topic to be discussed in the scope of this thesis is urban warfare. It describes the type of combat carried out in urban areas. As a result of the ongoing urbanization, discussed in the introduction, the importance of this military scenario is further increasing. Especially in 2022, the relevance of this is omnipresent, with the Russian invasion of Ukraine, including combats being held in urban districts like Kyiv [62] or Mariupol [18]. The military aspect of research is motivated to improve the precision and success of missions by using real-time and context-aware information, ultimately leading to information superiority. This describes the operational advantage in a military scenario, by collecting and processing a substantial amount of information. Military operations in urban areas require timely and context-aware information to increase their precision and success [39, 50]. Therefore, the collection of information from different data sources can support the Command and Control (C2) concept, widely used in the military field, to describe and understand the urban scenario and make accurate and context-aware decisions.

2.4 Understanding / Modeling of Traffic

Analyzing the data regarding different aspects converts it into understandable information that can be used to support traffic planning and ITS applications. Furthermore, we can model the state of a transportation network, by representing the real-world information through data features, such as traffic or speed. This also allows the development of applications, like traffic estimation or incident classification. In general, there are two major types of traffic models: i) Microscopic Traffic Model: Considers traffic features at a high level of detail, e.g., model based on characteristics of different vehicle movements (cars, buses, motorcycles, etc.) and ii) Macroscopic Traffic Model: Considers traffic characteristics, like speed flow and density, at a much lower level of detail, e.g., model for a big observed area [3]. Throughout this thesis we focus on the discussion of a macroscopic traffic model.

The complexity of those models depends on the amount of different features that are considered as an input. Many models use (spatiotemporal) correlation, which describes dependencies within the data. An example for this type of correlation is the relation among different types of traffic related features (e.g., a reverse correlation between speed and traffic). Furthermore, traffic data contains spatial and temporal dependencies, also referred to as spatiotemporal correlation throughout this thesis. An example for this is given through traffic at one area that influences the values from other connected areas. This dependency can be further observed under the consideration of a temporal aspect, with roads showing this correlation with a certain delay.

Machine Learning (ML) is a technique that is commonly applied in the context of traffic modeling, to improve the accuracy of a traffic estimation, for instance. There are various different approaches on how to use ML in this research context, which are going to be partially covered by the following related work chapter. In general, the ML algorithms can be used to learn from several traffic-related input features and utilize aspects like spatiotemporal correlation. This thesis is going to work with the two main concepts of supervised ML, Regression and Classification. Regression algorithms learn a mapping function, based on the input data to create continuous output variables. This is applicable for a traffic estimation approach, generating a future traffic value based on various input parameters. In contrast to this, classification attempts to map the input variables to a categorical output variable, such as incidents, for instance. In more detail, an observed traffic pattern, with features like level of traffic and speed, is going to be used as an input and classified to a certain type of incident [15].

3

Related Work

This chapter introduces a variety of literature related to the three main topics of interest. First, we cover some approaches within the research topic of data fusion to increase the availability and quality of information. Subsequently, we are discussing literature covering the idea of traffic estimation, utilizing various methods like data fusion, correlation, or ML. The last section discusses literature related to predicting and classifying different types of incidents. We conclude this chapter by providing a table summarizing the contributions of the discussed solutions compared to the thesis' contribution.

3.1 Data Collection and Fusion

Data fusion is a widely used concept that combines data from different sources to enrich the available information in a given situation [63]. Many ITS applications and solutions use data fusion to process information from multiple data sources [11]. However, besides the information advantage, the fusion of heterogeneous data requires additional data pre-processing, according to various data types and features which have to be combined [21, 25].

Increasing the amount of data in a given situation is the primary goal of data fusion. In [51], the authors provide a platform to gather, process, and export heterogeneous data from smart city sensors. Furthermore, they provide a variety of statistics and visualizations, but they do not consider data fusion to improve the quality of the provided information within the data platform. We share a similar motivation, but instead of solely focusing on the design of a data platform, we further provide a fusion approach to our collected data through DataFITS and propose two data applications in the context of ITS.

Emergency rescue operations are another use case that utilizes data fusion to improve the information provided in an emergency scenario. An exemplary approach to support such operations is given by Foresti et al. [13], introducing a system for

emergency management. Their presented approach uses data fusion to combine information from smart city sensors and reports from involved citizens via social media to increase the available information. Following the same research interest, but mainly addressing the path planning problem in an emergency rescue situation, the authors of [24] provide a methodology to dispatch and route emergency vehicles by fusing available heterogeneous information, such as casualties and road/traffic conditions. A further application of emergency rescue planning is presented in [57], proposing a co-evolutionary algorithm that solves the path planning problem. The underlying methodology considers the temporal and spatial characteristics of traffic flow in the urban network.

Furthermore, the data fusion concept is commonly applied to solve the problem of incomplete data and information overload in the military context. More precisely, it supports military information systems to reach information superiority, a critical aspect of urban warfare. It describes the operational advantage as a result of collecting and processing information [39]. A part of the existing literature provides applications using multi-sensor data fusion in maritime surveillance [14, 20] and autonomous vehicle navigation [36]. As a result, the operational advantage in a military scenario can be strongly supported with supplementary information from the civilian area. The collection and processing of available information support the rescue of people or defense against potential hostile adversaries. A practical example of the first scenario is the combination of social media data and information acquired by smart city sensors. The social media data, for instance, shared by local users, can be used to locate groups of people, injured or hiding, which need to be rescued. This aspect is further supported by stationary sensors on buildings and surveillance cameras, capable of performing human tracking to identify the location of a person.

3.2 Traffic Estimation

Traffic estimation is an application that is widely covered by the literature, discussing a variety of different solutions. Besides traditional approaches utilizing a single dataset of traffic information, there is an ongoing development, including the addition of data fusion, spatiotemporal correlation, and ML.

The most general form of traffic estimation can be utilized using one traffic data feature from a single dataset to estimate a future value. However, due to the ongoing trend of big data and concepts like OpenData (OD), there is an opportunity to combine information from multiple data sources to ensure a high-quality estimation of traffic states [23]. The methodology of using heterogeneous data fusion to support a traffic estimation approach mainly describes the combination of static data (e.g., from cameras or loop detectors) and data provided by probe vehicles (e.g., cellular data or GPS). An example of this is given by Anand et al. [2], using traffic flow values obtained from video, together with travel time, calculated via GPS data, to improve the accuracy of their proposed traffic estimation approach. The data fusion is implemented using a Kalman filter to provide a solution to estimate traffic density values based on the fused information. They achieve better results than using non-fused data from video sensors. Similarly, the authors of [27] provide a traffic estimation solution using data fusion to combine traffic information acquired

from stationary underground loop detectors and probe vehicles (taxis equipped with GPS sensors). Compared to other existing approaches, both of these approaches show an improvement in traffic estimation accuracy due to the heterogeneous traffic data fusion.

A further approach to support the estimation of traffic values is the consideration of spatiotemporal correlation within the data. This can improve the accuracy of a traffic estimation model by identifying specific hidden structures within the data and the road network. The authors of [17] provide a method to capture the correlation between road segments using a modified version of the *Pearson correlation coefficient*. They identify the most influential roads in a correlated road network by calculating a *Cross Correlation* between single road segments that also considers temporal aspects. Furthermore, the dependencies within the road network can be used together with probe data to fill gaps in the data and estimate traffic data on locations without any available information [29, 67]. An example for this is given in [29], creating a model that extracts the correlation within the road network and uses data from 50 probe vehicles to estimate the traffic of a complete, city-scaled network. Another solution to this problem is given in [34], using data fusion techniques to overcome the problems of error-prone and sparse data.

The usage of ML models to estimate future traffic states is widely discussed by recent literature [1, 10, 35, 47, 55, 65, 66]. The authors of [44] provide a survey introducing many different traffic estimation approaches, categorized by the estimation approach (model-driven, data-driven, or streaming-data-driven), traffic flow model (the underlying physics-based mathematical model representing the traffic dynamics) and the used input data (characterized by collection method, data representation, and temporal condition). An explicit example for ML in the context of traffic estimation can be found in [1], providing an auto-regressive model to estimate future traffic flow values up to 30 minutes ahead in time. The model can adapt to unpredictable events like accidents or road closures. The proposed evaluation uses data from a traffic simulator that generates traffic flow information based on historical data. Abadi et al. provide results with varying estimation errors, dependent on the time, from 2% for estimations up to 5 minutes ahead and 12% for 30 minutes estimation time.

Additionally, some of the ML approaches utilize the spatiotemporal correlation from the data, to improve the estimation quality further, outperforming the results of other existing approaches. Such an example is given in [65], providing a neural network-based estimation approach using a Graph Convolutional Network (GCN) and a Gated Recurrent Unit (GRU). The GCN can learn complex topological structures and can be used to capture the spatial dependencies from the network. Moreover, the GRU can detect dynamic changes in the traffic data and capture the traffic data's temporal dependencies. Compared to other existing traffic estimation models, which do not consider spatiotemporal aspects of the traffic data, the authors of this approach prove that the usage of correlation significantly benefits the accuracy of the traffic estimation. Some other similar neural network-based approaches, which also use spatiotemporal correlation [10, 47], show a similar improvement in the estimation accuracy. As a result of the broad literature coverage of ML algorithms, we summarize a variety of further traffic estimation approaches utilizing various kinds of ML techniques: i) [55] proposes a deep learning framework capable of capturing spatial and temporal features using GCN to estimate network-wide traffic multiple

steps ahead in time; ii) [35] describes the extension of a Stochastic Cell Transmission Model (SCTM) with a linear predictor and the consideration of spatiotemporal correlation; iii) [66] introduces the Graph Multi Attention Network (GMAN), using an encoder-decoder architecture to provide long-term traffic estimation up to 1 hour ahead in time.

Finally, there is a limited amount of literature discussing the combination of data fusion, spatiotemporal correlation, and ML, providing a similar methodology to this thesis's proposed traffic estimation application. The authors of [45] propose a model to fuse heterogeneous traffic data acquired through stationary and dynamic sensors and use spatiotemporal correlation of the traffic states from each road segment. The combination of these different information types is provided as an input to a multiple linear regression model that improves the traffic state estimation accuracy. Wang et al. [54] introduce a fusion of fine-grained and coarse-grained traffic data, considering the traffic states of single road segments and areas due to the lack of coarse-grained traffic data in existing traffic estimation models. The traffic estimation is given through the utilization of a GCN that captures the spatial and temporal correlation of the underlying road network and data from various datasets. In contrast to this thesis, the authors of the last two discussed approaches solely rely on traffic data reported from various sensors but do not consider different data types. Lastly, the authors of [64] provide a general platform for spatiotemporal data fusion to support a traffic estimation application. The approach discusses a fusion method to improve the accuracy of traffic state estimation with multi-sensor data. They use directly and indirectly traffic-related data as inputs for two different ML models. The output values of both models are then fused and used for the traffic estimation application. Data features like weather and points of interest are considered for the indirectly-related traffic data. They are used to improve the estimation quality by fusing it in the proposed spatiotemporal data fusion framework. In contrast to the aspect of indirectly-related traffic data, the presented study mainly focuses on details in the fusion process. Furthermore, the authors mainly consider points of interest, type of zone, and others. as indirectly-related traffic data, whereas this study focuses on incident-related data.

3.3 Incident Classification

In the course of the high interest regarding ML-based traffic estimation applications, there is a wide range of literature discussing similar methodologies focusing on incident classification. The goal within the research area of incident classification or prediction is to reduce the danger and damage of future traffic incidents. To enhance road safety in urban areas, proposed applications include traffic management, warning systems, or emergency rescue operations. Recent investigations in the literature [30, 31, 33, 43] propose various methodologies contributing an application for incident detection and most of them are also providing improvements for traffic management to support rescue operations in urban areas (e.g., controlling traffic lights via a RTF transmitter placed in emergency vehicles [43]).

Many studies utilize a deep-learning approach to create an incident prediction model. An example of this is *TAP-CNN*, a model based on a Convolutional Neural Network

(CNN) to predict road traffic accidents [58]. Using a state matrix containing traffic features that influence accidents, together with the CNN, the authors achieve a higher accuracy in predicting traffic accidents than other approaches, like a back propagation model. However, one limitation of this approach comes from the lack of training data, a problem that could be resolved using data fusion to enrich the available information, like the accident prediction model discussed in [37]. Park et al. [37] propose a big data approach using the Hadoop framework[1], providing an efficient way of combining the collection, pre-processing, and analysis of the available traffic data, including information, which is not directly incident-related. Furthermore, the study provides a classification analysis to classify the data entries into several groups of different accidents. While the usage of data fusion shows certain benefits in this approach, the model should consider spatial and temporal aspects of the traffic incidents to further increase the model's accuracy. The authors of [41] introduce a deep learning model that combines incident data with the spatiotemporal correlation characteristic of traffic accidents. It can predict accidents, try to prevent their occurrence, and reduce the resulting damage. It has high accuracy and can be applied to a traffic accident warning system or integrated into intelligent traffic control systems, improving traffic management. The consideration of only directly-related incident data is the main limitation of this approach, and a fusion with other traffic-related features (e.g., traffic flow, weather, etc.) could be used to increase the model's accuracy. Lastly, Wang et al. introduce a hybrid approach for automatic incident detection based on a combination of Time Series Analysis (TSA) and ML [53]. The presented methodology uses TSA to estimate the traffic at a specific time. It combines it with ML to detect incidents based on observable differences between real-world data and the estimated value from the TSA. Evaluating their approach, the authors show that the accuracy of detecting incidents is higher and faster than other state-of-the-art approaches.

Concluding our provided literature review, we summarize the previously discussed frameworks and models, showing their key aspects and functionalities in Table 3.1. The first column gives a general description of the approach together with the main aspects listed in the second column. The functionalities of the solutions are provided in the remaining columns, according to the following labels: i) *Data Col.* denotes whether the approach offers a solution for the problem of collecting data, ii) *Data Fus.* denotes the use of a data fusion technique, iii) *Tra. Est.* denotes that the approach describes a traffic estimation, iv) *Inc. Cla.* denotes that the approach describes an incident classification, v) *ML* denotes that the approach uses ML, vi) *Data Cor.* denotes the solution using a form of data correlation. Comparing the traffic model presented in this thesis to the other presented approaches specifies the main advantage of our solution: Providing a complete set of features to support ITS applications, starting from the collection and fusion of data and resulting in two different data applications. The other approaches currently presented in the literature only combine some of the listed features but instead focus on specific aspects, offering specialized solutions in the context of ITS.

[1] https://hadoop.apache.org

Topic	Lit	Key Aspects	Dat. Col.	Dat. Fus.	Tra. Est.	Inc. Cla.	ML	Dat. Cor.
Smart City Data Platform	[51] [22] [9]	Collect data from smart city sensors; platforms to provide heterogenous data	✓					
Emergency Rescue Operations	[13] [24] [57]	Increase available data in emergency case; path planning		✓			✓	
Data Fusion in Military	[14] [20] [36]	Support military information systems; Surveillance and reconnaissance on air, maritime and ground surface		✓				
Traffic Estimation using Data Fusion	[2] [27]	Heterogeneous data; Spatio-temporal data fusion; increased performance in comparison to using a single source		✓	✓			
Traffic Estimation using Correlation	[17] [29] [67]	Correlation of traffic data features; Correlation between streets and areas; Fill data gaps			✓			✓
Traffic Pred. using ML	[44] [1] [32]	Using modern ML approaches to create accurate estimates			✓		✓	
Traffic Pred. using Data Fusion and Correlation	[34] [4]	Fusing data from mult. sources and use correlation of features		✓	✓			✓
Traffic Pred. using Correlation and ML	[65] [47] [10] [55] [35] [66]	ML can benefit from correlation aspects within the data; Better results compared to other ML solutions			✓		✓	✓
Traffic Pred. using Data Fusion, Correlation and ML	[45] [54] [64]	Combining benefits of all three techniques to further improve estimation quality		✓	✓		✓	✓
Incident Detection / Classification	[58] [37] [41] [53]	Inc. Prediction; Inc.Detection based on traffic patterns; warning systems, etc.		✓		✓	✓	✓
Thesis		Collect and process data; Het. Data Fusion; Data Characterization; Traffic Estimation; Incident Classification	✓	✓	✓	✓	✓	✓

Table 3.1 Comparison: Solutions from literature and our thesis

4

Problem Statement

This chapter introduces the problem statement, discussing the issues and challenges addressed by this thesis. The goal is to answer the following questions: i) How to collect and fuse heterogeneous data from multiple sources in a spatiotemporal way? ii) how to design a traffic estimation application using fused data? and iii) how to classify incidents based on observed traffic patterns?

4.1 Data Fusion

We start by introducing the main problems related to data fusion. We discuss the combination of heterogeneous data in a spatiotemporal way, providing enriched information to support and improve ITS applications. Figure 4.1 describes the workflow of the heterogeneous data fusion, including civilian and military information. There is a massive amount of data being collected in the discussed context (e.g., through real and virtual sensors in smart cities), reported from different sources and represented through different types and formats. Moreover, part of the information is only accessible through a paid subscription and another part is provided through free public access. In Figure 4.1, the different layers of data represent the available data sources, e.g., traffic, incident, weather, social media, vehicular and military, containing data in various types and with different features.

Furthermore, the presence of various data types increases the complexity to provide robustness against errors in the acquired data. For example, wrong measurements, missing values, outliers, conflict, incompleteness and vagueness. Therefore, the combination of data from asynchronous sensor operation, including sensor errors and a particular level of noise, creates a problem with a high complexity [42].

The first task towards a data fusion solution is the collection of available data from a variety of sources. We include information about traffic, incidents, weather and vehicular statistics. Based on our investigations, we notice that the available information is either distributed via commercial providers or offered freely to the public.

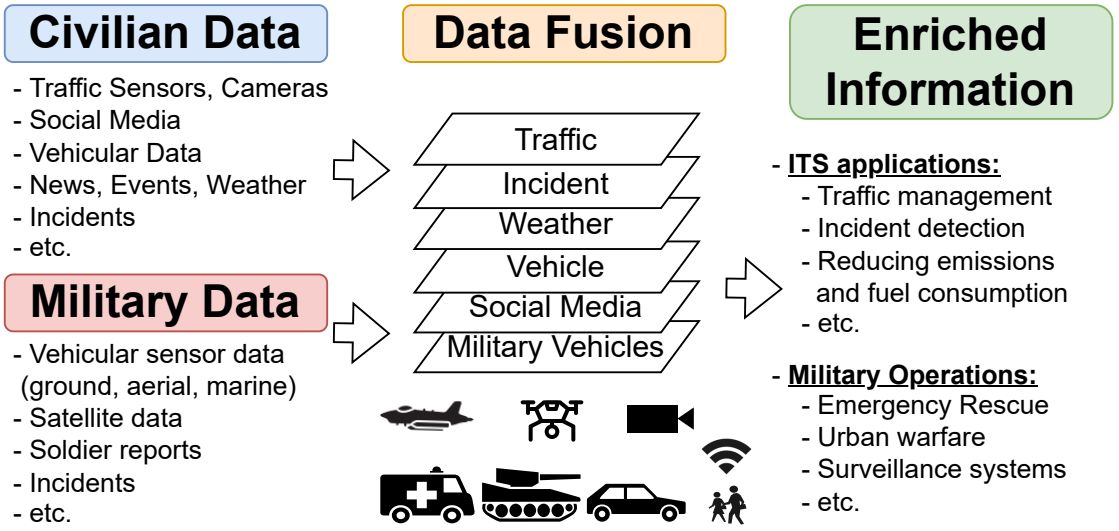


Figure 4.1 Data fusion of civilian and military information

In both cases, there is no uniform way to acquire the available data, requiring various techniques to collect the information. Furthermore, the collectible data in a smart city, for instance, is reported with different spatiotemporal aspects.

The combination of data in a spatiotemporal way is a further challenge in the context of this thesis. We discuss how to combine data that is reported from numerous sensors and distributed over various locations at different points in time. E.g., the combination of data from a stationary traffic sensor and a probe vehicle that traverses the same location within a given time. Therefore, we need to provide a solution for spatial fusion using the provided GPS locations. For the temporal fusion, we combine data entries that are reported within the same situation but at a different level of time granularity. Additionally, we provide several aggregations for the temporal data, to group it by aspects such as day of the week, time interval, season, etc.

Moreover, we argue that data fusion can be used to support urban warfare, by enriching the available data and improving data coverage used by the military. However, the fusion of civilian and military data has an additional complexity. The reason for this is given by the different levels of data such as from the ground level (e.g, soldiers, vehicles, etc.), up to the level of air or satellite data. Furthermore, due to the restricted access to military data, we are just able to demonstrate the scenarios that could benefit from the data fusion concept. Considering a threat (e.g. terrorism, incident, disaster) in an urban area involving civilians that are possibly injured, we argue that the fusion of civilian data from different sensors available in a city can support military operations as well as smart city applications.

In the scope of the discussed problems, this thesis provides a data fusion framework that combines different types of spatiotemporal data from a list of available data sources.

4.2 Traffic Estimation

Next, we introduce the issues related to the estimation of traffic values. The ability to estimate which roads are congested at a certain point in time is highly relevant, e.g., being used by navigation systems to improve path suggestions. The corresponding problem is defined by estimating the status of any given road or area, calculating a value that describes data features like traffic or speed [61].

Combining this problem with the concept of data fusion, using information from a variety of data sources, it could support a more reliable and robust traffic estimation. This is due to the fact that the behavior of traffic is influenced by many things, including weather, street type, time, events and the occurrence of traffic incidents. This increases the complexity of the traffic estimation problem but can also improve the estimation.

Moreover, traffic features generally show correlated behavior on a spatiotemporal level. The traffic situation on a given road can influence the state on surrounding locations. Extracting this type of behavior from the underlying road network and including it to a traffic estimation approach is challenging, as it requires the analysis of data features in a spatiotemporal way. Additionally, we require a method to combine the reported traffic areas if they show an intersection of covered street segments, to obtain a set of unique traffic areas.

Finally, choosing the right model/parameters to implement a traffic estimation is another demanding task. The literature shows a variety of solutions utilizing ML, statistical approaches and correlation. In this thesis, we are going to discuss the process of finding an applicable model for the problem of traffic estimation and present our solutions. Therefore, we implemented two different models, one based on a statistical approach and the other one based on a ML approach. They provide solutions to the problem of traffic prediction at a different level of quality and complexity.

4.3 Incident Classification

In order to provide an extensive traffic model, we also address the problem of classifying different types of incidents based on traffic patterns. Therefore, the model needs to identify certain patterns within the data, using the traffic features, observed over time. This problem has a high complexity according to unclear traffic patterns that are reported as an incident and a hard distinctness between certain types of incidents, such as accident and congestion.

Incidents have an effect on the traffic situation at the respective location and require the collection of all traffic data that is related to a given incident situation. The main issue is the aggregation of data in a spatial and temporal way to cover all relevant data and provide it to the model.

Furthermore, the duration of each incident is varying and may depend on other aspects like weather, time, etc., which requires the definition of a reasonable time interval for the input data, that can represent the complete traffic behavior of each incident case. However, the data sources solely contain a start time of each incident

which may be inaccurate due to a potential delay or some sensor noise. Therefore, we need a method to accurately identify the start and end point of an incident.

Additionally, the given data does contain some incorrect incident reports that consist of too much noise and show an unrealistic traffic pattern that is not related to any incident. Including these reports to the model training would lead to biased results and therefore requires an incident validation mechanism, which is able to detect and remove those erroneous data samples.

This thesis introduces a data application that is capable to solve a binary and multi-class incident classification problem, utilizing proper data filtering and processing methods.

5

Design

This chapter proposes a solution to the discussed problems, by defining a traffic model. The design covers two main parts, presented in Figure 5.1, such as the creation of a dataset through collecting and spatiotemporally fusing data from multiple sources. Furthermore, it provides two applications related to ITS, a traffic estimation and an incident classification, which are using the enriched set of data. The estimation application is implemented in two different ways, based on naive statistics and ML, whereas the classification is provided through a ML-based approach.

5.1 Data Fusion Framework

In this section, the design of the proposed data fusion framework DataFITS[1] is explained, focusing on the methods for collecting, preparing and fusing heterogeneous data from the transportation scenario. The main goal of this solution is to improve the quality of transportation-related data and therefore, enhancing applications in the context of ITS and military (e.g., emergency rescue or urban operations). As presented in Figure 5.1, transportation data includes a variety of heterogeneous data types, such as surveillance data, incident reports, vehicular statistics and weather conditions. Fusing the data in a spatiotemporal domain increases the amount and quality of information and supports the development of data applications.

5.1.1 Data Acquisition

The initial part of the framework is the data acquisition process, illustrated in Figure 5.2 (1). The framework requires a set of parameters, defined within a configuration file, to initialize the data collection process. The geographic area is represented

[1] `https://github.com/prettore/DataFITS`

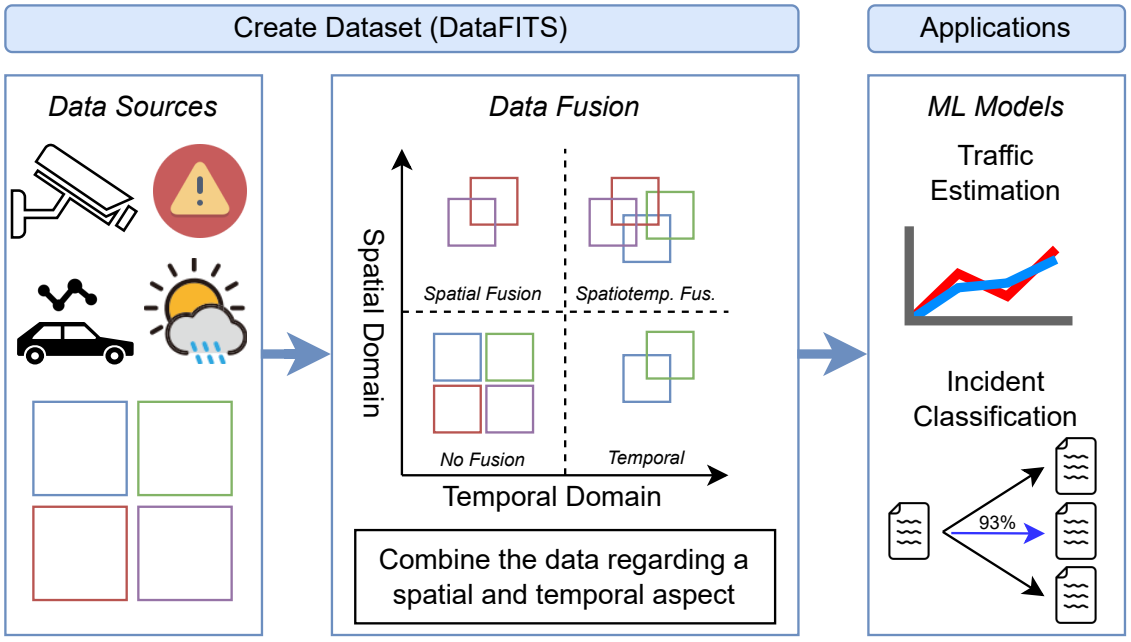


Figure 5.1 Design of the proposed traffic model

through a Bounding Box (BB), defined by two latitude and longitude values. Furthermore, the frequency of requests (delay between the acquisitions) is interchangeable, allowing to adapt the collection process based on potential request limits, given by the data providers. The framework offers a predefined list of currently implemented data sources, which can be selected for the data collection process. Depending on the chosen data sources, there may be a requirement of credentials, which can be added in the corresponding configuration file.

The framework collects transportation-related data, including traffic, incident, vehicular and weather data, via multiple Application Programming Interfaces (APIs) and web crawling methods, based on the defined setup. The acquisition step follows a modular application design, ensuring an easy expandability of the framework functionalities and allowing the specification of new data sources and more data types. However, it is noticed that, according to the different data sources, there is a heterogeneity in the information structure and data type (e.g., json, csv, xml, etc.). DataFITS solves this problem by using various parsing methods, converting the information into a unique csv format, storing the files based on the data source and corresponding date.

5.1.2 Data Preparation

Following the data collection, the available dataset is prepared as illustrated in Figure 5.2 (2). First, two dates are defined, spanning a time frame for the preparation parser. All information belonging in this temporal window is going to be loaded and processed by the framework, allowing to set a limit to the computational power required for the data processing step. Moreover, the data is processed individually per source to reduce the computational cost and time.

Due to the data source heterogeneity, the features are varying in names (e.g., 'Time' or 'Timestamp', etc.) and type (e.g, numerical, textual description, factor, etc.),

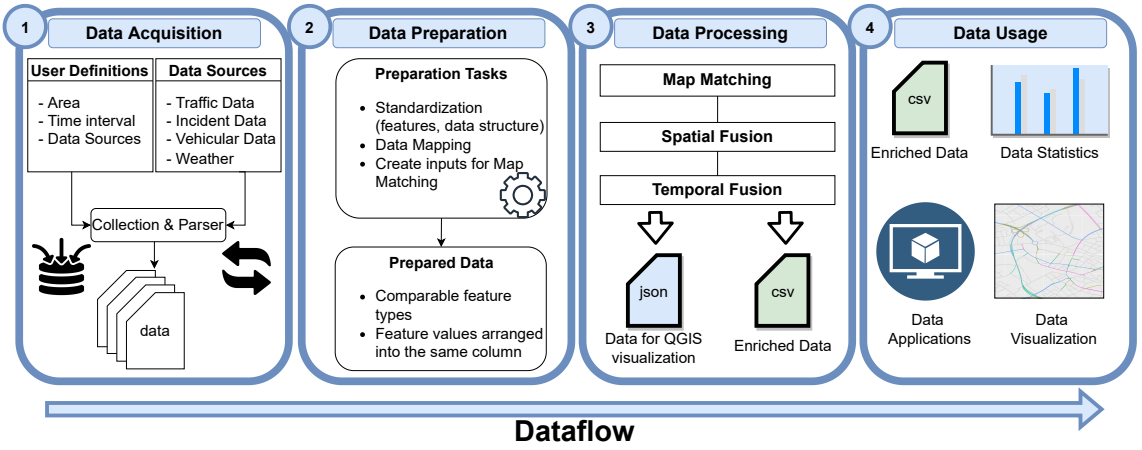


Figure 5.2 Workflow of DataFITS

although they are reporting the same information. As a consequence, DataFITS standardizes the information, addressing the data fields that report the same features by a unique identifier. Furthermore, the information about each timestamp is converted into the universal time format UTC, as it is the worldwide primary time standard. As a solution to the problem of various data types within the same feature, the framework implements different forms of data mapping to the respective information types. The two most common examples are given through a mapping of descriptive traffic values into a numerical representation and a more simplified information mapping regarding the different weather conditions. Therefore, DataFITS contains multiple dictionaries for those mappings, which can be changed in the configuration file, according to individual user preferences. This is illustrated in Table 5.1, providing an exemplary mapping of traffic values reported from different data sources and the respective mapping.

DataFITS contains a method to collect a Shapefile (SHP) from OpenStreetMap (OSM), according to the BB that has been defined in the data acquisition. The SHP provides a road network, required for the map matching procedure. Each SHP contains a list of road segments, creating the road network within the specified BB, allowing to identify each individual segment by an *fid* value. The amount of information that can be extracted by the SHP varies based on the purpose. In our case the framework collects the transportation-related information, like the road type or the maximum allowed speed. The underlying implementation uses OSMNX [7], a python package to download geospatial data from the OSM, which offers a free geographic database without any restrictions.

Additionally to the geographical road network data, the map matching process requires a special data format to match the collected data. Therefore, the framework provides a method to extract the relevant information (a unique identifier and the geometric data) and converts the GPS coordinates into Well-known Text Format (WKT). Furthermore, in case of traffic data, the method monitors the number of unique reported traffic observations to only match the data once, as the data is provided on static and non-changing locations. This provides a great improvement to the performance of the map matching process, which is discussed in the next section.

Textual Description		Numerical Traffic Value
Normal Traffic	→	1
Increased Traffic	→	5
Traffic Jam	→	10

Table 5.1 Exemplary data mapping of traffic values

5.1.3 Data Processing

The data processing procedure, described in Figure 5.2 (3), is structured into three different sub-tasks. First, we use a map matching algorithm to prepare the spatial fusion of data according the corresponding road network. Furthermore, the information is fused in the temporal domain, to complete the process of spatiotemporal fusion. In the following, we are giving a description about the most important aspects of the data processing, including a selection of methods and algorithms from DataFITS.

5.1.3.1 Spatial Fusion

The proposed solution regarding the spatial fusion uses a combination of map matching and data processing. Map matching is a technique that takes a set of GPS coordinates and matches them onto a new set of coordinates within a predefined GPS error, given an underlying road network. Using data from various sources reporting their GPS data with different precision, the DataFITS uses this procedure to reduce the divergence in precision under consideration of a potential GPS error. Map matching is widely applied in various map services, such as Google Maps, Bing Maps and HERE, in order to match the GPS coordinates onto the road network. Therefore, we investigate a variety of available map matching algorithms in scope of our research, using different methodologies to solve the problem of matching the locations, such as the Noiseplanet[2], Mapmatching[3], and the Fast Map Matching (FMM)[60]. Evaluating the different approaches, we chose to implement FMM as the map matching part of DataFITS, because it is a fast open source tool written in C++ and Python, offering two different algorithms to achieve the optimal performance depending on the size of the given road network.

The FMM uses the trip file and the SHP, created during the previous preparation step of the framework, to match all GPS points. More specifically, the trip file contains the necessary data for the process, given by the id of each data entry (*id*) and the corresponding geometry (*geom*), in WKT format. In most cases, e.g., for the reported traffic data, the geometry is given through a *Linestring*, a geojson class representing two or more geometric points that are collected through a line, describing the road path. In contrast, the GPS data reported from the incident-related data sources is given by a *Linestring* that solely contains the start and end point of each incident location. FMM offers a variety of configuration parameters, including the candidate size, search radius and GPS error, which affects the resulting path matched to each trajectory and the overall performance. In our proposed setup,

[2] `https://github.com/arthurdjn/noiseplanet`
[3] `https://pypi.org/project/mapmatching`

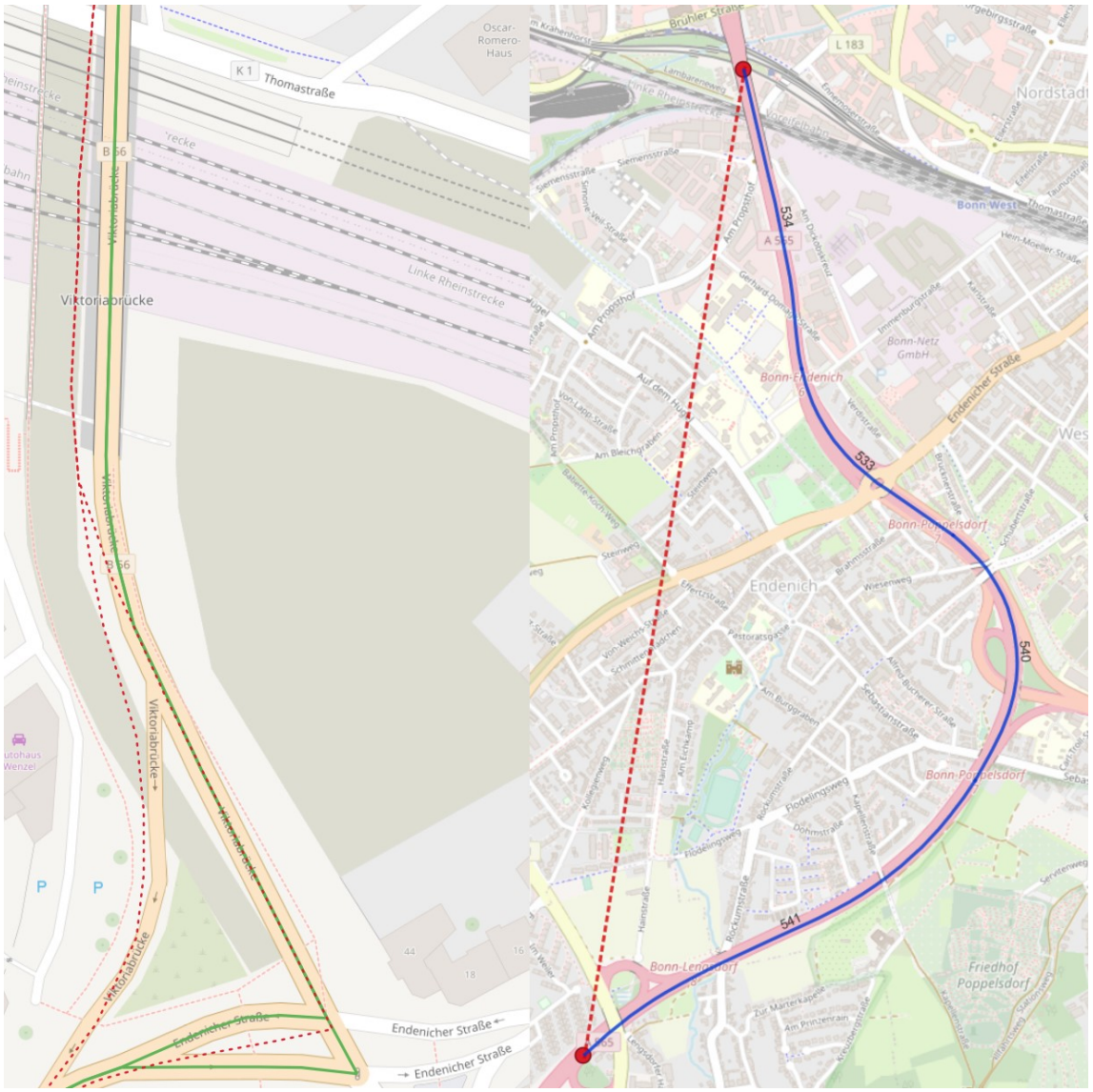


Figure 5.3 Example of the map matching process

the algorithm decides between eight candidates within a search radius of 400 meters, using 20 meters of GPS sensor error.

Figure 5.3 illustrates an exemplary map matching process, showing a path (left side) and the trajectory between two points (right side) matched onto the correct path, given the underlying road network. The first case represents a data entry reported from a traffic data source, and visualizes the capability of the map matching to fix imprecise GPS coordinates, due to the differing precision between the data sources. As visible in the figure, the original reported path (red dotted) is not matching the road network of the city, but using map matching, we get a realistic description of the geometry (green). The right side of the figure shows the correct matched path (blue) between the two points (marked in red) given by the initial data source. Instead of using the trajectory between those two points (red dotted), the map matching algorithm provides this improved output, which is further used to support the spatial fusion process. The output of FMM contains four values for each matched road trajectory (*id, opath, cpath and mgeom*), we are using to enrich the data in different levels:

- *id*: A unique identifier for each input entry, used to merge the map matching data with the other given data features
- *opath*: An Array containing all roads (fid values) which got matched to each point from the input trajectory
- *cpath*: An Array containing the path of roads (fid values) that is traversed by the input trajectory
- *mgeom*: An Array containing all GPS points matching the given cpath values.

This additional information is merged back to the initial acquired set of data using the id values created in the data preparation step. The usage of unique identifying values is required when processing the data from both traffic data sources, as the framework is not matching the complete set of information in order to save computing power.

We perform the spatial data fusion by combining the data and create groups based on the *cpath* value. Algorithm 1 shows the process of spatial data fusion for n defined data sources, given an underlying road network. It returns one csv file that contains the complete dataset, including the enriched information from the map matching process. The presented algorithm is defined through three different parts: i) REDUCE_TRIPS: Generate the trip file using a subset of the data, based on number of unique reports (only traffic data sources) ii) SPATIAL_FUSION: Merges the map matching output to the initial data iii) MAIN: Runs the other functions and re-groups the data on a level of fid values. The main function (line 26-46) loads the respective data and trip files for each of the implemented sources and creates an empty set to store all unique fid values (line 28-30). In case of a source providing traffic data, the method *REDUCE_TRIPS* is called, providing an optimization approach to reduce the computational requirement for this type of data. It decreases the size of the trip file, which is possible due to the static SHP and the traffic sources reporting data for the same locations on every acquisition. The respective function (line 1-8) stores the first acquisition time (line 2) and iterates over the complete set of data, testing for a change of the acquisition timestamp. On the detection of such a change, the function uses the current index to create a subset of data, including the trips from all unique locations that are covered by each traffic data source (line 4-6). We provide a brief run-time analysis, considering a time frame of six months (180 days) with 144 acquisitions per day and an average of 5,000 GPS points that have to be matched within one set of traffic observations. This results in a total of 129,600,000 points ($144 \cdot 180 \cdot 5000$) to be processed by the map matching algorithm, which is capable of processing 500 points per second on average. Based on this numbers, the map matching process would take approximately 72 hours to finish the task. In contrast, using our optimization approach matches the data once, taking 10 seconds to process the 5,000 distinct GPS points. However, we are just able to use this approach on the traffic data due to the static locations. For the incident- and vehicular-related data these locations are changing with each data observation, and therefore we cannot apply this method.

Next, the spatial fusion is described (line 9-25), taking the trips, SHP and collected data as an input, returning a set of all unique fids (*fid_arr*) and the data which got merged together with the information from the map matching process. By

Input: shp_file, sources
Output: fused_csv

```
1: procedure REDUCE_TRIPS(trips, data)
2:     first_timestamp = data[0][time]                    % First acquisition time
3:     for i=0 to length(data) do
4:         if (data[i][time] != first_timestamp) then     % New acquisition time
5:             return trips_cut = trips_cut[0:i]          % Return unique set of trip data
6:         end if
7:     end for
8: end procedure
9: procedure SPATIAL_FUSION(trips, shp, data)
10:    fid_arr = []
11:    config = k, radius, error
12:    matched = map_matching(trip_file, shp_file, config)
13:    data_len = length(matched)
14:    for i=0 to length(data) do
15:        cpath_arr = matched[i%data_len][cpath]
16:        data[i][cpath] = cpath_arr
17:        data[i][cords] = matched[i%data_len][mgeom]
18:        for fid in opath_arr do
19:            if (not fid in fid_arr) then
20:                fid_arr += fid
21:            end if
22:        end for
23:    end for
24:    return fid_arr, data
25: end procedure
26: procedure MAIN
27:    for src in sources do                              % Repeat for every source
28:        data = read_csv(src_data)                      % Load data entries
29:        trips = read_csv(src_trips)                    % Load map matching input
30:        fids = {}                                      % A set storing unique fids
31:        if (TRAFFIC_SOURCE) then                       % Case: Traffic data source
32:            trips ← REDUCE_TRIPS(trips, data)
33:        end if
34:        fids, data ← SPATIAL_FUSION(trips, shp, data)
35:        for fid in fids do
36:            for entry in data do
37:                if (fid in entry[opath]) then
38:                    fused += entry
39:                end if
40:            end for
41:        end for
42:        append_csv(fused)                              % Add fused data to the final csv file
43:    end for
44:    return fused_csv
45: end procedure
```

Algorithm 1 Spatial fusion

configuring the variables k (number of candidates), *radius* (the search radius in meters) and *error* (GPS sensor error in meters) (line 11), the user can impact the map matching process. The respective function (line 12) calls the FMM and returns the dataframe *matched*, storing the matched coordinates (*mgeom*) and the fid values from the underlying path (*cpath*) for each data entry within the trip file. The data fusion is described in the lines 13-23, iterating over both datasets, containing the new information and all data features. The identifier of the matched data is equivalent to calculating iterator i modulo the length of the dataset (line 15), because of the optimization approach for the traffic-related sources. This will ensure, that each entry is matched to the correct original data entry with the corresponding id. To fuse the information, all *fid* values, contained in the *opath* array, are extracted from the map matching output (line 15) and added to the data entry in a new *cpath* column (line 16). This is a mandatory step regarding the spatial fusion, facilitating the extraction of *fid* values from the array of each data entry. Furthermore, the coordinates of the acquired data are replaced by the matched geometry from the map matching output (line 17). The last step adds all unique *fid* values to the array *fid_arr* (line 18-22), which is returned together with the dataframe containing a new column describing the *cpath* values of each entry (line 24).

The return of SPATIAL_FUSION is further used to create a new set of data, grouped by the *fid* values (line 35-41). The algorithm examines every unique *fid* value (line 35) for existence in each data entry (line 36), creating a new row in the set of fused data and adds all features of the data entry (line 37-39). This ensures the possibility to access each single road segment by an identifier, rather than having only a *cpath* variable, representing the list of involved roads for each data report. This extraction and regrouping of the data, using individual road segments, significantly increases the memory usage. Therefore, the framework splits the data in chunks of 100,000 rows and processes one of them at a time, drastically reducing the memory requirement and allowing the use of multithreading to speed up the process. Finally, each chunk of fused data is appended to the csv output file (line 42), representing the output of the spatial fusion, by providing one set of data that contains the spatial grouped information of all data sources according the same road network.

5.1.3.2 Temporal Fusion

In contrast to the spatial fusion, the combination of data in a temporal domain does not require any type of map matching, but rather processes the available information related to the aspect of time. As a result of the data preparation, the information is provided in a uniform data format of UTC. Therefore, the complete set of information, addressable through the fid values, can be grouped regarding various aggregations over time (e.g., hour or day). We provide freely configurable data aggregations within our R scripts which are also used to conduct various types of data analysis. R is an open source software environment for statistical computing and graphics, widely used in the scope of data analysis being a powerful tool with great expandability due to user-created packages adding functions to the R language. The configuration of the spatiotemporal fused data is given through a data grouping in a 10 minute time aspect and the possibility to access the data either due to the fid values, reflecting single road segments or by aggregating over the cpath, providing the set of connected roads from each observation.

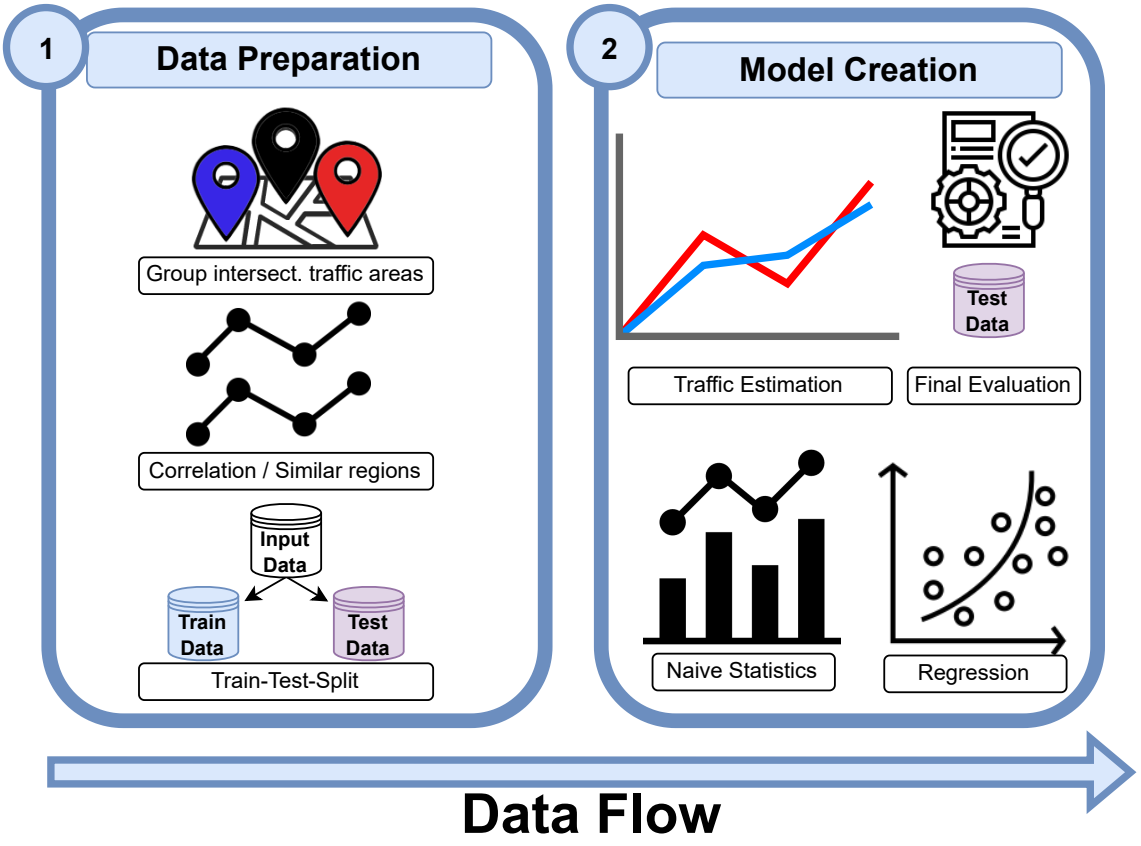


Figure 5.4 Design of the traffic estimation application

5.1.4 Data Usage

The last step, visualized in Figure 5.2 (4), shows a variety of use cases that are applicable to the dataset returned by the DataFITS framework. The enriched data offers many possibilities in military and civilian fields, by increasing and enhancing the information in a given urban area, supporting decision-making in the context of smart cities and urban military operations. At this stage, we create different types of statistics and visualizations, showing spatiotemporal data analysis, e.g., visualizing the data coverage. Moreover, the spatial analysis provides heat maps and density plots separated by each source and based on different features such as the number of observations, traffic, speed and incidents. On the temporal analysis, DataFITS provides time-series statistics regarding a specific time window and shows the correlation between different features. In scope of the presented thesis, the main aspect of data usage is to provide access to a big amount of high quality data allowing the creation of different data applications.

5.2 Traffic Estimation

Following the description of creating a heterogeneous fused database, we present the design of our proposed traffic estimation application, shown in Figure 5.4. First, we introduce a variety of data preparation steps to process the fused data, including grouping the data based on intersecting traffic areas, find similar regions based

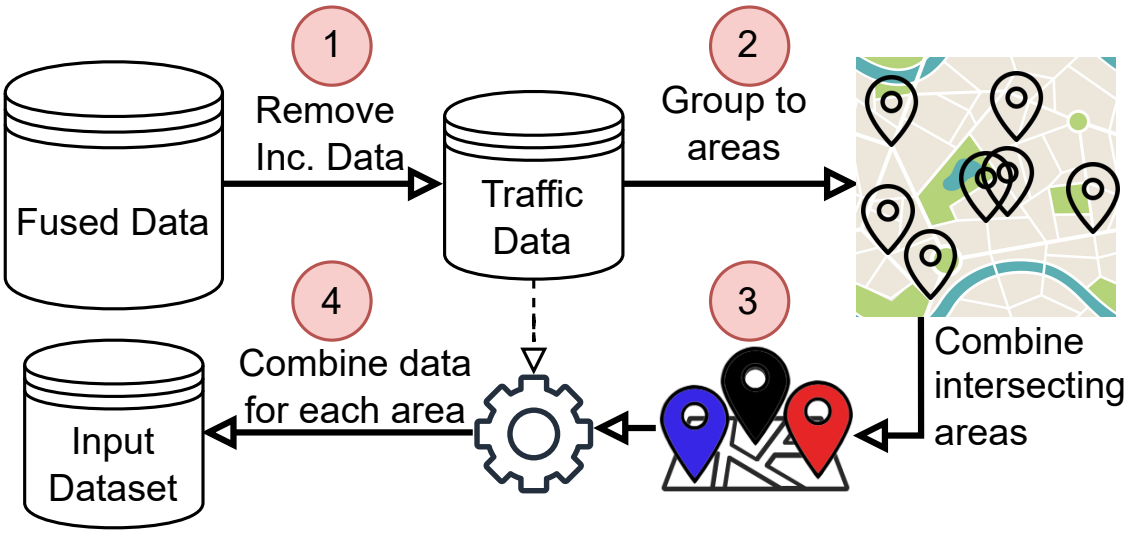


Figure 5.5 Group traffic data by a set of unique areas

on correlating traffic behavior and perform a train-test-split on the data (step 1). Subsequently, the traffic estimation model is created and evaluated using the test data (step 2). We designed the respective application based on two different models, using naive statistics and a ML-based regression.

This section introduces two traffic estimation models provided in scope of the thesis. First, we developed an approach based on naive statistics (mean value) to create an estimation for traffic values. The model predicts traffic values for a single region on a defined weekday and uses traffic data from correlating regions for the estimation. Furthermore, we propose the methodology of a robust ML model that provides an accurate traffic estimation for a given region under certain parameters (e.g., weather, street type, etc.). The model uses a regression algorithm, trained on the heterogeneous database of traffic data.

In the following, we are going to explain the design of both models in detail, starting with the preprocessing procedures which are the same for both approaches. Subsequently, the creation of both models is described, focusing on the different aspects of each respective model.

5.2.1 Preprocessing

The fused data, generated by the DataFITS, is processed by several methods, before it is used as an input to the estimation models. First, a subset of data is created, by removing the incident reports, as they are not required for the estimation. The application provides a procedure to calculate the intersection between road segments and combines them accordingly to group the data by individual areas. Moreover, we provide a method to calculate data similarities using the correlation of data features, to include this information in the estimation of each respective area.

The initial data preprocessing steps are visualized in the flow diagram shown in Figure 5.5. First, because the traffic estimation procedure solely requires data containing traffic information, the data is filtered, removing all rows and columns that contain incident-related information (step 1). This step is necessary to reduce the

memory requirement, according to the particularly high amount of data. Next, the data is grouped by the spatial aspect of areas that contain one or multiple road segments. Using a data aggregation over the *cpath* variable, we create a list of areas contributing traffic information to the dataset (step 2). In fact, we observed that there exist some areas that have an overlapping behavior, basically describing the same traffic region but with a minor difference in the covered road segments. This is a result of the data fusion, combining traffic areas from individual data sources. Therefore, we provide a method to combine those intersecting regions, resulting in a set of unique and distinct traffic areas (step 3). The underlying function iterates through all existing areas, calculates pairwise intersections and combines them if the amount of overlapping road segments is above a defined threshold. Finally, the initial set of fused data is re-grouped according to the new set of combined areas, resulting in the input dataset that contains all information combined for each area (step 4). Therefore, the final input dataset can be grouped by each individual traffic area and filtered by any arbitrary parameter, providing the input to both traffic estimation models.

Furthermore, the proposed design contains a procedure to increase the amount of input data for each area. We provide a function to add further data points from other, similar behaving regions, to the area of interest, using the aspect of data similarity. Therefore, we are going to utilize a modified version of the *Pearson Correlation Coefficient* and the DTW, to identify correlated regions that show a similar traffic behavior. The correlation between two time series was defined in [17], and adapted for our proposed methodology:

$$X_{i,j} = \frac{\sum_{t=1}^{L}(S_i(t) - \bar{S}_i)(S_j(t) - \bar{S}_i))}{\sqrt{\sum_{t=1}^{L-t}(S_i(t) - \bar{S}_i)^2} \cdot \sqrt{\sum_{t=1}^{L-t}(S_j(t) - \bar{S}_j)^2}} \tag{5.1}$$

Using the correlation defined in Equation 5.1, we can calculate the respective value between two time series of any traffic data feature $S_i(t)$ for two regions i and j. We calculate this value between every pair of regions, and define a threshold of correlation th_{cor} to identify similar regions.

However, this type of correlation can solely describe a linear relation between two variables and therefore, results in high values for two time series with a similar traffic pattern but at a different level of values. An example for this is given by two areas that show the same increase of traffic at the same point in time, e.g., from 1-5 and 5-9 respectively, would be considered highly correlated using this approach. However, due to the much higher level of traffic, the information from the second area is not feasible as additional information for the first area. Therefore, we use the DTW algorithm to measure the distance between two time series, and set a threshold of DTW th_{dtw}, to ensure that both correlating areas are on a comparable level. DTW is an algorithm to measure the similarity between two time series that are not synchronized. More precisely, DTW is capable of using a temporal alignment of the data pattern resulting in a more similar comparison than using e.g., the Euclidean distance, comparing timestamps regardless of the feature values [48].

Calculating both, correlation and DTW for the data features of traffic and speed, we define the following threshold for identifying similar regions:

$$(cor_{traf} \geq th_{cor} \wedge cor_{speed} \geq th_{cor}) \wedge (dtw_{traf} \leq th_{dtw} \wedge dtw_{speed} \leq th_{dtw}) \tag{5.2}$$

As a result we can generate a set of traffic regions, that show a similar and correlating behavior for all areas within our dataset. This information can then be used to generate a larger set of input data for each area, described in the next section.

We use the complete set of data, filtering it accordingly to a single traffic area to generate the input data for the traffic estimation. Furthermore, a subset of input data is created, according to the chosen set of parameters regarding time frame, weekday, street type and weather. Before the initial creation of the respective traffic estimation models, we require a method called *Train-Test-Split*, generally used in ML applications. Using this method allows us to evaluate the performance of both models. Therefore, the full dataset is split into a set of data points that are used to train the model (train dataset) and another one to test the model and evaluate the performance (test dataset). The split is performed by randomly sampling all available data points for each area, and split them according to the defined parameters. Using a data splitting procedure allows to simulate each models' performance on a new set of information that is unknown to the model. Testing a traffic estimation model on the same set of data that it was trained on could lead to a biased result and overfitting, which denotes that the model is too close to the particular set of training data.

5.2.2 Model Creation

Subsequently to the explained data preprocessing steps, this section provides a description regarding the creation of both individual estimation approaches: The statistical-based model and the ML-based model.

5.2.2.1 Naive Statistical Model

Initially, we designed a simple methodology, based on a naive statistical approach, to estimate traffic values. The model takes a subset of data entries from the train dataset, that serves as the underlying input data for the estimation, based on a selected time interval and one respective day of the week. The approach utilizes a naive and intuitive calculation of estimation values. It groups all data points that represent the same time, neglecting the information about the date, and calculates the mean value. Moreover, in case of existing similar regions for the respective area, the approach computes the average traffic value from this additional data points. The final estimated value is generated by calculating the mean of those two average results, representing 50% of data from the observed region and 50% from all similar areas. Within the time frame of this thesis, we presented this naive approach in a publication [69].

$$Y(t) = \overline{x(t)} + \frac{1}{n_{corr}} * \sum_{i=1}^{n_{corr}} \overline{x_i(t)} \tag{5.3}$$

Equation 5.3 describes the calculation of $Y(t)$, representing an estimation value for a time point t. Additionally to the mean of the original region at time t, $x(t)$, we also add the average from all corresponding regions $i \in 1, ..., n_{corr}$, represented by

the second part of the equation. Subsequently to this naive approach we further wanted to propose a more refined solution, leading to the investigation of ML within the context of traffic estimation.

5.2.2.2 ML Model

In contrast to the statistical approach, the ML-based solution adds more complexity to the generation of the estimated traffic value. Supervised ML approaches can be categorized into two types of estimation algorithms: *Regression* and *Classification*. Regression solves the problem of predicting continuous variables according to the data labels, representing the same type of variable (e.g., use traffic values to estimate a new traffic value). In contrast, a classification task is used on data with class labels and solves the problem of categorizing data samples into different classes. The proposed traffic estimation model uses regression to estimate continuous traffic values (e.g., traffic level and speed), based on historical data.

A regression algorithm can be implemented in various types, e.g., through a linear or polynomial regression. Following the same goal of modeling the relationship between two variables (explanatory and response variable), the type of regression is determined by the underlying technique. In case of the simple linear regression, the model is fitting a linear equation to the input data.

$$Y = \theta_0 + \theta_1 x + \epsilon \tag{5.4}$$

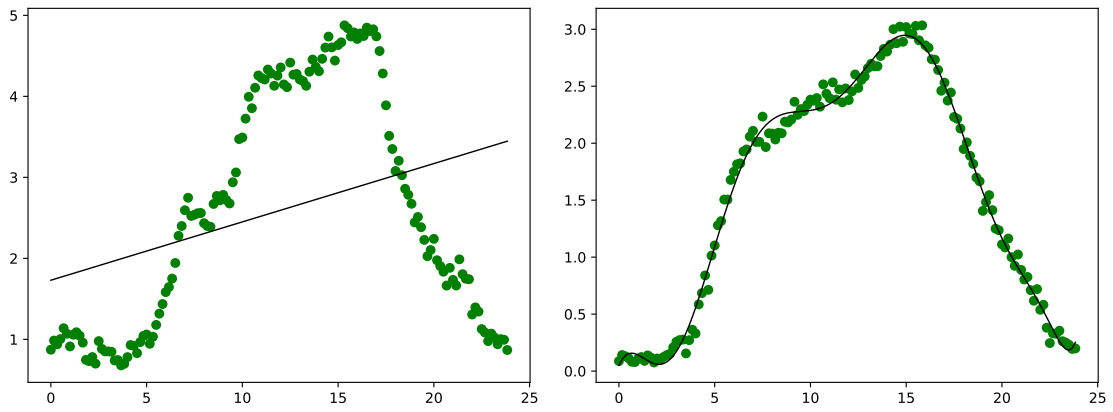
Equation 5.4 describes the approach to find an intercept θ_0 and slope θ_1 on the explanatory variable X to represent the dependent variable Y in the best way. ϵ represents a random bias variable, adding noise to the fitting process. The underlying estimation method of the model reaches this goal by minimizing the sum of residual squares between the observed targets in the dataset and the estimation [38]. The residual sum of squares in a model with one explanatory variable X is defined by:

$$RSS = \sum_{i=1}^{n} (y_i - f(x_i))^2 \tag{5.5}$$

Therefore, Equation 5.5 describes the sum of squares between all predicted points $f(x_i)$ and the corresponding variable to be predicted y_i, which has to be minimized by fitting a linear line. A main problem of the linear regression algorithm is given by its linearity which often results in an underfitting of the data. This denotes that the one-dimensional complexity of the linear model is not suitable to represent certain datasets. The problem is visualized in Figure 5.6 a), showing the linear regression line not being able to accurately represent the traffic behavior from an exemplary area within our dataset.

Due to the non-linear aspect of traffic data, linear regression is not suitable for our data application. Therefore, we preserve the problem of underfitting, by increasing the complexity of the regression model and use a polynomial regression:

$$Y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + ... + \theta_d x^d + \epsilon \tag{5.6}$$



(a) Simple Linear Regression **(b)** Polynomial Regression

Figure 5.6 Comparison: Simple linear regression and polynomial regression

Equation 5.6 shows a higher order polynomial, up to a degree d, that can be used to create a representation of the dependent variable Y at a much higher level of quality. This fact is also visualized in Figure 5.6 b), showing a regression line with an exemplary degree of 10 that provides a much better estimation regarding our available traffic data. As a result, we use a polynomial regression approach in the provided ML-based traffic estimation model. The implementation calculates an ordinary least squares linear regression, resulting in an estimation that minimizes the sum of squares between the dependent and independent variable.

The proposed model is setup to estimate traffic values on an arbitrary time frame (up to 24 hours) for a set of defined parameters: Data feature (traffic, speed, etc.), weekday, weather and road type. An example for such an estimation is given by calculating the traffic level on a Monday in a selected motorway area for the whole day, resulting in 144 estimated data points (1 per 10 minutes). However, before we can conduct an estimation based on the available information within the train dataset, we need to process the timestamp of each data entry. First, we neglect the information about day, month and year, because the model solely considers time information related to minute, hour and day of the week. Furthermore, the time information, represented through a python datetime object, is converted into a numerical value, to be used as an input for the regression model. Next, the converted data is reduced to a subset which is used to train the model, filtering entries by the defined parameters and the area of interest.

The function *ESTIMATE*, presented in Algorithm 2 describes the calculation of the estimated values. Using the area and a polynomial degree as an input, the function creates a polynomial regression model, fits the training data and returns the estimated points of the dependent variable Y. As previously discussed, the initial step filters the input data by the defined parameters (line 2) and separates the data points into a train and test dataset (line 3). The explanatory time variable x is converted from a datetime object into a numerical representation for both datasets (line 4-6). Lines 7-12 describe the creation of the regression model using a set of functions, starting with the *PolynomialFeatures* function that generates a polynomial feature matrix of a certain degree d. This type of data transformation is required to represent the given input in a higher order feature space, e.g., a 2-dimensional feature space (X_1, X_2) is transformed to $(1, X_1, X_2, X_1^2, X_1 \cdot X_2, X_2^2)$. Therefore, the newly

Input: area, deg
Output: estimation

```
1: procedure ESTIMATE(area, deg)
2:     input = data[parameters]
3:     X_train, X_test, y_train, y_test = train_test_split(input, test_size=0.2)
4:     % Convert time to numeric objects
5:     X_train = np.array([x.hour + x.minute / 60 for x in X_train])
6:     X_test = np.array([x.hour + x.minute / 60 for x in X_test])
7:     % Create Polynomial Regression Model
8:     poly = PolynomialFeatures(degree=deg)          % Create feature matrix
9:     poly_features = poly.fit_transform(X_train)    % Fit the data to the matrix
10:    regression = LinearRegression()      % Ordinary Least Squares Regression
11:    regression.fit(poly_features, y_train)                % Train the model
12:    estimation = regression.predict(poly_features)
13:    return estimation
14: end procedure
```

Algorithm 2 Traffic estimation

created feature contains the bias value of 1, all values raised to the power for each degree $\in 0, ..., d$ and all combinations between every pair of features. Next, the input values stored in the *X_train* array are fitted to the data, and transformed afterwards (line 9). The variable *poly_features* contains a polynomial feature matrix and allows to use the *LinearRegression* class to implement a polynomial regression. Therefore, an instance of the linear regression is stored (line 10) and we use the *fit* function to train the model using the polynomial representation of our test data (line 11). Finally, we can use the regression model, to estimate the response variables Y for an arbitrary set of input points (line 12), comparing them to the variables contained in the test dataset to measure the performance of the model. Furthermore, to determine the best value for d, we are going to compare the performance of the model, using a range of degrees from 1 to 20.

In Chapter 6, we are going to illustrate the achieved performance of both presented models, through a variety of metrics on a set of different experimental setups. Furthermore, we provide a detailed comparison of both approaches and evaluate the usage of additional correlating data points from similar regions.

5.3 Incident Classification

The second data application of our proposed model implements a classification of incidents based on traffic data. We are utilizing a modified version of the k-NN algorithm, capable of learning different patterns from historical traffic data, and categorizing them as a certain type of incident. The design of our proposed application is depicted in Figure 5.7. The initial part of the design implements a variety of data preprocessing methods making the data applicable to be used as an input to the classification model (step 1). In a second step, the input data is generated, adding data from normal traffic situations to the input data of the model, interpolates the data if required and performs a train-test-split. Finally, the model is

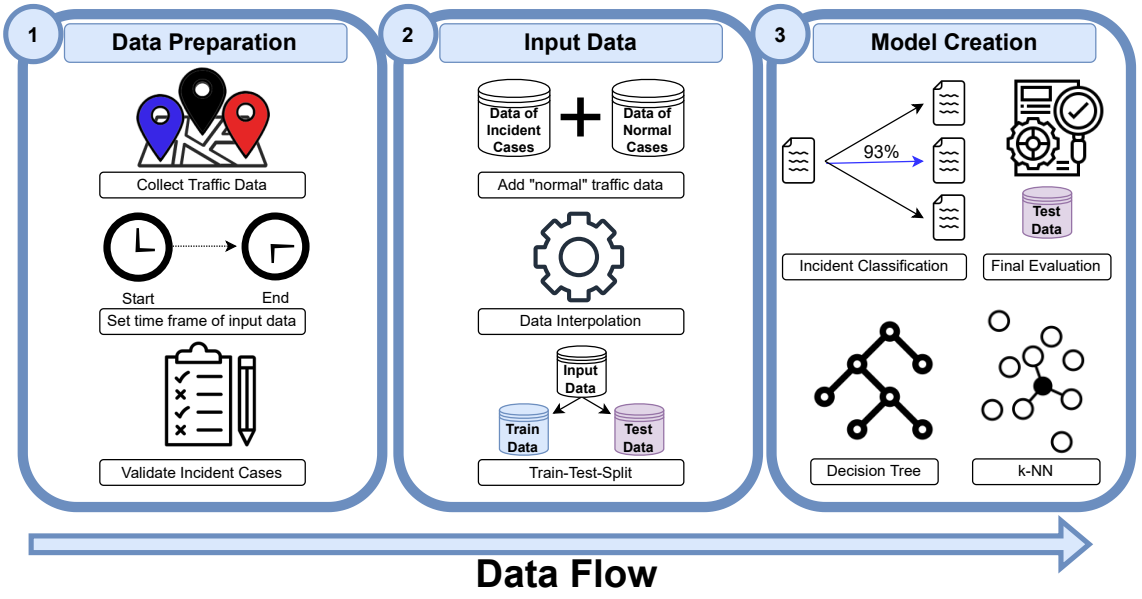


Figure 5.7 Design of the incident classification application

created, providing two different approaches (step 3), that we investigated in scope of this thesis: A binary and multi-class classification (three different types of incidents). The evaluation of the model is going to be discussed later in Chapter 6.

5.3.1 Preprocessing

The first task of the data preprocessing collects all data related to an incident, a procedure described in Figure 5.8. Initially, we filter the data, solely taking entries that provide incident information (step 1). Subsequently, we iterate over all data entries, extracting the incident location (by fid value), depicted in the second step. Because the incident-related data sources do not provide information about traffic data features (e.g., traffic and speed), we add this type of information using the spatiotemporal fused data. This is a main part of the data pre-processing, due to the requirement on traffic data from our proposed classification approach. To get all information related to one specific incident, we provide a method to extract all traffic areas that are intersecting with the incident area (step 3). The underlying procedure is explained in Algorithm 3, using the set of all incidents, the unique regions and an overlapping threshold to output a list that contains the intersecting areas for each incident observation (step 4).

Our proposed algorithm starts by initializing an array to store the required information (*intersecting_areas*), described in line 2. The main part of the algorithm (line 3-15) contains a nested loop, iterating over all incident entries to calculate the ratio of identical roads contained within the *cpath* array. Initially, the outer loop defines the first area, using the cpath value of the respective incident case (line 3), and iterates over all traffic regions, setting the respective cpath values to the second area (line 7). The list of intersecting road segments is stored in the array *inter* (line 8) and, through dividing the length of this array by the amount of street segments covered by each respective area, we calculate the proportion of overlapping roads (line 9-10). If this value matches a defined threshold, the corresponding area is

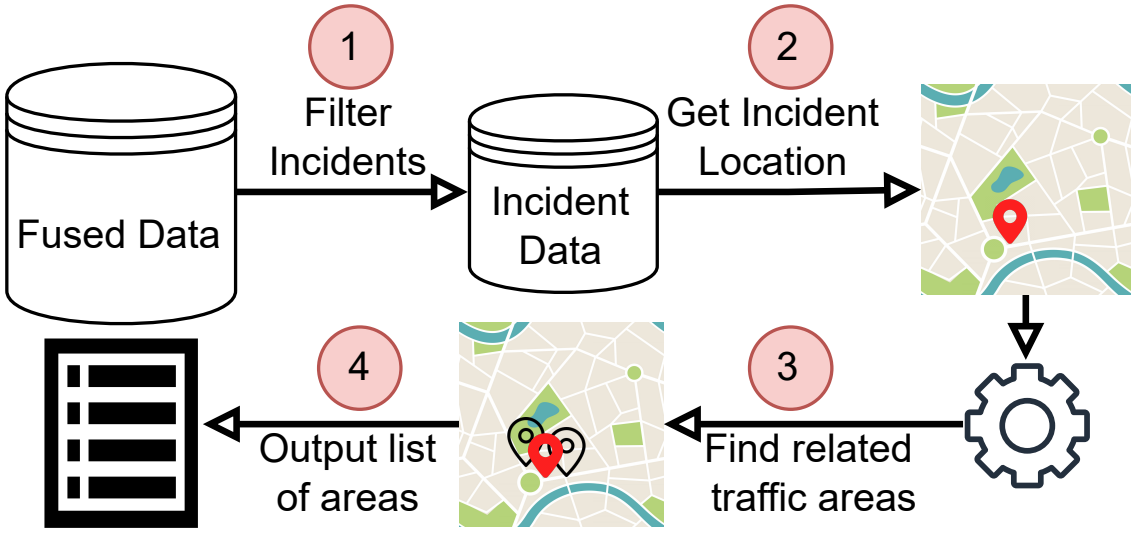


Figure 5.8 Collect traffic data for each incident

added to the array *intersect_one_area*, storing all intersecting areas for the observed region (line 11). On completion of the inner loop, the new information of intersecting areas is added to the array *intersecting_areas* (line 14), which also represents the final output of the algorithm, containing all intersections for each area in the set of incidents (line 16). The runtime of the given algorithm is $O(n^2)$, but only needs to be executed once during the data preprocessing to provide essential information regarding the further steps of the proposed methodology.

5.3.1.1 Input Strategy

Furthermore, to obtain all data related to one single incident observation, we include the temporal aspect of incident duration. This is necessary, because incidents are reported over a certain period of time. Therefore, we include data from a time frame starting four hours before the incident report and ending four hours after, which provides a sufficient amount of traffic information for each observed incident. We selected this wide time frame due to potential delays in the reported start time of an incident, which was noted in a few cases when inspecting the dataset. To be precise, a few incident cases contained a delay between the reported start time and a visible effect on the traffic values. Due to these time delays, we are working with two different approaches, when defining the start time for each incident report: i) *Original start time*: Using the original start time reported from the data source and ii) *Estimated start time*: Iterating over the traffic data, to find significant changes in traffic and set the start point based on this observation. We are going to evaluate the model using input data based on these two different approaches, to see if our approach provides more realistic representation of the incident start time, leading to a better model accuracy. Furthermore, we are going to compare the model using data from a time interval 90 minutes prior and after the incident time and 120 minutes respectively.

Input: data, unique_cpaths, overlap
Output: intersecting_areas

```
1: procedure INC_INTERSECT_AREAS(incidents, unique_cpaths, overlap)
2:     intersecting_areas = []                          % Initialize output dataframe
3:     for i=0 to length(incidents) do                  % Iterate over all data entries
4:         area1 = incidents[i].cpath
5:         intersect_one_area = []                      % Intersections for one area
6:         for j=0 to length(unique_cpaths) do          % Iterate over all cpaths
7:             area2 = unique_cpaths[j]
8:             inter = intersect(area1, area2)
9:             if (length(inter) / length(area1) >= overlap &
10:              length(inter) / length(area2) >= overlap) then
11:                intersect_one_area.append([area2])
12:            end if
13:        end for
14:        intersecting_areas.append(intersect_one_area)   % Add areas to output
15:    end for
16:    return intersecting_areas                % Returns all intersections for each area
17: end procedure
```

Algorithm 3 Intersecting areas

5.3.1.2 Incident Validation

Moreover, we introduce a method to validate an incident observation. Therefore, we process the traffic data, related to the incident, in order to identify cases that contain too much sensor noise. In other words, incident reports that show no corresponding effects on the traffic behavior have no positive contribution to the training of the model and add a potential bias to the accuracy. Therefore, these cases are detected using the method presented in Algorithm 4 and removed from the input data set.

The proposed algorithm describes three different strategies to validate each incident case. A first indicator for an instructive incident report is given through a visible difference between the traffic level at the start and end point of the observation. We define the end of each incident by the first point in time which is not showing an incident report anymore. In other words, the database does not report the end time of an incident. As described in line 3, the average traffic value is taken at the start and end time respectively, measuring the absolute difference between both values. If the difference lies above a certain threshold, the respective incident observation is validated and used as part of the input data (line 4-5).

Moreover, we test the standard deviation over the complete time interval of traffic values. Based on previous data analysis, we argue that most cases of an accident or congestion show a high standard deviation due to the substantial change in traffic behavior. Therefore, the algorithm calculates the standard deviation of traffic for the given time interval, and tests if it lies above a certain threshold (line 8). In the positive case, the observation is validated and again marked to be usable for the input data of the model (line 9).

Finally, in case of the incident showing no clear traffic variation in the previous tests, we provide a last method: Iterating over a time period close to the reported incident

Input: start, end, incident_data
Output: True/False

```
 1: procedure VALID_INCIDENT(start, end, incident_data)
 2:     % Check absolute difference between traffic at start and end
 3:     diff = abs(mean(incident_data[start]) - mean(incident_data[end]))
 4:     if (diff >= absolute_treshold) then
 5:         return True                                  % Valid incident report
 6:     end if
 7:     % Check standard deviation within the traffic over the time
 8:     if (StandardDeviation(incident_data) >= sd_threshold) then
 9:         return True                                  % Valid incident report
10:     end if
11:     % Sweep over the points after the incident
12:     for i=0 to 8 do                                  % Iterate max. 9 times
13:         prev = incident_data[end + i]
14:         next = incident_data[end + i + 1]
15:         if (abs(prev - next) >= absolute_threshold) then
16:             return True                              % Valid incident report
17:         end if
18:     end for
19:     % Sweep over the points before the incident
20:     for i=0 to 8 do                                  % Iterate max. 9 times
21:         prev = incident_data[start - i]
22:         next = incident_data[start - i - 1]
23:         if (abs(prev - next) >= absolute_threshold) then
24:             return True                              % Valid incident report
25:         end if
26:     end for
27:     return False                                     % No valid incident report
28: end procedure
```

Algorithm 4 Incident validation

start time, examining the data for traffic variation. First, the algorithm sweeps over the data entries, starting at the end time of the incident, until it reaches a maximum of 90 minutes after (line 12-17). We defined this time limit, arguing to see any kind of change in traffic within this time frame on a valid incident. Within this iteration, we check for a change in traffic behavior, by calculating the absolute difference between two consecutive data points and compare it against another threshold variable (line 13-15). In case of the condition being successful, the entry is finally validated (line 16). In a contrary case, the sweep is accomplished in the other direction, testing for a change of traffic up to 90 minutes before the occurrence of the given incident (line 20-26). Using this three different methods, we validate each incident to be used as input to our model. In case of every method failing to validate the incident (line 27), the respective data entry is removed from the dataset. It is important to remove these cases from our data, as they lead to a reduction of the model quality. The method has a constant time complexity $O(1)$ on the validation of a single incident, resulting in a total runtime of $O(n)$ to validate n incidents.

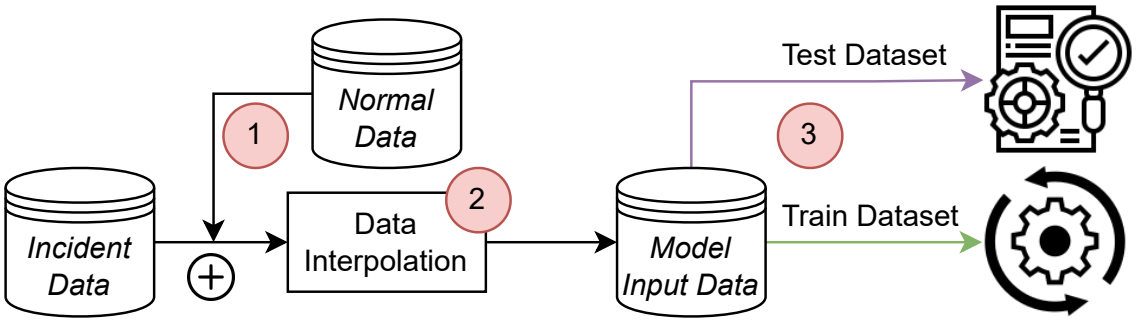


Figure 5.9 Create input data for the incident classification

5.3.1.3 Further Preprocessing

Lastly, to create the final dataset for the classification model, three further data treatments are performed as shown in Figure 5.9:

- 'Normal' traffic data: The addition of non-incident, or 'normal' traffic observations to the input data is essential, because the model requires to classify between accidents, congestion and a normal traffic situation. The procedure is visualized in the first step of Figure 5.9, showing the normal data observations being appended to the incident data. To get the most comparable data, we add observations that are similar in time, weekday and location. Therefore, these reports can accurately reflect a non-incident situation to every incident that is contained in the dataset.

- Data Interpolation: As a result of measurement errors or problems within the data collection, there is a possibility of missing traffic values within the incident duration. Therefore, we use a simple linear data interpolation, equalizing the shape of each data entry (step 2). This linear interpolation approach works by taking two data points, calculating the mean and taking the result as a representation of the missing value. We rely on this simple method as it does not add a lot of overhead and is solely required in very few cases.

- Train-Test-Split: Finally, step 3 shows the process of splitting the total input data into the train and test dataset. The train dataset is used to fit the ML model, allowing it to be generalized. Furthermore, the test dataset is used to evaluate the classification quality. Equally to the estimation in the regression model, the incident cases are randomly sampled and either used within the train or test dataset.

5.3.2 Model Creation

Subsequent to completing all previously explained data preprocessing tasks and creating the train and test datasets, we are now going to describe the creation of our proposed incident classification model.

The defined problem relates to categorizing a labeled set of data, and therefore requires a supervised learning approach. Each incident entry contains multiple data

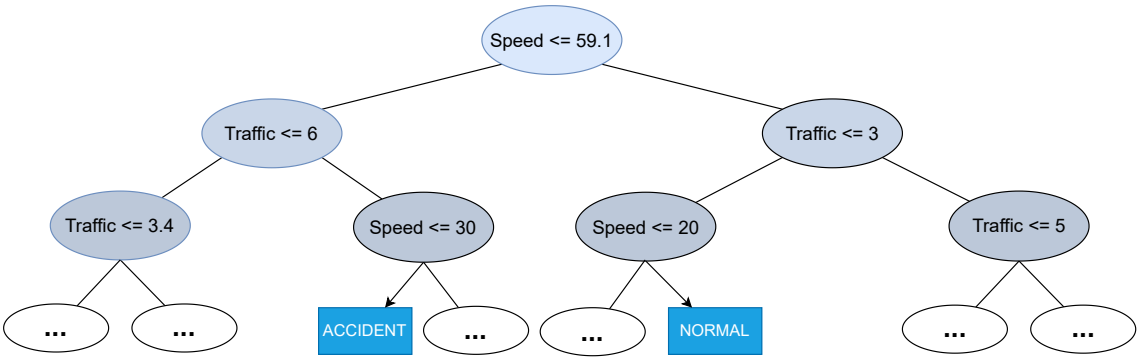


Figure 5.10 Example: Decision tree classifier

features and a label referring to a certain incident type (accident, congestion or no incident). The data features represent a number of time entries (time series) together with the corresponding traffic level, speed (absolute and relative to the maximum allowed speed) and road type, for each data entry. We propose a design for classifying incidents based on two different algorithms, comparing them to see which one is more applicable to the given task: i) Decision tree classifier and ii) k-nearest neighbors (k-NN).

A decision tree classifier is a supervised learning algorithm, capable of deciding between categorical variables using a variety of input features. Figure 5.10 shows an exemplary visualization of a decision tree classifier, with branches representing conditions of the different features, leading a path to the class labels (leaves). We restructured our data from representing a complete time series in a single row to exactly containing one time value per row, to meet the required data structure of the algorithm. An initial evaluation, using the traffic data features on an 80-20 train-test split, resulted in a poor model quality. Furthermore, testing the other two data features could not improve this performance, neither did the approach of using multiple data features at a time. Therefore, we conclude that this model type is not applicable for our use case of time series data and is not further investigated in this thesis.

The next discussed algorithm is a modified version of k-NN, able to work with time series data. In general, k-NN classifies each data entry, based on the label represented by the majority of the k nearest neighboring elements. A representation of the algorithm is visualized in Figure 5.11, showing the $k = 5$ (dashed circle) nearest neighbors of one data sample ('X' in orange). Based on those data entries, the majority of neighboring elements is labeled as *Accident* (red points), which also represents the resulting classification. By changing the distance used in the algorithm, the approach can be adapted to our time series input data. We are using two different distance approaches for the k-NN algorithm:

- DTW: An algorithm to measure the similarity between two time series that are not synchronized. We already explained this approach in Section 5.2.

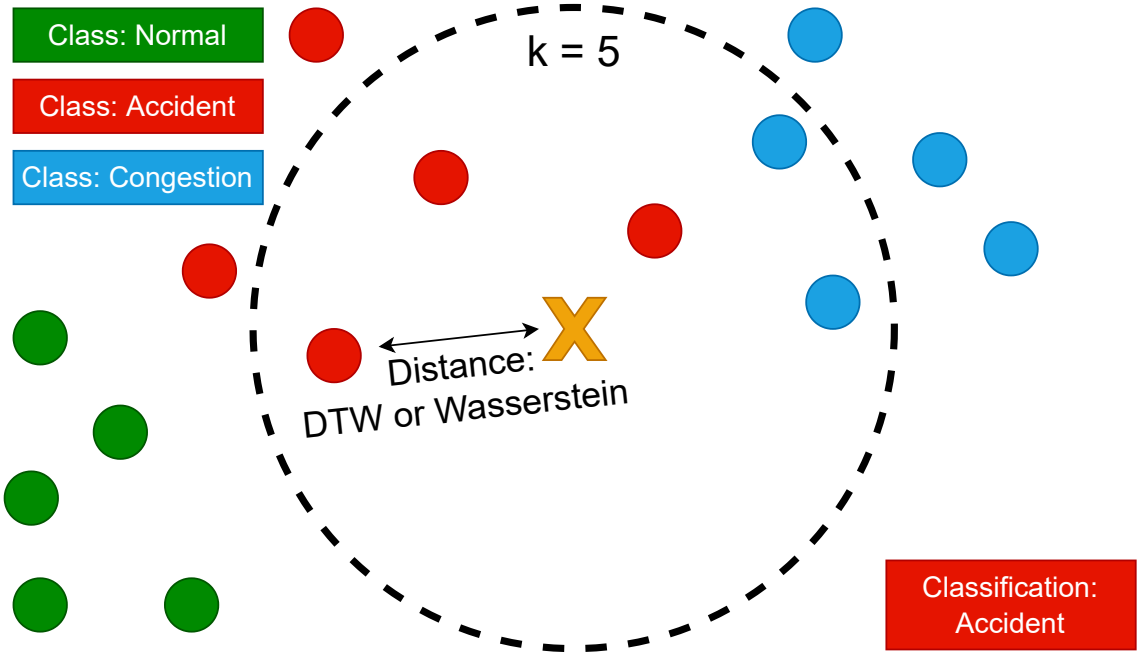


Figure 5.11 Example: k-NN classifier

- Wasserstein metric: A function to calculate the distance between two probability distributions μ and ν, defined in Equation 5.7. Intuitively, it describes the minimum cost of turning one distribution into the other [26].

$$W_p(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{\mathbb{R}x\mathbb{R}} d(x,y)^p d\gamma(x,y)\right)^{\frac{1}{p}} \tag{5.7}$$

Both approaches are applicable on our input data (either as a time series or a distribution) and we are going to evaluate both of them in Chapter 6.

The proposed model is designed based on the k-NN-Implementation *K Nearest Neighbors with Dynamic Time Warping*[4]. More precisely, we are using the *KnnDtw* class and extend it with further functionalities, such as the usage of the Wasserstein metric. The class takes a parameter n (the number of neighbors) and the maximum warping window for the DTW, limiting the number of elements to compare and therefore, offering the possibility to reduce the execution time. Furthermore, the class contains functionalities to fit the data to the model, classify a given input and calculate the DTW distance between two time series.

Furthermore, additional data entries that match the input parameters, are loaded into a dataframe to train the model. As a reminder, we are using parameters such as the start time (original or estimated) and the size of the observed time frame ($\pm$ 90/120 minutes to the start time). The model is initialized with the parameters k (number of neighbors for the k-NN) and *warping_window* (number of entries to compare for the DTW). Furthermore, the data feature (traffic, speed, speed_relative) and metric (DTW or Wasserstein) are defined.

The final step, prior to the model creation, is the usage of data sampling to reduce a given under-representation of certain data classes. Our available input dataset

[4] `https://github.com/markdregan/K-Nearest-Neighbors-with-Dynamic-Time-Warping`

is imbalanced, as the number of accidents is much lower compared to the other data classes. Instead of training the model using an imbalanced set of data, there is the possibility to use either oversampling or undersampling to reduce the data imbalance: i) *Oversampling* describes the process of adding more samples of the under-represented class to the training data, using the information from the already existing data points. ii) *Undersampling* provides the contrary part, removing samples from a majority class. In general, to retain a big amount of available data, oversampling is employed more frequently than undersampling. In consequence, we implemented two different techniques for oversampling[5] and one for undersampling[6] within our classification model:

- *Random oversampling*: A simple approach that randomly duplicates samples from the under-represented class and adds them to the train dataset.
- *SMOTE*: The Synthetic Minority Oversampling Technique (SMOTE) synthetically creates new examples for the minority classes in the given feature space. More precisely, the algorithm selects a random example from the minority class, finds the k nearest neighbors and chooses one of them to create a new data point in between those two data samples. Because we are working with time series data, this method is used on every point within the series, resulting in a new, synthesized sample for the training set.
- *Near-Miss Undersampling*: This undersampling technique balances the dataset by removing entries from the larger classes that have the shortest distance to the smaller classes (based on k-NN).

We are going to provide an evaluation of these various data sampling approaches in Chapter 6 to see the benefits regarding the quality of the model in comparison to using an imbalanced train dataset.

Finally, we explain the creation of our incident classification model, based on the underlying implementation presented in Algorithm 5. The model is created using the previously defined parameters *k*, *warping_window* and *metric* (line 2). Next, the model is trained with the respective data samples from the train dataset (line 3). The test data samples are classified using the *predict* function, returning the label for each data sample and a corresponding probability from k-NN (line 5). Within this function, the algorithm calculates a distance matrix using the specified metric, containing the respective distance between all data samples (line 10). In case of the DTW, the algorithm uses the existing implementation, and regarding the Wasserstein metric we implemented a version for the distance measure in the 1-dimensional feature space. The matrix is sorted by the distance argument (line 11) and the respective labels accordingly (line 12). Using the sorted array of labels, the most common values are determined (line 14), and the corresponding labels are extracted (line 15). Furthermore, the method calculates the probability of the given classifications and returns two arrays containing the classified label and probability for each data entry in the test dataset (line 17). Taking those returned values, we can measure the performance of our model, using different metrics related to ML, which are described in the following chapter.

[5] `https://imbalanced-learn.org/stable/over_sampling.html`
[6] `https://imbalanced-learn.org/stable/under_sampling.html`

Input: X_train, y_train, X_test, k, warping_window, metric
Output: labels, probabilities

```
1: procedure INC_CLASSIFICATION(incident_data, unique_cpaths, overlap)
2:     model = KnnDtw(k, warping_window, metric)        % Initialize the model
3:     model.fit(X_train, y_train)          % Fit the model using the train dataset
4:     % Classify samples from the test dataset
5:     labels, probabilities = model.predict(X_test, k)
6:     % Return arrays with classified labels and respective probabilities
7:     return labels, probabilities
8: end procedure
9: procedure PREDICT(X_test, k)
10:    dm = dist_matrix(X_train, X_test)   % NumPy array of the distance values
11:    knn_idx = dm.argsort(n_neighbors)                    % Sort by distance
12:    knn_labels = y_train[knn_idx]                  % Sort labels accordingly
13:    % Returns array of the modal (most common) values in the given array
14:    result = mode(knn_labels)
15:    labels = result[0]                                % Get most common label
16:    probabilities = result[1] / k                % Get corresponding probability
17:    return labels, probabilities
18: end procedure
```

Algorithm 5 Incident classification

This concludes the design section, which introduced the proposed traffic model, containing the DataFITS framework, to collect and fuse heterogeneous data in a spatiotemporal manner. Furthermore, we provided two data applications related to ITS. First, a traffic estimation using either i) a naive statistical approach or ii) a polynomial regression, and we designed an incident classification, using a modified version of the k-NN algorithm and the Wasserstein metric. The next chapter is going to provide a detailed performance measurement on all parts of the proposed solution of the thesis, evaluating the complete traffic model.

6

Evaluation

This chapter evaluates the contributions of our proposed traffic model that is based on heterogeneous data fusion. We are combining data analysis and characterization to show the benefits of data fusion through the DataFITS. Furthermore, we conduct an extensive evaluation on the proposed traffic estimation and incident classification applications.

6.1 DataFITS

6.1.1 Experimental Setup and Performance

The acquisition process started on December 1st 2021 and covers a time frame of nine months, to ensure a high data availability. It provides heterogeneous information of two German cities, Bonn and Cologne, but we further collected data for another 10 cities in scope of this thesis, ensuring a large database for potential future analysis and applications. The fusion process was performed using a virtual machine running Ubuntu 18.04 on 4 GB of RAM and an Intel Core i5 12600k processor with a clock speed ranging from 3.7 to 4.9 GHz.

DataFITS has a good performance, due to an optimization of the map matching and data fusion methods, as they require the highest amount of computation power. First, the map matching performance is high, according to the C++ implementation of FMM, using multiple optimization approaches [60]. Furthermore, by selecting the best algorithm regarding the scale of the network, and tweaking the input parameters, we were able to achieve an average map matching speed of 500 GPS points per second. Within the data fusion procedure, the algorithm iterates over the matched data, extracts single roads from each data entry and extends the existing data with one row for each of the contained roads. To reduce the memory usage, the creation of the final dataset is implemented by processing chunks of data with a size of 100,000 entries. This prevents the requirement of loading an enormous amount of data into

the memory and instead, reduces it to a manageable size. Based on the hardware used in this experimental setup, the time to process one chunk is around two and a half minutes, resulting in a total computation time of just under six hours for nine months of data, for the Bonn dataset. Furthermore, we processed the data in monthly subsets to reduce the number of entries that have to be processed at the same time.

6.1.2 The Data

The acquired heterogeneous data from Bonn has a total size of 14 GB, whereas the neighboring city Cologne has a corresponding data size of 31 GB.

All data sources covered by the data acquisition are shown in Table 6.1, including the data features and some general statistics about the time frame and number of entries. The '*' symbol highlights the data features we are using from each data source. *Traffic HERE* represents data from the commercial map service HERE[1], that allows a limited collection of traffic information through an API. The limitation regarding this service is given through a maximum number of requests per API-key. It provides the highest amount of data within our experimental setup, given 6,038,496 data entries for Bonn and more than 15 million for Cologne in the given acquisition time frame. The data contains a variety of features, including speed (average speed of observed vehicles), a traffic value (range: 0-10) and a GPS coordinate. The other traffic-related data is provided from the OD service. Similarly to the commercial

[1] `https://www.here.com`

Source	Features		Time Frame	Entries (Bonn)	Entries (Cologne)
Traffic HERE	Speed*	Traffic*	2021-12-01 - 2022-08-31	6,038,496	15,616,800
	StreetDesc*	Cords*			
	Street Info	Current Flow			
	Free Flow				
Traffic OD	Speed*	Traffic*		3,852,144	3,921,552
	Cords*	RoadID			
Incident HERE	Inc. Type*	IncidentID*		949,709	1,752,953
	Comments*	Cords*			
	Road Close	Criticality			
	Verification	Roadway			
Incident BING	Inc. Type*	IncidentID*		1,876,353	2,189,202
	Comments*	Cords*			
	Road Close	Criticality			
Construction OD	Inc. Type*	IncidentID*		957,398	5,215,535
	Desc.*	Cords*			
	District	Adress			
	Blockage	Carrier			
Meteostat (Weather)	Condition*	Temperature		6,576	6,576
	Precipitation	Snow			
Envirocar	Speed*	Consumption*	2014-05-22 - 2022-08-31	9,917	13,443
	CO2*	Throttle Pos.*			
	RPM*	VehicleID*			
	Cords*	Engine Load			
	Temp	Phone GPS			

Table 6.1 Features reported by the different data sources.

alternative, this source reports the most important data features, including speed, traffic (categorical description with four levels) and GPS coordinates. More precisely, the descriptive values from OD regarding both cities offer a categorization of traffic into *no measurement*, *normal condition*, *increased traffic* and *traffic jam*. Comparing the amount of data entries to the commercial service, the OD has a significantly lower quantity, offering 3.8 million data entries for Bonn, and 3.9 million for Cologne.

The incident-related information is collected through a variety of data providers, again offering commercial and free accessibility. Starting with the commercial sources, *HERE* provides the type of incident, an identifier value, further comments on the incident and the GPS coordinates for each data observation. *BING*[2] is the second data source, that is another commercial competitor in the class of map services, distributed by Microsoft and offers the same type of data features. Comparing the number of data entries from each data source shows a superior amount of information that is reported from the BING service. It doubles the amount for Bonn to a total of 1.8 million compared to *Incident HERE*. A similar observation is made comparing the number of data entries for Cologne, with 2,189,202 reports given by *BING* to 1.75 million from the other incident data provider. In contrast to the commercial data sources, OD provides information about constructions or blocked roads due to events that are happening in the city. The Bonn data contains a total amount of 957,398 entries, which is significantly lower compared to more than five million data entries for Cologne. This set of data includes information, like the type of incident, together with an identifier, some additional information and the coordinates of the respective location. We want to point out, that the number of data entries is not referring to the amount of unique construction sites. In contrast to other incident reports (e.g,. accident or congestion), that typically have a maximum duration of a few hours, a construction site is constantly existing over a time frame of multiple days, months or years. This naturally increases the number of data reports within this data source.

In addition to the traffic-related data, we include information about the weather condition, using the *Meteostat* API[3], an open source project offering hourly weather data. The service provides a variety of data features, like temperature, precipitation and snow, but we are mainly interested in the reported weather condition. The API returns one of 27 different weather condition codes, describing the weather state in a very precise way. Therefore, we are mapping this condition to seven more general descriptions of weather (clear, rain, snowy, etc.), increasing the data granularity for our required use case. Acquiring the data at a hourly base results in a total of 6,576 data entries for both cities respectively.

Finally, we collect vehicular data from the open community platform *Envirocar*[4]. As previously discussed, the data provided by this service has a low spatiotemporal coverage, reflected by the number of available data entries for each city. For Bonn, we observed a spatial coverage of 9,917 matching data entries over the complete time frame starting in 2014, with Cologne offering a total of 13,443 entries. The data quality of this service is high, offering vehicular data at a very precise level, including data features like vehicle speed, fuel consumption, CO2 emissions, RPM

[2] `https://www.bing.com/maps/`
[3] `meteostat.net`
[4] `https://envirocar.org`

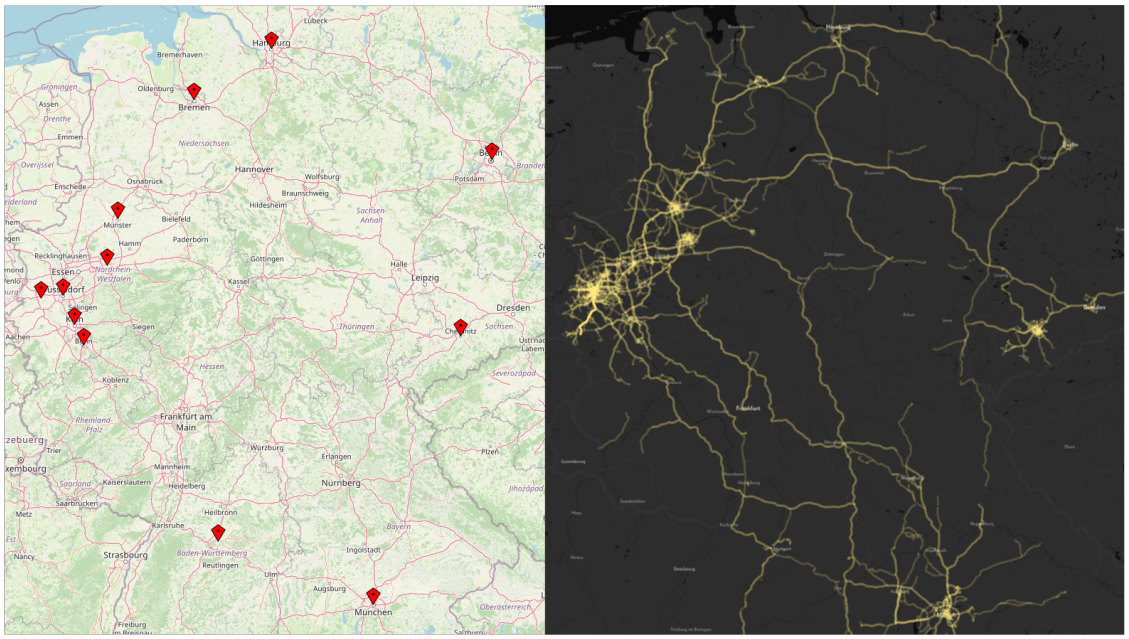


Figure 6.1 Cities of the data acquisition compared to Envirocar data availability

and more. This data is useful to understand the traffic pattern and the driver's behavior.

In general, transportation data can be collected for many cities through a variety of data sources. The main data providers, used within this context, are commercial map services like Google, HERE or Bing. However, these services follow a financial motivation and consequently, there is a limited access per user, which can be increased through paid subscriptions. In contrast to this, there are projects like OpenData (OD) providing open access to big data from multiple information categories to everyone. This creates a collaborative data infrastructure, that can be used by industry, academia and civilian people, matching the idea of a smart city concept. This concept is widely distributed in Germany and officially supported by the government, that also adopted a strategy to improve the existing OD ecosystem [6].

The acquisition process of DataFITS covered 12 of the largest German cities, but provided poor results on data availability, with only two cities (Bonn and Cologne) providing easy accessible OD. Furthermore, six out of the 12 cities offer an OD platform, but currently do not provide access to transportation data (München, Bremen, Münster, Mönchengladbach, Stuttgart, Frankfurt). The remaining four cities did offer access to OD, however it was not applicable in the context of DataFITS. From our research, we conclude that the OD service in Germany requires further development, currently offering a low data availability, that is related to the transportation scenario. Increasing the OD coverage supports the development of ITS applications in a strong way, by offering high quality information to all users, and should be stronger supported by any smart city.

The data collected from the open community platform *Envirocar*, provides access to user generated vehicular data at a very high level of detail. The main drawback regarding this source is the low amount of available data, resulting in a bad spatiotemporal coverage, compared to the other types of collected information. This is

Source	Bonn			Cologne		
	Total Roads	Unique Roads	Fusion Portion	Total Roads	Unique Roads	Fusion Portion
Traffic HERE	684	339	20.94%	2940	1379	27.14%
Traffic OD	581	195	12.04%	914	173	3.40%
Incident HERE	206	53	3.27%	946	370	7.28%
Incident BING	597	256	15.81%	1944	821	16.16%
Construction OD	52	31	1.91%	193	86	1.69%
Envirocar	433	178	10.99%	905	245	4.82%
Overlapping	**567**	**567**	**35.02%**	**2007**	**2007**	**39.50%**
Total		**1619**			**5081**	

Table 6.2 Covered roads by data source

a result of *Envirocar* being a cloud platform for user-collected car data, requiring external hardware (e.g., an OBD adapter) that is associated with additional cost to contribute information. Furthermore, the amount of provided data is increasing at a slow rate, based on our investigation. Currently the service provides 24,300 routes tracked by 1,100 user accounts in total, showing an addition of exactly 1,000 routes over the time frame of 1 year. Figure 6.1 shows a spatial comparison regarding the cities we chose within the acquisition process (left) and the amount of data entries from Envirocar (right). This explains the decision for choosing these cities in Germany to start the data acquisition, in order to increase the probability of also collecting vehicular data.

6.1.3 Data Fusion

To analyze the general benefits of heterogeneous data fusion, we quantify the amount of roads covered by each source in Table 6.2. It contains information about the total number of roads, roads that are solely covered by the respective source and the proportion within the fused data, comparing the cities of Bonn and Cologne. Each data source is contributing a different amount of information to the fused dataset, with *Traffic HERE* having a proportion of more than 20% to the fused data in both cities. However, especially for the incident-related sources, we can observe a major amount of additionally covered roads that are added to the dataset, contributing 20-25% of new information. The number of overlapping road segments reaches 35% for Bonn and 38% in case of Cologne, revealing the potential of further information enrichment by using heterogeneous data fusion. Basically, nearly 40% of roads are provided with information by multiple data sources, that can be used to give a better description on the respective roads. In general, we conclude that using heterogeneous data fusion improves the amount of information compared to only using a single data source. Compared to the source with the highest information quantity, we enriched the 684 roads by 137% to a total of 1619 unique roads covered by the fused data, and 173% for Cologne respectively.

Traffic Level	Street Type	Entries	Traffic	Speed	Speed (rel.)
Low (1)	Main Road	2,670,574 (80.85%)	0.86	37.74	68.60%
Normal (4)	Main Road	527,150 (15.96%)	2.21	31.88	56.64%
Increased (7)	Main Road	61,169 (01.85%)	5.07	17.68	32.11%
Jammed (10)	Main Road	44,047 (01.33%)	9.79	15.37	30.72%
Low (1)	Motorway	2,029,821 (80.50%)	0.41	83.79	86.86%
Normal (4)	Motorway	296,694 (11.77%)	2.11	79.72	80.77%
Increased (7)	Motorway	166,481 (06.60%)	4.91	53.78	57.27%
Jammed (10)	Motorway	28,643 (01.14%)	8.74	27.42	28.67%
Low (1)	Residential	286,654 (93.68%)	0.88	27.04	53.93%
Normal (4)	Residential	285 (00.09%)	2.27	13.76	27.85%
Increased (7)	Residential	16,218 (05.20%)	5.00	13.26	27.05%
Jammed (10)	Residential	3,062 (00.98%)	10.00	3.31	03.36%

Table 6.3 General traffic data statistics

6.1.4 Traffic Data Characterization

Next, we provide the data characterization, starting with the traffic data analysis containing subsections that provide an overview about the traffic statistics and spatiotemporal visualizations. The second part of this section conducts the same characterization using the incident data.

6.1.4.1 Traffic Overview

First, we provide different statistics and visualizations, to understand the general traffic behavior in dependence of certain aspects, like time of the day or the street type. We group the traffic values into four unique levels, representing low traffic (0-1), normal traffic (>1-4), increased traffic (>4-7) and traffic jams (>7-10). Furthermore, due to the high amount of different street types that are contained within the SHP, we provide a grouping of roads on a higher level of granularity: i) Motorway: roads with high speed limits (>100 km/h) or none, ii) Main Road: Fast roads with speed limits from 50 km/h to 100 km/h and iii) Residential: Small roads in a residential area with a speed limit from 30 km/h to 50 km/h.

The statistic provided in Table 6.3 shows the number of data entries for each traffic level on different street types, including the average traffic, speed and relative speed (:= *speed/speed limit*). Noticeably, there is a similar distribution of traffic for the different street types, with the highest proportion of entries given in a low traffic state. However, there is a difference between the distributions related to the different street types. On main roads, nearly 97% of all data entries represent a low or normal traffic level, whereas on motorways the amount is just at 92%, showing a higher proportion of increased or jammed traffic states. The traffic on residential areas is mainly represented in a low (93.68%) or increased state (5.20%). Furthermore, Table 6.3 depicts the traffic and speed (absolute and relative) in dependence of the current traffic level. Considering the information about relative speed, there is a strong

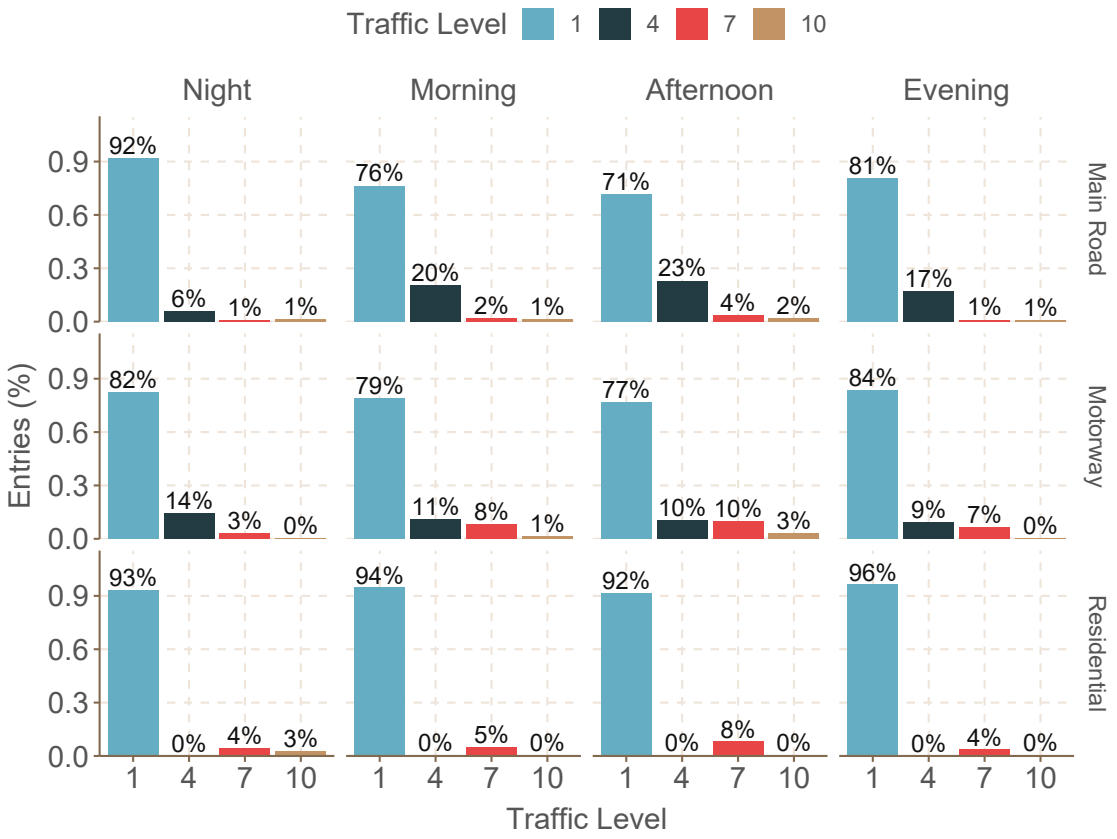


Figure 6.2 Distribution of traffic based on street type and time of the day

variety between the different street types. On a main road, this value decreases from nearly 70% in a low traffic status to 30% in case of a jammed road. A similar behavior is given on motorways, although representing higher relative speeds at the first two levels of traffic. However, in residential areas, the speed reaches a maximum of 54% and significantly decreases to a value of three percent in a jammed traffic status. Therefore, the traffic in Bonn is in a low state most of the time (>80% of all data entries). Higher states of traffic are given in seven percent of all entries, with the proportion of congestion states representing one percent. This depicts a realistic behavior, as congestion generally emerges solely during rush hours or in case of certain incidents. Moreover,Table 6.3 shows a significant reduction of speed for the high levels of traffic (seven or more), especially in case of the residential areas. From these numbers we also notice that the average speed on main and residential roads is close to 50%, resulting in no efficient use of the street with increased fuel consumption, emissions and waste of time.

Figure 6.2 depicts the distribution of traffic levels over the time of the day, based on different street types. We define *Morning* from 05:00-11:59, *Afternoon* from 12:00-16:59, *Evening* from 17:00-20:59 and *Night* from 21:00-04:59. Noticeably, the highest proportion of data entries represents a traffic level of 1, but there is a significant difference in the distribution of traffic levels between the different street types. On main roads, there is a visible increase of higher traffic states throughout the day, compared to the night. A similar observation is made on motorways, showing a difference of 3.1% to 10% on the increased traffic state, from the night compared to the afternoon. On residential streets, the traffic state is in a level of 0-1 for more

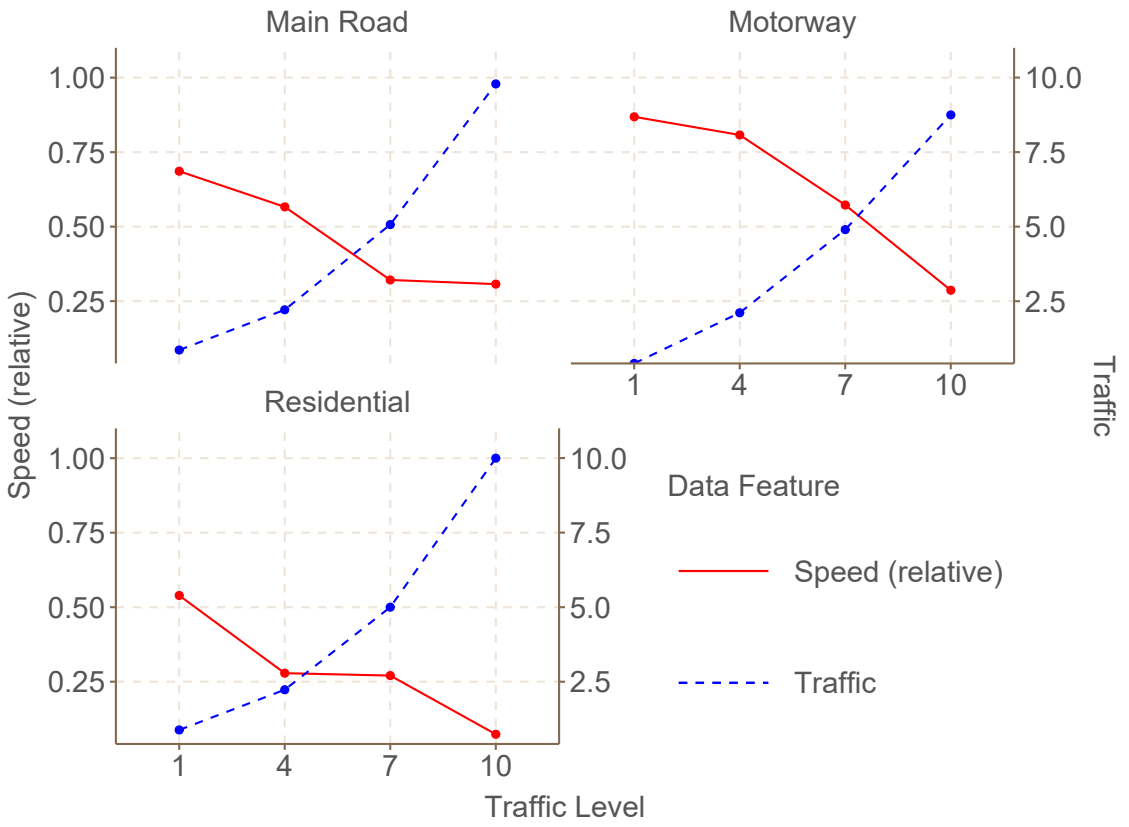


Figure 6.3 Relation of speed and traffic on different street types

than 90% throughout the complete day. Furthermore, the distribution of traffic is more equal on residential areas, compared to main roads and motorways, where it shows a significant change based on the time. Therefore, the traffic level is strongly influenced by the time of the day and the respective street type, demanding different traffic strategies based on these parameters.

Next, we analyze the relation between the relative driven speed and the traffic status for different street types. Figure 6.3 depicts this information, showing the traffic level on the x-axis, relative speed (solid red line) on the left y-axis and the traffic value (blue dotted line) on the right y-axis. There is a strong correlation between both traffic features, with a varying level of dependency according to the different street types. Therefore, in a residential area, the relative speed shows a strong decrease, starting at the second traffic level and reaches nearly zero in the state of level 10. On a main road the speed is decreasing at a much slower rate, resulting in 25% for the highest levels of traffic. Lastly, the motorways are showing relative speed values of more than 50% for nearly all different levels of traffic, except for the congestion state in level 10 which has a relative speed of 25%. This analytical results show the inverse correlation between this two traffic features, but additionally states that the level of correlation significantly differs between the depicted street types.

Concluding this general characterization of traffic data, the city demands different strategies of traffic management for various locations to improve minor problems in the transportation system.

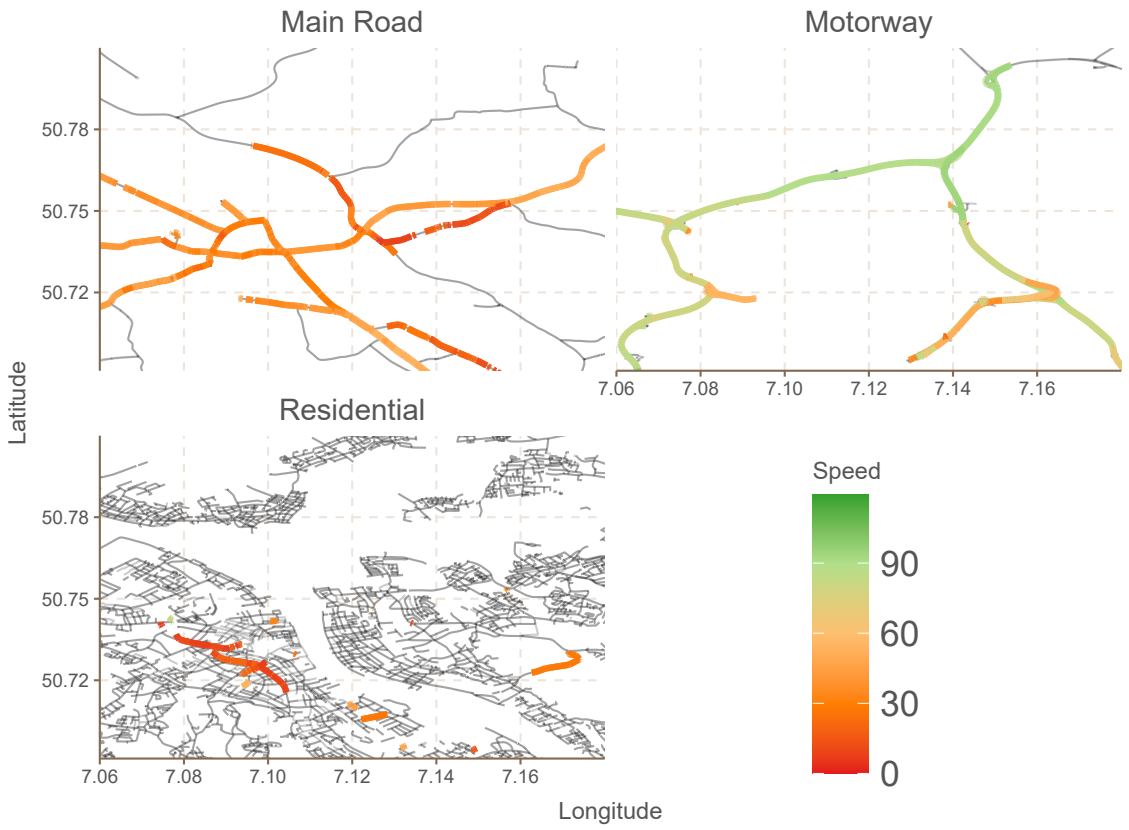


Figure 6.4 Street coverage and relative speed of each road segment

6.1.4.2 Spatiotemporal Visualization

Furthermore, we provide an analysis focusing on the spatiotemporal aspects of the traffic data. Figure 6.4 depicts the data coverage of different street types, in comparison to all existing streets. Furthermore, all covered road segments are highlighted by their average speed. Therefore, motorways provide a close to perfect coverage of data, as it is the primary and most important road type used for transportation. Furthermore, the plot shows that most of the main roads are also covered by our collected data, however, the proportion is significantly lower compared to the motorways. In contrast, only a very small subset of residential streets is covered by the data. Furthermore, Figure 6.4 depicts the speed for every road segment covered by the dataset. Additionally, we provide some statistics about the average speed driven on each street type: Motorway - 81 km/h, Main Road - 36 km/h and Residential 26 km/h. Overall, the speed driven on a large proportion of highways is high and close to the limit, with some occasions of orange colored roads representing a speed of around 60 km/h. Considering the majority of main roads and residential areas, there is a significant lower speed value, especially close to the center of Bonn. In conclusion, the aspect of coverage indicates the importance of each road type to the transportation system. The poor coverage of residential areas can be explained by the large amount of small, unique street segments. Therefore, equipping each of those segments in an urban area with traffic sensors results in high cost and is not profitable, according to the low relevance to the overall transportation system. The visualization of speed per road helps to identify problematic traffic areas that require improvement of the traffic to increase the efficiency of transportation.

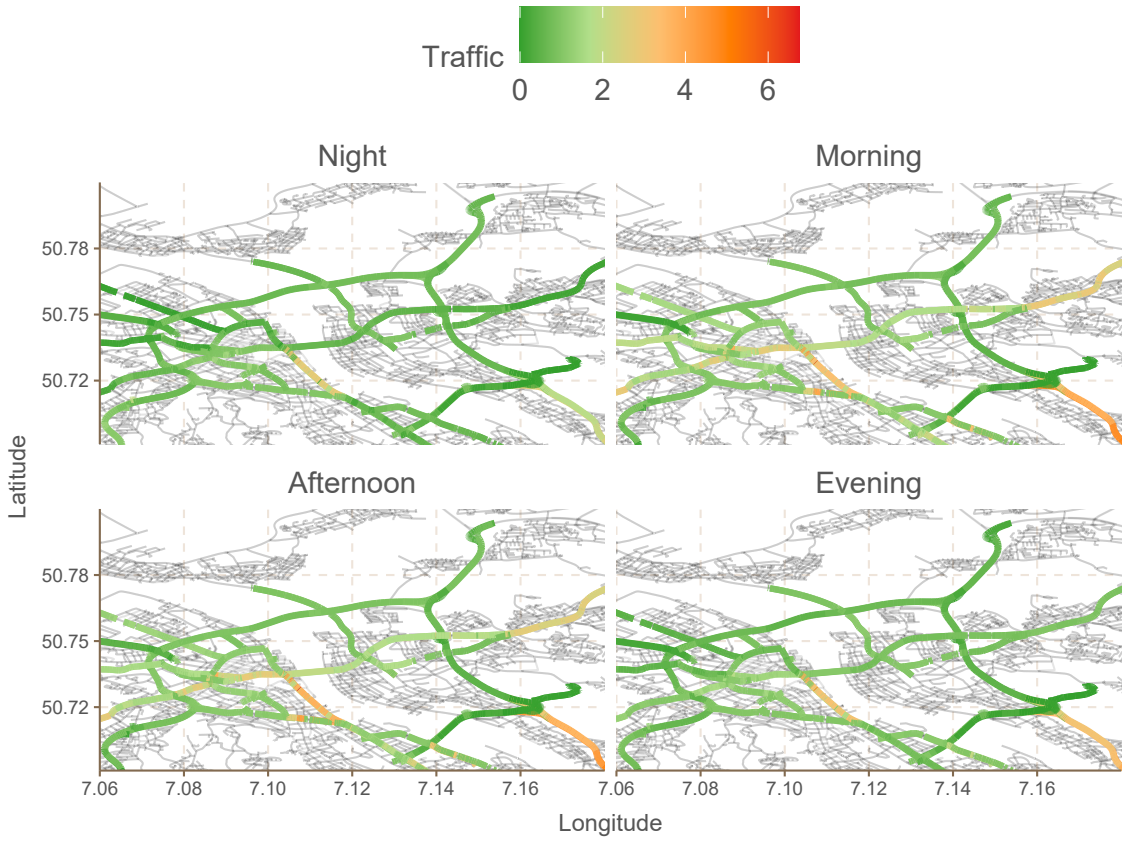


Figure 6.5 Spatiotemporal traffic view

Next, Figure 6.5 presents the traffic on multiple times of the day, showing differences for many areas throughout the day. During the night and evening, the traffic level is low on all locations, except a few roads located near the city center and highway parts that show a slightly increased traffic. However, in the morning and afternoon, multiple road segments show higher traffic values, especially main roads in the center and a few motorway segments. In conclusion, there is a spatiotemporal dependency of traffic values, illustrated by the spatially differing change of traffic values over time. However, the plot shows that on average, most roads in the observed area show a low traffic value, just reaching maximum values of up to five or six in a few locations during the day.

6.1.5 Incident Data Characterization

In this section, we provide an extensive analysis of incident data within the discussed area of Bonn, including four different incident types: Accident, Congestion, Disabled Vehicle and Road Hazard. Similar to the preceding section, we first discuss some general statistics related to the observed incidents and further provide a spatiotemporal analysis, including an investigation about the effect of single incidents to the traffic on surrounding areas.

Incident Type	Street Type	Entries	Dur. (avg.)	Dur. (max.)
Accident	Main Road	4 (01.48%)	10.00 min	10 min
Accident	Motorway	260 (95.94%)	15.92 min	170 min
Accident	Residential	7 (02.58%)	25.71 min	120 min
Congestion	Main Road	326 (07.37%)	12.98 min	90 min
Congestion	Motorway	3892 (88.03%)	19.15 min	730 min
Congestion	Residential	203 (04.59%)	11.43 min	50 min
Dis. Vehicle	Main Road	9 (02.58%)	10.00 min	10 min
Dis. Vehicle	Motorway	328 (93.98%)	15.88 min	210 min
Dis. Vehicle	Residential	12 (03.44%)	18.33 min	60 min
Road Hazard	Main Road	11 (02.49%)	10.91 min	20 min
Road Hazard	Motorway	412 (93.42%)	15.29 min	110 min
Road Hazard	Residential	18 (04.08%)	18.33 min	110 min

Table 6.4 General incident data statistics

6.1.5.1 Incident Overview

Table 6.4 shows the amount of data entries, regarding each different type of incident. Furthermore, the statistic is grouped by street types and contains information about their average and maximum duration. Comparing the different incident types, we notice that the amount of congestion reports surpasses the number of all other reports. This is a result of the fact, that congestion is a re-occurring event, generally emerging due to high traffic. Moreover, the majority of reports occurs on motorways, with a proportion of around 90%. Furthermore, the average and max duration shows a minor variance between the different street types. However, the maximum duration significantly differs between the respective type of incident, reaching up to 730 minutes for a congestion but not more than 210 minutes for the other types. The average duration of congestion exceeds the duration of the other types in general, showing a slightly higher duration. However, this is not observable in residential areas, showing a longer time span of incidents in general, except for congestion situations. Summarizing the table, the highest proportion of incidents is given through congestion reports, mainly occurring on motorways and showing a higher maximum duration compared to the other incidents. Furthermore, we argue that the data imbalance, which is shown by Table 6.4, may reduce the significance of the presented duration values, when comparing different incident types, due to an over-representation of incident values.

The set of bar plots, provided in Figure 6.6, shows the distribution of incidents for different times of the day, separated by each individual street type. Intuitively, the majority of incidents does occur during daytime (morning and afternoon) depicted by the black and red bars. However, there is a noticeable difference regarding the type of incident, as just a minor proportion of congestion occurs during the night or evening ($\geq 6\%$). This cannot be reported on the other street types, showing a wider distribution of each incident throughout the day. This analysis confirms the trend of incidents having a higher occurrence during daytime but shows that there is a significant difference between each unique incident type. A more predictable

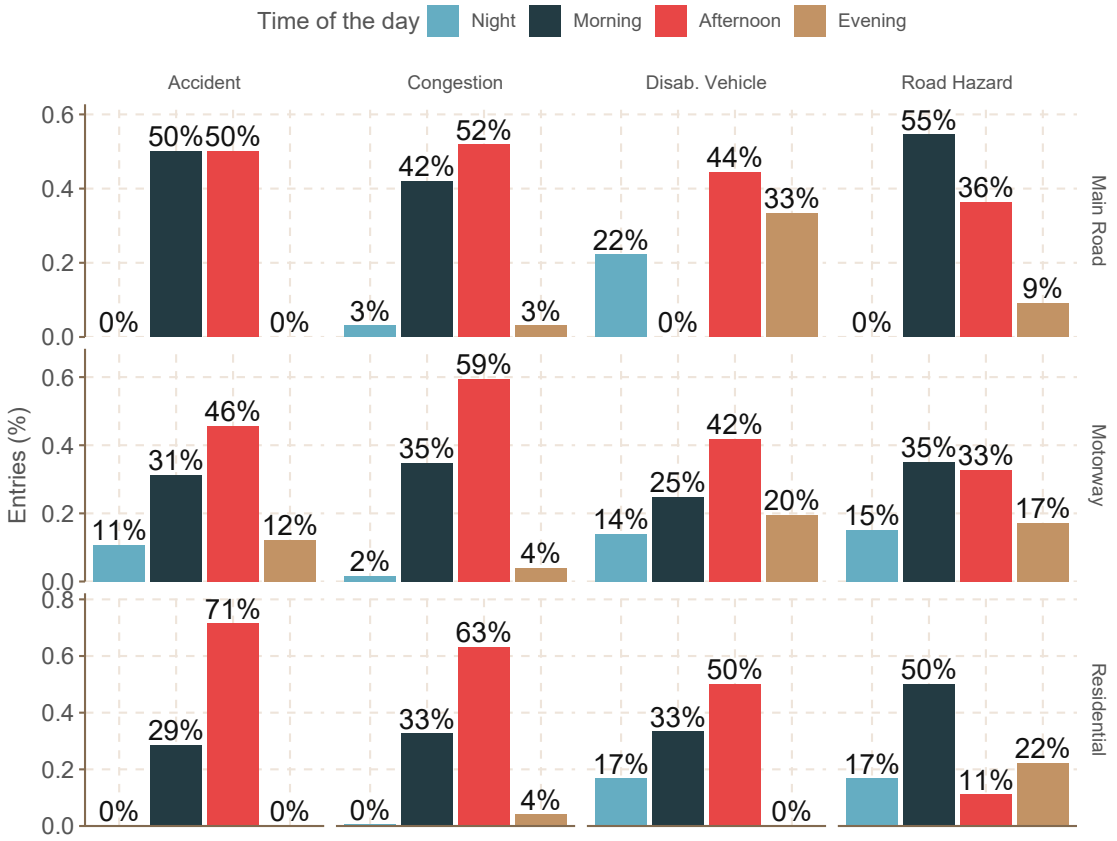


Figure 6.6 Incidents based on road types and time of the day

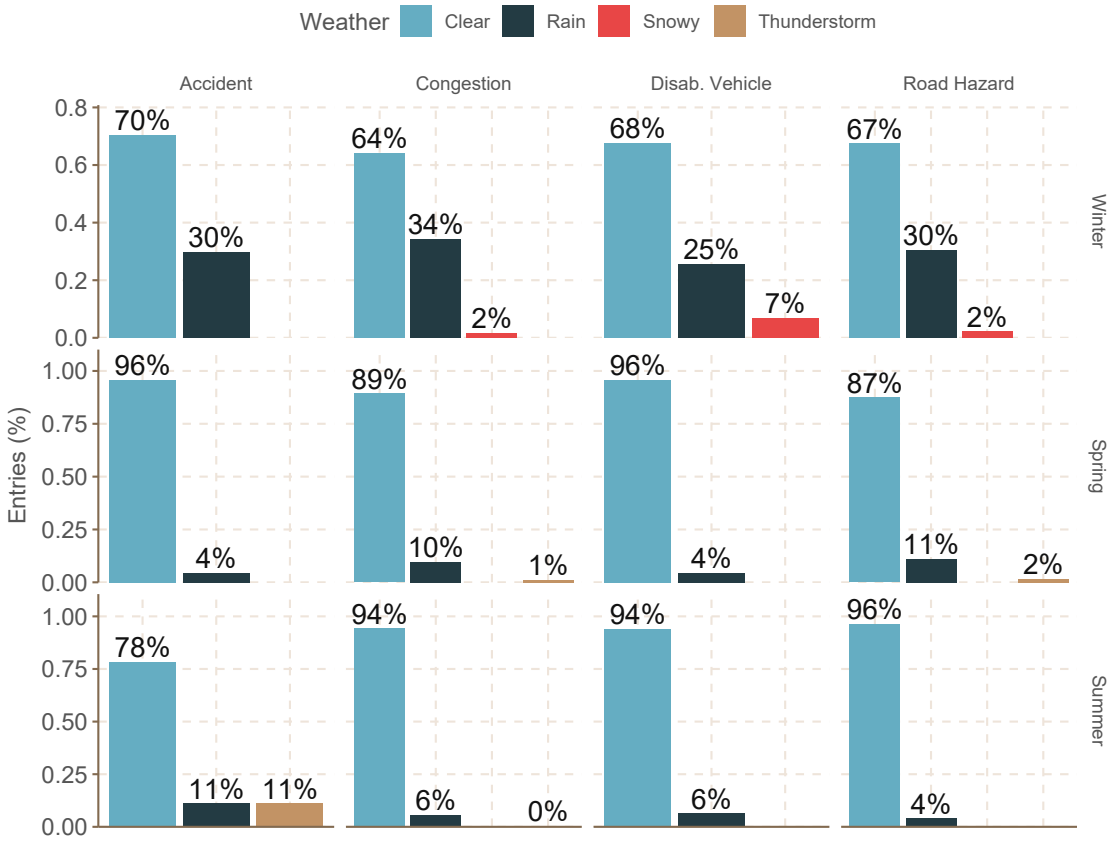


Figure 6.7 Incidents based on weather and season of the year

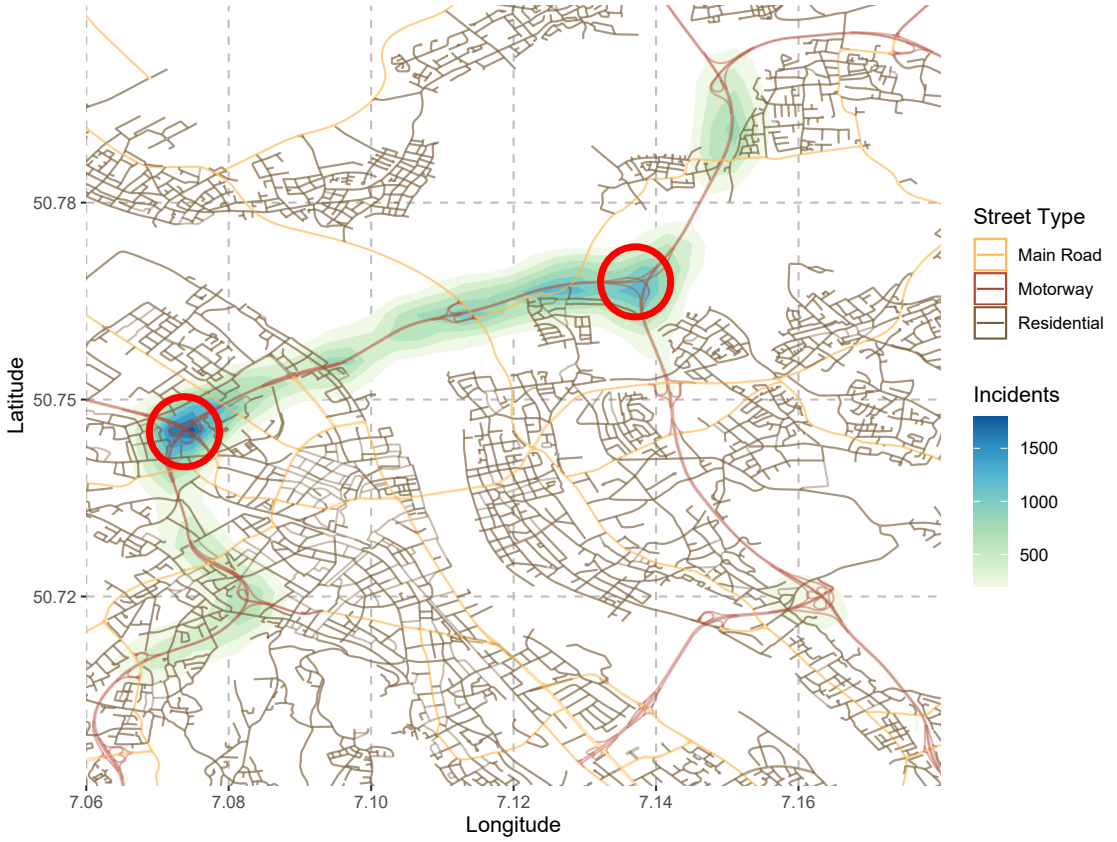


Figure 6.8 Density of all incidents

incident type, e.g., a congestion, is almost solely reported during daytime, whereas the unpredictable events occasionally are reported during the night and evening as well.

Finally, we measure occurrences of incidents based on weather condition and season of the year. The bar plots in Figure 6.7 show the distribution of incident types for three different seasons: Winter, Spring and Summer. A main conclusion drawn from this analysis is made according the 'worse' types of weather (rain and snow) being more present in winter. On average 30% of all incidents are reported during a rainy weather condition and up to seven percent during snow. In contrast to this, about 90% of incidents, during the spring season, are reported on a clear weather condition, with an additional 5-10% of rain conditions and just 1-2% on thunderstorms. A similar observation is made in the summer season, although 11% of all accidents happen during a thunderstorm, indicating a potential correlation as none of the other incidents happened on this condition. In general this shows that there is a minor correlation between the weather and certain types of incidents.

6.1.5.2 Spatiotemporal Visualization

This section shows the occurrence of incidents within a geographical aspect. Figure 6.8 visualizes the density of all incidents in a spatial way, with a unique color scheme to illustrate the individual street types. As visible, most incidents are occurring on the motorway areas, mainly at the two main interchanges *Bonn-Nord* (red circle on the left) and *Bonn-Nordost* (red circle on the right), indicated by the blue level of

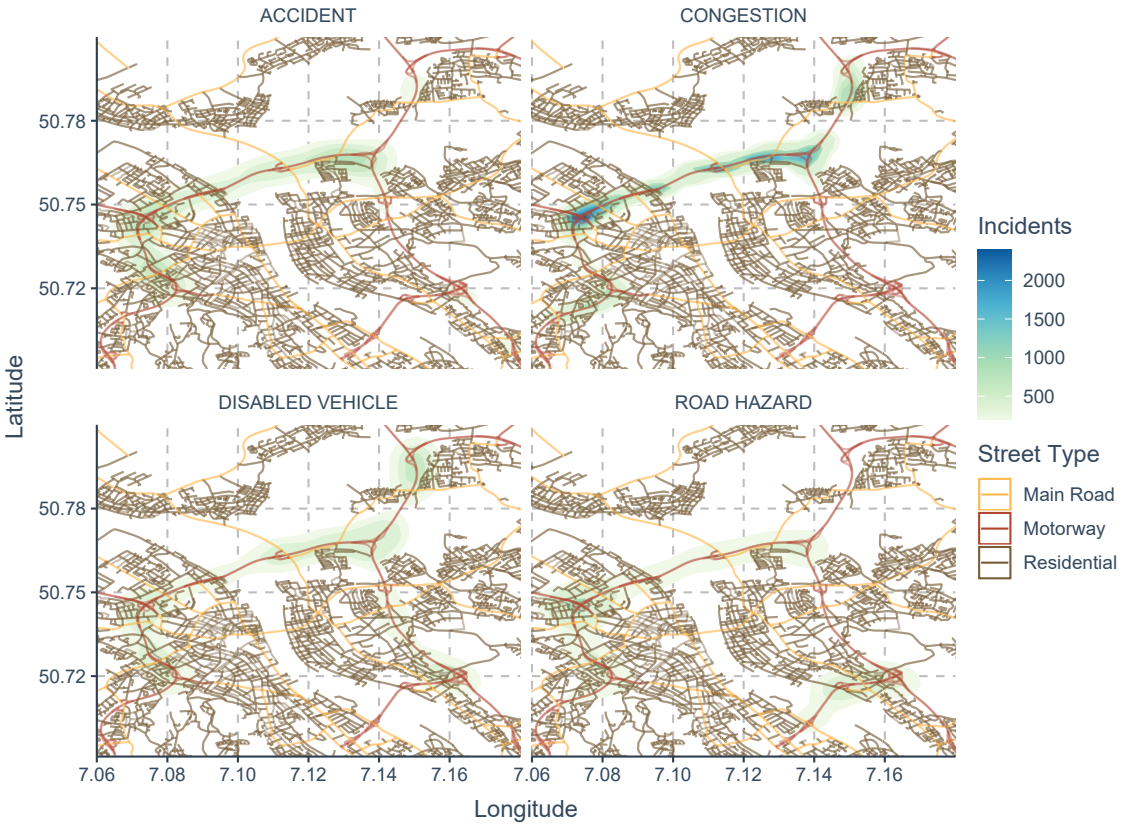


Figure 6.9 Density of different incident types

density that refers to more than 1,500 entries. Both interchanges serve as a crucial part of the transportation system, connecting several different motorways that connect the cities of Bonn and Cologne, daily used by many road users. Therefore, this areas should be considered for potential traffic optimizations, given the significant higher amount of incident occurrences.

Additionally, Figure 6.9 shows the same information with with the additional facet of individual incident types. Therefore, each type of incident has a unique distribution in terms of space, visualized by the minor change in coverage from each respective density layer. However, the majority of incidents occurs on different motorway locations, independent on the type of incident.

Finally, we analyze the effects of an incident on the surrounding area. Figure 6.10 visualizes an accident (marked by 'X'), that happened on a motorway at 17:30. Furthermore, it provides information regarding the traffic level of the surrounding roads over time. There is a visible increase of traffic on the directly connected areas surrounding the accident location. The increase starts at 17:20, just before the incident is reported by the respective source. The condition can be observed for a total of 30 minutes, slowly going back to a lower traffic state at 17:50. Therefore, the given accident has an impact on the traffic behavior of many neighboring roads, especially visible in the time of 17:30 and 17:40. Within this time, a major proportion of the connected motorways shows a significantly higher traffic value, ultimately correlating with the observed accident. Conducting this type of analysis on different incidents shows a variety of different traffic behaviors related to the unique incidents. The majority of accidents and congestion reports provided a similar behavior as

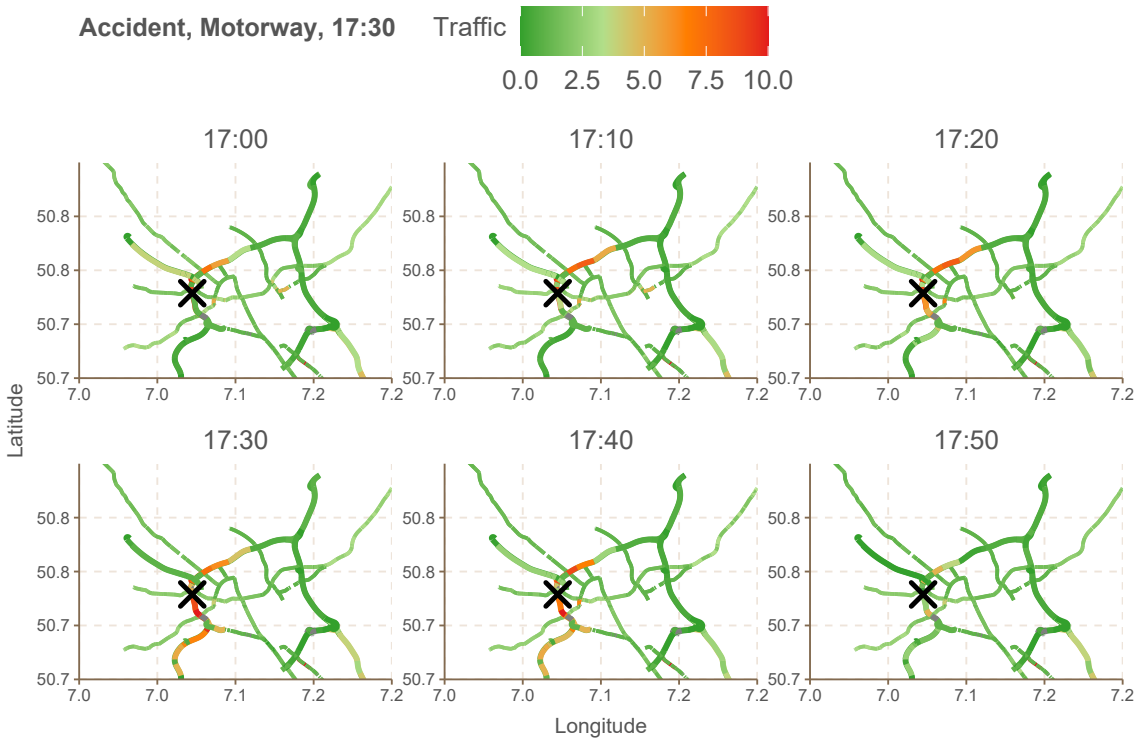


Figure 6.10 Incident effect on traffic behavior

shown in Figure 6.10. However, the inspected congestion cases did show a smaller impact on the surrounding areas in general. This shows the correlation of traffic features in order to classify incidents based on traffic patterns and motivates the creation of our presented incident classification model.

This specific analysis concludes the data characterization. We show multiple insights regarding the data availability and information enrichment as a result of the heterogeneous data fusion. Using the fused dataset, we are able to conduct a variety of analysis tasks, providing insights on multiple levels of detail for both, traffic and incident data. We can use this data characterization to provide a better understanding of the traffic behavior on an urban area and further support the development of applications in the context of ITS.

6.2 Traffic Estimation

This section evaluates the performance of the proposed traffic estimation application. We start by providing a detailed performance measurement on the model that uses naive statistics, and continue with the regression approach. Finally, we compare the performance of both models on two different datasets.

To measure the performance of the designed traffic estimation models, we calculate a variety of effective and commonly used performance metrics, which are explained in the following:

- Coefficient of Determination (R^2): The metric is defined by

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{6.1}$$

  and represents the proportion of variance between the true value y and predicted value $\hat{y}$. It can be used to measure the performance of the proposed model estimation. The formula calculates the squares of residuals (numerator) and divides it by the total sum of squares (denominator). An optimal estimation would be represented by $R^2 = 1$, a score of 0 is given to a model that always predicts the average value of y. The score can also be represented through a negative value, as the model can be arbitrary worse.

- Mean Absolute Error (MAE): The MAE is an error metric that describes the sum of absolute errors between the real values and estimations from the model, with a desired value of 0, defined by:

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{6.2}$$

- Root Mean Squared Error (RMSE): The RMSE describes the root of the Mean Squared Error (MSE), a measurement for the average squared distance between the estimated values by the model and the real values within the dataset. It also has a desired value of 0 and is defined by:

$$RMSE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{6.3}$$

6.2.1 Naive Statistical Model

First, we discuss the results of our initial traffic estimation application, based on naive statistics. Initially, we evaluated the statistical approach using a dataset containing two months of traffic data, comparing the estimation quality on four different data samples: i) Fused Data with Grouped Regions based on correlation (Fused GR) ii) Fused Data from a Single Region (Fused SR) iii) Raw Data (RAW) from *Traffic HERE* and iv) RAW from *Traffic OD*.

Figure 6.11 presents four examples that compare the estimation accuracy, using the different data samples, to the ground truth. *Example 1* shows that using Fused GR (red line) and Fused SR (orange line) provides estimated values close to the ground truth (black line). This indicates that including data from correlating regions slightly improves the traffic estimation in this case. Moreover, the benefits of heterogeneous data fusion (red and orange line) suggest a higher accuracy compared to the samples using raw data (blue and green). Similarly, *Example 2* shows that the fused data achieves the best estimation result with Fused GR and Fused SR being the same values, due to no available similar regions. The RAW samples are worse, indicating that the usage of fused data can provide a benefit to the accuracy of the estimation. The results presented in *Example 3* demonstrate an opposite case, where the data fusion leads to a worse estimation compared to using RAW-HERE. The

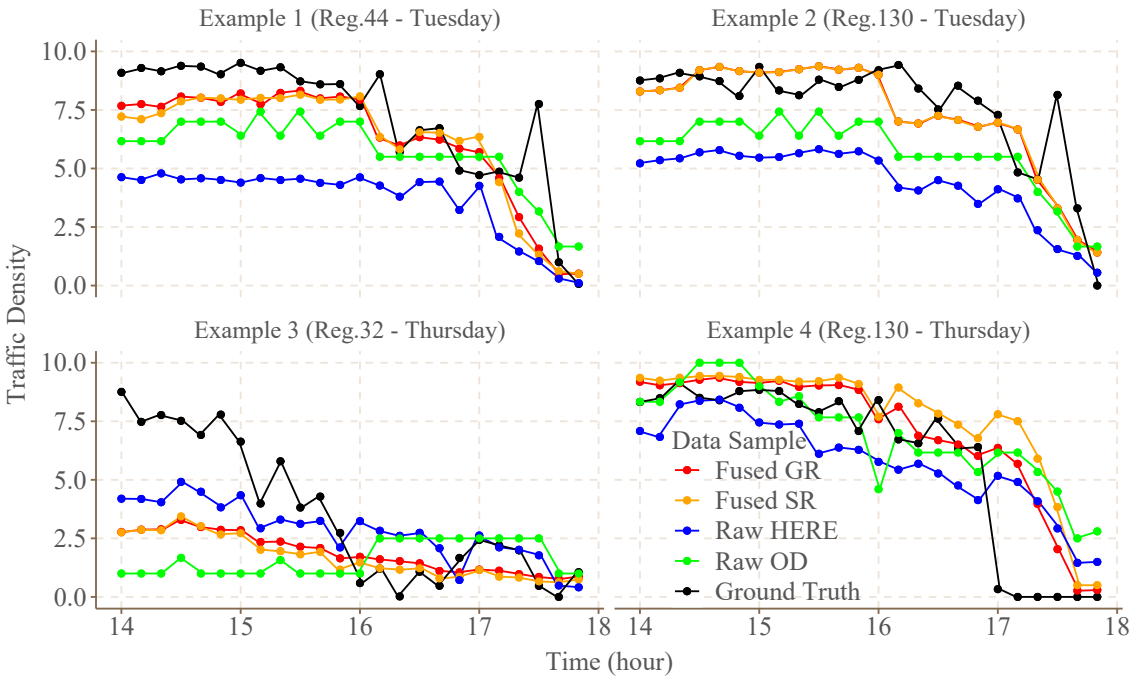


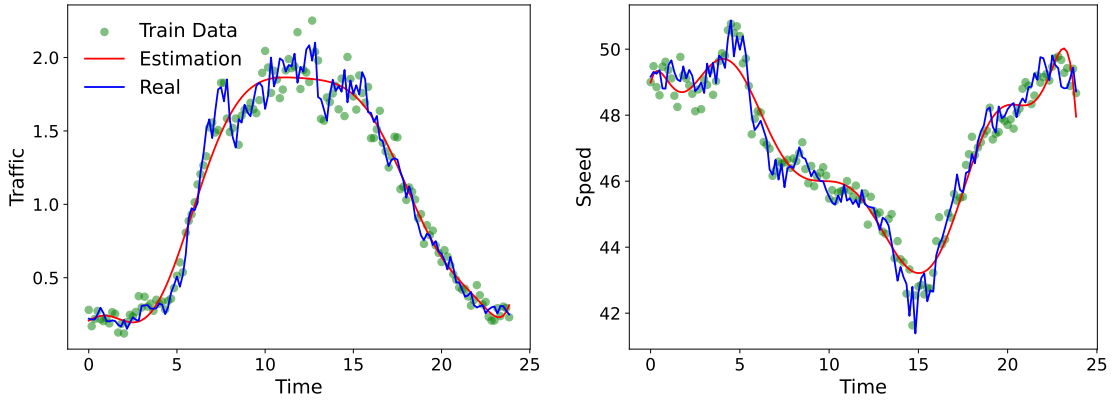
Figure 6.11 Estimations using the naive statistical model

plot shows that the estimation based on RAW-OD is very coarse and has a strong bias, compared to the ground truth. Fusing the data from OD and HERE improves the estimation accuracy, but it remains lower compared to using only data from RAW-HERE. *Example 4* shows that all estimated values are close to the ground truth, with Fused GR achieving the overall best result. In conclusion, both fused samples have a better accuracy compared to RAW-HERE and especially RAW-OD, furthermore revealing the benefits of our proposed approach.

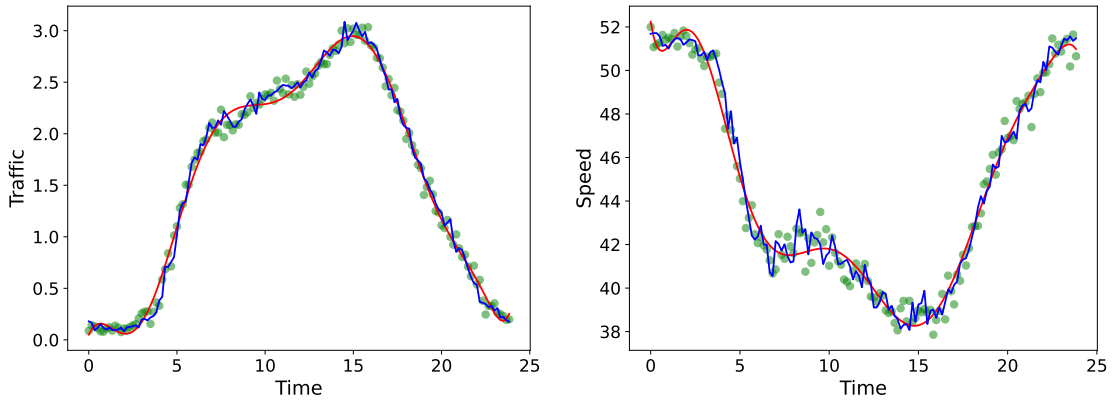
Data Sample	R2	MAE	RMSE
Fused GR	-0.11	1.19	1.50
Fused SR	-0.14	1.21	1.52
RAW-HERE	-0.25	1.31	1.58
RAW-OD	-1.36	1.67	2.06

Table 6.5 Performance of the naive statistical model

Table 6.5 shows the evaluation results on all areas. Therefore, using the Fused GR data sample achieves the overall best performance. It has a minor advantage compared to Fused SR, showing a better R2 score of -0.11, compared to -0.14 and slightly better error measures. In comparison to RAW-HERE, there is a stronger improvement, achieving a R2 score of -0.25 and slightly higher error metrics. Compared to using RAW-OD, our approach provided significantly better results, increasing the R2 value by 1.25 and both error metrics by 0.48 and 0.56 respectively. However, the overall quality of the presented model was not particularly high, which motivated us to further develop a regression based model.



(a) Estimation and real data for traffic (left) and speed (right) on Region A



(b) Estimation and real data for traffic (left) and speed (right) on Region B

Figure 6.12 Estimations using the polynomial regression

6.2.2 Polynomial Regression Model

This Section evaluates the traffic estimation using a ML-based polynomial regression model. The configuration, presented in Table 6.6, is the same as in the naive statistical approach, but further includes a polynomial degree and a larger set of input data (nine months). The model can estimate traffic for each of the 181 areas contained in the dataset, referring to 7,324,198 data entries. The overlap threshold value for intersecting regions is set to 0.5, and the corresponding thresholds for the correlating areas are given by 0.25 for DTW and 0.90 for Pearson Correlation. The regression is based on a polynomial degree of 10.

Total Entries	Areas	Overlap	DTW Thresh.	Corr Thresh.	Pol. Degree
7,324,198	181	0.5	0.25	0.90	10

Table 6.6 Setup of the traffic estimation model

Figure 6.12 depicts four examples of our proposed traffic estimation for two different areas. In Figure 6.12 a), the left plot shows the estimated regression line (red) on the set of training data (green points). We want to mention, that the points from the train dataset are grouped to 144 points, each representing a mean value, for better visibility instead of showing >10,000 data points. The blue line depicts

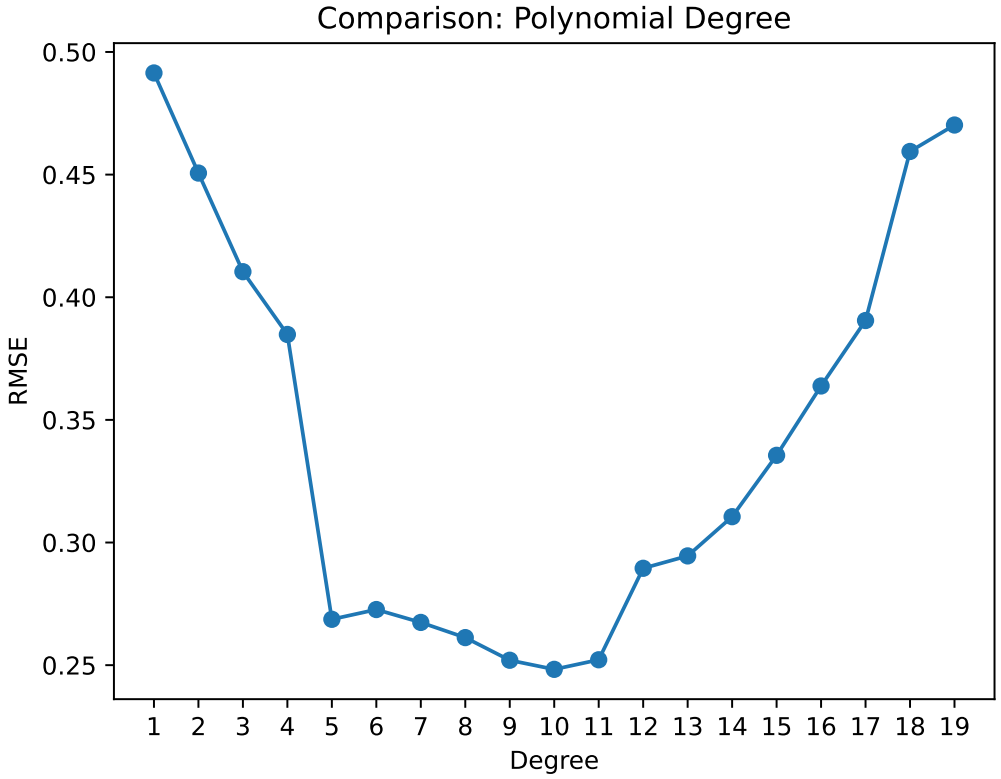


Figure 6.13 RMSE for different polynomial degrees

the test data, representing a real world traffic data observation. Noticeably, the estimation is showing a high precision on both, traffic and speed data, indicating a great traffic estimation. The second area depicted in Figure 6.12 b), shows a similar result, scoring an even higher performance and provides a closer estimation to the ground truth.

The regression model can be setup using the following parameters: Data feature (traffic, speed or speed relative), weekday, weather and street type. Furthermore, we can vary the polynomial degree and train-test-split to achieve an optimal performance. To find the best possible polynomial degree for the underlying regression model, we conducted a variety of tests with different values for d in range of 1 to 20. The result of this is presented in Figure 6.13, showing that using a degree of $d = 10$ achieves the lowest RMSE value and therefore, the highest accuracy. Furthermore, we use a train-test split with 40% of training and 60% of test data, according to the performance results depicted in Figure 6.14, showing a minor performance advantage compared to most other combinations of the train-test-split. Moreover, there is visible decrease in performance when using 70% of training data or more.

Table 6.7 presents the results of a further exploratory investigation, regarding the best model configuration. First, it shows the performance measures of our model using the full set of data, for each data feature (first line). In total, the model can achieve a high R^2 score of 0.84, using speed or relative speed, but also performs good using the traffic data feature. Furthermore, both error measures are low for each respective data feature, represented by values below 0.10. Furthermore, Table 6.7 evaluates the performance of using different street types and weather conditions,

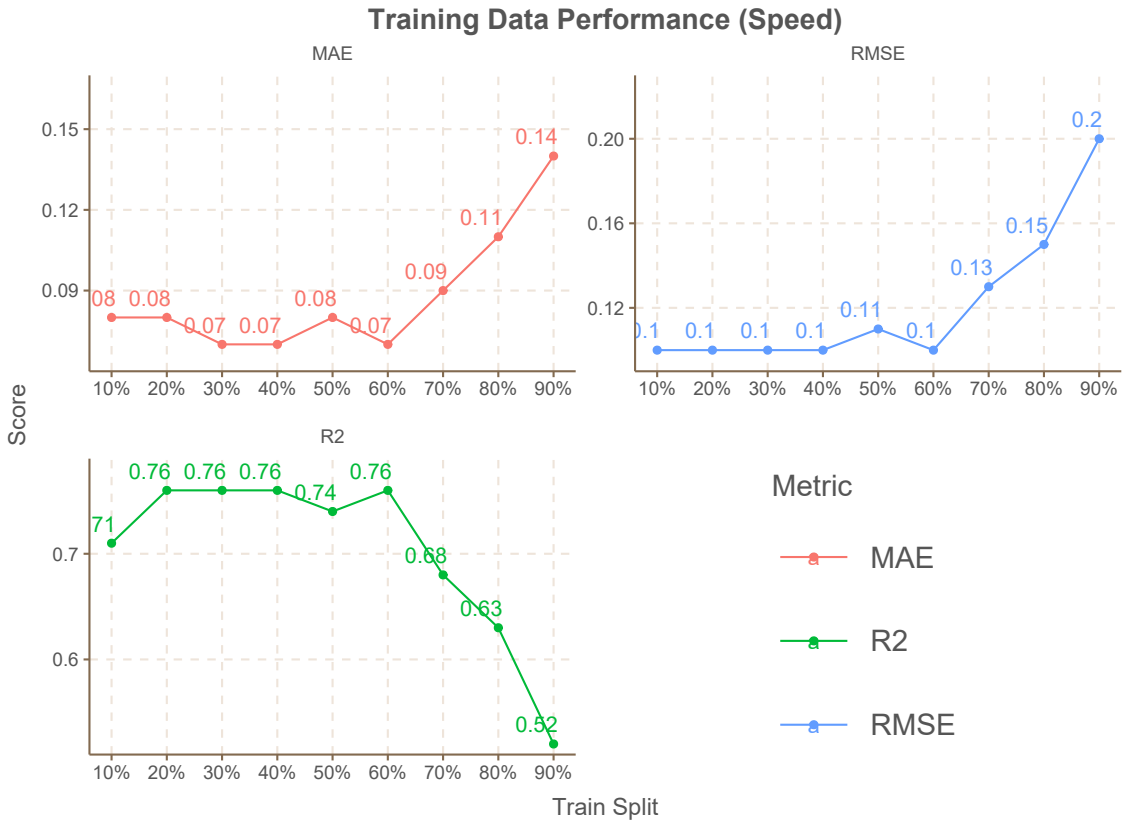


Figure 6.14 Performance for different sizes of training data

comparing it to the general performance (first line), with the green color showing an improvement, black no change and red a decrease. We also included some indicators (represented by the arrows and circles), that show the general performance of the respective set of input data. The model achieves the best performance estimating traffic on main roads with no specified weather condition, with R^2 scores of 0.91 using the speed data features and 0.81 on traffic. The general performance on motorways is a bit lower compared to both, main roads and residential areas, reaching an R^2 score of up to 0.84, therefore marked by an orange circle. We conduct the same measurement using a variety of different weather conditions, with the results shown in the last three rows of the table. Noticeably, the performance is significantly lower estimating traffic in case of rain, and especially snow, indicated by the red arrow. This is due to the low amount of traffic observations that happened during rain or snow, significantly reducing the train dataset. Therefore, the estimation uses many interpolated data points, instead of real values, reducing the overall quality. However, the model is still capable of providing an accurate estimation for the clear weather condition. Based on this exploratory investigation, we argue that the model achieves a great performance on most input data parameters, however, using separate models to estimate the traffic based on a specific road type achieves the overall best results. In contrast, creating a model based on the weather conditions rain and snowy reduces the quality of our model.

Finally, we evaluate the usage of additional data from correlating areas in the train dataset. To ensure that we are just considering data from strong correlating areas at a similar value level, we define the following thresholds for the correlation and

	Street Type	Weather	Data Feature	R2	MAE	RMSE
↑	All	All	Traffic	0.76 ±0.00	0.07 ±0.00	0.09 ±0.00
			Speed	0.84 ±0.00	0.06 ±0.00	0.08 ±0.00
			Speed (rel.)	0.84 ±0.00	0.06 ±0.00	0.08 ±0.00
●	Motorway	All	Traffic	0.68 −0.08	0.08 +0.01	0.10 +0.01
			Speed	0.71 −0.13	0.08 +0.02	0.11 +0.03
			Speed (rel.)	0.71 −0.13	0.08 +0.02	0.11 +0.03
↑	Main Road	All	Traffic	0.81 +0.05	0.06 −0.01	0.08 −0.01
			Speed	0.91 +0.07	0.05 −0.01	0.07 −0.01
			Speed (rel.)	0.91 +0.07	0.05 −0.01	0.07 −0.01
↑	Residential	All	Traffic	0.75 −0.01	0.07 ±0.00	0.10 +0.01
			Speed	0.91 +0.07	0.05 −0.01	0.07 −0.01
			Speed (rel.)	0.91 +0.07	0.05 −0.01	0.07 −0.01
↑	All	Clear	Traffic	0.74 −0.02	0.07 ±0.00	0.09 ±0.00
			Speed	0.82 −0.02	0.06 ±0.00	0.09 +0.01
			Speed (rel.)	0.82 −0.02	0.06 ±0.00	0.09 +0.01
↓	All	Rain	Traffic	0.55 −0.21	0.08 +0.01	0.11 +0.02
			Speed	0.69 −0.15	0.08 +0.02	0.11 +0.03
			Speed (rel.)	0.69 −0.15	0.08 +0.02	0.11 +0.03
↓	All	Snowy	Traffic	0.23 −0.53	0.09 +0.02	0.15 +0.06
			Speed	0.28 −0.56	0.11 +0.05	0.16 +0.08
			Speed (rel.)	0.28 −0.56	0.11 +0.05	0.16 +0.08

Table 6.7 Performance of the regression model on various streets and weather conditions

DTW: $th_{cor} = 0.90$ and $th_{dtw} = 0.25$. For the naive statistical approach, the usage of correlation showed minor improvements to the models performance, e.g., increasing the overall R^2 score from -0.14 to -0.11 and reducing the MAE by 0.02. However, for the regression model we achieve no improvement on using this additional data from similar areas. In contrast, the approach decreased the overall performance by five percent on average. This is due to the large size of available data for each area. Therefore, including more points to the input dataset adds a minor degree of bias when using the regression model. We argue that the use of data from correlating areas is beneficial for the naive statistical model on a smaller dataset, but is not applicable to a regression algorithm used on a large train dataset.

Based on this initial evaluation we conclude that our proposed traffic estimation approach, based on a polynomial regression, is achieving good results and is useful to predict the traffic for single areas using different parameters. However, due to the lack of data on certain weather conditions (e.g., snow) the model performance is significantly lower, due to the requirement of data interpolation. Furthermore, the usage of additional data from correlating regions could not benefit the performance of the model like it did for the naive statistical approach.

6.2.3 Comparison

This section conducts a detailed comparison between the two presented models for traffic estimation on two different sets of data: i) two months of traffic data (this

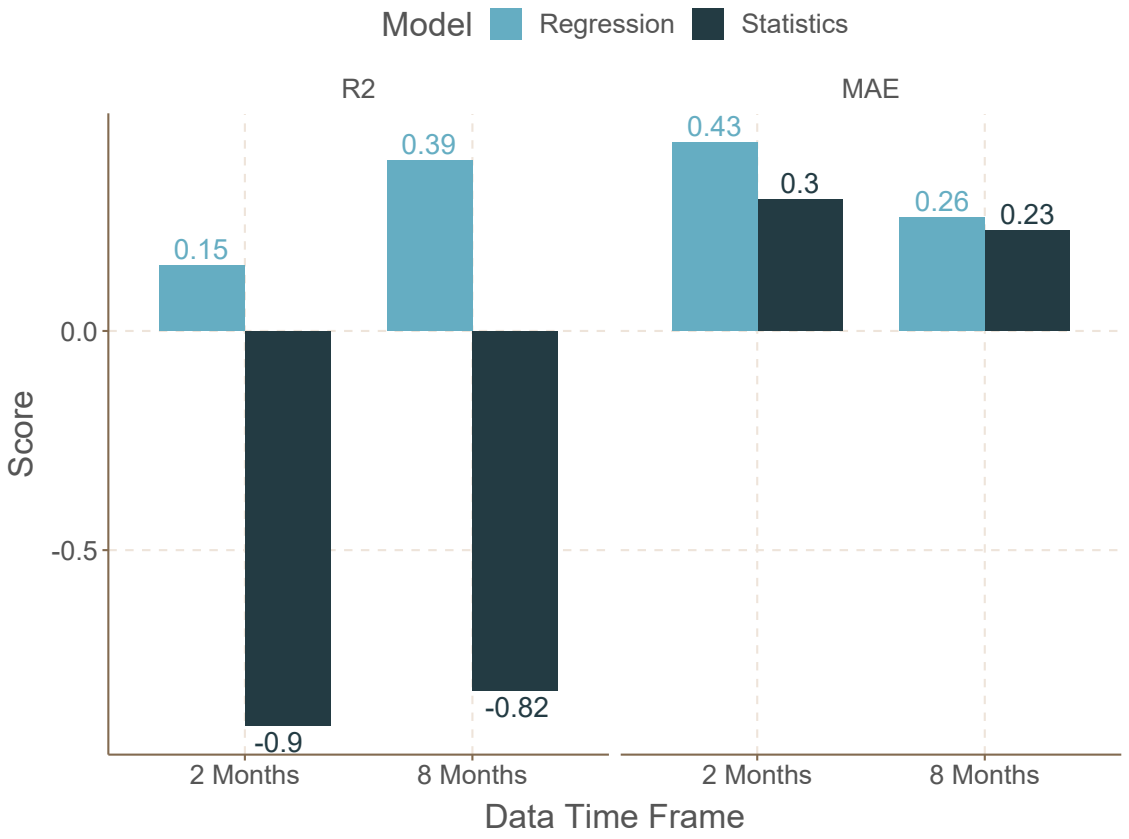


Figure 6.15 Comparison: Statistical model and polynomial regression

is the dataset used on the initial evaluation of the statistical model), from July to August 2021 and ii) nine months of traffic data, from December 2021 to August 2022

To compare both models, we use a train-test-split of 80-20 and estimate traffic values on all areas for a single weekday. Figure 6.15 shows a comparison between the corresponding models on both datasets, evaluating the R^2 and MAE scores. We start by comparing the performance on the initial dataset of two months, showing a significant performance difference regarding the R^2 score. The naive statistical approach provides an overall score of -0.90, whereas the regression model achieves a significantly higher score of 0.15. The MAE value is more close for both models, however, the statistical model shows a slightly better performance. We conclude, that the polynomial regression outperforms the statistical model, due to the significantly better R^2 score. Equally, the regression achieves a much better performance, comparing both models on the nine months dataset. Moreover, we observe that the ML model could significantly improve the performance on the larger dataset, improving R2 from 0.15 to 0.39 and MAE from 0.43 to 0.26. Similarly, the statistical approach showed a minor improvement within both metrics.

In conclusion, the presented traffic estimation approach achieved a great performance using the ML-based regression approach. The naive statistical model is applicable to estimate the traffic for most areas, with a low overall error and good R^2 measurements. We showed that using fused data together with information from correlating areas could improve the quality of the model. However, the overall performance on the total set of data is not ideal for this approach, providing an R^2 score slightly below average and high error metrics. Using the more advanced regression model,

we achieve a much higher overall accuracy and provide a robust traffic estimation application, reaching an R^2 score of up to 0.91 and very low error metrics. While the usage of additional data from similar regions provided a benefit to the naive statistical approach, it added a small bias to the regression model, resulting in an overall minor performance decline.

6.3 Incident Classification

In this section, we provide a detailed evaluation of the proposed incident classification. The designed model classifies three different classes of incidents (Accident, Congestion and Normal), but we are also providing a binary classification (Incident or No Incident), to compare the performance of both approaches. All results presented in the next two subsections refer to the multi-class model. The application utilizes a k-NN algorithm, using two different methods to compare time series and distributions (DTW and Wasserstein) and is evaluated using a variety of classification metrics:

- Confusion Matrix: Represents the instances in an actual class (rows) and the predicted class (columns), visualizing the amount of correct and wrong classifications. The results are categorized in the following way:
  - True Positives (TP): Test samples correctly classified to their class
  - False Positives (FP): Test samples wrongly classified to their class
  - False Negatives (FN): Test samples correctly classified to another class
  - True Negatives (TN): Test samples wrongly classified to another class
- Accuracy: Represents the proportion of correct classifications to the total number of cases.
$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{6.4}$$
- Precision: Represents the proportion of test cases that were classified correctly to the specific class. It should be used in case of the requirement to be very sure about the classification.
$$\text{Precision} = \frac{TP}{TP + FP} \tag{6.5}$$
- Recall: Represents the proportion of test cases from the actual class that are classified correctly. It can be used in case of a model that tries to capture as many correct classifications as possible.
$$\text{Recall} = \frac{TP}{TP + FN} \tag{6.6}$$
- F1 Score: Represents the harmonic mean between precision and recall to indicate the balance of both values. A model can reach a high accuracy but have a low precision and/or recall at the same time. The F1 score can be used to identify those cases.
$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6.7}$$

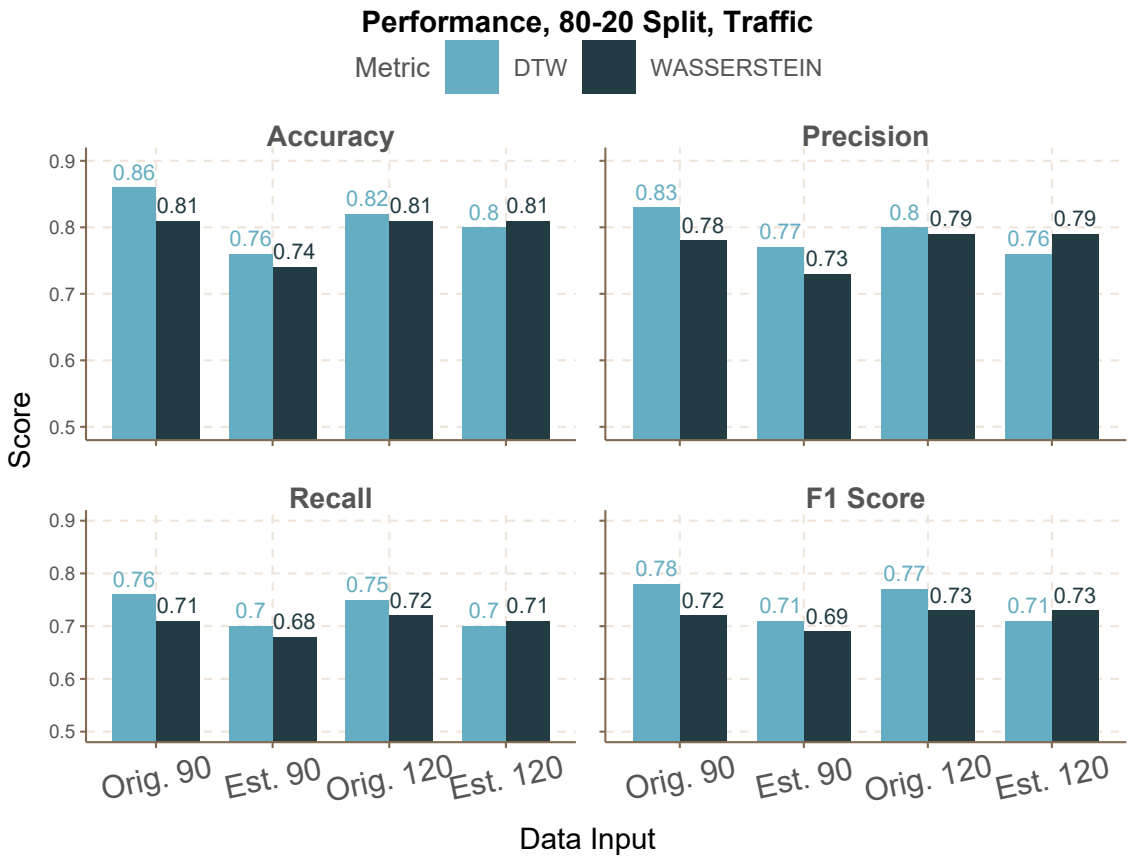


Figure 6.16 Comparison: Different input strategies

When evaluating the model performance, we are going to discuss all of the presented metrics. We start by evaluating our different input strategies and validation concept, concluding the chapter by providing an evaluation of the final binary and multi-class model and the usage of data sampling techniques.

6.3.1 Input Strategy

First, we compare the model's performance using input data based on the original reported start time of each incident, against using the estimated starting point, obtained by iterating over the data. Furthermore, we conduct an evaluation regarding two different time frames, comparing using data from 90 or 120 minutes prior and after to the incident start time.

The bar plots shown in Figure 6.16 provide the score of each performance metric regarding various input datasets. In general, using the 90 minute time interval achieves the highest overall scores, with an accuracy of 0.86 for DTW and 0.81 for Wasserstein. For this time interval, we furthermore see an advantage of using the actual start time, compared to applying our provided estimation. In contrast, the a larger set of input data, namely 120 minutes before and after the incident start time, shows a much closer performance between using the original and estimated starting point. Moreover, applying the Wasserstein metric achieves a better performance on the 120 minute input intervals, scoring better results than the DTW in case of the estimated incident start time. In conclusion, the final model uses input data

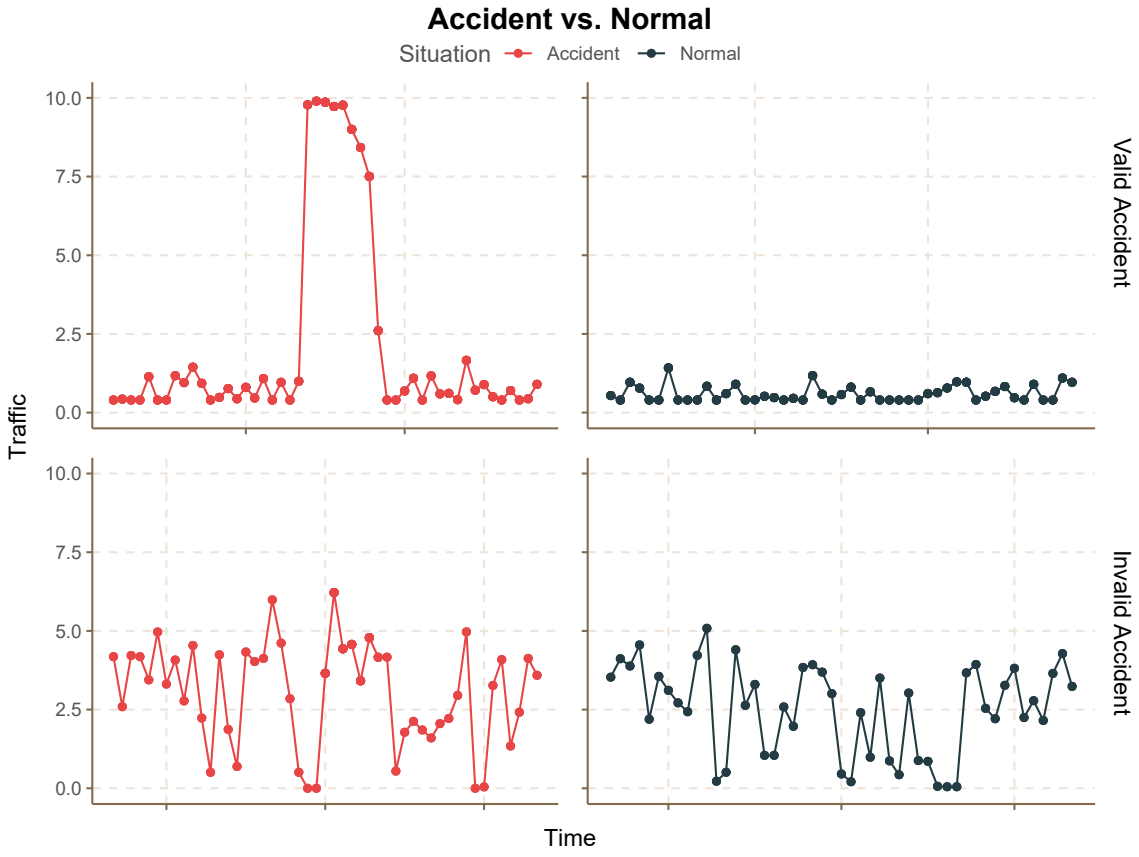


Figure 6.17 Example: Biased incident observation

covering a 90 minute time frame prior and after the original reported incident time, as this provides the overall best results. Using the iterative approach to estimate a more realistic start point of an incident does not benefit the accuracy of the model. Moreover, increasing the input time interval to 120 minutes shows a minor decrease in performance overall, but benefits the model in all test setups that use the Wasserstein metric, especially when working with the estimated start time.

6.3.2 Incident Validation

Subsequently, we conduct an evaluation on the presented procedure to validate each incident case. The method filters the data, by removing incident cases that contain too much noise. An example for this is given in Figure 6.17, representing the traffic on an accident (red line plots) and a normal situation (black line plots) on the same location. The first row of plots presents a valid accident, showing a significant difference between the accident (left) and normal situation (right), with the traffic reaching a peak value of 10 and gets back to a low value over time. In the second row, the left plot shows an example for a reported accident that was not validated, as there is no major change in the traffic behavior within the observed time frame. We cannot see any indicator of an accident based on the traffic level, comparing the accident to the normal situation. An input like this adds too much noise to the model and leads to biased results, explaining the requirement of our proposed incident validation method.

Figure 6.18 Comparison: Validated and not validated data

Next, we evaluate the respective performance scores using DTW on an 80-20 train-test-split, comparing a validated subset against the complete set of input data (not validated), to measure the benefits of the validation approach. The results of this evaluation are shown in Figure 6.18, presenting all performance metrics for each data feature. The model achieves a major performance improvement using the proposed method to validate and filter the incident cases. Regarding the accuracy, there is an overall improvement of 29%, with the strongest performance increase of 38.7%, regarding the incident classification on the traffic data feature. A similar level of improvement is achieved on all other metrics, with 25% on the precision, 22% on the recall and 25% on the F1 score. Therefore, our proposed incident validation approach improves the overall performance by 26%. We conclude that our dataset contains noisy incident samples that bias the model input leading to a worse performance. By providing a method to filter the input, removing those cases, we achieve significant better results on the model.

6.3.3 Model Performance

Finally, we evaluate the classification model using the parameters shown in Table 6.8, which represent the optimal setup, based on our previous discussed results. The validated dataset contains 1,838 data samples and uses an overlapping threshold of 50% to identify intersecting traffic regions. Furthermore, we utilize the input strategy of taking the original reported incident starting point and include data 90 minutes prior and after to that. The k-NN is setup using a total of five neighbors

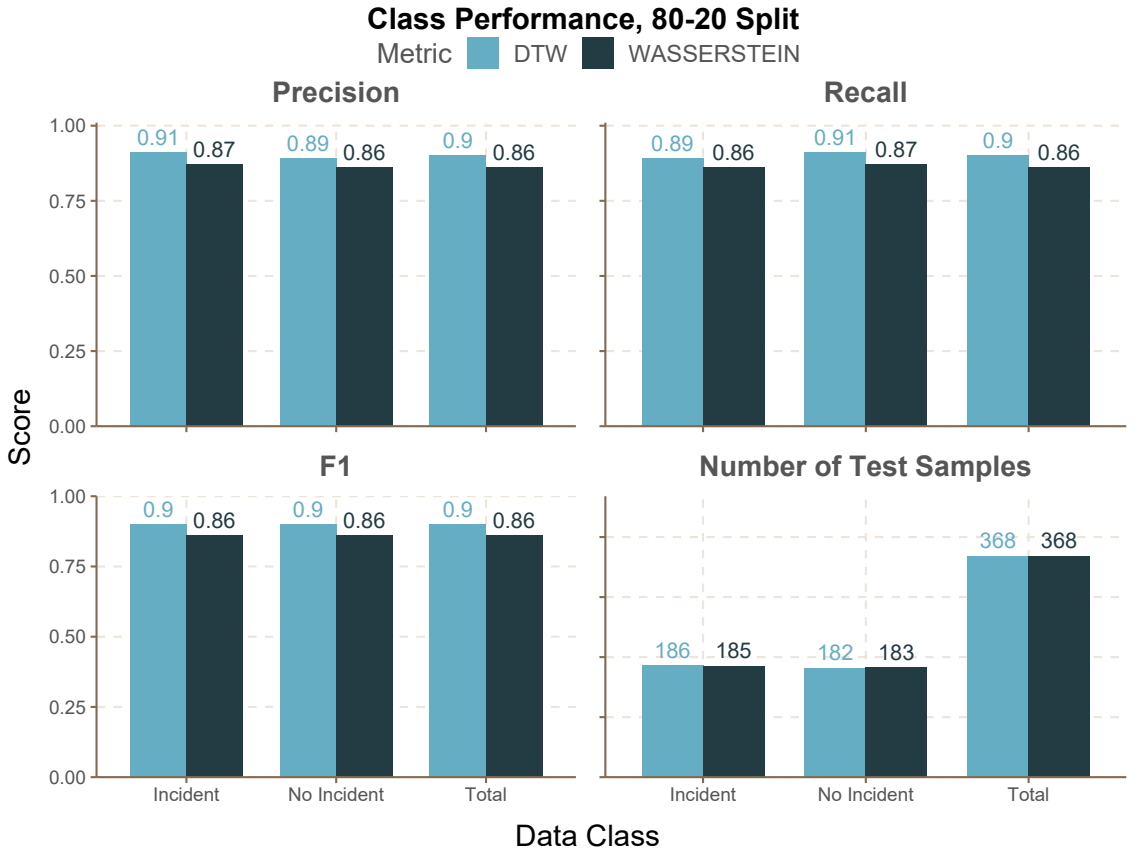


Figure 6.19 Binary model performance for each class

(k=5) and a warping window of five regarding the DTW (w.w = 5), as this provided the overall best results in our conducted experiments. Furthermore, we use SMOTE oversampling, but the benefits of this will be discussed in the following.

Data Entries	Overlap	Start Point	Time	Valid.	k	w.w.	Sampling
1,838	0.5	Original	90 min	✓	5	5	SMOTE

Table 6.8 Setup of the incident classification model

First, we evaluate the performance of our model providing a binary classification, to detect any case of incident or a normal situation. Overall, the model achieves an accuracy of 90% and moreover, Figure 6.19 presents the performance metrics for each class using the traffic data feature. Therefore, the model achieves high scores of around 90% for all data classes in every metric, with DTW showing a minor advantage compared to applying the Wasserstein metric. Furthermore, we see a balanced set of test data, with an equal amount of *Incident* and *No Incident* data samples. We conclude that the model achieves a great performance for the binary classification of incidents. To increase the complexity of the problem, the model is further tested by classifying multiple types of incidents, namely *Accident*, *Congestion* and *No Incident*.

In general, the classification of different incident classes scores an accuracy of 86%, presenting a minor decrease compared to the binary classification. However, when considering the performance of each individual class, presented in Figure 6.20, we notice major differences between the classes. Therefore, the precision, presented in

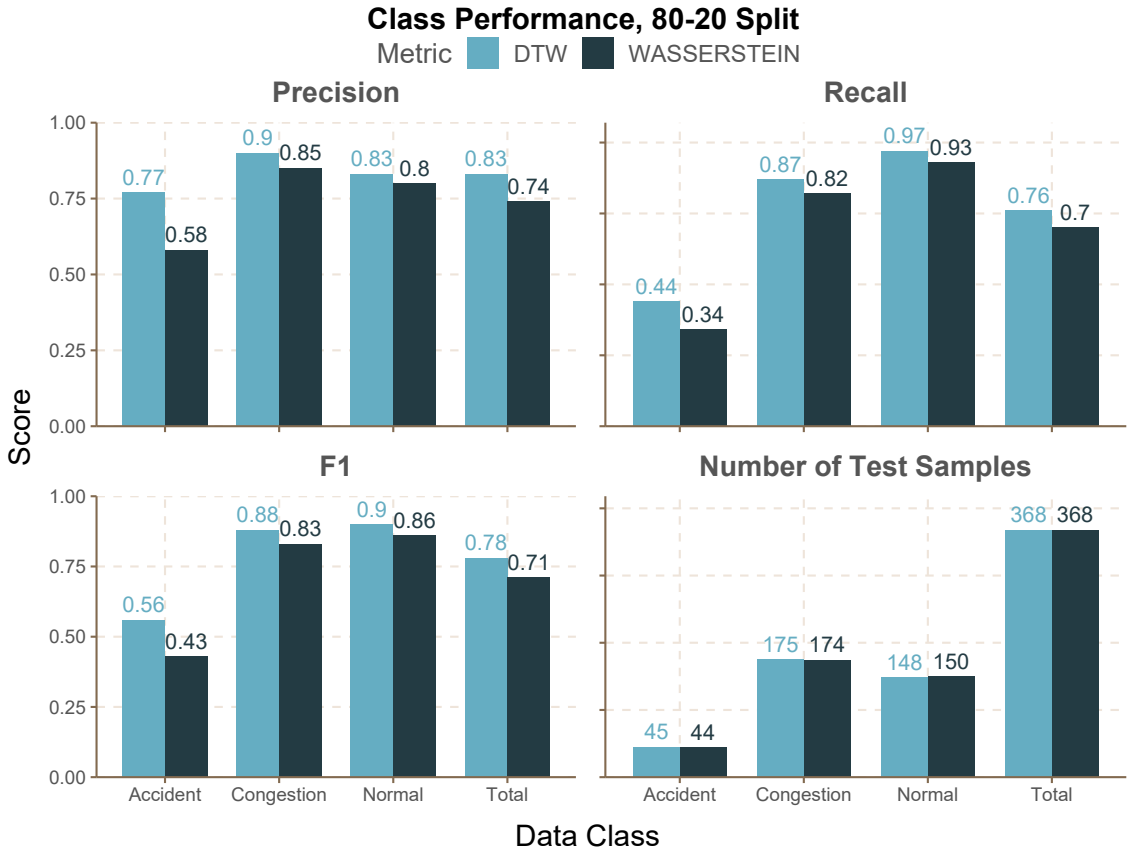


Figure 6.20 Model performance for each class

the upper left plot, shows a lower score for the accident class compared to the others, especially using the Wasserstein metric. Furthermore, the recall score of accidents is significantly lower, which explains the poor F1 score. Both other data classes achieve high scores, comparable to the ones presented from the binary classification model. A possible explanation for this observation is given by considering the number of test samples, showing major differences between the classes, indicating an imbalance of our input dataset. Therefore, only 12% of all test samples represent an accident, with 48% congestion and 40% normal samples respectively. In conclusion, the multi-class model shows a worse performance compared to the binary classification, which is a result of the imbalanced input dataset, showing a major under-representation of accident data cases. The presented results utilize an oversampling technique that could reduced the severity of this problem. In the following, we provide an extensive evaluation about such data sampling methods. The problem of under-represented samples from a certain data class is hard to solve, but we use a variety of sampling techniques to reduce the problem severity, like *Random Oversampling*, *SMOTE Oversampling* and *Nearmiss Undersampling*. The final results of the model, just presented, show the classification using *SMOTE Oversampling*, as a result of the following discussion.

Figure 6.21 provides a comparison of precision and F1 score, regarding the unique data classes and shows significant differences using various sampling techniques. With no sampling on the data, the model achieves the best results regarding the classes *Congestion* and *Normal*, but provides a poor score in case of the classified accidents. Using either of the two discussed oversampling techniques shows that the

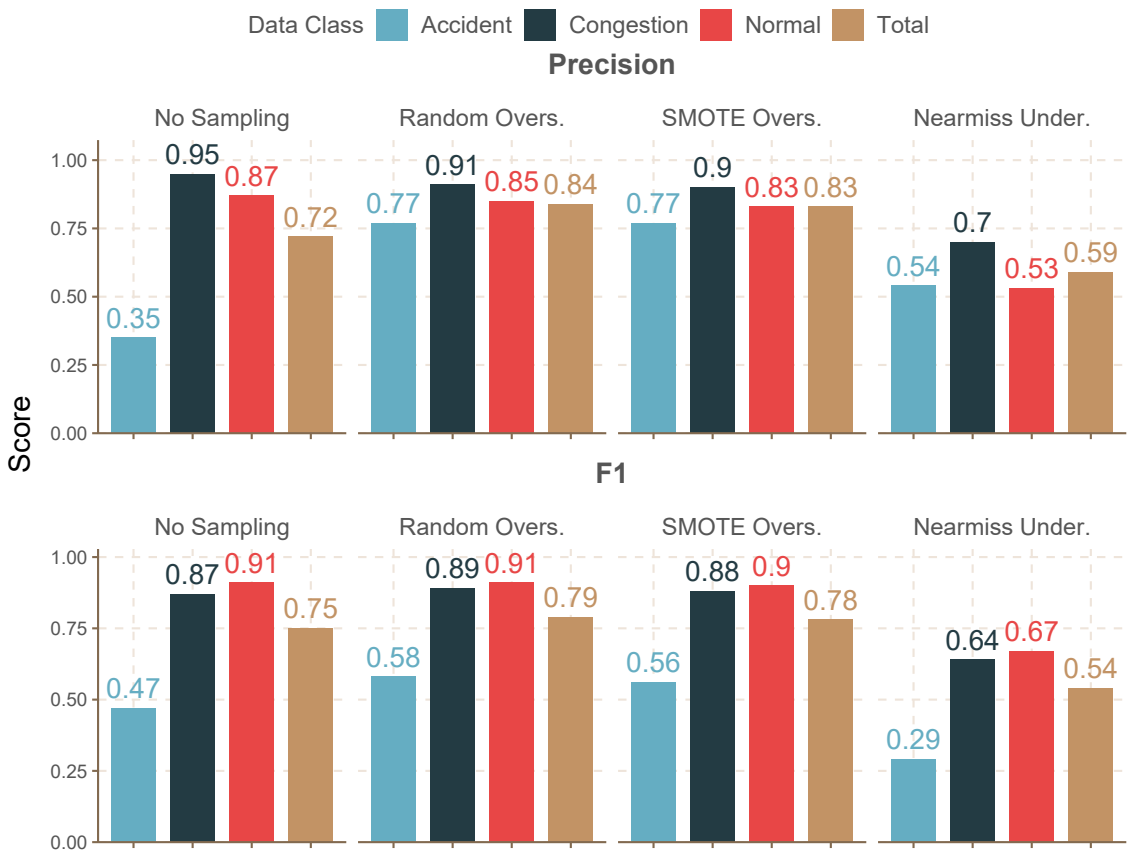


Figure 6.21 Precision and F1 Score for different sampling methods

total precision of the model can be increased from 72% to around 84%, and significantly improves the value for accident classification from 35% to 77%. A drawback of using oversampling is the minor reduction of precision regarding both other data classes. However, overall it provides a substantial benefit to the performance of the model. Furthermore, the F1 score shows an improvement for all data classes using the oversampling techniques. In case of the provided undersampling technique, we see a minor improvement of the precision, however, the improvement is significantly lower, compared to using oversampling. Furthermore, the precision scores of both other data classes are reduced, resulting in a lower total precision score of 59%. Equally, the F1 score is drastically reduced within all data classes using the undersampling method.

The benefit of oversampling is furthermore shown in Figure 6.22, depicting the number of data samples within the train and test dataset. The first row of bar plots shows a significantly imbalanced train dataset, when using no sampling method. It contains 112 different accident cases, which refers to 7.5% of all samples. Using one of the discussed oversampling techniques generates more samples of all underrepresented classes, resulting in an equal distribution of 746 samples for each class. This increases the total number of data samples from 1,488 to 2,238. Using the larger set of training data has a positive effect on the quality of the model. In contrast to this, undersampling selects the class with the lowest number of data samples and reduces the sample size from all other classes accordingly. This causes a significant reduction of the train dataset size, containing 112 samples for each class and ultimately results in a bad performance. Moreover, the set of bar plots in the second

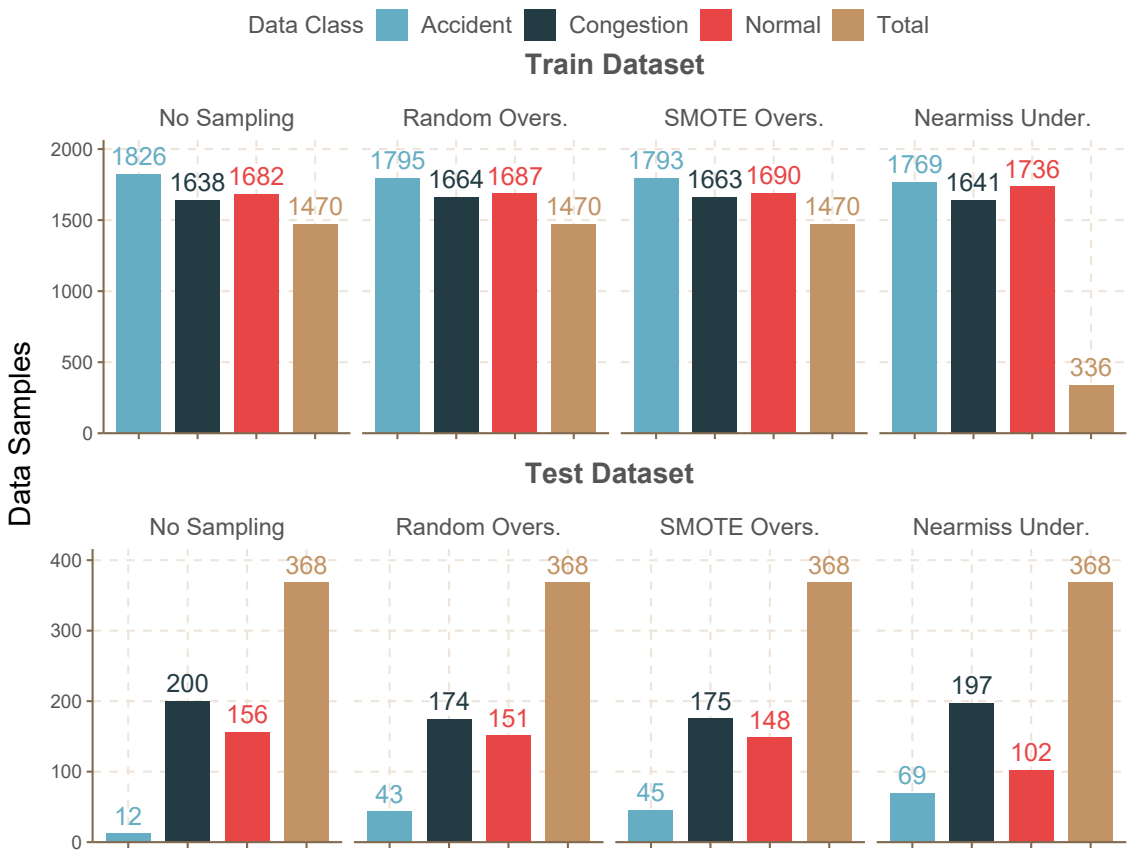


Figure 6.22 Number of train and test data samples for different sampling methods

row of Figure 6.22 visualizes the size of the test dataset for each sampling technique. It is observable, that, while not providing full balance to the test dataset, using oversampling results in a much more balanced set of data classes.

Finally, we evaluated a combination of both sampling approaches, first partially reducing the amount of over-represented data samples and then using *SMOTE Oversampling*. Therefore, we conducted an experiment, taking a subset of congestion and normal data samples and measured the performance of each data class accordingly. Figure 6.23 shows the scores of each data class for multiple subsets including different amounts of congestion and normal data samples. Focusing on the performance of accident data samples, we notice that using a larger amount of congestion and normal samples reduces the recall score significantly. From 0.77 when using a subset of 20% congestion and normal cases from the total data samples, to 0.46 in case of using 80% of the data samples. This also influences the F1 score, showing a similar behavior. However, the precision score shows minor improvements when increasing the amount of congestion and normal samples, leading to an imbalanced set of data as we noticed before. Based on this observation we can manually reduce the imbalance of the dataset, to improve the performance of each class. A setup using 20% of congestion and normal samples refers to 111 accident, 126 congestion and 138 normal class data entries in the train dataset, before oversampling. This model provides a more balanced performance, showing values of 80% for all metrics: Accuracy, Precision, Recall and F1 Score. Moreover, the performance of each class is much closer compared to results of the imbalanced input data.

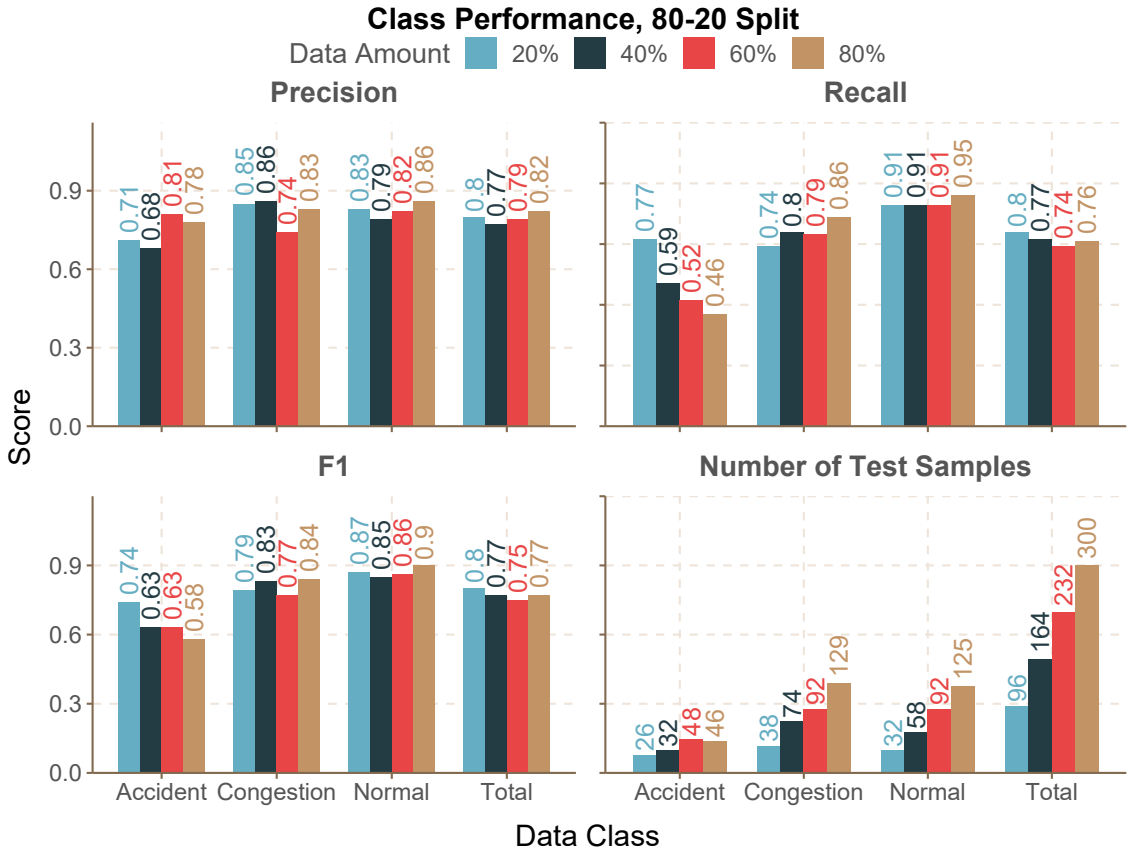


Figure 6.23 Model performance for subsets of over-represented data classes

In conclusion, the performance of our model is great on the binary classification of an incident situation, reaching an accuracy of 90%. Increasing the complexity, by classifying three different types of incidents, reduced the accuracy to 86% but furthermore showed a worse performance for the classification of accident cases, due to an imbalanced set of input data. We improve this behavior by using various sampling techniques and also manually balancing the classes within the set of input data. On a balanced dataset, the model shows an accuracy of 80% for all cases, which is a good result, however, a better way to balance the data is given by collecting more accident cases. Moreover, the Wasserstein metric achieved a marginally lower performance in general, but in certain cases provides an advantage compared to DTW. Overall, we provide a robust incident classification based on heterogeneous data fusion, capable of detecting various types of incidents derived from different traffic data patterns.

7

Conclusion

This thesis provided a traffic model composed by a traffic estimation and an incident classification that combines the collection and fusion of heterogeneous data. The goal was to introduce a framework for heterogeneous data fusion and utilize the collected information to support a robust implementation of the model.

We started with the hypothesis that a traffic model based on heterogeneous data fusion is more likely to be robust and reliable in the estimation of traffic and classification of incidents. Introducing Data Fusion on Intelligent Transportation System (DataFITS), we provided a solution to collect and combine heterogeneous data in a spatiotemporal manner, improving the amount and quality of information. The proposed framework uses the map matching technique and is extendable in terms of supported data types and sources.

Furthermore, we designed the traffic estimation application based on a naive statistical approach and a ML-based approach, using a polynomial regression. The incident classification combined incident- and traffic-related information to classify traffic patterns in a binary way (detect incidents) and categorize them to three different types of incident.

We evaluated our proposed traffic model, by conducting an extensive data analysis on the information provided by DataFITS and measured the performance of both data applications. The data fusion showed a significant enrichment of information, improving the amount of covered roads by 137% and fusing data from multiple sources on 40% of all streets. Based on the enriched information, we could provide a detailed data characterization, that can be used to detect important aspects of the traffic behavior within an urban area. To evaluate the traffic estimation application, we measured the performance of both provided models. The naive statistical approach showed good estimation results on several areas, but only provided average results on the complete set of areas, with an average R^2 value of -0.11 and error metrics above 1. In contrast, the polynomial regression approach, achieved a significantly better performance, scoring an R^2 value of up to 0.91, MAE of 0.05 and

RMSE of 0.07. However, we were not able to further improve this results by using data from correlating areas, as this reduced the performance by 5% on average.

Finally, the proposed incident classification approach achieved a high accuracy of 90% on the binary classification of incidents. Solving the more complex task of performing a classification to multiple types of incidents, the model accuracy got reduced to 86% but also showed significant differences between the performance of each class. More precisely, the classification of accidents showed a poor performance, due to the under-representation of respective samples in a general imbalanced dataset. This problem could be reduced by oversampling the train dataset, creating a more balanced representation of the data. Using a balanced dataset showed an accuracy of 80% for each data class which denoted a good result of the model. Furthermore, we provided an approach to validate each incident case, removing data samples with too much noise, which improved the model's performance by 29% on average.

7.1 Publications

The publications and contributions related to the present thesis are as follows:

- Zissner, P., Rettore, P. H. L., Santos, B. P., Lopes, R. R. F., and Sevenich, P. Road traffic density estimation based on heterogeneous data fusion. In *2022 IEEE Symposium on Computers and Communications (ISCC)* (2022), pp. 1–6
- Zissner, P., Rettore, P. H. L., Loevenich, J., and Lopes, R. R. F. A data fusion framework supporting urban military operations. Poster presented at 2022 International Conference on Military Communication and Information Systems (ICMCIS), Udine, Italy, 2022

7.2 Future Work

The proposed solution provides a complete traffic model, covering all steps from data collection, processing and application, showing the benefits of using data fusion within the context of ITS. However, we argue that there are further possibilities to improve the model and want to motivate some aspects that could be investigated in the future.

First, we consider the implementation of DataFITS on a database structure, instead of processing a set of csv files. We can store and organize the complete amount of data, making it accessible to different applications, by using a database. Within this context, we are also planning on adding a real-time processing of the data, which can be implemented on top of the database structure.

Moreover, we plan to improve the existing data characterization and proposed applications. The available data offers a lot of different opportunities for analysis, which can be conducted in the future to increase the knowledge about the traffic behavior. Furthermore, we can improve the traffic estimation application, by including

the incident information to the existing model and further investigate the correlation between this two data types. We could utilize a combination of both presented data applications, to predict the future status of traffic and detect certain types of incidents based on the estimation.

Finally, we are inspired to develop an application to solve the path planning problem, using the heterogeneous fused data. The information of traffic and incidents can provide an optimal path suggestion, dependent on the current situation of the road network. This type of application can be used in a civilian, but also military context, further supporting emergency rescue operations.

Bibliography

[1] Abadi, A., Rajabioun, T., and Ioannou, P. A. Traffic flow prediction for road transportation networks with limited traffic data. *IEEE transactions on intelligent transportation systems 16*, 2 (2014), 653–662.

[2] Anand, R. A., Vanajakshi, L., and Subramanian, S. C. Traffic density estimation under heterogeneous traffic conditions using data fusion. In *2011 IEEE Intelligent Vehicles Symposium (IV)* (2011), IEEE, pp. 31–36.

[3] Azlan, N. N. N., and Rohani, M. M. Overview of application of traffic simulation model. In *MATEC Web of Conferences* (2018), vol. 150, EDP Sciences, p. 03006.

[4] Bekiaris-Liberis, N., Roncoli, C., and Papageorgiou, M. Highway traffic state estimation with mixed connected and conventional vehicles. *IEEE Transactions on Intelligent Transportation Systems 17*, 12 (2016), 3484–3497.

[5] Bleiholder, J., and Naumann, F. Data fusion. *ACM computing surveys (CSUR) 41*, 1 (2009), 1–41.

[6] BMI - Bundesministerium des Innern und für Heimat. Open data. `https://www.bmi.bund.de/DE/themen/moderne-verwaltung/open-government/open-data/open-data-node.html`, 2021. Accessed: 2022-09-28.

[7] Boeing, G. Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems 65* (2017), 126–139.

[8] Camero, A., and Alba, E. Smart city and information technology: A review. *cities 93* (2019), 84–94.

[9] Cheng, B., Longo, S., Cirillo, F., Bauer, M., and Kovacs, E. Building a big data platform for smart cities: Experience and lessons from santander. In *2015 IEEE International Congress on Big Data* (2015), IEEE, pp. 592–599.

[10] Di, X., Xiao, Y., Zhu, C., Deng, Y., Zhao, Q., and Rao, W. Traffic congestion prediction by spatiotemporal propagation patterns. In *2019 20th IEEE International Conference on Mobile Data Management (MDM)* (2019), IEEE, pp. 298–303.

[11] El Faouzi, N.-E., Leung, H., and Kurian, A. Data fusion in intelligent transportation systems: Progress and challenges–a survey. *Information Fusion 12*, 1 (2011), 4–10.

[12] European Commission. Snart cities. https://ec.europa.eu/info/eu-regional-and-urban-development/topics/cities-and-urban-development/city-initiatives/smart-cities_en. Accessed: 2022-07-12.

[13] Foresti, G. L., Farinosi, M., and Vernier, M. Situational awareness in smart environments: socio-mobile and sensor data fusion for emergency response to disasters. *Journal of Ambient Intelligence and Humanized Computing 6*, 2 (2015), 239–257.

[14] Gad, A., and Farooq, M. Data fusion architecture for maritime surveillance. In *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002.(IEEE Cat. No. 02EX5997)* (2002), vol. 1, IEEE, pp. 448–455.

[15] Garbade, D. M. J. Regression versus classification machine learning: What's the difference? https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7, 2018. Accessed: 2022-07-12.

[16] German Federal Statistical Office (Destatis). Passengers carried in germany. https://www.destatis.de/EN/Themes/Economic-Sectors-Enterprises/Transport/Passenger-Transport/Tables/passengers-carried.html, 2022. Accessed: 2022-07-12.

[17] Guo, S., Zhou, D., Fan, J., Tong, Q., Zhu, T., Lv, W., Li, D., and Havlin, S. Identifying the most influential roads based on traffic correlation networks. *EPJ Data Science 8*, 1 (2019), 1–17.

[18] Harrison, V. Hundreds of ukrainian troops evacuated from mariupol steelworks after 82-day assault. *The Guardian* (2022). https://www.theguardian.com/world/2022/may/16/hundreds-of-ukrainian-troops-evacuated-from-azovstal-steelworks-after-82-day-assault.

[19] Hausfeld, L. Die dynamische straße. https://www.nw.de/nachrichten/thema/sicherheitswoche/21981346_Die-dynamische-Strasse.html, 2017. Accessed: 2022-07-12.

[20] Ho, G., Kim, E., Khattak, S., Penta, S., Ratnasingham, T., and Kirubarajan, T. Operator use of multi-sensor data fusion for airborne picture compilation. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (2020), IEEE, pp. 3188–3193.

[21] Jabbar, S., Malik, K. R., Ahmad, M., Aldabbas, O., Asif, M., Khalid, S., Han, K., and Ahmed, S. H. A methodology of real-time data fusion for localized big data analytics. *IEEE Access 6* (2018), 24510–24520.

[22] Jeong, S., Kim, S., and Kim, J. City data hub: Implementation of standard-based smart city data platform for interoperability. *Sensors 20*, 23 (2020), 7000.

[23] Jiang, W., and Luo, J. Big data for traffic estimation and prediction: a survey of data and tools. *Applied System Innovation 5*, 1 (2022), 23.

[24] Jotshi, A., Gong, Q., and Batta, R. Dispatching and routing of emergency vehicles in disaster mitigation using data fusion. *Socio-Economic Planning Sciences 43*, 1 (2009), 1–24.

[25] Kashinath, S. A., Mostafa, S. A., Mustapha, A., Mahdin, H., Lim, D., Mahmoud, M. A., Mohammed, M. A., Al-Rimy, B. A. S., Fudzee, M. F. M., and Yang, T. J. Review of data fusion methods for real-time and multi-sensor traffic flow analysis. *IEEE Access 9* (2021), 51258–51276.

[26] Kolouri, S., Park, S. R., Thorpe, M., Slepcev, D., and Rohde, G. K. Optimal mass transport: Signal processing and machine-learning applications. *IEEE signal processing magazine 34*, 4 (2017), 43–59.

[27] Kong, Q.-J., Li, Z., Chen, Y., and Liu, Y. An approach to urban traffic state estimation by fusing multisource information. *IEEE Transactions on Intelligent Transportation Systems 10*, 3 (2009), 499–511.

[28] Lau, B. P. L., Marakkalage, S. H., Zhou, Y., Hassan, N. U., Yuen, C., Zhang, M., and Tan, U.-X. A survey of data fusion in smart city applications. *Information Fusion 52* (2019), 357–374.

[29] Liu, Z., Li, Z., Li, M., Xing, W., and Lu, D. Mining road network correlation for traffic estimation via compressive sensing. *IEEE Transactions on Intelligent Transportation Systems 17*, 7 (2016), 1880–1893.

[30] Liu, Z., and Wang, C. Design of traffic emergency response system based on internet of things and data mining in emergencies. *IEEE Access 7* (2019), 113950–113962.

[31] Long, K., Liu, Y., and Luo, X. Emergency accident rescue system in freeway based on gis. In *2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)* (2008), vol. 2, IEEE, pp. 247–250.

[32] Meena, G., Sharma, D., and Mahrishi, M. Traffic prediction for intelligent transportation system using machine learning. In *2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE)* (2020), IEEE, pp. 145–148.

[33] Oskarbski, J. Automatic road traffic safety management system in urban areas. In *MATEC Web of Conferences* (2017), vol. 122, EDP Sciences, p. 03007.

[34] Ota, K., Dong, M., Zhu, H., Chang, S., and Shen, X. Traffic information prediction in urban vehicular networks: A correlation based approach. In *2011 IEEE Wireless Communications and Networking Conference* (2011), IEEE, pp. 1021–1025.

[35] Pan, T., Sumalee, A., Zhong, R.-X., and Indra-Payoong, N. Short-term traffic state prediction based on temporal–spatial correlation. *IEEE Transactions on Intelligent Transportation Systems 14*, 3 (2013), 1242–1254.

[36] Panicker, M., Mitha, T., Oak, K., Deshpande, A. M., and Ganguly, C. Multisensor data fusion for an autonomous ground vehicle. In *2016 Conference on Advances in Signal Processing (CASP)* (2016), IEEE, pp. 507–512.

[37] Park, S.-h., Kim, S.-m., and Ha, Y.-g. Highway traffic accident prediction using vds big data analysis. *The Journal of Supercomputing 72*, 7 (2016), 2815–2831.

[38] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[39] Pigeon, L. A conceptual approach for military data fusion. Tech. rep., DEFENCE RESEARCH AND DEVELOPMENT CANADA VALCARTIER (QUEBEC), 2002.

[40] Qureshi, K. N., and Abdullah, A. H. A survey on intelligent transportation systems. *Middle-East Journal of Scientific Research 15*, 5 (2013), 629–642.

[41] Ren, H., Song, Y., Wang, J., Hu, Y., and Lei, J. A deep learning approach to the citywide traffic accident risk prediction. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 3346–3351.

[42] Rettore, P. H., Maia, G., Villas, L. A., and Loureiro, A. A. F. Vehicular data space: The data point of view. *IEEE Communications Surveys & Tutorials 21*, 3 (2019), 2392–2418.

[43] Sanjana, K., Lavanya, S., and Jinila, Y. B. An approach on automated rescue system with intelligent traffic lights for emergency services. In *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)* (2015), IEEE, pp. 1–4.

[44] Seo, T., Bayen, A. M., Kusakabe, T., and Asakura, Y. Traffic state estimation on highway: A comprehensive survey. *Annual reviews in control 43* (2017), 128–151.

[45] Shan, Z., Xia, Y., Hou, P., and He, J. Fusing incomplete multisensor heterogeneous data to estimate urban traffic. *IEEE MultiMedia 23*, 3 (2016), 56–63.

[46] Statista. Number of registered cars in germany from 1960 to 2022. `https://www.statista.com/statistics/587764/number-of-registered-cars-germany/`, 2022. Accessed: 2022-07-12.

[47] Tang, J., Li, L., Hu, Z., and Liu, F. Short-term traffic flow prediction considering spatio-temporal correlation: A hybrid model combing type-2 fuzzy c-means and artificial neural network. *Ieee Access 7* (2019), 101009–101018.

[48] Tavenard, R. An introduction to dynamic time warping. `https://rtavenar.github.io/blog/dtw.html`. Accessed: 2022-09-14.

[49] United Nations, D. o. E., and Affairs, S. World urbanization prospect - the 2018 revision. `https://population.un.org/wup/Publications/Files/WUP2018-Report.pdf`, 2018. Accessed: 2022-10-13.

[50] Vautravers, A. Military operations in urban areas. *International Review of the Red Cross 92*, 878 (2010), 437–452.

[51] Vítor, G., Rito, P., and Sargento, S. Smart city data platform for real-time processing and data sharing. In *2021 IEEE Symposium on Computers and Communications (ISCC)* (2021), IEEE, pp. 1–7.

[52] Wang, C. The relationship between traffic congestion and road accidents: an econometric approach using gis. `https://repository.lboro.ac.uk/articles/thesis/The_relationship_between_traffic_congestion_and_road_accidents_an_econometric_approach_using_GIS/9455504`, 5 2010. Accessed: 2022-07-12.

[53] Wang, J., Li, X., Liao, S. S., and Hua, Z. A hybrid approach for automatic incident detection. *IEEE Transactions on Intelligent Transportation Systems 14*, 3 (2013), 1176–1185.

[54] Wang, S., Zhang, M., Miao, H., Peng, Z., and Yu, P. S. Multivariate correlation-aware spatio-temporal graph convolutional networks for multi-scale traffic prediction. *ACM Transactions on Intelligent Systems and Technology (TIST) 13*, 3 (2022), 1–22.

[55] Wang, X., Guan, X., Cao, J., Zhang, N., and Wu, H. Forecast network-wide traffic states for multiple steps ahead: A deep learning approach considering dynamic non-local spatial correlation and non-stationary temporal dependency. *Transportation Research Part C: Emerging Technologies 119* (2020), 102763.

[56] Weiland, R. J., and Purser, L. B. Intelligent transportation systems. *Transportation in the new millennium* (2000).

[57] Wen, H., Lin, Y., and Wu, J. Co-evolutionary optimization algorithm based on the future traffic environment for emergency rescue path planning. *IEEE Access 8* (2020), 148125–148135.

[58] Wenqi, L., Dongyu, L., and Menghua, Y. A model of traffic accident prediction based on convolutional neural network. In *2017 2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE)* (2017), IEEE, pp. 198–202.

[59] World Health Organization. *Global Status Report on Road Safety 2018*. World Health Organization, 2019. Accessed: 2022-07-12.

[60] Yang, C., and Gidofalvi, G. Fast map matching, an algorithm integrating hidden markov model with precomputation. *International Journal of Geographical Information Science 32*, 3 (2018), 547 – 570.

[61] Yuan, H., and Li, G. A survey of traffic prediction: from spatio-temporal data to intelligent transportation. *Data Science and Engineering 6*, 1 (2021), 63–85.

[62] Yuras Karmanau, Jim Heintz, V. I., and Porta, J. L. Street fighting begins in kyiv; people urged to seek shelter. *AP News* (2022). https://apnews.com/article/russia-ukraine-vladimir-putin-volodymyr-zelenskyy-boris-johnson-business-08f569df695831ee467979527ea2e241.

[63] Zhang, L., Xie, Y., Xidao, L., and Zhang, X. Multi-source heterogeneous data fusion. In *2018 International conference on artificial intelligence and big data (ICAIBD)* (2018), IEEE, pp. 47–51.

[64] Zhao, B., Gao, X., Liu, J., Zhao, J., and Xu, C. Spatiotemporal data fusion in graph convolutional networks for traffic prediction. *Ieee Access 8* (2020), 76632–76641.

[65] Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., and Li, H. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems 21*, 9 (2019), 3848–3858.

[66] Zheng, C., Fan, X., Wang, C., and Qi, J. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2020), vol. 34, pp. 1234–1241.

[67] Zhu, Y., Li, Z., Zhu, H., Li, M., and Zhang, Q. A compressive sensing approach to urban traffic estimation with probe vehicles. *IEEE Transactions on Mobile Computing 12*, 11 (2012), 2289–2302.

[68] Zissner, P., Rettore, P. H. L., Loevenich, J., and Lopes, R. R. F. A data fusion framework supporting urban military operations. Poster presented at 2022 International Conference on Military Communication and Information Systems (ICMCIS), Udine, Italy, 2022.

[69] Zissner, P., Rettore, P. H. L., Santos, B. P., Lopes, R. R. F., and Sevenich, P. Road traffic density estimation based on heterogeneous data fusion. In *2022 IEEE Symposium on Computers and Communications (ISCC)* (2022), pp. 1–6.

List of Figures

4.1 Data fusion of civilian and military information 18

5.1 Design of the proposed traffic model 22
5.2 Workflow of DataFITS 23
5.3 Example of the map matching process 25
5.4 Design of the traffic estimation application 29
5.5 Group traffic data by a set of unique areas 30
5.6 Comparison: Simple linear regression and polynomial regression . . . 34
5.7 Design of the incident classification application 36
5.8 Collect traffic data for each incident 37
5.9 Create input data for the incident classification 40
5.10 Example: Decision tree classifier 41
5.11 Example: k-NN classifier 42

6.1 Cities of the data acquisition compared to Envirocar data availability 48
6.2 Distribution of traffic based on street type and time of the day 51
6.3 Relation of speed and traffic on different street types 52
6.4 Street coverage and relative speed of each road segment 53
6.5 Spatiotemporal traffic view 54
6.6 Incidents based on road types and time of the day 56
6.7 Incidents based on weather and season of the year 56
6.8 Density of all incidents 57
6.9 Density of different incident types 58
6.10 Incident effect on traffic behavior 59
6.11 Estimations using the naive statistical model 61
6.12 Estimations using the polynomial regression 62

6.13 RMSE for different polynomial degrees 63
6.14 Performance for different sizes of training data 64
6.15 Comparison: Statistical model and polynomial regression 66
6.16 Comparison: Different input strategies 68
6.17 Example: Biased incident observation 69
6.18 Comparison: Validated and not validated data 70
6.19 Binary model performance for each class 71
6.20 Model performance for each class 72
6.21 Precision and F1 Score for different sampling methods 73
6.22 Number of train and test data samples for different sampling methods 74
6.23 Model performance for subsets of over-represented data classes 75

List of Tables

3.1 Comparison: Solutions from literature and our thesis 16

5.1 Exemplary data mapping of traffic values 24

6.1 Features reported by the different data sources. 46

6.2 Covered roads by data source 49

6.3 General traffic data statistics 50

6.4 General incident data statistics 55

6.5 Performance of the naive statistical model 61

6.6 Setup of the traffic estimation model 62

6.7 Performance of the regression model on various streets and weather conditions 65

6.8 Setup of the incident classification model 71