Ana Filipa Simão de Almeida

mCity: Utilização de dados de monitorização de uma cidade inteligente para caraterizar e melhorar a mobilidade urbana

mCity: Using smart city monitoring data to characterize and improve urban mobility

Ana Filipa Simão de Almeida

mCity: Utilização de dados de monitorização de uma cidade inteligente para caraterizar e melhorar a mobilidade urbana
mCity: Using smart city monitoring data to characterize and improve urban mobility

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor Ilídio Castro Oliveira, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e da Professora Doutora Susana Sargento, Professora Catedrática do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho à minha mãe.

o júri / the jury

presidente / president

Professor Doutor José Manuel Matos Moreira,
Professor Auxiliar, do Departamento de Eletrónica, Telecomunicações e Informática, da Universidade de Aveiro

vogais / examiners committee

Professor Doutor Pedro Miguel Alves Brandão,
Professor Auxiliar, do departamento de Ciências e Computadores, da Faculdade de Ciências da Universidade do Porto

Professor Doutor Ilídio Fernando de Castro Oliveira,
Professor Auxiliar, do Departamento de Eletrónica, Telecomunicações e Informática, da Universidade de Aveiro

agradecimentos / acknowledgements

Gostaria de começar por agradecer à minha mãe. Obrigada por me incentivares a seguir o ensino superior e por te esforçares para garantir que eu tinha acesso às oportunidades que tu não tiveste. Agradeço também à minha família, àqueles que me apoiaram e estiveram do meu lado nas várias etapas da minha vida.

Agradeço ao professor Ilídio Oliveira e à professor Susana Sargento pela orientação, sugestões e apoio dado. Agradeço também à Susana Brás, pela partilha de conhecimento e por todas as ajudas fornecidas. Agradeço ainda ao grupo de investigação Network Architectures and Protocols do Instituto de Telecomunicações de Aveiro, em particular ao Miguel Luís e ao Carlos Senna, por me terem acolhido no grupo e acompanhado o meu percurso académico.

Agradeço aos amigos e colegas que fizeram parte do meu percurso académico e que o enriqueceram com momentos de amizade e descontração.

Agradeço à Fundação Portuguesa para a Ciência pelo suporte financeiro através de fundos nacionais e europeus, no âmbito do projecto CMUP-ERI/TIC/0010/2014, S2MovingCity: Sensing and Serving a Moving City.

palavras-chave Mobilidade urbana inteligente, Fluxo de tráfego, Previsão, Aprendizagem profunda, Perfil de condução

resumo

O crescimento sustentável das cidades criou a necessidade de decisões melhor informadas, baseadas em tecnologias de informação e comunicação para sentir a cidade e quantificar o seu pulso. Uma parte importante no conceito de "cidades inteligentes" é a caracterização dos fluxos de tráfego. O objetivo deste trabalho é caraterizar a mobilidade em duas cidades diferentes: Porto e Aveiro. A estrutura e conteúdo dos respetivos datasets é muito diferente, permitindo dois casos de estudo, com casos de uso distintos relacionados com a análise de tráfego e a previsão.

Para o caso de uso do Porto, foi concedido acesso a sensores de tráfego instalados na estrada e dados de rastreamento de autocarros. Para a primeira fonte realizou-se um estudo e a pesquisa de padrões (por exemplo, o comportamento dos dias da semana). Dados históricos dos contadores de tráfego foram usados para prever fluxos futuros, usando métodos estatísticos e de aprendizagem profunda.

Descobrimos que não era possível encontrar uma relação clara entre a velocidade (dos autocarros) e a intensidade do tráfego, no entanto, quando a velocidade era alta, havia baixa intensidade e, quando havia alta intensidade, a velocidade era baixa. Existem padrões diários e semanais nos dados do fluxo de tráfego que permitem a previsão. Quando as anomalias no tráfego ocorrem, os métodos para previsão de curto prazo têm um desempenho melhor do que aqueles para previsão de longo prazo.

Para o caso de uso de Aveiro, o conjunto de dados inclui rastreamentos de autocarros, que foram utilizados para caraterizar o comportamento de condução, baseado na velocidade e aceleração. Esses dados foram mapeados na cidade para encontrar áreas problemáticas. As visualizações lado a lado ajudam na comparação do comportamento do tráfego em períodos selecionados. Foi observado que algumas estradas apresentam frequentemente os mesmos problemas, independentemente do dia ou da hora do dia. Em outras partes da cidade, os problemas podem ser encontrados com mais frequência em períodos específicos.

Os conjuntos de dados de Aveiro e Porto tinham amostras com diferentes frequências (a cada segundo e a cada minuto, respectivamente). Confirmamos, com simulações, que a análise feita para Aveiro não era possível com a granularidade do conjunto de dados do Porto (dado que algumas informações seriam perdidas).

A pipeline computacional para executar as análises de suporte foi totalmente implementada, bem como as integrações necessárias para obter programaticamente os dados das fontes de dados existentes. Foi desenvolvida uma pipeline de previsão de tráfego para o Porto. Para a análise do comportamento de condução, foi construída uma web dashboard, permitindo que os departamentos relevantes estudem possíveis áreas problemáticas na cidade de Aveiro.

keywords Smart Urban Mobility, Traffic flow, Forecasting, Deep Learning, Driving behavior

abstract The sustainable growth of cities created the need for better informed decisions based on information and communication technologies to sense the city and quantify its pulse. An important part in this concept of "smart cities" is the characterization of the traffic flows.
In this work, we aim at characterizing the urban mobility in two different cities, Porto and Aveiro. The structure and contents of the corresponding datasets is very different, enabling two case studies, with distinct use cases related to traffic analysis and forecasting.
For the Porto use case, we had access to road-mounted traffic sensors and the buses tracking data. The first source was studied and was looked for patterns (e.g.: weekdays behavior). Historic traffic counters data was used to forecast future flows, using both statistical and deep learning methods.
We found that it was not possible to find a clear relationship between (buses) speed and traffic intensity, however, when the speed was high, there was low intensity, and when there was high intensity, the velocity was low. There are daily and weekly patterns in the traffic flow data that enable forecasting. When the anomalies in traffic do happen, the methods for short-term forecasting perform better than those for long-term forecasting.
In the Aveiro use case, the dataset includes bus traces, that were used to characterize the driving behavior, based on speed and acceleration. These data were mapped into the city to find problematic areas. Side-by-side visualizations help with the comparison of the traffic behavior in selected time periods. We observed that some roads often present the same problems, independently of the day or time of the day. In other parts of the city, the problems can be found more often in specific periods.
The datasets for Aveiro and Porto were sampled with different frequency (each second and each minute, respectively). We confirmed, with simulations, that the analysis made for Aveiro was not possible with the granularity of the Porto's data set (as some information would be lost).
The computational pipeline to run the supporting analyses is fully implemented, as well the required integrations to programmatically obtain the data from the existing data sinks. For the driving behavior analysis, a web dashboard is deployed, enabling the relevant departments to study potential problematic areas in the city of Aveiro.

Contents

List of Figures

List of Tables

Acronyms

ADF *Augmented Dickey-Fuller*. 9, 37, 41, 43

AIC *Akaike Information Criterion*. 16, 17

ANN *Artificial Neural Network*. 12, 21, 23, 24, 52, 53, 54, 61, 62, 65, 70, 74

AR *AutoRegressive*. 10, 11, 12

ARIMA *AutoRegressive Integrated Moving Average*. 10, 11, 12, 20, 54

ARMA *Autoregressive Moving Average*. 11

BIC *Bayesian Information Criterion*. 16, 17, 56, 57, 101

CNN *Convolutional Neural Network*. 12, 21

DCU *Data Collecting Unit*. 6, 80

DHA *Deviation from Historical Average*. 20

DNN *Deep Neural Networks*. 21

DTC *Dynamic Temporal Context Neural Network*. 21

EWHA *Exponentially Weighted Historical Average*. 20

EWMA *Exponentially Weighted Moving Average*. 37

FCN *Fully Connected Network*. 15

FFNN *FeedForward Neural Network*. 12, 15, 17, 54, 61, 62, 65, 66, 67, 70

GARCH *Generalized Autoregressive Conditional Heteroskedasticity*. 54

GIS *Geographic Information System*. 7

GPS *Global Position System*. 5, 6, 19, 30, 31, 32, 33, 48, 76, 80, 81, 82, 83, 93

GRU *Gated Recurrent Unit*. 21

GTFS *General Transit Feed Specification*. 6, 7, 25, 31, 33, 34, 75, 76, 80

HA *Historical Average*. 21

HAF *Historical Average Forecast*. 20

IEETA *Institute of Electronics and Informatics Engineering of Aveiro*. 1

IoT *Internet of Things*. 1

IT *Institute of Telecommunications*. 1, 6, 76

ITS *Intelligent Transportation Systems*. 19

LSTM *Long Short-Term Memory*. 15, 17, 21, 22, 54, 61, 62, 63, 67, 69, 70, 72, 74

MA *Moving Average*. 10, 11

MAE *Mean Absolute Error*. 16, 63, 65, 66, 67

MAPE *Mean Absolute Percentage Error*. 16

MSE *Mean Squared Error*. 15, 16, 56, 57, 60, 62, 63, 65, 66, 67

NaN *Not a Number*. 30, 80, 83

NAP *Network Architectures and Protocols*. 1, 6

OBU *On-Board Unit*. 6, 79, 80

OSM *OpenStreetMap*. 7, 24, 80, 81, 82, 93

ReLU *Rectified Linear Unit*. 13

RMSE *Root Mean Squared Error*. 15, 16, 56, 63, 65, 66, 67

RNN *Recurrent Neural Network*. 12, 15, 21

RSU *Road Side Unit*. 6

SARIMA *Seasonal AutoRegressive Integrated Moving Average*. 10, 17, 20, 54, 55, 56, 72, 74, 101

STFSA *Spatio-Temporal Feature Selection Algorithm*. 21

SVR *Support Vector Regression*. 22

tanh *hyperbolic tangent*. 13, 63, 67

TCHA *Temporal Clustering and Hierarchical Attention*. 21

VPN *Virtual Private Network*. 76

Chapter 1

Introduction

This dissertation project studies the urban mobility in the cities of Porto and Aveiro and was developed in the context of the research activities of S2MovingCity project.

1.1 Context and Motivation

Smart cities use information and communication technologies to enhance decision making and, thus, trying to make cities better and more sustainable places to live. A smart city is a city endowed by the ability to make smart management of the resources in a smart way. The *smart* aspect comes from the ability to integrate sensing and analytics technologies to support decision making and actuations. For example, smart management of electricity by turning the lights on only when there are people nearby, smart management of water by watering gardens only when the soil is not damp, etc.

Sensing and communication capabilities would be deployed across the city that can collect data, transmit data, and perform actions. In this context emerged the concept of the *Internet of Things* (IoT).

The smart management of urban mobility is one of the many goals that several cities want to achieve. By taking advantage of the IoT, it is possible to collect urban mobility data, process and analyze the data and propose actions to improve urban mobility. Some examples are on-demand parking, responding to accidents in real-time, improving traffic flowing, etc.

The project S2MovingCity [1] aims to improve city management through the creation of decision support systems. It relies on the creation of a communication infrastructure to collect data from fixed stations, a vehicular network, and mobile collectors. The gathered information will be processed and analyzed in order to expose behavior patterns and make predictions.

Members of the *Network Architectures and Protocols* (NAP) [2] group from *Institute of Telecommunications* (IT) [3] and members from *Institute of Electronics and Informatics Engineering of Aveiro* (IEETA) [4] are developing this project. Several works contribute to this project, for example, Pereira et al. [2] created a decision support dashboard to help in the management of traffic by the competent authorities. Ricardo et al. [3] created a tool that

[1] s2movingcity.av.it.pt
[2] www.it.pt/Groups/Index/36
[3] www.it.pt/ITSites/Index/3
[4] wiki.ieeta.pt

builds smart bus schedules and estimates the bus time of arrival. Tavares et al. [4] create an application that performs the estimation of bus arrival time.

This dissertation is focused on the study of smart urban mobility in two cities of Portugal: Porto and Aveiro. This work aims study traffic flow and driving behavior. Identifying patterns in traffic flow data, and predicting traffic flow data even when anomalous conditions happen, can lead the responsible authorities to take measures when it is necessary. For example, if it is detected an increase in traffic in a location, it might be beneficial to change the infrastructure, like to change traffic signs, to allow a better flow. Identifying the locals' or periods associated with non-safe driving behavior can also lead to changes in the infrastructure.

1.2 Objectives

The main goal of this work is to study urban mobility by using traffic data obtained from buses and traffic counters. The present dissertation has the following objectives:

- Study the bus tracking dataset and additional deployed sensors (traffic counters) in the Oporto city to predict future traffic behavior,
- Use vehicle tracking data to characterize the driving behavior and look for safety patterns,
- Integrate the analytic methods in friendly end user-tools.

Besides buses, there are several sensors spread across the city that allow us to have a good picture of the evolution of traffic as time goes by. The first topic intends to study the collected data to highlight traffic problems. Besides that, traffic behavior can foresee potential problems, and help in the creation of alternative solutions to traffic.

The second topic aims at the creation of driving profiles, using as metrics the collected data. The driving profiles should contemplate safe driving profiles and non-safe driving profiles. Create a distinct line between the driving profiles is the first step to identify the sources that lead to non-safe driving. It is important to identify roads, zones, times of the day, etc.

The last topic is focused on the creation of a decision support tool for studying driving behavior in order to identify the causes that lead to non-safe driving. This topic is a complement to the previous one.

1.3 Contributions

The work developed in this dissertation can be summarized as follows:

- Creation of a dataset for road segments from Porto,
- Creation of a dataset with maximum bus speed for road segments in Aveiro,
- Creation of a dataset with bus data, road segment, acceleration, travel distance and behavior profile,
- Creation of a Fiware structure to receive and persist data
- Pipelines for preprocessing the data,

- Pipeline to forecasting the traffic flow observed,
- Classification of driving profile (safe or non-safe),
- Tool for comparison and study of the driving behavior.

As a result of this work, two scientific papers are being prepared, entitled "Using automatic traffic counter data to forecast traffic behavior in Oporto" and "Characterizing driving patterns from bus tracking data".

1.4 Dissertation Structure

This document is structured as follows:

- Chapter 1 - Introduction: Contains the dissertation motivation, context, objectives of the developed work, and contributions.
- Chapter 2 - Background concepts: This chapter introduces key concepts of the developed work. The key concepts are time-series, statistical methods for forecasting, and deep learning methods for predicting.
- Chapter 3 - State of the art: It presents the state of the art about statistical methods for forecasting the traffic flow, deep learning methods for predicting the traffic flow, and driving behavior analysis methods based on speed and acceleration.
- Chapter 4 - Use cases: It presents the requirements defined, the data sources,and the use cases.
- Chapter 5 - Forecasting the traffic flow: It describes the process to forecast traffic flow observed.
- Chapter 6 - Driving behavior: It describes the process to study the driving behavior.
- Chapter 8 - Results: It presents the results obtained from the implemented solutions.
- Chapter 9 - Conclusions and future work: It discusses the main conclusions and proposes future improvements.

Chapter 2

Background concepts

In this chapter, we review the essential background concepts supporting the data analysis methods used in this work. The first section contains a brief introduction to the type of data found in urban mobility studies, the challenges associated, and the type of components that characterize urban mobility data: time and space.

Since the data is very dependent on the component of time, and one of the main goals is to forecast or predict future values, this chapter contains tree more sections for each one of the key aspects: time-series, statistical methods for forecasting, and deep learning methods for predicting.

The second section explains what is a time-series and how it can be studied. The third section presents the statistical methods that can forecast future values. The statistical methods are very important in the early stages of this study. The fourth section explains the machine learning methods used for predicting the traffic flow. The last section presents the evaluation metrics used.

Note that, in this context, forecast and prediction are very similar. However, infer something is not necessarily about the future; for example, I may want to predict if an object is a pencil or a pen, based on a set of features.

2.1 Urban Mobility Data

The urban mobility data used in this work refers to traffic data, which contains spatial and temporal information that can be used to characterize mobility flows in the city.

2.1.1 Data sources for traffic description

The information about traffic can be from the infrastructure, like traffic flow sensors, traffic speed sensors [5], pedestrian sensors, speed sensors, beacons, security cameras and traffic lights, or can be from vehicles. In our case, the information comes from both and have heterogeneous sources. In the literature, we also find information that comes from video [6].

In Porto, there is a network of buses that provide information about their speed, *Global Position System* (GPS) coordinates, and heading every minute. Besides buses, it can also be used on taxis [7], garbage trucks, etc. There are also traffic flow sensors that measure the number of vehicles crossing a segment per time unit.

Earlier this year, in Aveiro, a similar infrastructure to the Porto bus infrastructure began to be built. In AveiroBus [1] buses it was connected *On-Board Unit* (OBU)s, and *Data Collecting Unit* (DCU)s. Some *Road Side Unit* (RSU)s were also added at strategic points in the city. The infrastructure in Aveiro was created by the NAP group of the IT of Aveiro.

Veniam [2] and AveiroBus provide information about the buses, such as GPS data and speed. The traffic flow information was given by Porto Digital [3]. Both datasets have two key features, space and time. It is expected to observe patterns resulting from the evolution of time, but that depends on the location. The gathered information can be processed using historical methods, statistical methods, or machine learning methods. In some cases, even just the visualization of the data can be useful.

Besides, we receive a *General Transit Feed Specification* (GTFS) that provides information about the buses like routes, trips, stops, agency and calendar dates. However, this information was given just about the network of Porto. For Aveiro, it was possible to get a similar type of information using available online datasets.

GTFS represents public transport information through a set of text files organized in a similar way to a relational database [8]. It is possible to obtain all the bus routes and the planed places where buses can go. It is also possible to calculate, for a given street and in a period, the number of buses that are expected to pass there. GTFS files have an expiration date, usually just a few months.

The *shapes.txt* file contains GPS coordinates organized in a way that makes it possible to discover the paths where buses go through, but it is not feasible to work with GPS coordinates. To determine which street or segment the GPS coordinates belong, or which segment is closest to a GPS position, it was decided to create a spatial database that allows performing geographic and geometric queries. However, first we need to convert GPS coordinates into geographic features.

The most significant difficulties arise from the characteristics of the datasets, namely the bus speed dataset of Porto. Bus mobility patterns do not match the general traffic mobility patterns. Buses do not represent the behavior of the remaining vehicles, as they have a pre-established route. They make stops along their route, and given their dimensions, they may be forced to travel at lower speeds, and they may sometimes have their own traffic lanes.

2.1.2 Geospatial data visualization

If we look at each of the nodes individually (one bus or one traffic flow sensor), we realize that they can be represented through a time-series, where the sequence of values matters and cannot be changed.

It is expected to observe patterns as a result of the evolution of time. There are two types of patterns to consider: cyclic and seasonal. A seasonal pattern is a result of seasonal factors and has a fixed frequency. In a cyclic pattern, there is not a fixed frequency, and the patterns observed depend on factors that are not influenced by the calendar. Beyond helping find patterns, time-series are very useful to predict future values [9].

To achieve the desired goals, it is necessary to simplify the problem. Working with geographic data can be more difficult than it seems. Some authors ignore their presence by working with one geographical point [10] or two freeway locations [6]. However, other authors

[1] www.aveirobus.pt/
[2] www.veniam.com
[3] www.portodigital.pt

suggest a solution to deal with geographical data. Qimei Cui et. al. makes the mapping of geographical data into a grid [7]. A different approach is proposed by Yiming Xing et. al. as they divide geographical data into road segments [11]. The approach of analyzing by segments, bus lines, or regions of interest seems to be the best to follow.

Spatial data requires special attention in order to be manipulated. *Geographic Information System* (GIS) is a framework that maps the world in a two-dimensional plane. In other words, GIS deals with flat map projections. It allows to gather, store, analyse, and visualize spatial data [12]. There are several GIS Softwares available, the two most popular being ArcGIS [13] and QGIS [14]. Unlike ArcGIS, QGIS is a free software, open-source and cross-platform. One advantage of using QGIS is the capability of importing and exporting geographic data in different formats. QGIS has other features such as visualization, exploration, data processing, etc. In addition to the key features, many others can be added by adding plugins [14].

With QGIS it is possible to convert the *shapes.txt* from the GTFS files in a shapefile. The tables were also created automatically. The shapefile contains geographic features, representing points, lines, or polygons.

PostGIS is a relational and spatial database based on PostgreSQL. Likewise QGIS, PostGIS is free, open-source and cross-platform. A spatial database stores data quite similarly to normal databases, but also has spatial functions that support the creation of geometric and geographic queries. Note that a spatial database allows the storage of non-spatial data. Besides PostGIS, there are some similar alternatives based on Oracle, SQLite, MongoDB, etc [15].

By doing segment analysis, instead of having an infinitude of points, we will have around 40000 segments. These can be grouped in the future, making it possible to analyze a larger segment, a road or a route. Thus, we no longer have the spatial component to influence the analysis, reducing in this way the complexity of the data.

OpenStreetMap (OSM) [16] is an open-source project that provides geographic data for free, for people to use it as they want as long as they credit OSM and its contributors. OSM counts with the contribution of a big community. OSM contains information about roads, bus routes, trails, etc. Besides that, it also provides GTFS information and can help in its construction [16].

There are several web applications, libraries, APIs, and other tools (Java, C/C++, Python, and Javascript) that allow us to work with OSM data. Some of those can give us, for example, the roads with information about the maximum speed. Such information can be very useful if we want to study, for example, driving behavior, etc.

2.2 Time-series

A time-series is a discrete function whose value depends on time, as shown in Figure 2.1. Because of that, it is important to preserve the sequence of values. A time-series shows the evolution over time of a certain value. For example, it can show the evolution of the temperature through the day, month, year, etc. It is important to take into consideration the metrics that we are working with. Some types of data are continuous values, like temperature or speed; others can be counts like the traffic flow observed or the number of sold items.

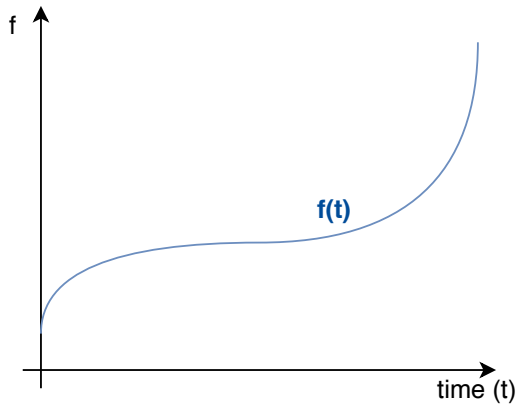


Figure 2.1: Time-series

2.2.1 Components of a time-series

A time-series can be decomposed in four components: trend (t), seasonality (s), cyclic (c) and residual (r) [17]. The components are the result of the existing patterns in the time-series. A time-series can be expressed as:

$$f(t) = t(t) + s(t) + c(t) + r(t) \tag{2.1}$$

There are two types of models to decompose time-series: additive models and multiplicative models. Equation (2.1) assumes that the model is additive. In a multiplicative model, instead of adding components, we multiply them, making the model sensitive to zero values. If seasonal variation is constant over time we use the additive model. If the model is multiplicative, the components increase or decrease over time [9].

If a time-series has an upward or downward evolution, then it has a trend. This evolution can be linear or non-linear and can be obfuscated by seasonal fluctuations and noise [17]. The seasonal component depends on seasonal patterns. A seasonal pattern is a consequence of seasonal factors and has a fixed frequency. For example, the school calendar can influence more or less traffic nearby schools in specific times of the year. The cyclic component is related to cyclic events that occur from time to time without being related to calendar events. In a cyclic pattern, there is not a fixed frequency. When we eliminate these three components from the data, it can remain some residual data, so we get the last component, the residual component [9].

2.2.2 Measures of dependence between variables

One of the most popular ways to measure dependence between variables is by using correlation. If a variable is very related to another, their relationship is strong, which means that the correlation coefficient will be very close to 1. On the other hand, if they have nothing in common, their relationship is weak, which means that the correlation coefficients will be very close to 0.

Pearson correlation

Pearson correlation is one of the many types of correlating variables. It is better used to detect linear relationships, and assumes a Gaussian distribution of the data [18].

Correlation of time-series

Cross-correlation compares two time-series and tries to detect a correlation between features with the same maximum and minimum values. Autocorrelation compares the same time-series but at different times. This type of correlation can detect patterns or seasonality. Normalized cross-correlation is similar to cross-correlation but can compare metrics with different value ranges. Normalized autocorrelation is the same as normalized cross-correlation, but for autocorrelation [17, 19].

2.2.3 Stationarity

Stationarity in a time-series implies that, when it occurs a shift in time, the distribution of the data is the same. When a time-series is nonstationary, that is because of the existence of unit root. If the time-series is not stationary, then we must make it stationary. This is a crucial step to forecast future values using statistical methods or make predictions using deep learning methods.

To evaluate if a time-series is stationary or not, visualization techniques can be helpful; however, it is not always enough. In this context emerge the *Augmented Dickey-Fuller* (ADF) test [20].

Augmented Dickey-Fuller test

The ADF test is a statistical test, more precisely, a unit root test, that tests how strongly a time-series can be characterized by a trend. If the time-series is not stationary, it is necessary to apply differencing to make it stationary [20].

The differencing technique will subtract the previous lag to the actual lag. This type of differencing is called first-order differencing. In some cases can even be necessary to apply second-order differencing, which means to apply for the second time the first-order differencing [17].

This test has a null hypothesis that assumes that the time-series can be represented by a unit root. So, as null hypothesis is defined that the time-series is nonstationary. Therefore, if the null hypothesis is rejected, it can be concluded that the time-series is stationary. The hypothesis test is verified by the calculated p-value. If the p-value is lower than a threshold, then the null hypothesis is rejected meaning that time-series is stationary [20]. Usually, the threshold values used are 0.01 (for a confidence of 99%) or 0.05 (for a confidence of 95%).

2.2.4 Smoothing

Sometimes, time-series data can have several small fluctuations. Those small fluctuations make the time-series difficult to study, and in some cases, even impossible to study. To overcame these fluctuations, it was proposed several smoothing methods over the years. A smoothing method aims to remove noise from data without changing the core features.

The choice of the best smoothing method can be difficult. It can depend on the type of analysis that we want to make and the type of results that we want to achieve. The best way to choose a smoothing method is to apply it and then check what happens to the data.

There are several ways of smoothing data. For example, *Moving Average* (MA) smoothing allows the removal of small variations between time steps. This type of smoothing is simple and easy to calculate. It is necessary to define the window size that is going to be used to calculate the MA process. To apply this type of smoothing, the time-series should be stationary [20].

Some smoothing methods can be based on interpolation of points. It is possible to do interpolation using a linear function, a cubic function, a spline function, etc.

Savitzky–Golay smoothing

The Savitzky–Golay smoothing [21], also known as Savgol smoothing, fits a set of points without changing the signal tendency, applying locally a polynomial function. The number of points that is applied is configurable and is called the window size. Usually it is preferred polynomials of a lower degree, and the window size needs to be bigger than the polynomial degree.

2.3 Statistical Methods

Time-series can enhance several types of studies. One of the most used is the forecasting of future values. To perform forecasting it start to appear several methods; like historical methods, and statistical methods. While historical are based only on what was observed previously; statistical methods are based on the study of statistical features. This section presents the statistical method used: seasonal *AutoRegressive Integrated Moving Average* (ARIMA).

2.3.1 AutoRegressive Integrated Moving Average (ARIMA)

ARIMA [9] is a statistical model for analyzing time-series. There are two types of ARIMA models: non-seasonal ARIMA (also know just as ARIMA) and seasonal ARIMA (also know as *Seasonal AutoRegressive Integrated Moving Average* (SARIMA)).

The ARIMA model, expressed in equation 2.2, can be subdivided in three models: *AutoRegressive* (AR), Differencing (represented as I), and *Moving Average* (MA). For each one of these partes there are a parametric component that can be studied: p, d, and q. [9]

$$ARIMA(p, d, q) = AR(p) + I(d) + MA(q) \tag{2.2}$$

The parameter p referes to the order of the AR part of the model. The parameter d is the degree of the first differencing involved. The parameter q referes to the order of the MA part of the model.

2.3.2 AutoRegressive (AR) model

The AR model can be represented as $AR(p)$. AR models are good at learning the trend of time-series data [17]. If the order of the model is 1, then the model can be described as

[22]:

$$X_t = \phi_1 X_{t-1} + \epsilon_t \tag{2.3}$$

The AR model is based on the concept that, if two observations are related to each other, and one happens before the other, then they are correlated. A consequence of the model is that, if an observation t depends on the previous observation $t-1$, and if the previous observation $t-1$ also depends on a previous observation $t-2$, then the observation t also depends on the observation $t-2$, and so on. The equation 2.4 expresses the previous observation $t-1$, if the order of the model is 1 [22].

$$X_{t-1} = \phi_1 X_{t-2} + \epsilon_{t-1} \tag{2.4}$$

The letter ϕ represents a coefficient that can correlate an observation with a previous observation. ϕ_k means that the degree of the polynomial is k. In the case of equations 2.3 and 2.4, the degree of the polynomial is 1. The ϵ_t represents the white noise that can exist.

If the order of the model is p, then the equation is expressed as 2.5

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + \epsilon_t \tag{2.5}$$

As shown in equation 2.5, if we increase the order of the model, that means that an observation is more dependent on the previous observations.

2.3.3 Moving Average (MA) model

The MA model is similar to the AR model, but it is focused on the white noise, using the previous white noise to forecast the next observation. The MA model can be represented as $MA(q)$, being q the order of the model. When the order of the model is 1, the model can be expressed as in equation 2.6 [22].

$$X_t = \epsilon_t + \theta \epsilon_{t-1} \tag{2.6}$$

If the order of the model is q, then we get an equation as in 2.7.

$$X_t = \epsilon_t + \theta \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + .. + \theta_q \epsilon_{t-q} \tag{2.7}$$

2.3.4 Differentiation

There is a model based on the AR and the MA model called *Autoregressive Moving Average* (ARMA). ARIMA adds an extra component to an ARMA process: the differencing model. Differencing a time-series can be important, if the time-series is non-stationary, as it was previously mentioned. Since the ARIMA model deals with differencing, verify if the time-series is stationary or not becomes insignificant. The differencing model will eliminate the trend and seasonality from the data, and focus on the remaining aspects of the data [17, 22].

If the order of the model, represented as $I(d)$, is 1, then it can be represented as:

$$\Delta_{X_t} = X_t - X_{t-1} = \epsilon_t + \theta \epsilon_{t-1} \tag{2.8}$$

2.3.5 Seasonal ARIMA

The seasonal ARIMA can be represented as:

$$ARIMA(p, d, q)(P, D, Q)m \tag{2.9}$$

In the seasonal ARIMA model, there are 2 parts to take into consideration. The first one ((p, d, q)) refers to the non-seasonal part of the model. The second one ($(P, D, Q)m$) it is for the seasonal part of the model, where the parameter P is for the order of the seasonal AR part of the model, and the same happens to the D and Q part. The parameter m is for the number of periods per season. Usually, all these parameters assume values like 0, 1, or 2. More complex models can assume other values, those values have always to belong to the set of natural numbers $\mathbb{N}$.

2.4 Machine learning

Machine learning is the ability of computers to learn using big volumes of data, and without being programmed. There are three types of machine learning. The first one is supervised learning, and it is characterized by having labeled data, feedback, and the goal is making a prediction. The second one is unsupervised learning whose goal is to find hidden patterns in the data. The third type is reinforcement learning and aims to learn a set of actions. [23] Deep learning is a subset of machine learning and belongs to the supervised learning algorithms. In this dissertation it will be used deep learning algorithms, designated by *Artificial Neural Network* (ANN)s, to predict the traffic flow observed.

As the name suggests, ANNs [24] are inspired by the concept of neural networks present in neurobiology. There are several types of ANNs, like *FeedForward Neural Network* (FFNN) [25], *Recurrent Neural Network* (RNN) [25, 26], *Convolutional Neural Network* (CNN) [25, 26], etc. In order to use ANNs to make predictions, it is necessary to train the network before.

A simple example of an ANN is a single neuron, also known as the perceptron. The perceptron model can have several inputs where each input is multiplied by a weight. The obtained values are summed and go through an activation function. We have also a bias term that will make a shift to the activation function, and it will be produced an output. In this example, we have one hidden layer with one activation function [27].

Usually, an ANN is formed by three types of layers, as represented in figure 2.2. The first type of layer is the input layer, the second one comprises the hidden layers (one or more), and the last one is the output layer. Each layer can have one or more neurons. In this case, the first layer has four neurons, the second has five neurons, the third has five neurons, and the last one has one neuron. The arrows represent the connections between neurons in different layers. Each neuron is fully connected with the neurons of the next layer.

The input layer receives the data, transforms the data multiplying it by weights, and then does the feed-forward propagation of the data to the next layer. The weights start at random values in the input layer, and because of that, there will be different outputs for the same inputs and network configurations [24].

The data can be propagated by several layers that belong to the hidden layers' group. In each layer, the neurons are multiplied by a weight. At the end of each step of the training process, it will be made a prediction that it will be compared with the actual value using a loss function. The loss function will produce a score that is important to adjust the weights

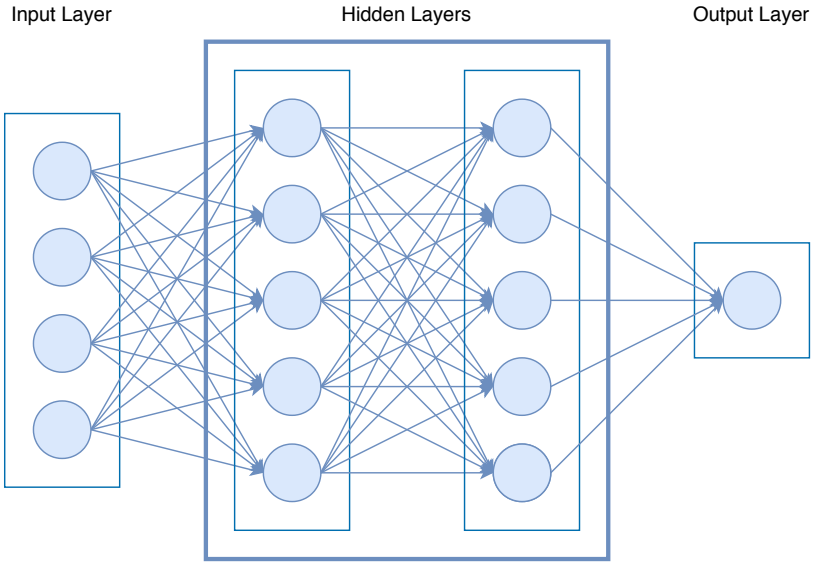


Figure 2.2: Example of an ANN.

---

of the hidden layers in the next epoch. The loss score increases as the difference between the true value and the predicted value increases. The optimizer is responsible to make this correction.

An epoch is a step in the training process. It is important that the dataset passes several times through the network, because the network can not learn if there is only one epoch. If the network does not learn, then we will have an underfitting of the data.

Underfitting is one of the problems that may occur. We can also have overfitting of the data. Overfitting happens when the model can perfectly describe the data. It may sound good, but it is not. The problem with overfitting is that the model is not generalized. Sometimes it is necessary to make the dropout of some neurons, this is, some connections are "turned-off" to ensure that it does not happen the overfitting of the model, as shown in figure 2.3 in the third layer. In other words, there are not always fully connected layers (dense layers) as it is presented in figure 2.2.

Activation functions should be non-linear, because they allow non-linear transformation to the data, but this does not mean that they have to be complex. This is important because, if the data is not linearly separable, then it will not be possible to separate it using linear functions. Some common activation functions used are *Rectified Linear Unit* (ReLU), sigmoid, *hyperbolic tangent* (tanh) and softmax. The first three functions are represented in figure 2.4.

Activation functions allow the separation of data classification in neural networks. The ReLU [24] function is very popular in neural networks and is 0 if the values are negative, and is the value otherwise. The function can give any value in $\mathbb{R}_0{}^+$, being a sparse function. The sigmoid [24] function limits the output between 0 and 1. For values close to 0, the sigmoid function will present very different values, for negative values, the sigmoid function tends to 0 and for positive values, the sigmoid function tends to 1. This function presents a small dispersion of the data when compared with the ReLU function. The tanh [24] function curve is similar to the sigmoid function, but the range of the values go from -1 to 1. Sigmoid functions and tanh functions are very used in RNNs [17]. The softmax [24] function gives

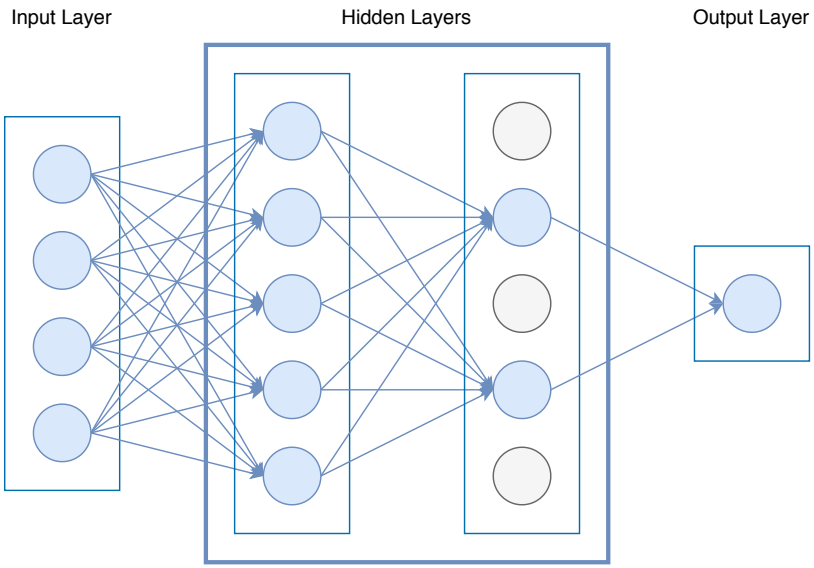


Figure 2.3: Example of an ANN with dropout.

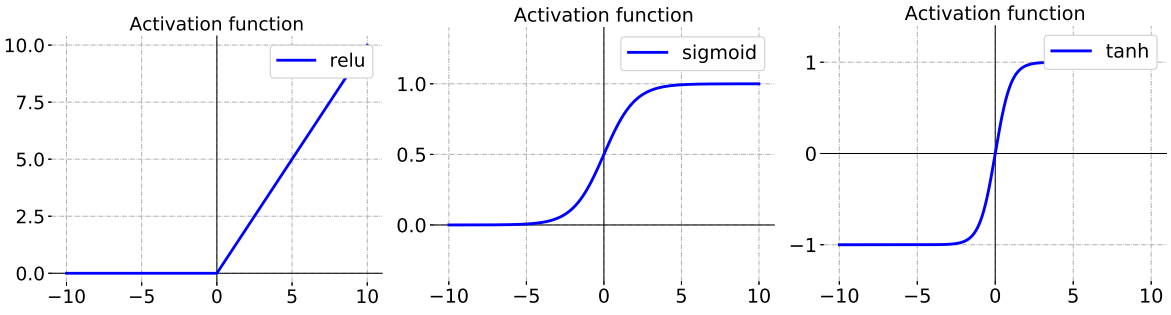


Figure 2.4: Activation functions

the probability distribution, for that reason the values belong to the interval between 0 and 1; however, if we can have several distinct output values, the function will give very small values. Sigmoid and tanh functions seem the best function to work with time-series.

The simplest type of neural network is the FFNN. As the name might suggest, in this type of neural network the information goes forward through the network and does not make any type of cycle in the hidden layers. Some types of networks perform cycles between hidden layers or even in the same neuron. There is a special case of a FFNN called a *Fully Connected Network* (FCN), which is basically a FFNN with dense layers.

RNNs are a type of neural network used to make predictions, being more suitable for sequential data. The process is made by iterating over data and saving a state in memory. The major difference between FFNN and RNNs is that RNNs can perform loops in the hidden layers. There are several types of RNNs, being *Long Short-Term Memory* (LSTM)s [25] one of the most used, especially for sequential data.

One of the most common problems associated with RNNs is the vanishing gradient problem. This problem is characterized by the incapability of changing the weight values as time goes by, creating a vanishing effect. LSTMs were created to solve this problem. The network can save information across many timesteps, to use it later [24]. The LSTM networks present loops not only in the hidden layers but also in the neurons. A consequence of these loops is that LSTMs become more complex computationally.

2.5 Evaluation Metrics

Evaluation metrics measure the models performance, being helpful to choose the best model. It will be presented some of the metrics that can be used.

Mean Squared Error

Mean Squared Error (MSE) [9, 25] can be described as in equation 2.10. Y_i represents the observed value, and $\hat{Y}_i$ represents the predicted value. MSE is given by the sum of the square of the differences between the predicted values and the true values, and then it is divided by the number of elements, represented by n.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.10}$$

The MSE value is always larger or equal to zero. If the predicted value and the true value are equal, the error is zero. As the difference increases, the error increases. A good model has a MSE close to zero.

Root Mean Squared Error

Root Mean Squared Error (RMSE) [25] is the square root of MSE, as shown in equation 2.11. If the value of MSE is lower than 1, then the value of RMSE is bigger than the MSE value. Otherwise, the value of RMSE is lower than the value of MSE. Similar to MSE, RMSE

value is always bigger or equal to zero, and a good model has a RMSE close to zero.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} = \sqrt{MSE} \tag{2.11}$$

Mean Absolute Percentage Error

Equation 2.12 gives us the *Mean Absolute Percentage Error* (MAPE) [9] formula. Looking for the sum part, MAPE divides the absolute error value between the predicted value and the true value by the true value. This can have huge consequences because, if the true value is zero, then we will get infinite. Since the traffic flow observed can have zero values, this metric is not good. Besides that, all it takes is a small true value for the MAPE value to be huge.

$$MAPE = \frac{1}{n}(\sum_{i=1}^{n}\frac{|Y_i - \hat{Y}_i|}{Y_i}) * 100\% \tag{2.12}$$

Mean Absolute Error

Mean Absolute Error (MAE) [9] gives the average of the absolute error values, being always bigger or equal to zero. MAE does not penalize big errors, as much as MSE [28].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \tag{2.13}$$

Explained variance

Explained variance measures the dispersion of the data. Var represents variance. For a good model, the obtained value should be close to 1.

$$ExplVar = 1 - \frac{Var(Y - \hat{Y})}{Var(Y)} \tag{2.14}$$

Coefficient of determination

The coefficient of determination is also known by R^2-Score and can be calculated as in equation 2.15. $MSE_{baseline}$ is similar to MSE, but instead of using the predicted value, it uses the mean of the observed. A good model should have a score close to 1. The maximum value of $R^2 - Score$ is one, and the minimum value is -∞. However, any value that is not positive means that the model is worst than predicting the mean [28].

$$R^2 - Score = 1 - \frac{MSE}{MSE_{baseline}} \tag{2.15}$$

Akaike Information Criterion

Akaike Information Criterion (AIC) and *Bayesian Information Criterion* (BIC) are two evaluation metrics very used in statistical methods. They select the best model based on

the log-likelihood and complexity of the model. AIC is based on frequentist probability and penalizes complex models less [29].

AIC is expressed in equation 2.16. $\hat{L}$ is the likelihood and p is the number of estimated parameters [9].

$$AIC = 2p - 2\ln(\hat{L}) \tag{2.16}$$

Bayesian Information Criterion

BIC is based on Bayesian probabilities and penalizes the model with a bigger complexity. A very complex model has fewer probabilities to be chosen. [29]

BIC can be obtained using AIC, as shown in equation 2.17. [9]

$$\begin{aligned} BIC &= AIC + p(\ln(n) - 2) \\ BIC &= 2p - 2\ln(\hat{L}) + p * ln(n) - 2p \\ BIC &= -2\ln(\hat{L}) + p * ln(n) \end{aligned} \tag{2.17}$$

2.6 Summary

This chapter introduces the type of data used, the problems associated, present methods to perform forecasting and prediction, and presents some evaluation metrics. These problems are related to the components time and space. It is possible to simplify the space component by using a solution based on road segments. To simplify the time component we use time-series.

Once the data is ready, it is necessary to perform analysis to identify patterns and characteristics associated with the data, decomposing the time-series can be very useful. The use of cross-correlation allows the identification of similar time-series and the use of autocorrelation allows the identification of similar time periods. If the data present to significant noise it can be used smoothing techniques for a more focused study.

To forecast time-series data there are two types of methods that can be used. The first type is the statistical one, and an example is the SARIMA model. SARIMA models are deterministic and computationally lightweight; however, they are very limited in the prediction they can make. The second type is deep learning methods, and we can use FFNN or LSTM. LSTM is very used with time-series data because it can save old information in the network and use it when it is necessary. Deep learning methods are computationally heavier but allow long-term forecasting.

Chapter 3

State of the art

The work developed in this dissertation covers the process of dealing with temporal data, GPS data, forecast and predict the traffic flow observed, detecting anomalies, classifying driving behavior, and detecting problematic areas. For a more comprehensive study, this chapter will present the state of the art on this area.

This work deals with several topics from different areas. The first section of this chapter contains the related work associated with smart urban mobility. The second section presents some works developed to forecasting urban mobility data. The third section presents some works with the goal of classifying driving behavior. The last section presents a discussion about the several mentioned works.

3.1 Smart Urban Mobility

Smart cities aim at offering to their citizens a more sustainable, optimized and safe city: a place that can offer a better quality of life. Smart cities are urban spaces where data is collected from sensors in order to better manage the resources and available services. For this to be possible, the data will be processed and analyzed.

Traffic jams, lack of parking, air pollution, noise pollution, and road safety are common problems in many cities and tend to get worse. For that reason, planning urban mobility becomes necessary. In order to improve it, we can propose changes in the infrastructure, like expanding the number of roads or lanes, but cities cannot always implement such changes. Even if they can, it is not always enough. The best thing we can do is to improve how we use the actual infrastructure. In this context, *Intelligent Transportation Systems* (ITS) emerge. ITS aims to improve the mobility of people and cargo, safety, productivity, efficiency, and decrease pollution in transportation [30].

The study of urban mobility data is essential in ITS. Urban mobility data can have different sources, it can be from different types, and it can have different risk levels (sensitive data, quasi sensitive data, and public data) [31]. Liu et al. [31] propose a framework to manage smart city data with the described characteristics. It was necessary to perform data anonymization and have different forms of storage, publish, and retrieve.

It is possible to estimate urban mobility indicators by using telecommunication data. Vidovic et al. [32] estimate the number of trips, travel time, and distance time using voice calls, text messaging, and internet access data.

Pagani et al. [33] performed a knowledge discover mechanism to extract features from

car-sharing data services to calculate travel time, vehicle flow, identify congestion zones, etc. They verify that cars have a lower speed during the day, and the fuel consumption is bigger during the day.

Dontu et al. [34] used Weigh-in-Motion system data to estimate vehicle dimensions (volume and weight) and the type of vehicle. They used this information to estimate the level of pollution. They verified that car consumption is bigger at lower speeds, which agree with the conclusions made in [33].

To minimize traffic jams, reduce stress, and decrease pollutant emissions, the authors in [35] proposed a method for optimizing traffic light green phase.

This chapter will be focused on how we can analysis the vehicular traffic in the city of Porto and Aveiro, and the detection of patterns, anomalies and the forecasting of some metrics. Urban mobility patterns and anomalies are the result of a set of individual choices. Those choices are made taking into account the source and destination, the fastest route versus the most economical route, etc. Sometimes it can be influenced by individual, social, or cultural preferences.

3.2 Urban Mobility Forecasting

Several approaches have been proposed to address the forecasting of traffic flows, ranging from historical methods to machine learning applications.

Historical Methods

Historical methods predict future values naively using just historical data. In *Historical Average Forecast* (HAF) [6] only the previously observed patterns are taken into account. It can be given more weight to recent events, in order to improve the model. Another way to control the events that are considered is done by using a fixed moving window. This model can be seen as a fitted ARIMA, like Billy Williams et. al. [6] mentioned. This happens because one of the parameters of ARIMA can be recalculated for each time of the day as a weighted average leading to become very similar to HAF.

Statistical Methods

The traffic flow forecast using statistical methods is not a new topic. It has been studied for over 35 years. In 1984, Okutani et al. [36] published a paper about how they could use Kalman filtering theory to predict traffic volume, even with minimal computational resources. In the following, it will be presented some of the most recent statistical studies done on forecasting traffic flow.

In 2003, Williams et al. [6] evaluated traffic flow data from two different locations aggregated in 15 minutes intervals, and divided the data from each location into two groups: test and validation. It then analyzed vehicles per hour and vehicles per hour per lane, comparing the real results with the ones forecasted using SARIMA, *Exponentially Weighted Historical Average* (EWHA), and *Deviation from Historical Average* (DHA). The best results were obtained with SARIMA.

Clustering

Clustering techniques aim to group data into clusters with certain common features. The final goal is to predict which cluster a certain point belongs.

In 2019, Liu et al.[5] developed an approach based on *Temporal Clustering and Hierarchical Attention* (TCHA), to make short-term traffic speed predictions. Temporal Clustering is used to obtained several clusters and group data with similar traffic patterns. The similarity is calculated using the Pearson correlation function. The model uses two attention mechanisms, one to capture spatial features, and another one to capture temporal relations. At each time step, it is determined the most relevant features. The proposed method was compared with a *Historical Average* (HA) model and a *Gated Recurrent Unit* (GRU) model. According to the authors, the TCHA method could retain more temporal and spatial information.

Artificial Neural Networks

Machine Learning comprises several areas of study, including Deep Learning. The main goal of Deep Learning is to extract patterns from data. Deep learning models are inspired by the biological nervous systems and are called ANN [27]. An ANN can be composed of several layers of neurons, whose information flows through the layers [27].

In 2018, Wu et al [37] developed a *Deep Neural Networks* (DNN) based traffic flow prediction model that takes into account the spatial-temporal characteristics of the data. In the beginning, the data goes through an attention model that will determine how correlated the past data is with future data. Then, the data will be divided. The model is a mix of two types of DNNs. For the spatial component, it is used a CNN. For the temporal component, it is used an RNN. At the end of the network, the model used a regression method to link both networks. This method takes into consideration the data from the last day, last week, and future data.

In 2019, Guowen et al. [38] proposed a model using a five-layer GRU network, which is a type of RNN with a gating mechanism to make a short-term traffic flow prediction. Due to the bad results obtained when compared with CNN, they performed a spatio-temporal feature selection using a *Spatio-Temporal Feature Selection Algorithm* (STFSA) before applying the GRU model. The authors used Pearson correlation to make a spatial correlation analysis and a temporal correlation analysis. The spatial correlation analysis allows the selection of the best spatial points, and the temporal correlation analysis allows the selection of the best periods.

Also in 2019, Bartlett et al. [10] investigated the influence of the use of short and long term patterns to get a more accurate prediction, and developed a *Dynamic Temporal Context Neural Network* (DTC) framework. The DTC model uses short and long term patterns as features, and determines the most relevant through online learning. Online learning has some issues associated, as time goes by, some long term patterns can be lost. However, this model can dynamically determine the most relevant patterns for the regression GRU model used. They achieve better results using both temporal patterns when compared with a GRU model. One major limitation of this work is that they do not include spatial dependencies, only one geographic point was analyzed.

Xiaolei Ma et, al. [39] proposed, in 2015, a LSTM neural network to capture nonlinear traffic dynamics to make short-term traffic prediction. This type of network can determine the optimal time lags, which can improve performance. Longer time lags lead to better

performance. The ability to choose the optimal time lags can have a huge impact on the accuracy of the model. The authors implement several other methods, like a *Support Vector Regression* (SVR) [39]. The SVR can obtain accurate prediction results if the parameter settings are well chosen. However, the LSTM neural network obtained better accuracy and stability results.

3.3 Driving behaviour and safety

Mobility has a fundamental role in human life: in social interactions, it has economic factors associated, cultural impacts, etc. In the last decades, it is notorious the significant increase in mobility, and with that increase, it became necessary to study it. One of the most discussed topics is driving safety.

There are many factors that can contribute to safe or unsafe driving behavior. Some of those factors are related to the driver's attitude towards driving such as speed, acceleration, maneuver signaling, safety distance, the respect for traffic signs, etc [40]. However, many other factors can influence safe driving, those factors are resumed in Table 1.

Table 3.1: Factors that influence safe driving

Types of factors	Factors
Human	Age, genre, emotional state, fatigue, sleepiness, consumption of alcohol, medication, and other substances [40].
Environment	Visibility, road grip, stability (influenced by the wind), road condition, and vehicle condition [40].

For a complete study of the driving behavior and driving analysis, it would be required a wider knowledge about the driver, the vehicle, the road, the weather conditions, etc. For example, some of the studies found used mechanisms to detect drivers distraction [41]. Knowing our dataset helps us focus on just the studies that use the same type of data that we have access.

One of the ways of study driving behavior is through the study of speed and acceleration to understand if the driver presents a safe or unsafe driving behavior. But, how can we describe safe driving behavior? A conductor that has a safe driving behavior respects the maximum limits of speed imposed, and does not make abrupt changes on the speed that leads to sudden accelerations or decelerations.

Studies based on speed and acceleration usually are based on the G-G diagram [42]. The G-G diagram gives the maximum acceleration that is possible to achieve for a given speed. It takes into consideration the longitudinal and lateral accelerations. However, there are other types of models.

Derbel et. al [42] propose a system to evaluate the driving risk. This system is based on three types of factors. It is made a fusion of information about the driver, the vehicle, and the environment. The fusion happens at two levels. The first level is related to the type of factor and is made using Dempster-Shafer Theory [43]. The second level of fusion is a global fusion and is based on a Fuzzy theory [44] to designate basic probability assignment functions. The information about the driver that is considered relevant is the age and genre of the driver, since statistical studies reveal that they influence the probability of having an accident. They call vehicle information, the information about speed and acceleration, and

use it as an indicator of the aggressiveness level. The environment information is based on statistical studies about the place, the time of the day, and the day of the week [42]. The vehicle factors are studied at two different levels. The first one is based on the Euclidean acceleration norm [45] and the speed of the vehicle. The second one is based on the G-G diagram, with lateral and longitudinal acceleration to identify danger left and right turns [42].

Eboli et. all [46] based their study on speed, acceleration, and friction. Once again, the starting point is the G-G diagram. This model is more ethical than the previous one, because statistics can not define a person.

Equation 3.1 to 3.5 explains how the authors created the model, starting from the G-G diagram. The starting point is from equations 3.1 to 3.3.

Equation 3.1 gives the acceleration modulus.

$$|\vec{a}| = \sqrt{a_{lat}^2 + a_{lon}^2} \tag{3.1}$$

Equation 3.2 is the second law of Newton, F_s is called by stimulated force, being m the mass of the vehicle.

$$F_s = m * |\vec{a}| \tag{3.2}$$

Equation 3.3 presents the resistant force (is a frictional force), μ is the coefficient side fiction and g is the gravitational weight. The side friction coefficient depends on the road material and meteorological conditions.

$$F_r = w * \mu = m * g * \mu \tag{3.3}$$

Using the three equations they get the result presented in equation 3.4. In a safe driving behavior, the F_s should be smaller than F_r.

$$F_s = F_r \Leftrightarrow m * |\vec{a}| = m * g * \mu \Leftrightarrow |\vec{a}| = g * \mu \Leftrightarrow a_{lat}^2 + a_{lon}^2 = (g * \mu)^2 \tag{3.4}$$

They divide the friction into two components and obtain the maximum friction coefficient to consider a safe driving. Then they make the necessary substitutions and get the equation 3.5. Note that, they assume a dry pavement and rural road.

$$|\vec{a}| = g * [0.198 * (\frac{v}{100})^2 - 0.592 * (\frac{v}{100}) + 0.569] \tag{3.5}$$

Using this equation, the authors can draw the limits for acceleration given the speed, to classify a driving behavior safe or unsafe. The work developed in chapter 6 is based on this paper.

3.4 Discussion

The first section is focused on the study of smart urban mobility. This field is broad and it was developed many studies over time. We will focus on two types of studies: the ones related to traffic flow, and driving behavior.

This work will implement statistical and ANN models to predic the traffic flow. The statistical models are more transparent and reproducible. ANNs are seen as 'black boxes'. In some cases, statistical models can obtain similar results to ANNs. One of the advantages of

using ANNs is because they can handle multi-dimensional data and are very adaptable, as well as handling outliers, missing data or noisy data [47]. The statistical methods presented are good to detect patterns related with time.

Working with ANNs can be difficult. It begins with finding the weights of the inputs. The training process will update the model weights in each iteration; however, the optimization algorithm used may not lead to the minimum error or loss, and can lead to overfitting. The training process can last for days or even months. ANNs also require a lot of information and great computational power.

Driving behavior studies can be very complex; however, we focus on the ones that use the same type of information that we have access to. The model chosen is the one that only depends on speed, acceleration, road material, and meteorological conditions. This model has some limitations, namely, the limited geographical location. Using data from OSM is possible to extend our analysis to all the city.

Chapter 4

Use cases

This work considers mobility data available from two "living labs", in the cities of Aveiro and Porto. The two datasets, with parts in common, are substantially different. The differences in the datasets and the existence of some previous work, originated different opportunities for each case. This chapter describes the context and use cases for each case study.

4.1 Forecasting use cases for Porto

Porto LivingLab is a project that aims to study urban dynamics and is deployed in Porto. This project contemplates a multi-source sensing infrastructure, capturing information from a vehicular network, weather sensors, environment sensors, and people flows sensors. This data is characterized by having a spatio-temporal component. There is a common backend infrastructure to ensure consistency in the data models used. Data sharing is achieved by using different services like publish-subscribe middleware, RESTful API, etc. This project has three monitoring platforms (SenseMyCity, UrbanSense, and BusNet), provides free Wi-Fi service to public bus users, and allows the development of several research works [48].

Several works were developed in the NAP group related to this project. There are works about content gathering and content dissemination strategies [49] using Porto's bus network. Some relevant developed works were mentioned in Section 1.1 and are related to the creation of decision support dashboards, estimation of the bus time of arrival, etc.

4.1.1 Data sources and existing city infrastructure

We used three data sources from Porto. Veniam gives information about the buses, PortoDigital provides information about several sensors like traffic flow, air quality, noise level, weather, and some other types of data; however, only the traffic flow was used. PortoDigital also provides the GTFS information about the bus network of Porto.

Porto's bus network is complex and has several bus lines, stops. Figure 4.1 contains part of the network, including traffic flow sensors. The colored lines are the bus lines, the circles on the top of those lines are the stops, the markers are the traffic flow sensors and the clusters of markers are traffic flow sensors that were clustered because they were close.

The map was built with the information present in the GTFS files about the buses, and the geographical information of the traffic flow sensors.

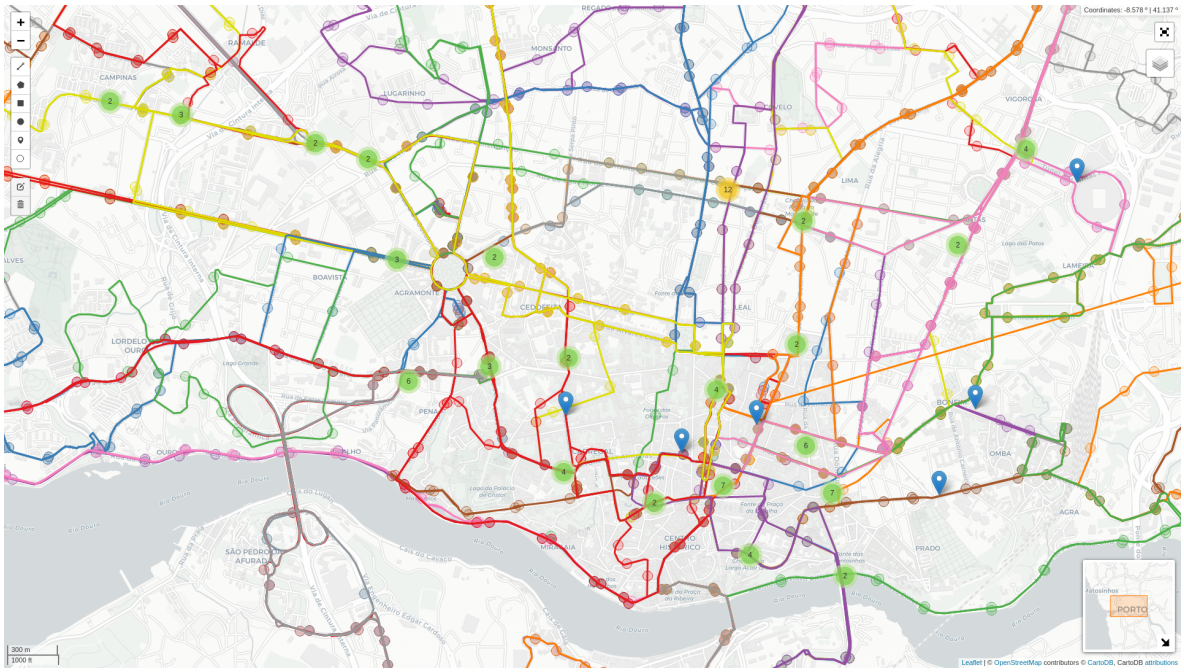


Figure 4.1: Bus lines, stops and traffic flow sensors

4.1.2 Selected use cases

Managing urban mobility is a difficult task that has to be performed by city managers. Sometimes, they can only identify problematic areas when the problems assume very big proportions. Predict traffic problems before they happen, or in early stages is one of the main goals. Since urban mobility data is complex, city managers would benefit from a computational system that was able to find patterns in traffic and predict traffic conditions.

This can be achieved by using the data from the buses, the traffic flow sensors, and the GTFS data. From the buses we have information about their speed and GPS position, from the traffic flow sensors we have the count of vehicles and the GPS position. Using the GTFS data we can associate GPS positions to road segments.

The main use case for the data from Porto is to predict and analyze traffic conditions. With this in mind, the following use cases were planned:

- Study the relationship between the buses speed and the traffic flow intensity,
- Analyze patterns in the traffic flow observed,
- Forecast traffic flow observed,
- Detect anomalies in traffic flow observed,
- Forecast traffic flow observed even in anomalous situations.

A good starting point is to explore if bus speed is correlated with the intensity of traffic flow sensors and try to understand, for example, if an increase in traffic leads to lower bus speed, and when/where there is more congestion. Detecting the congested roads could lead

to changing bus routes in order to get to the destination faster (note that there are many overlaps in the routes), or can highlight the need for adding more buses. Comparing temporal snapshots is also useful, for example, comparing the same interval in different years, as well as studying the impact of calendar related events. In this way, we can draw conclusions about the evolution of traffic over time. Several events can be considered, such as the school season versus school holidays, workweek versus weekend, holidays, etc. Using road segmentation, or by examining an individual road or route can highlight individual patterns.

4.2 Driving behavior use cases for Aveiro

In Aveiro, there is a sensing infrastructure constituted by buses, environmental sensors and weather sensors. This infrastructure was created by the NAP group to create a "living lab" for Aveiro.

4.2.1 Data sources and existing city infrastructure

In Aveiro, the project Aveiro STEAM City [1] aims the development of an urban platform and services that enable the management of the city based on a 5G infrastructure [50].

In the beginning of the year, members of our group start to deploy sensors and collecting units in some of the city buses. We already had some sensors and collecting devices in some strategic points of the city. Figure 4.2 contain some of the installed sensors, like the Smart Lamppost installed in Aveiro, and the ones that will be installed, etc.

Figure 4.2: Aveiro STEAM city sensors [1].

[1] https://www.it.pt/Projects/Index/4613

4.2.2 Selected use cases

Increase road safety is one of the big concerns of city managers. To increase road safety it is necessary to avoid accidents, mainly severe accidents. Sensitize drivers is the first step, but accidents can be caused by circumstantial reasons. Understand the reasons that lead to non-safe driving behavior plays a key role.

The main use case for the dataset for Aveiro is to analyze driving behavior. This use case can be subdivided in the following ones:

- Classifying safe driving behavior versus unsafe driving behavior,
- Identify road segments and zones that can be problematic,
- Compare temporal snapshots:
  - Compare different hours of the day of the same day,
  - Compare different periods like earlier in the day, midday, and end of the day,
  - Compare different days and different sets of days.

One of the most interesting applications is driving quality monitoring, such as hard acceleration and braking, speeding, degraded pavement detection, frequent braking locations, and the most dangerous areas. In this work we will focus on the driving quality aspect.

4.3 Summary

The architectures, presented in this chapter, were separated because it was created two different studies. One for the dataset from Porto and the other for the dataset from Aveiro. These were motivated by the existence of different information for each one of the cities, which made necessary the existence of different use cases.

Both datasets have bus information, being the network from Porto much richer in the number of buses, and trips. However, the network from Aveiro has a smaller periodicity. From Porto, we have also traffic flow information.

Both datasets have data from the infrastructure. This data makes it possible to perform a more complete study. It is possible to analyze the bus data and the traffic flow data by road segment, and it is possible to get the maximum speed for road segments.

The main uses cases can be resumed as predict traffic flow observed and studying driving behavior.

Chapter 5

Forecasting the Traffic Flow

We evaluated both statistical and machine leaning methods to forecast traffic flow, based on previous observations. Such methods were explained in Sections 2.3 and 2.4.

It is possible to apply the statistical methods without doing any additional preprocessing besides the one mention in section 5.1. However, the same can not be applied to deep learning methods. Because of that, section 5.2 explains the need steps to predict the traffic flow. The third section presents the pipeline elements, including the developed algorithms (statistical and deep learning). The fourth section presents the process to forecast traffic flow using SARIMA and the results obtained. Section 5.5 describes the process to predict using deep learning methods, and the results. Section 5.6 contains the methods to detect anomalies and predict traffic when an anomaly happens. Section 5.7 presents system implementation. The last section presents the chapter summary.

5.1 Data set preparation

Chapter 2 mentioned that the data came from different sources, correspond to different types of information, and, besides that, had different formats. For those reasons it was necessary different pipelines for preprocessing the data.

Table 5.1 contains a resume of the data types, sources, and location. The data has 2 different sources, which can be subdivided into 3 types of data (bus speed data, traffic flow data, and infrastructure data). In this chapter, it was only used data from Porto.

Table 5.1: Data sources

Data source	Data type	Data collection location
Veniam	Bus data	Porto
PortoDigital	Traffic flow observed data	Porto
PortoDigital	GTFS data	Porto

The different data will be used in different ways. The traffic flow data will be used to make predictions of the future values of traffic flow observed. The data from the buses from Porto will be used to attempt to establish a relationship with the traffic flow data. The infrastructure data will give support to the previous tasks.

Table 5.2 contains the five weeks chosen to study the traffic flow. The first four weeks were used to study the data, and to train and test the predicting modules. Note that, the last

week was just used to observe anomalies, study them, and try to predict them since November first is a holiday.

Table 5.2: Calendar for studying traffic flow.

October 2020							
Week	Mon.	Tue.	Wed.	Thur.	Fri.	Sat.	Sun.
1	30	1	2	3	4	5	6
2	7	8	9	10	11	12	13
3	14	15	16	17	18	19	20
4	21	22	23	24	25	26	27
5	28	29	30	31	1	2	3

5.1.1 Veniam data

Veniam gives us data about the buses in Porto with the fields present in table 5.3. The most important fields are the speed, the GPS coordinates, and the timestamp when the data was collected. Data from buses arrives every minute.

Table 5.3: Table *node_data* from Veniam data source

Field	Data type	Description
id	integer	Identification of the database entry.
node_id	uuid	Identification of the node.
location_id	integer	Represents a vehicle or a collection of access points.
head	double	Heading, the direction in which the bus is traveling.
lon	double	Longitude of the bus.
lat	double	Latitude of the bus.
speed	double	The speed of the bus.
ts	timestamp	Timestamp when the data was collected.
write_time	timestamp	Timestamp when the data was written.

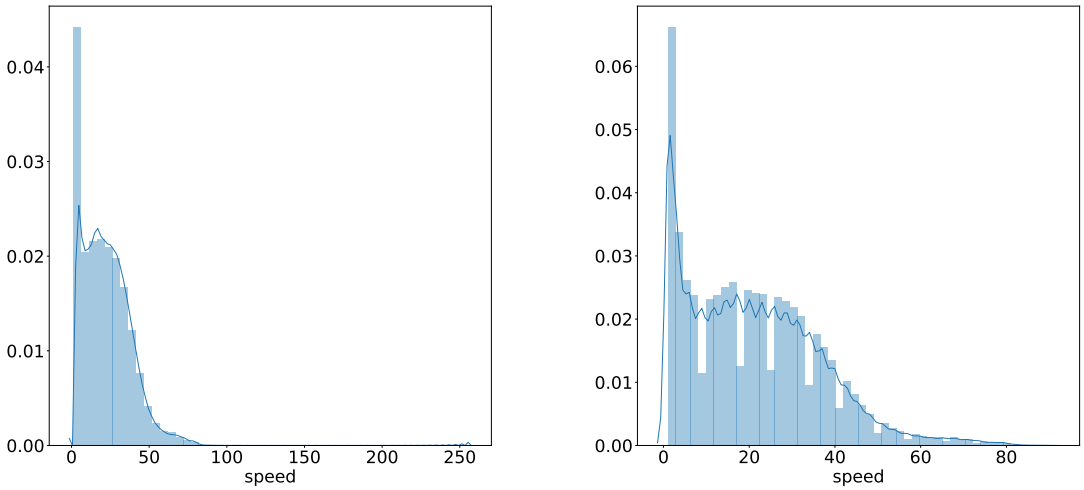
A first look into the data reveals some problems; table 5.4 includes some descriptive statistics. Those statistics summarize the central tendency and dispersion of the data, excluding *Not a Number* (NaN) values. There are some missing data, and others have to be discarded, as its values are unusual. Another problem is the imprecision of the GPS sensors, which can result in inaccurate positions or speeds.

As can be seen from the histogram in figure 5.1a, which shows the bus speed values, there are some very high values. If we remove the high values, and because they are rare events, and most likely, outliers, they can be excluded without having a huge impact. By removing them we obtain the histogram in figure 5.1b. Note that each graphic has a histogram with several bars and the bin size of the bars is calculated automatically. The curve is the application of a kernel density estimate function that gives the probability density at different values if the variable was continuous. The speed histograms show that the majority of the values are concentrated from 0 to 95, which means that the buses can reach speeds from 0 to 95 km/h.

Note that from the buses data, almost 50% of speed values are null, raising several questions as to why this happens. By comparing Veniam's information with information from the

Table 5.4: Statistics about speed data.

	speed
count	3693883
mean	21.000
std	15.589
min	1
25%	8
50%	19
75%	31
max	255



(a) Histogram of speed without a limit.

(b) Histogram of speed with a limit.

Figure 5.1: Histogram of speed

GTFS files, as can be seen in figure 5.2, it is possible to verify that the number of buses to circulate obtained from Veniam datasets approximates the expected number. The graphic was obtained after removing the null values. This may indicate that the null values obtained may occur because the vehicle is parked at the bus station or the drivers are taking a break. Note that, at the beginning of the graphic the number of buses given by Veniam is bigger than the number of buses given by the GTFS information, which can happen due to existing buses that arrive late to the station.

5.1.2 PortoDigital data

Table 5.5 contains the data available by PortoDigital regarding the traffic flow observed; table 5.6 contains the descriptive statistics.

The traffic flow dataset has also its flaws. There are some missing data, it has imprecise GPS sensors, and some of the values are abnormally high, as can be seen in histogram 5.3a. Briefly, the intensity graphic shows that the majority of the values are concentrated from 0

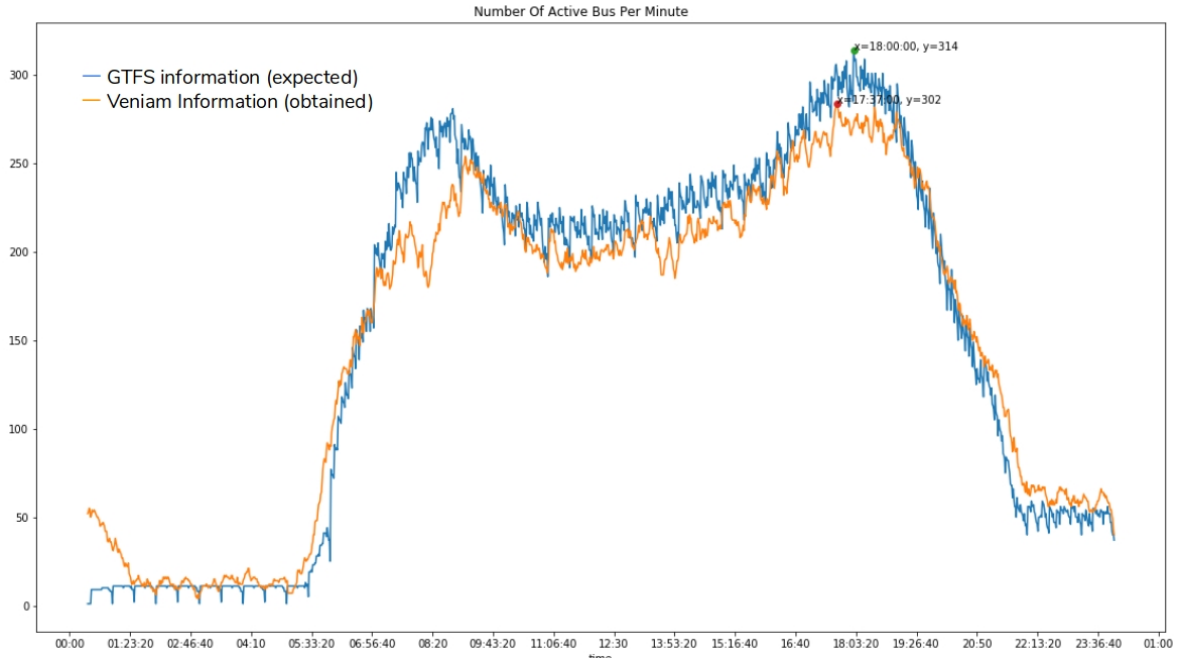


Figure 5.2: Comparison of the number of active buses (GTFS and Veniam information)

Table 5.5: Table *TrafficFlowObserved* from PortoDigital data source

Field	Data type	Description
id	Integer	Unique identifier.
type	String	Entity type (TrafficFlowObserved).
dateModified	DateTime	Last update timestamp of this entity.
dateObserved	DateTime	Contains two separate attributes: dateObservedFrom, dateObservedTo.
dateObservedFrom	DateTime	Observation period start date and time.
dateObservedTo	DateTime	Observation period end date and time.
intensity	Integer	Total number of vehicles detected during the observation period..
laneId	Integer	Lane identifier.
location	GeoJSON geometry	Location of this traffic flow observation.

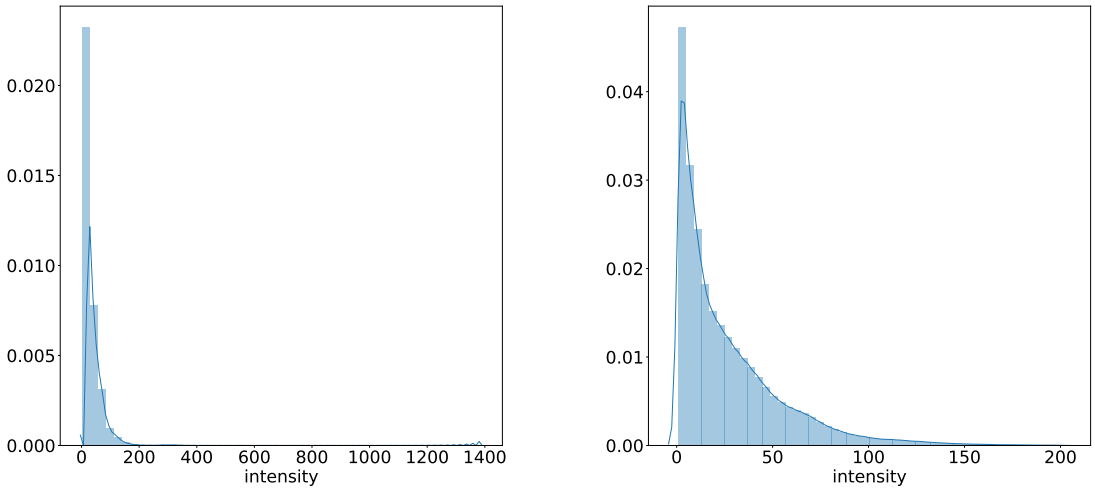
to 200, which means that the count is 0 to 200 vehicles per 5 minutes interval. The maximum value of intensity is 1386, as it was given in table 5.6, meaning that in 5 minutes there were 1386 vehicles, which gives an average of 4,62 vehicles per second.

The information about lanes is not trustworthy. In figure 5.4 there are four lanes; however, there are six traffic flow sensors in the same GPS position and with lane number from one to six. There is also some uncertainty as to whether a traffic flow sensor measures the intensity along the lane or just on one traffic lane. Finally, the traffic flow data is even more sparse than the bus speed data from Veniam, because it only sends the information every five minutes.

Figure 5.5 shows the GPS problems associated with the data. One of the traffic flow sensors is on top of a building. However, this sensor has normal values, as shown in figure

Table 5.6: Statistics about traffic flow observed data.

	intensity
count	727220
mean	30.015
std	56.910
min	1
25%	6
50%	18
75%	39
max	1386



(a) Histogram of intensity without a limit.

(b) Histogram of intensity with a limit.

Figure 5.3: Histogram of intensity.

5.6. This sensor presents some outliers that can be discarded. The outliers correspond to the six peaks in the graphic. These can be considered outliers because their values are much higher than the previous values and the next values. The sensor in node CT4Z3, although appearing to be well-positioned, has abnormal values; therefore, the sensor will not be used in the data analysis.

The preprocessing task is explained with more detail in section 5.3. Briefly, we perform a process of data reduction (select only the important data), data cleaning (remove missing data, outliers and noise), and data transformation (normalize). Besides that, we also can calculate the road segment of the sensor, if we need it.

5.1.3 GTFS Porto

The GTFS data available by PortoDigital was useful to solve the GPS problems associated with data from Veniam and PortoDigital. This data had to be manipulated because some of the segments were too big for our goals, and it was necessary to have small segments. Looking

Figure 5.4: Zone with 4 lanes and 1 point that contains 6 traffic flow sensors.

Figure 5.5: Traffic flow sensors location

---

at the GTFS data, it is possible to see the complexity of the Porto bus network, as can be observed in figure 5.7.

Figure 5.8 presents the structures of the given files. Note that there can be small structure modifications between GTFS files made by different entities. There is not a rigid fixed structure. This information was valid from June 15, 2019 to January 1, 2020.

This information allows us to know the buses that are supposed to be active and associate other sensors' positions to road segments.

5.2 Preparatory traffic flow data analysis

The traffic flow is a specific type of data called count data. We selected a specific traffic flow sensor to focus the study in one time-series. That sensor has an identification of CT1Z8. As can be observed in figure 5.9, the traffic flow presents a high variability, which may hide existing patterns in the data. The figure represents the first week of data, and the vertical red lines symbolize the different days. By observing the figure, we can see patterns between days, and also seasonality. Nevertheless, the excessive variability presented, which can be categorized as noise, may be hiding other patterns and also making difficult to model the behavior of the traffic flow.

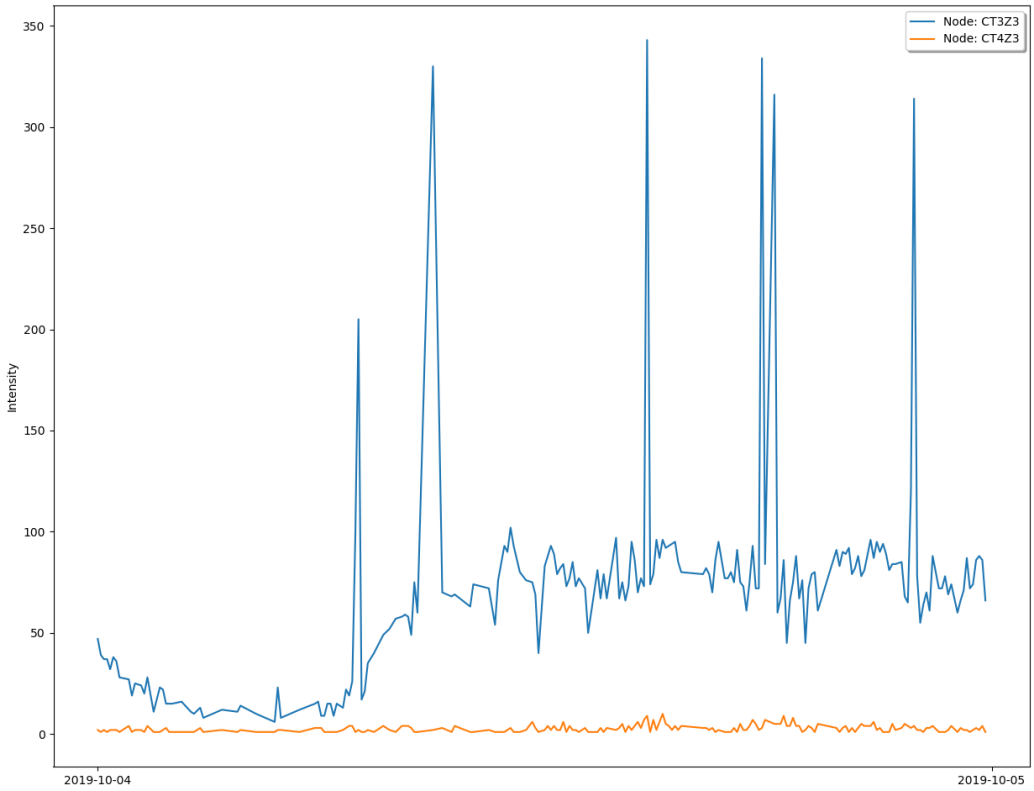


Figure 5.6: Comparison of the intensity of two traffic flow sensors

Figure 5.7: Visualization of the Porto bus network, defined by the GTFS dataset.

We try to study the traffic flow without using any more types of preprocessing; however,

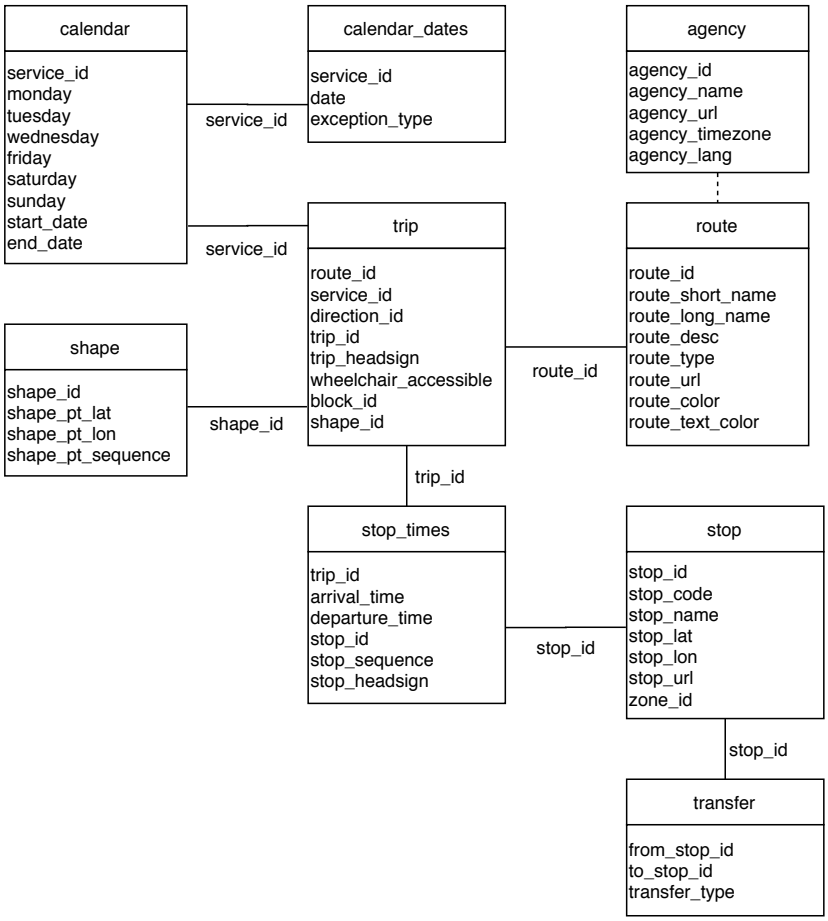


Figure 5.8: GTFS Porto - files structure.

---

the data was just too complex to analyze. One way to resolve this problem is to reduce data frequency; however, we want to analyse as much information as available, to maintain important features and characteristics of the data. Therefore, a preprocessing step was added to the study, in order to smooth the data.

5.2.1 Time-series smoothing

The smoothing of a time series is important to remove extreme values, likely caused by artifacts (section 2.2.4). As was observed in figure 5.6, there are several small fluctuations that can be seen as noise. In order to perform the forecasting task, it was necessary to smooth the data.

All the analyses related to traffic flow information are done after the smoothing has been applied. Thus, we will get values that can be unrealistic; for example, we can have a count of 6.7 vehicles in the five-minute interval. Even not being realistic values, it is more important to have a predicted value that is close with the real value, but its meaning is different, than having a value that is realistic but is very distant from the real value.

Several smoothing methods were tested, being the Savgol smoothing the one that pre-

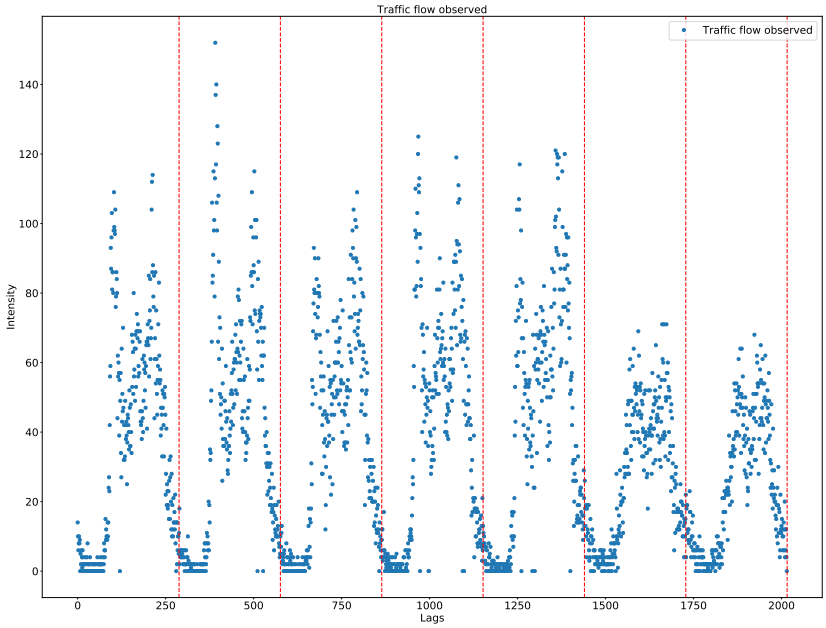


Figure 5.9: Traffic flow observed.

---

sented the best results. The Savgol smoothing was the only one that was tested that preserved the real behavior of the time-series.

The smoothing method was defined on the data from the first week of data. Then, the method was tested between September 30 of 2019 and October 27 of 2019, evaluating the effect of smoothing on the data.

In order to apply smoothing methods appropriately, it is necessary to verify the stationarity of the time-series. It was verified that the time-series is stationary, since a significant p-value was found (<0.001) in the ADF test.

Due to the intrinsic characteristics of the data, it is not intuitive which is the best smoothing method to use. So, several methods were tested; however, Savgol smoothing was the one that seems more suitable. In figure 5.10, some of the tested smoothing methods are represented.

Figure 5.10a contains the curve corresponding to the application of one-dimensional interpolation smoothing using a linear function. It was also tested the application of a cubic function, and using approximation methods (nearest, previous, and next) as functions of the one-dimensional interpolation.

The *Exponentially Weighted Moving Average* (EWMA) smoothing method is in figure 5.10b. To evaluate which method better describes the data, several combinations of parameters were tested. Some of the parameters specify decay in terms of center of mass (com), others in terms of the span, or half-life, etc.

Figure 5.10c presents the application of smoothing using rolling methods. Once more, it was tested with multiple parameters. The orange curve is the result of the rolling mean of the values, and the dotted green curve uses the euclidean distance.

The last figure, figure 5.10d, uses a spline function to smooth data. All the smoothing applications presented in figure 5.10 have similar performance, indicating a clear behavior of

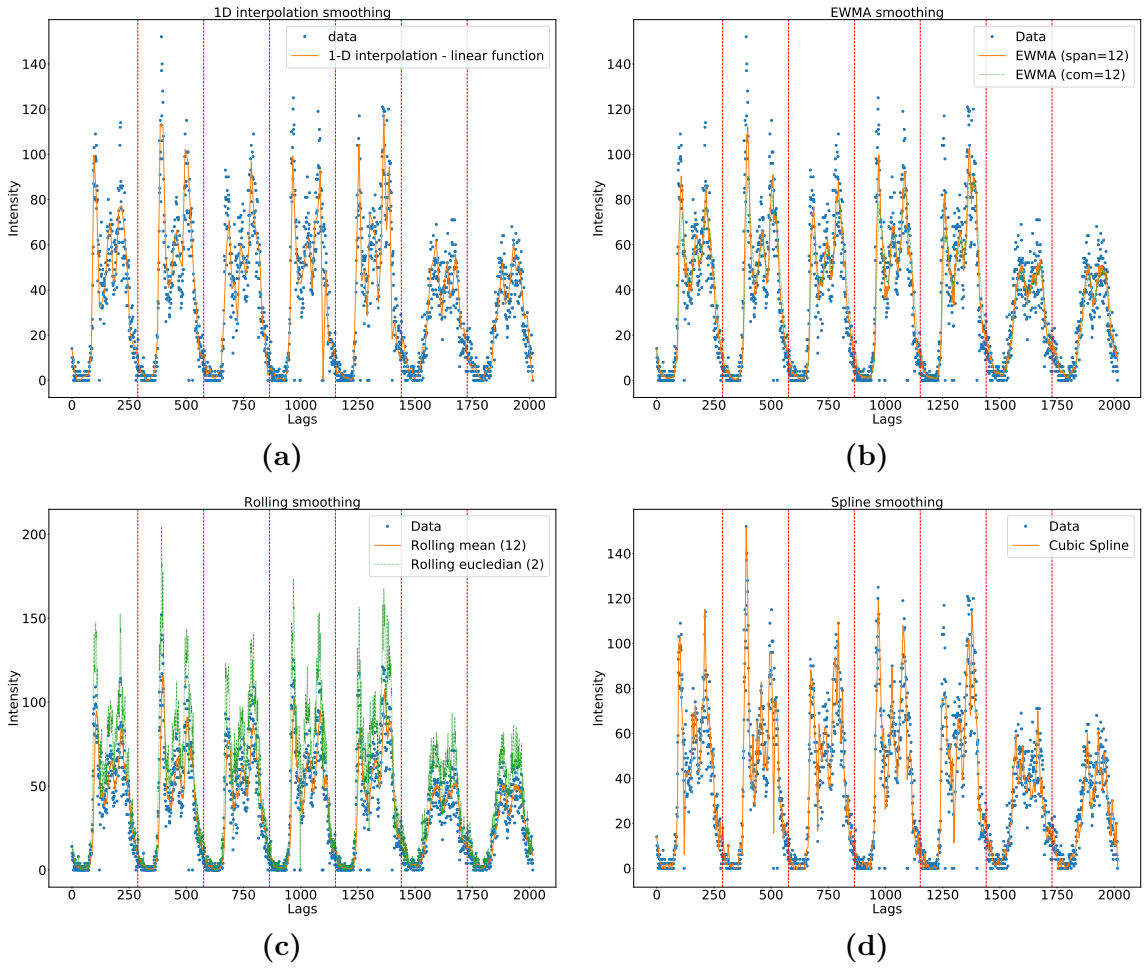


Figure 5.10: Application of the different smoothing methods to the traffic flow of the first week of October **(a)** 1D interpolation **(b)** EWMA **(c)** rolling **(d)** spline.

the data. Savgol was selected, since this method does not introduce discontinuities on the time-series. Several parameters were tested, as exemplified on Figure 5.11. The one that seems to better fit the data is the smoothing done with the parameters window size 41 and polynomial order 3.

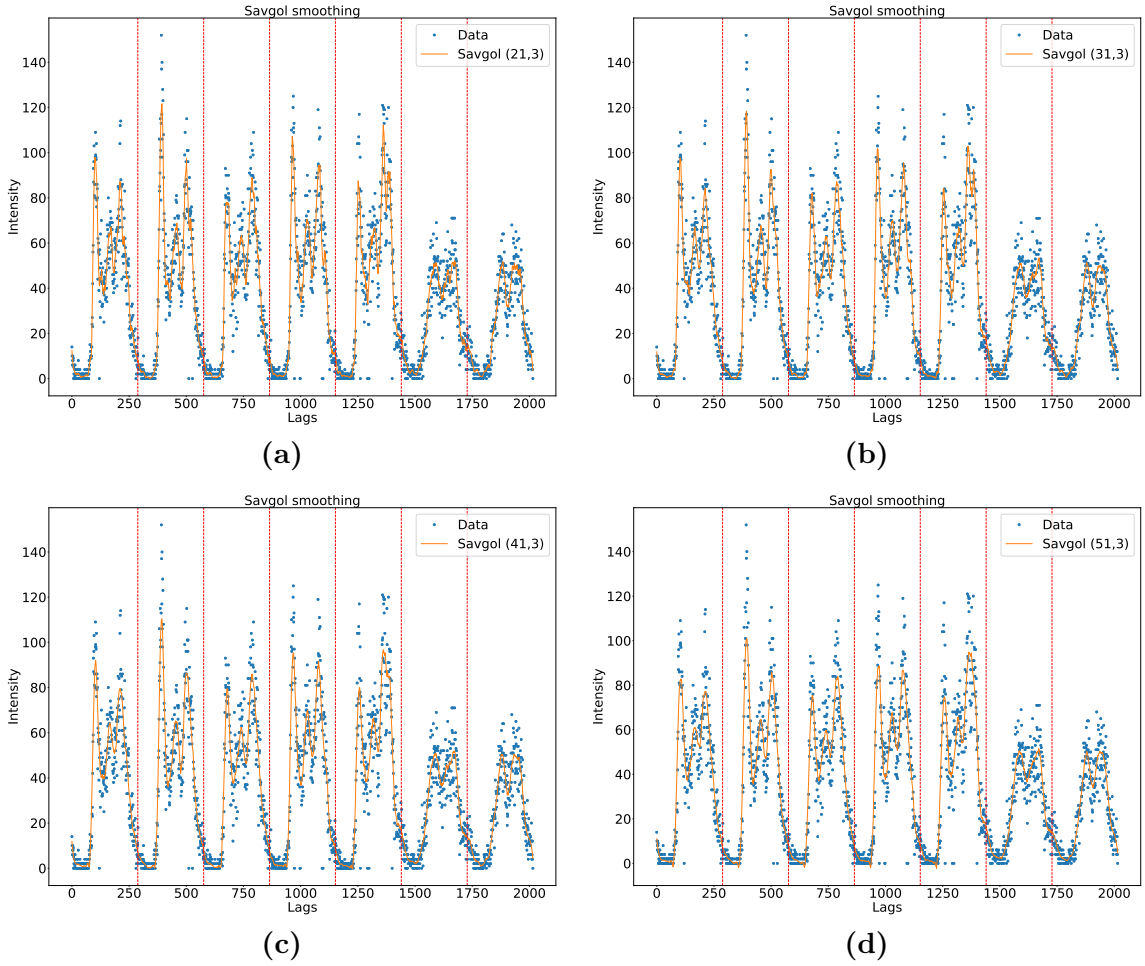


Figure 5.11: Savgol smoothing applied to the traffic flow time-series **(a)** window size 21, polynomial order 3 **(b)** window size 31, polynomial order 3 **(c)** window size 41, polynomial order 3 **(d)** window size 51, polynomial order 3.

With the application of Savgol smoothing, the pointed peaks disappear. Figure 5.12 presents the application of Savgol smoothing to the four weeks that will be considered to latter forecast the traffic flow. The first week corresponds to the first line, the second week to the second line, and so on.

All studies related to the traffic flow observed in this chapter will consider that the data was previously smoothed.

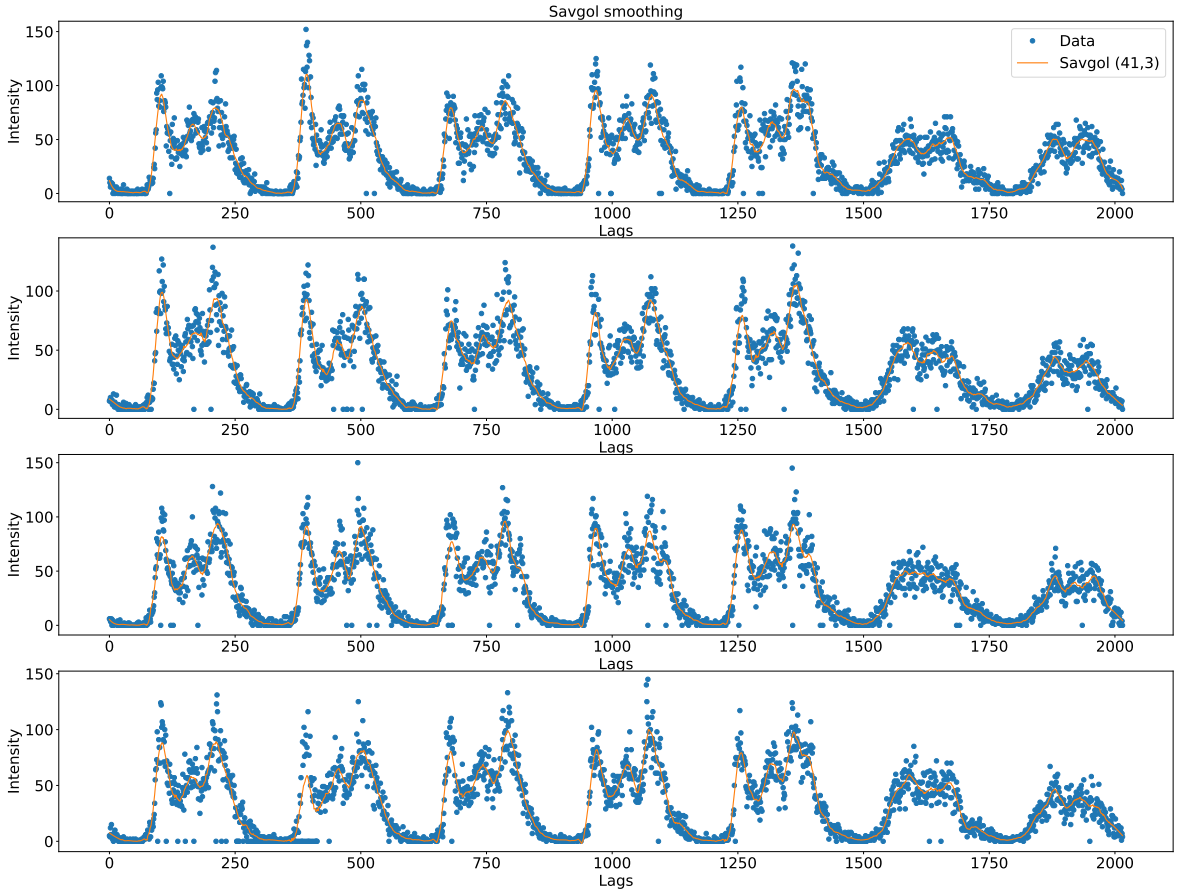


Figure 5.12: Savgol smoothing traffic flow.

5.2.2 Assessing Stationarity

Smoothing should not alter time-series characteristics. The stationarity was evaluated before smoothing the time-series. As mentioned, the time-series is stationary ($p<0.001$, in the ADF statistical test).

Table 5.7: Stationary test

ADF Statistic		-9.429390
p-value		0.0000001
Critical Values	1%	-3.431
	5%	-2.862
	10%	-2.567

The ADF statistic value should be negative; if the value is negative, we will probably reject the null hypothesis. Besides that, the ADF statistics value is lower than any of the critical values, even the 1% value. This means that the statistics are maintained, giving credibility to the statistical test.

5.2.3 Timeseries decomposition

Time-series decomposition is made based on the time-series frequency of patterns. Table 5.8 contains possible values of patterns frequency, since the data sampling frequency for the traffic counters is 1 sample per 5 minutes. For example, if a pattern happens every day, we would use a frequency of 288.

Time period	Number of sample
5 minutes	5 minutes / 5 minutes = 1
1 hour	60 minutes / 5 minutes = 12
1 day	24 hours * 60 minutes / 5 minutes = 288
1 week	7 days * 24 hours * 60 minutes / 5 minutes = 2016
1 month (4 weeks)	4 weeks * 7 days * 24 hours * 60 minutes / 5 minutes = 8064

Table 5.8: Time periods and number of samples

Figure 5.13 presents the additive decomposition of the traffic flow time-series. Note that, a multiplicative decomposition can not be used because the traffic flow can be zero, meaning that the presence of a unique zero value would make it impossible using the multiplicative decomposition.

Figure 5.13a shows that, using a frequency of 12, it is not possible to observe a pattern in the seasonal component. The pattern appears in the trend component, meaning that a lot of information about the seasonal component was not retained. Besides that, there are still some patterns in the residual component.

In figure 5.13b it is possible to observe that the daily pattern is retained in the seasonal component. However, the weekly pattern is not retained, and we can see that there is a pattern that remains in the trend component. There is still significant information in the residual component, but it is less significant than in figure 5.13a.

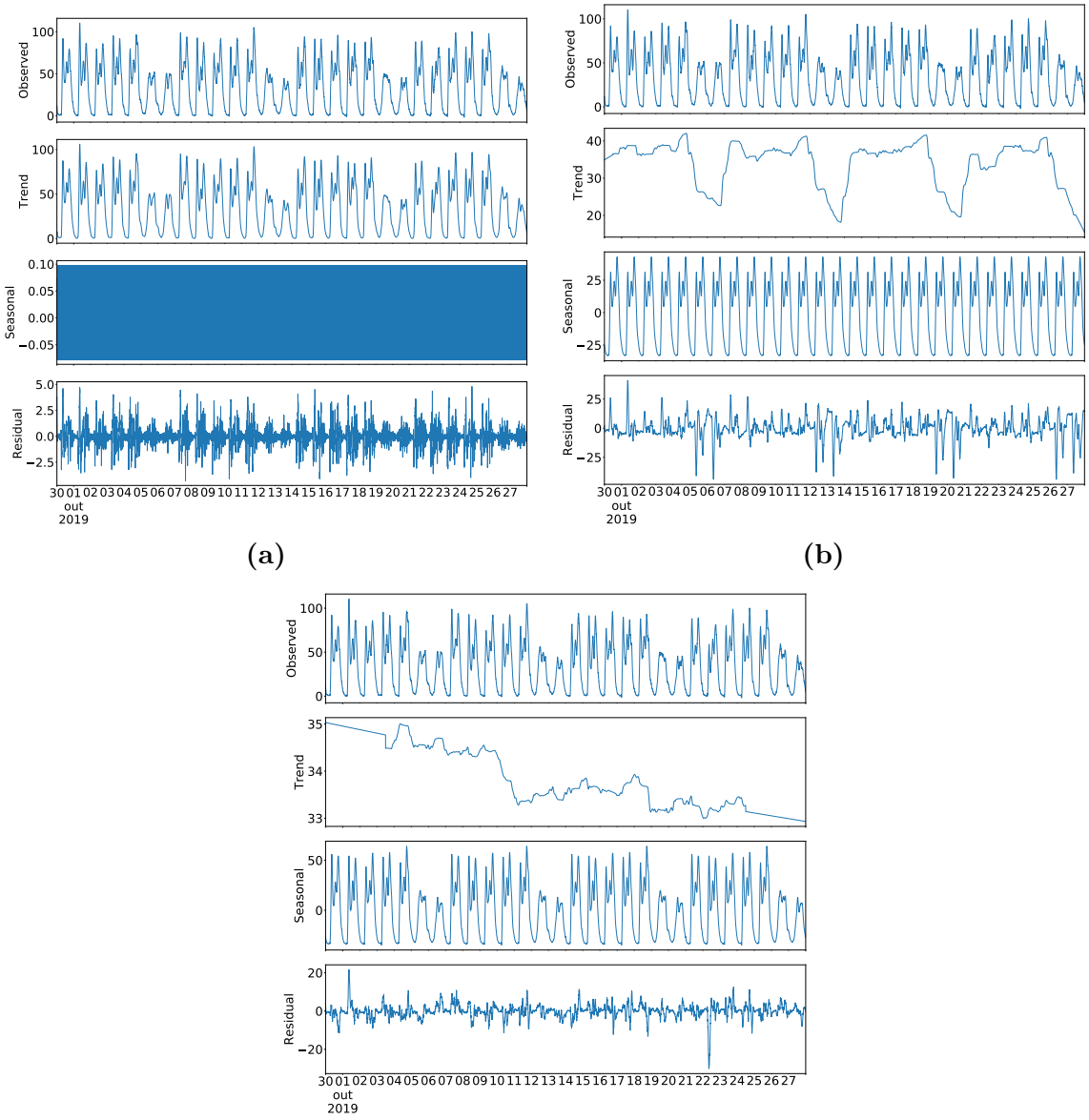


Figure 5.13: Time-series additive decomposition **(a)** frequency = 12 **(b)** frequency = 288 **(c)** frequency = 2016.

In figure 5.13c with a frequency of 2016, the daily and the weekly patterns are retained in the seasonal component. We have a trend component free of patterns, and in the residual component, it remains some information being even less significant than in figure 5.13b. The trend component does not have a linear growth, meaning that it is probably stationary. The behavior of the trend supports the results of the ADF tests.

Note that, sometimes the seasonal decomposition model can not be able to separate the noise from the trend, meaning that the trend can contain noise.

Auto-correlation and partial auto-correlation

Correlating the time-series with itself can help us to choose the best lags to use in the deep learning models. Figure 5.14 contains the autocorrelation graphic for the first day (288 lags). Autocorrelation values belong to the interval between -1 and 1. In the figure, we can see that lag 0 has the biggest autocorrelation value. This happens because the lag 0 is compared with itself. The blue area presented in both images is the confidence interval that is 95% by default. This confidence interval around the correlation value indicates a statistical significance of the obtained correlation values.

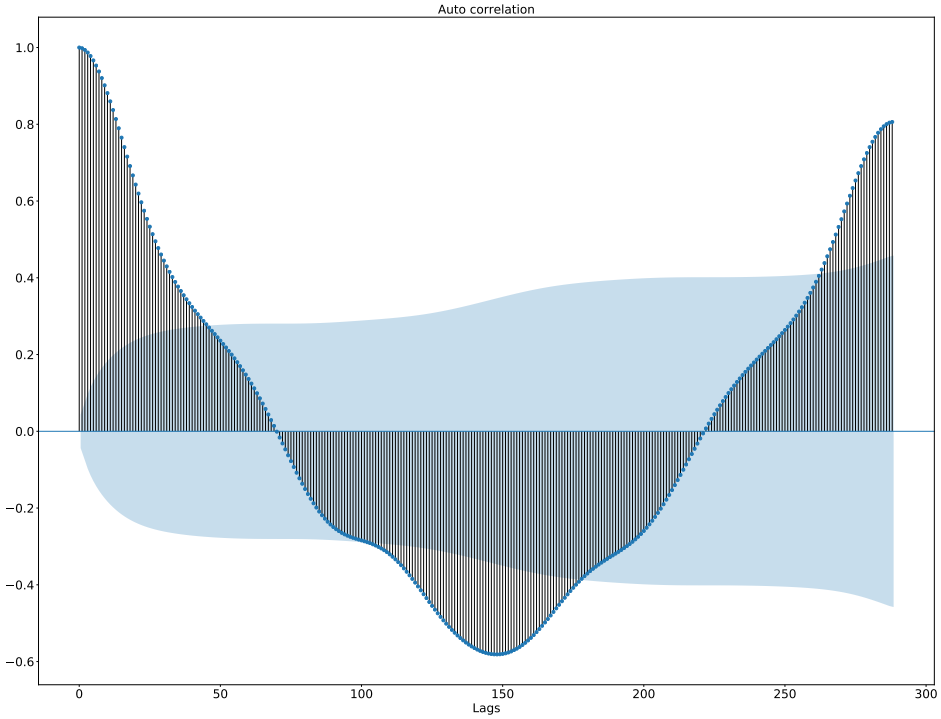


Figure 5.14: Auto correlation - 288 lags

If we look into figure 5.15, it is represented 2016 lags, that correspond to 7 days (a week). In this figure it is possible to observe the seasonality that exists in the time-series. For each one of the days, for 288 lags in 288 lags, there is a local maximum; because, by comparing 2 days, the maximum value for correlation happens for the 24-hour lag.

Figure 5.16 is the autocorrelation function for the four weeks of study. In this figure, it is possible to observe that the local maximums decrease and then increase, reaching the

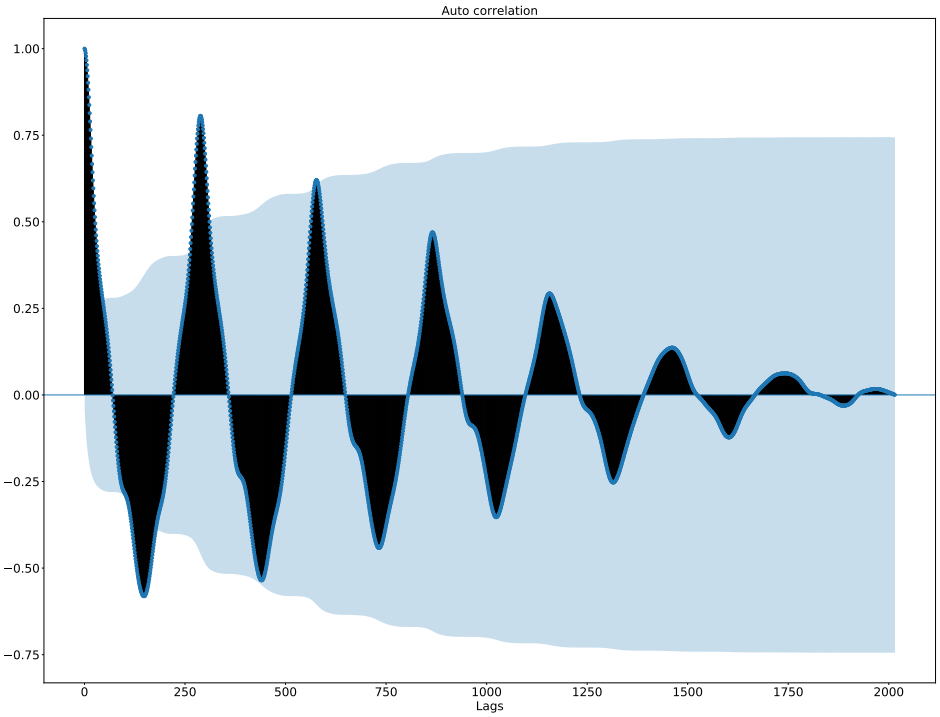


Figure 5.15: Auto correlation - 2016 lags

---

maximum at each week. It is possible to observe the weekly seasonality.

At the beginning of each week, it is reached the maximum value for that week, due to the comparison of the same week day (Monday in this case). If we started to compare in a different week day, it would happen the same. This shows that there are a strong relation between what is happening in the present week, with what has happened in the week before. The daily patterns are observed by the consecutive peaks that have maximums at the same distance (approximately), meaning that they happen at the same hour. The decreasing tendency of the maximum values is due to the existence of less information to calculate the correlation as time goes by.

The blue area increases as time goes by, and the maximum values of autocorrelation decrease. This makes it more difficult for autocorrelation to be considered relevant over time, and it means that the model assumes that the importance of a lag becomes less significant over time.

An interesting aspect is the fact that, after every local maximum there is a local minimum of negative proportions, and together they create a pattern. These local minimums happen twelve hours later, and they mean that past values have a big negative correlation with the future.

Figure 5.17 contains the parcial auto correlation graphic for the first day. Partial autocorrelation removes correlations of closer lags and it will indicate the relation between the lag and the observation. In figure 5.17, it is possible to observe a maximum at lag 240 (approximately).

If we expand the number of lags, we can see that the maximum happens at lag 400 (approximately), as it is represented in figure 5.18.

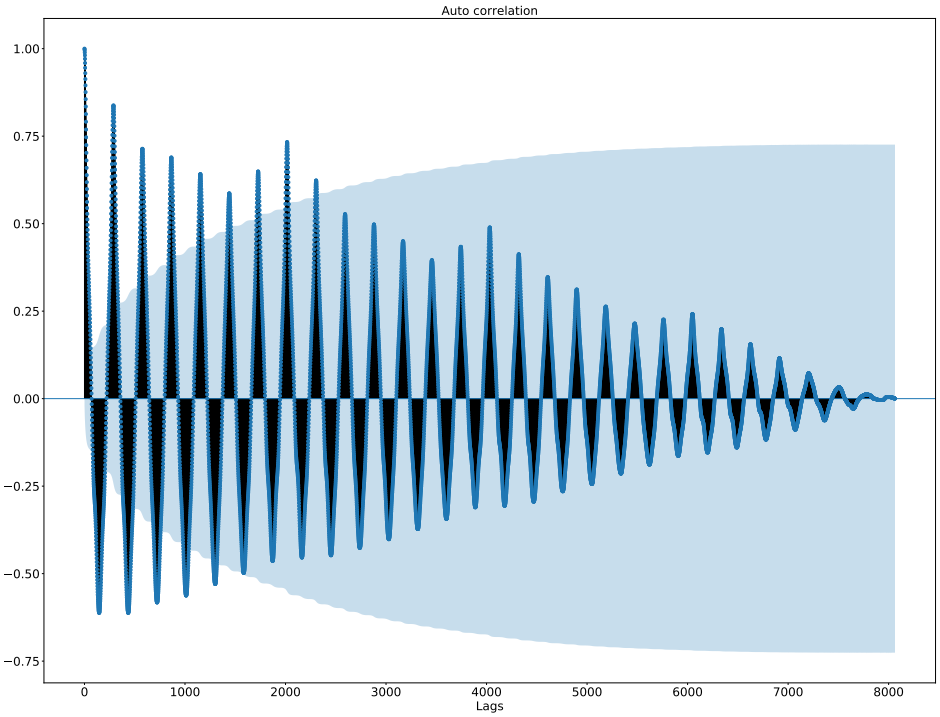


Figure 5.16: Auto correlation - 8064 lags

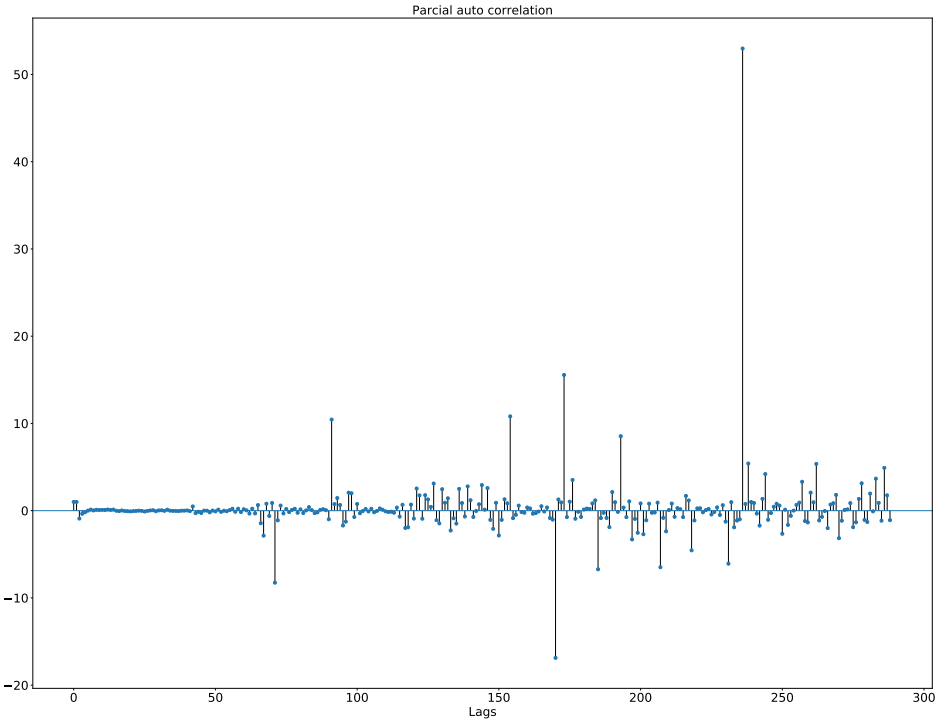


Figure 5.17: Parcial auto-correlation - 288 lags

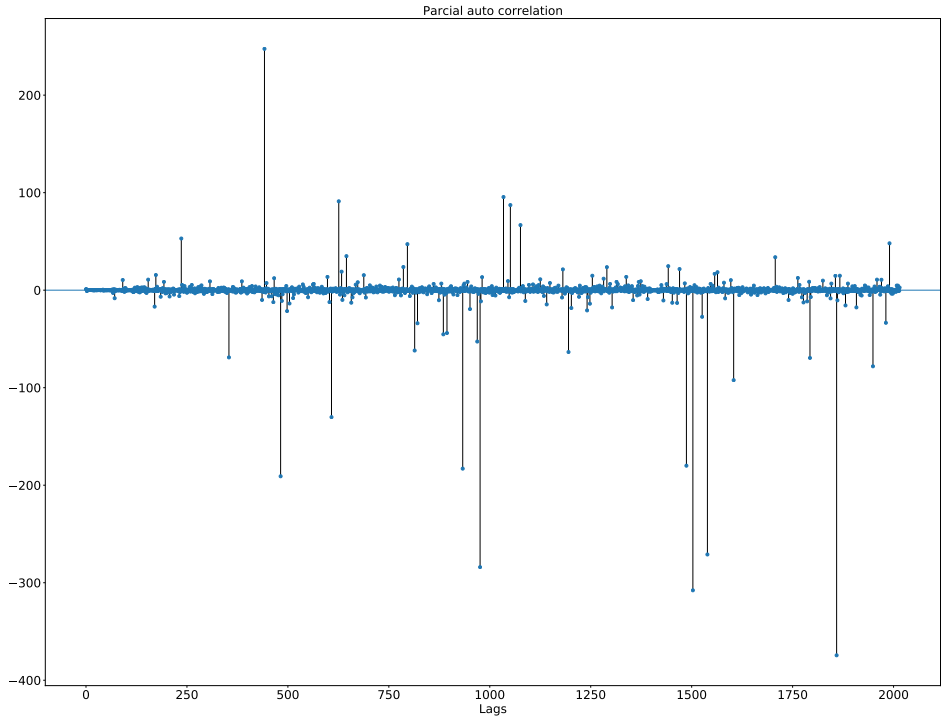


Figure 5.18: Parcial auto-correlation - 2016 lags

Cross-correlation

Correlating the time-series with another time-series can indicate if we can apply the same methods and parameters to perform, for example, the prediction of future values. It was performed the correlation of the traffic flow sensor with another that is close to this sensor. The sensor had an identification of CT2Z8. Figure 5.19 presents the obtained result for the first week. It was performed the correlation with the parameter unbiased as true. This means that the autocovariance is adjusted. The visual effect is that, if the unbiased was false, we would see a decrease in the local maximums, as we see in the autocorrelation plots.

Figure 5.20 presents the cross-correlation for four weeks. Once again, we can see the weekly pattern effects on the correlation. Both sensors have nearby locations and are very correlated since they present maximum values very close to 1. Through the observation of these images, we can conclude that there is a very high probability that we can apply the same models that work for the traffic flow that is being analyzed to similar ones.

5.2.4 Relationship between sensed traffic flow and bus speed data

There are two sources of data concerning traffic: the installed traffic flow meters (traffic counters), and the bus tracking information. A question that was raised was to which degree the information of the two series is related. This could inform additional decisions, such as if the bus network can ”replace” the deployment of physical traffic counters. In order to associate traffic flow data with the buses speed, we had to perform the steps in figure 5.21 to determine the buses that could be counted by one sensor:

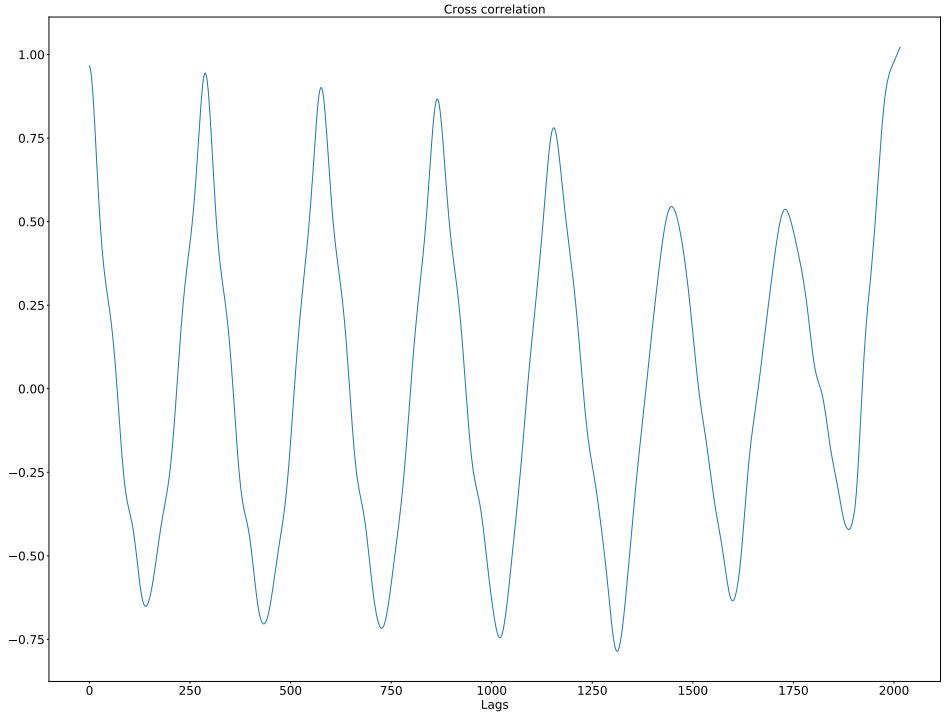


Figure 5.19: Cross-correlation - 2016 lags

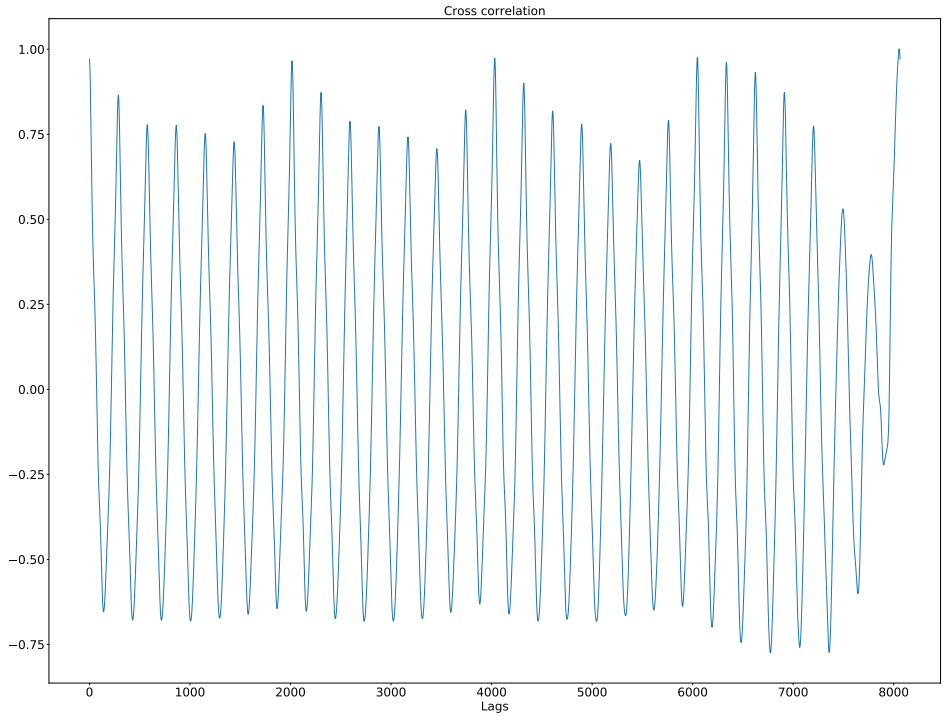


Figure 5.20: Cross-correlation - 8064 lags

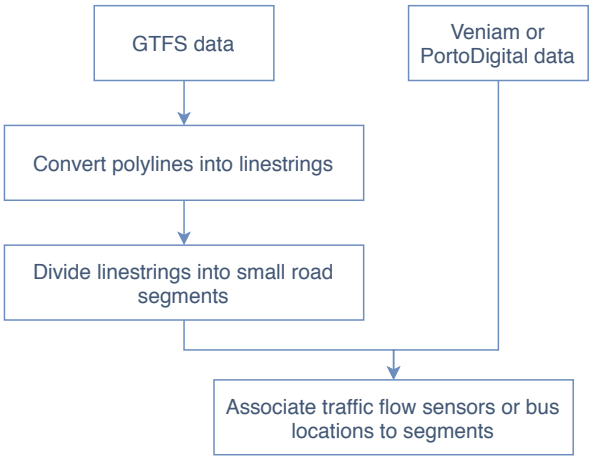


Figure 5.21: Associating traffic flow data or bus speed data to road segments

---

GTFS files provided road segments by having polylines. Polylines are sets of linestrings, for that reason they were split so we could obtain linestrings. Since some linestrings were too long, we divide them to achieve smaller segments. By performing geometric and geographical queries it was possible to associate sensors or buses to road segments.

As can be observed in figure 5.22, the left segment can have multiple points, in this case, the segment has eight points (from A to H), so it can be subdivided into seven subsegments. Through a combination of geometrical and geographical queries, it was possible to determine the closest segment to a GPS position. It will be calculated the traffic flow sensor is closer to subsegment GH than the others. This type of calculus is expensive. This selection is represented in figure 5.22 on the right. Note that, this is just an example and does not represent a real example neither a real subdivision.

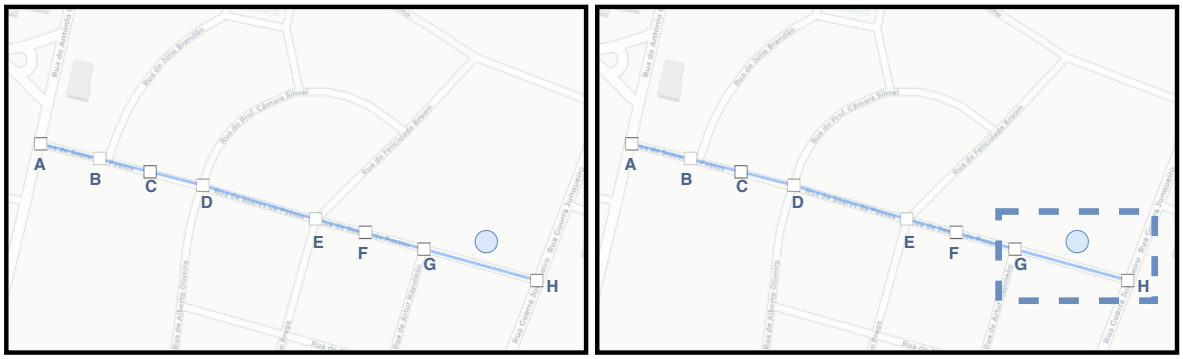


Figure 5.22: Road segments, subdivision and traffic flow sensors

After this, it was decided to choose a traffic flow sensor in a popular bus segment. It was made an algorithm to find the best combination. The first step was to count how many buses go to each one of the segments. The second step ordered the segments, by decreasing order. The final step was to find the first segment where was located a traffic flow sensor. The sensor chosen was CT5Z5.

Figure 5.23 contains the scatter matrix for the traffic flow observed, the bus speed, and the number of buses. Note that the traffic flow observed data was smoothed before the comparison; however, this process is explained in the following subsection. The scatter matrix compares every pair of features. The diagonal is the feature compared with itself using a kernel density estimation.

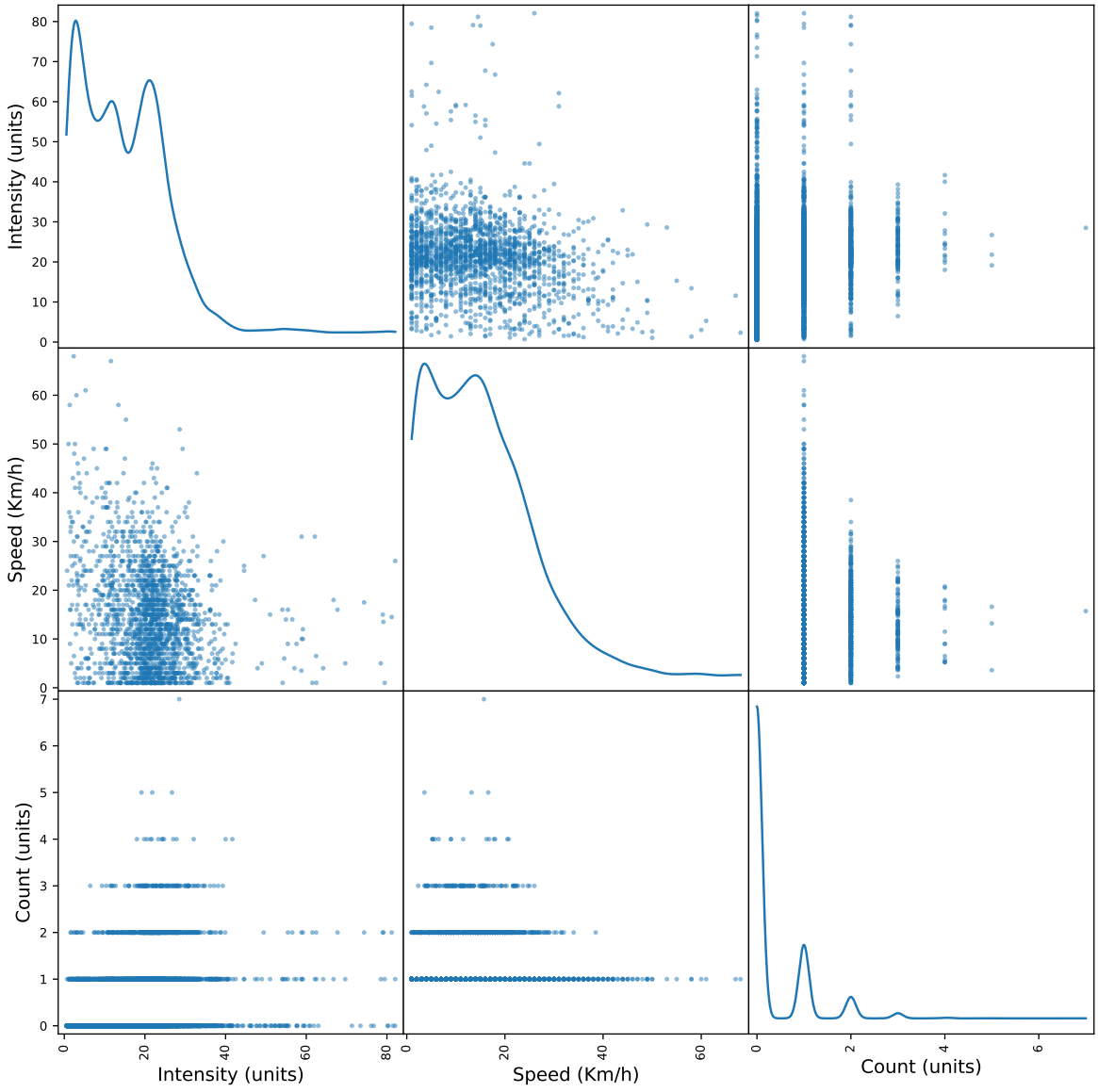
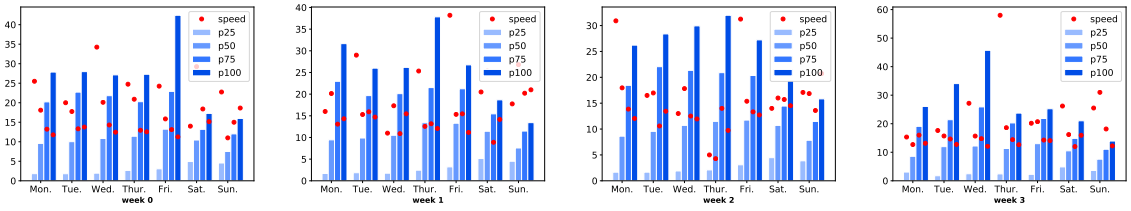


Figure 5.23: Scatter matrix - Traffic flow observed, speed, and count

Despite any particular pattern was found, it is visible that when we have high speed, we have low intensity. When we have high intensity, we have low speed. The majority of the points are concentrated in the area of low intensity and low speed. There are not any points in the area of high intensity and high speed.

Given the observed patterns, we try to find if could observe patterns analyzing the data

quartiles. The data was divided by week, and then, by weekday. The results are presented in image 5.24 and do not show relevant patterns.



(a) **(b)** **(c)** **(d)**

Figure 5.24: Traffic flow observed versus speed **(a)** week 0 **(b)** week 1 **(c)** week 2 **(c)** week 3

Several methods were tested, but the results were very limited. Besides that, even choosing a segment by bus popularity presents some problems. There is an enormous quantity of null values because most of the time there was not even one bus that went through the segment.

Buses behavior can not represent normal traffic behavior (namely cars). This might be the main reason why a clear relationship between bus speed and traffic flow intensity was not observed. Nevertheless, it was observed that when we have big values for speed, we have small values for the intensity, and when we have big values for the intensity, we have small values for speed. For this reason, the speed values will not be used in the traffic flow prediction.

5.3 Elements of the forecasting pipeline

5.3.1 Forecasting pipeline

Forecasting the traffic flow can be a difficult task. Before any manipulation of the data begins, the data must be prepared. That analysis can involve visualizations, statistics, and the study of the data types. All these three types of analysis were made as it was discussed in previous chapters. It was developed a pipeline, as can be observed in figure 5.25.

Once we understand our data, we can begin to prepare it. Preprocess the data includes data cleaning, data transformation, and data reduction.

Data cleaning aims to eliminate the problems related to missing data, noisy data, and outliers. Since the traffic flow observed had few missing data, we could replace it by zeros, and through smoothing, the impact of the missing data was not significant. Smoothing also helped in the case of the existence of noise and outliers.

Data transformation is usually made to ensure that there are a maximum and minimum defined. One of the ways of doing it is through normalization, scaling the data with a min-max scaler from -1 to 1.

Data reduction can be made by aggregating data, performing a reduction in the data that is studied, or through dimensionality reduction. In the case of traffic flow data, it was not necessary to aggregate it. However, this study is focused on just one traffic flow observed sensor. Relatively to dimensionality reduction, some of the original features were not used, and it was performed a feature selection between time lags.

Feature selection is very important because it helps reducing overfitting, improves accuracy, and reduces training time.

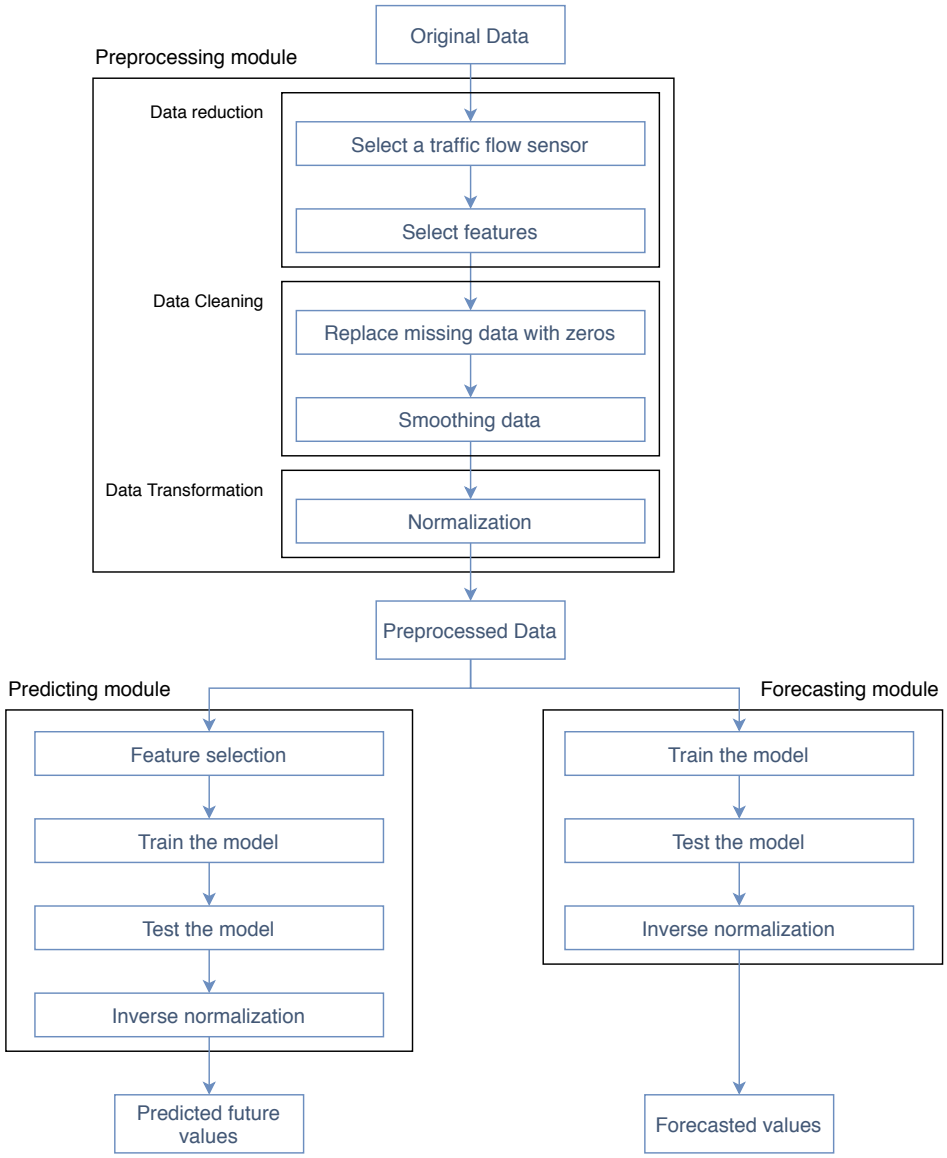


Figure 5.25: Forecasting pipeline

When the preprocessing phase is done, we can test and evaluate models. It is important to choose metrics to help choose the best models. Once we choose it, we can decide if we want to improve the results. For example, since it was noticed that the increase in the number of neurons has an impact on the decrease of the error in deep learning methods, it was increased the number of neurons beyond the initially foreseen. Subsection 5.6.5 presents these results in a more detailed explanation.

The final phase is present the results that were obtained by the process. Note that, when we scale data, we will have to make an inverse scale before evaluating the model.

The use of external features to predict traffic flow data can improve the model [51]. For that reason, it will be studied the relationship between bus speed and traffic flow.

The characteristics of the machine where the tests where made are described in table

5.9. Note that, it was used more machines with different characteristics; however, to get the average running time, it was used just the times from the tests performed in this machine. Since the number of tests was large, to avoid overheating problems, it was imposed a limit of CPU usage at 600%. The computer had 8 cores, having a total capacity of 800%. Only one of the five machines used had a GPU. For that reason, it was only used the CPU to train the ANN models.

Table 5.9: Machine configurations

Operating System	Ubuntu 18.04 LTS
Architecture	x86_64
CPU op-mode(s)	32-bit, 64-bit
CPU(s)	8
Thread(s) per core	2
Core(s) per socket	4
Model name	Intel(R) Core(TM) i7-7700 CPU @ %3.60GHz
Memory	8 GB

5.3.2 Features selection

A time-series can be expressed as a function of time (discussed in section 2.2). In deep learning problems, we need to have input features and output features. A naïve approach would be using the time as an input feature and the value as an output feature. However, deep learning algorithms do not work in that way. At this stage arises a very important question: "How can we transform a time-series to use it in a deep learning algorithm?".

Time by itself is not useful to predict future values, and in a time-series, past occurrences have an impact on future occurrences. Therefore, past occurrences will be the input features and the actual occurrence will be the output feature, as represented in table 5.10.

Table 5.10: Time-series and lags

	Features		
	Output feature	Input features	
t	f(t)	f(t-1)	f(t-m)
0	f(0)		
1	f(1)	f(0)	
...	...	...	...
n-2	f(n-2)	f(n-2-1)	f(n-2-m)
n-1	f(n-1)	f(n-1-1)	f(n-1-m)

Note that not all past occurrences influence future occurrences in the same way. That is why we need to select which past occurrences will have the biggest impact. The past occurrences are often called lags.

Knowing the characteristics of the data is important to choose the best lags, this is denominated "expert knowledge". Traffic flow patterns depend on seasonality patterns. A combination of what happened in the last hour, a week before, or even a month before can be useful.

To avoid a biased opinion about which lags should be used, we can compare some lags with a correlation matrix. A correlation matrix is obtained by calculating the correlation between each pair of input features. For that reason, the correlation matrix is a square matrix in which the upper triangle is symmetric to the lower triangle. In the diagonal, all values have the maximum correlation value because it corresponds to the correlation of the feature with itself.

The data needs to be transformed in order to create and use ANN models. It is necessary to adapt the time-series into a set of input features and an output feature. In this specific case, the features are the lags, so they need to be selected.

To choose the best lags to train the models, we performed feature selection. The lags to perform feature selection are chosen based on the knowledge acquired from the study of the dataset.

In figure 5.26, the diagonal presents the highest values because every lag is most correlated with itself. The matrix is symmetric, being the diagonal the symmetrical axis, meaning that if lag a has a certain value of correlation with lag b, then lag b has the same value of correlation with lag a.

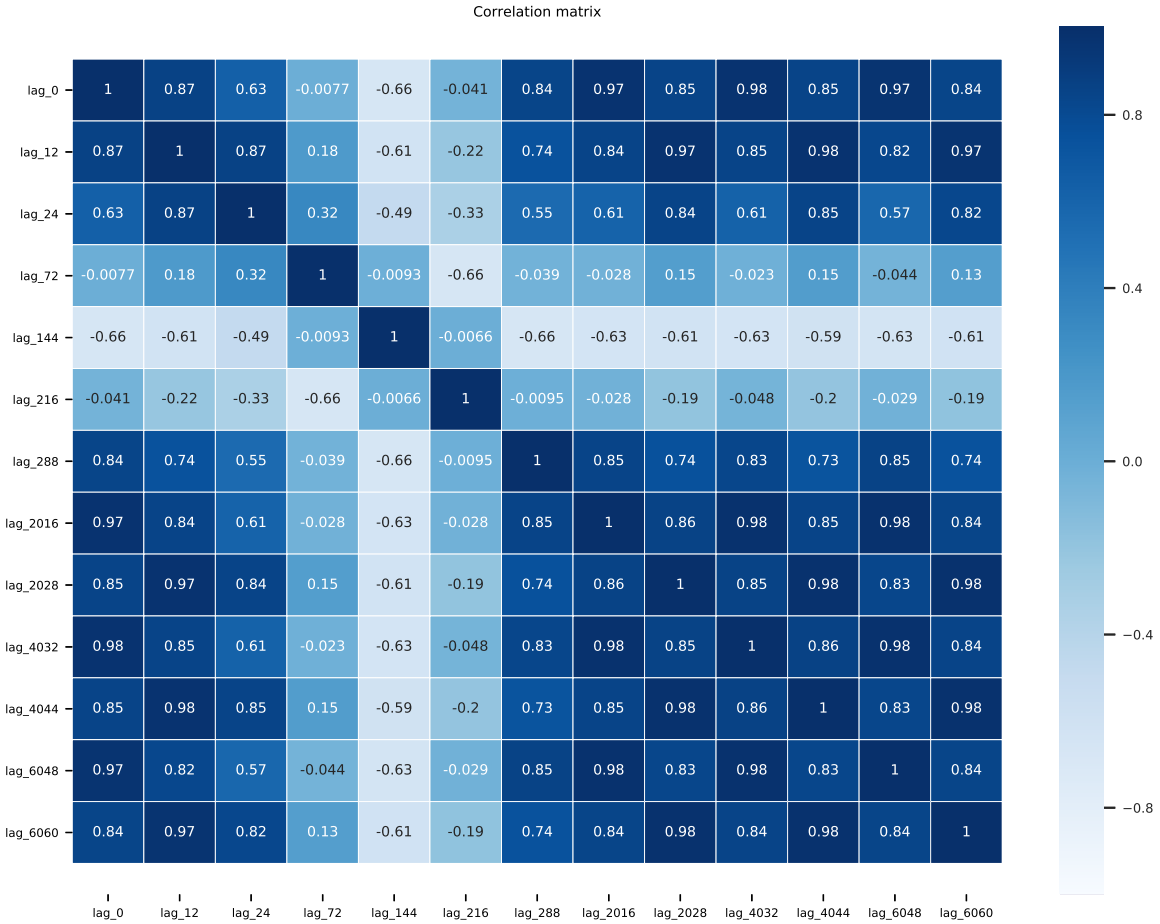


Figure 5.26: Feature selection matrix of the traffic flow time lags

We will focus on comparing the lag 0 with the remaining because we want to predict the actual value. Lag 0 is most correlated with lag 2016, lag 4032, and lag 6048. Those lags correspond to the previous values one week earlier, two weeks earlier and three weeks earlier, and the correlation value is very close to 1 in all of them (0.97, 0.98, 0.97), meaning that they are strongly correlated with lag 0.

The next lags that are very correlated with lag 0 are lag 12, lag 288, lag 2028, lag 4044, and lag 6060. These lags correspond to the previous hour, the previous day, the previous week plus an hour, the previous two weeks plus an hour, and the previous three weeks plus an hour, and present values between 0.84 and 0.87.

Choosing the best lags, we should take into consideration multiple factors like, how much time to predict, how much information we plan to use, and how correlated is the lag. To predict one week of data, we cannot use the previous hour or day. To use less information, we will just focus on the first two weeks. With this in mind, it was chosen the lags: 2016, 2028, 4032, and 4044.

5.3.3 Algorithm selection

We decided that it would be done the comparison between the use of statistical methods and machine learning methods, namely deep learning methods. In terms of algorithms, for the statistical methods, it was used SARIMA.

It was also done an attempt to use one other statistic method name *Generalized Autoregressive Conditional Heteroskedasticity* (GARCH) [19], however, the model was not better than a mean model. The GARCH model was not adjusted to the traffic flow observed. One of the possible explanations is that this type of model does not take into account the seasonality of the data. One of the characteristics of GARCH models is that they deal within the changes in the variance, and in the volatility, unlike ARIMA based methods.

Two types of ANNs were chosen to make the predictions: FFNN and LSTM. In both types of networks, we have an input layer at the beginning and an output layer at the end. However, the content of the hidden layers changes.

In the case of the FFNN, it were considered three versions. The first version is represented in figure 5.27 and has one dense layer, which can be followed by dropout. The second version is similar to the first, but is followed by an extra dense layer followed by dropout. The third version is an extension of the second, having one more dense layer followed by dropout. The dropout does not happen always, since it is configurable. Before the output layer, it was necessary a flatten layer that flattens the data.

In the case of the LSTM, we have just a LSTM layer followed by dropout, as can be observed in figure 5.28. The dropout is configurable.

5.4 Forecasting the traffic flow with SARIMA

Considering that the analysed time-series is seasonal, the SARIMA model has been used. Statistical methods like SARIMA allow us to perform short term forecasting. SARIMA was the first method being tested. All parameter values that were tested are summarized in table 5.11. There are 729 possible configurations.

One of the big advantages of using statistical methods like SARIMA is that SARIMA is a deterministic method. This means that we can just run tests one time because the results are

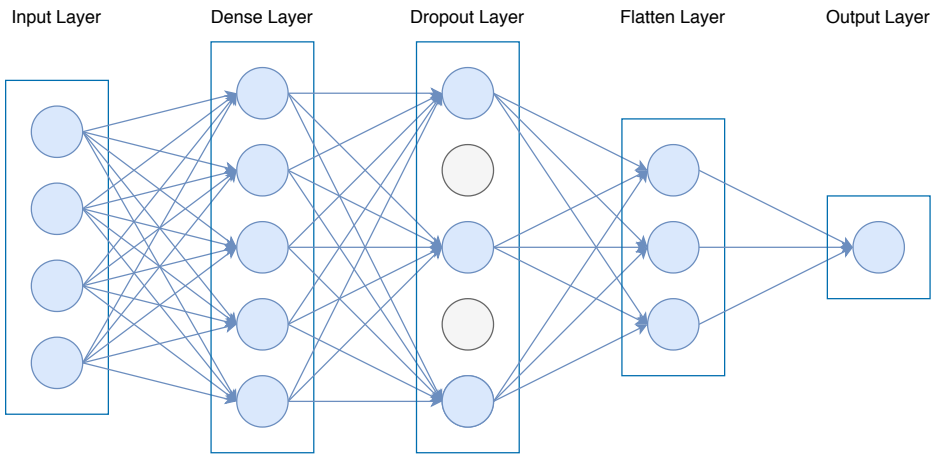


Figure 5.27: FFNN - Diagram

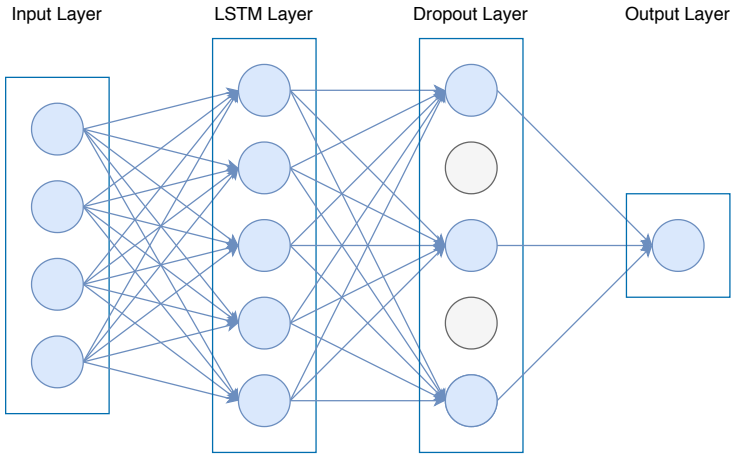


Figure 5.28: LSTM - Diagram

Table 5.11: Configurations details of SARIMA

	Configurations	Values
Order	p	0, 1, 2
	d	0, 1, 2
	q	0, 1, 2
Seasonal Order	P	0, 1, 2
	D	0, 1, 2
	Q	0, 1, 2
	m	12

always the same. However, this type of model is more limited than a deep learning model, since it only allows to perform short term forecast.

With SARIMA, we can just predict 12 steps that correspond to 1h, as can be observed in

table 5.8. It is not possible to predict more steps given the nature of the model and the data. We tested other values for the m parameter, but the behavior of the curve being forecasted was worst, or the model was too complex (when m was too big) that could not be calculated. Because of that, it was defined that it would be used one model for each day of the week.

To forecast the traffic flow, we used 75% of the data for training and another 25% for tests. For example, to predict the last Monday it was used the first three Mondays to train SARIMA.

The best model for each day of the week was chosen based on the results of the first forecasted hour. To forecast a hour it was necessary, on average, 3.583 seconds. Relatively to the chosen evaluation metrics to choose the best model, initially it was used MSE. MSE is a good metric for many cases, but this one was not one of them because it provided overfitting of the model to the data. The models with the best MSE values for the first hour do not always presented good forecastings for the remaining hours.

The goal was to forecast more than one hour given the limitation of the model, even if that meant redoing the training task. Using an evaluation metric like BIC meant that the model had bigger errors, but we could reuse the model. Tables 5.12 and 5.13 present the results obtained when the chosen metric was MSE, versus BIC for the first forecasted hour.

Table 5.12: SARIMA results - Choosing the best model MSE

Day	Configurations							Metrics			
	Order			Seasonal Order							
	p	d	q	P	D	Q	m	MSE	RMSE	AIC	BIC
Mon.	2	2	0	1	1	1	12	0.017	0.132	1970.081	1993.724
Tue.	1	0	0	2	1	1	12	0.138	0.371	3402.825	3426.414
Wed.	0	0	0	2	0	0	12	0.495	0.703	6758.493	6772.693
Thur.	0	0	0	1	2	0	12	0.312	0.559	7337.251	7346.689
Fri.	2	0	2	0	2	1	12	10.122	3.181	3566.309	3594.602
Sat.	1	1	1	2	1	0	12	10.006	3.163	1410.093	1433.676
Sun.	2	2	0	1	2	2	12	10.086	3.175	1573.185	1601.389

Table 5.13: SARIMA results - Choosing the best model BIC

Day	Configurations							Metrics			
	Order			Seasonal Order							
	p	d	q	P	D	Q	m	MSE	RMSE	AIC	BIC
Mon.	2	1	2	0	0	2	12	6.908	2.628	1855.097	1888.198
Tue.	2	1	2	1	1	1	12	3.176	1.782	1893.512	1931.235
Wed.	2	1	2	0	0	2	12	3.726	1.930	1782.029	1815.130
Thur.	2	1	2	0	0	2	12	10.210	3.195	1906.963	1940.063
Fri.	2	1	2	0	0	2	12	77.296	8.791	2252.751	2285.851
Sat.	1	1	1	0	0	2	12	16.500	4.062	1214.998	1238.647
Sun.	1	1	1	0	0	2	12	7.799	2.792	1400.868	1424.517

Choosing BIC as a metric can increase a lot the error, even if we used other ways to measure it, besides MSE or RMSE. However, as can be observed by comparing table 5.12

with table 5.13, the values of parameters are more consistent using BIC as an evaluation metric. In table 5.12 none of the days share all the parameter values. In table 5.13 there are only three combinations. One of them is for the weekend days, and another one is for all days of the week except for Tuesday. Tuesday has different seasonal parameters than the other days of the week, meaning that were observed different seasonal patterns for Tuesday. Besides that, it is possible to observe that, for the entire week except for Tuesday, all seasonal parameters have the same value.

Figure 5.29 shows the impact of the choice of the chosen evaluation metric for the first twelve steps for Wednesday. The model chosen using BIC seems a lot worse than the model chosen using MSE. The major impact is when we want to retrain the model without having to calculate once more the best parameters.

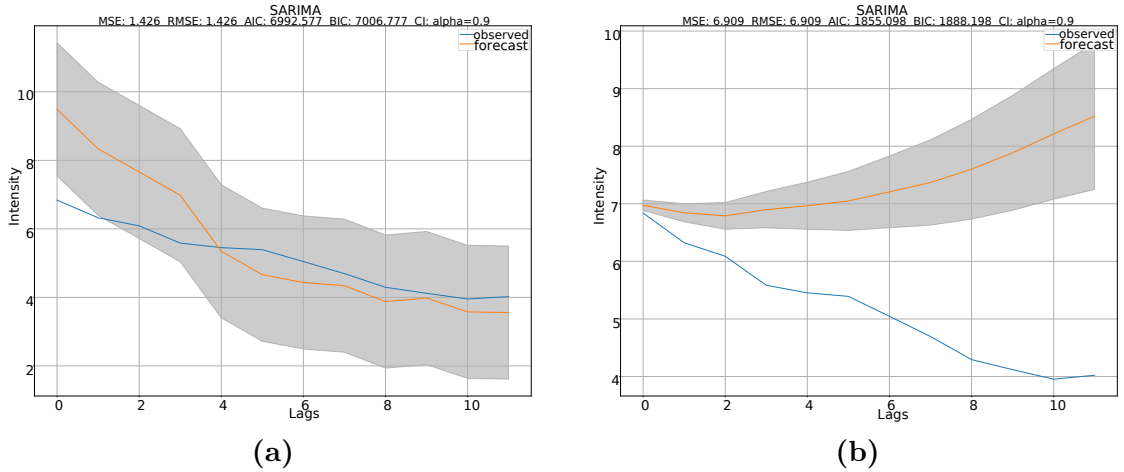


Figure 5.29: Choosing the best model using different evaluation metrics. **(a)** MSE **(b)** BIC.

To predict more than one hour, it was initially experimented to retrain the model using the predicted values as true values; however, the model was not able to make good predictions, as can be observed in figure 5.30. Using MSE, it was clear that the model was doing overfitting and it was not able to adapt. Using BIC, the model would starts to tend to a specific value, even on the second train. This confirms the limitations of statistical methods.

To achieve the goal of making predictions for a day, the only way that was possible was to retrain the model using the true values. As can be observed in figure 5.31, both models can make good predictions. The model chosen by using MSE does overfitting and presents a delay of the forecasted values. This shows that, choosing BIC as an evaluation metric leads to the choice of a good model, because the model does not make overfitting of the data for the first hour.

To observe the impact of training from twelve in twelve steps versus smaller steps, it was done the comparison with the divisors of twelve. Figure 5.32 contains the obtained values for Monday using 1, 2, 3, 4, 6, and 12 steps. As can be observed, increasing the number of steps leads to an increase in the error. One step forecasting is extremely precise. However, one step forecasting only allows us to predict five minutes ahead. In conclusion, fewer steps lead to predicted values with the smallest errors; however, the predicted time is decreased.

Figure 5.33 contains four subplots that analyze the data for the first hour of the forecasted

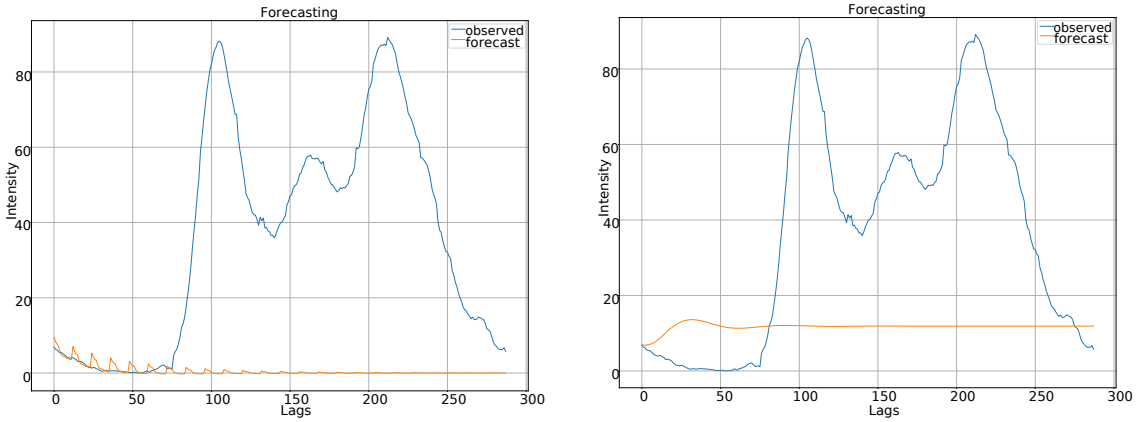


Figure 5.30: Forecasting an entire day training with forecasted values **(a)** MSE **(b)** BIC.

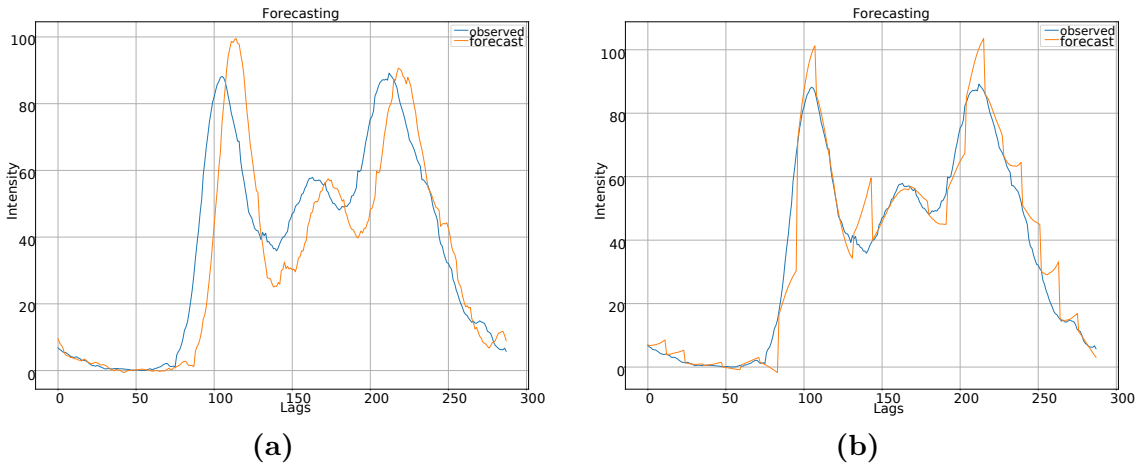


Figure 5.31: Forecasting an entire day, retraining with the true values. **(a)** MSE **(b)** BIC.

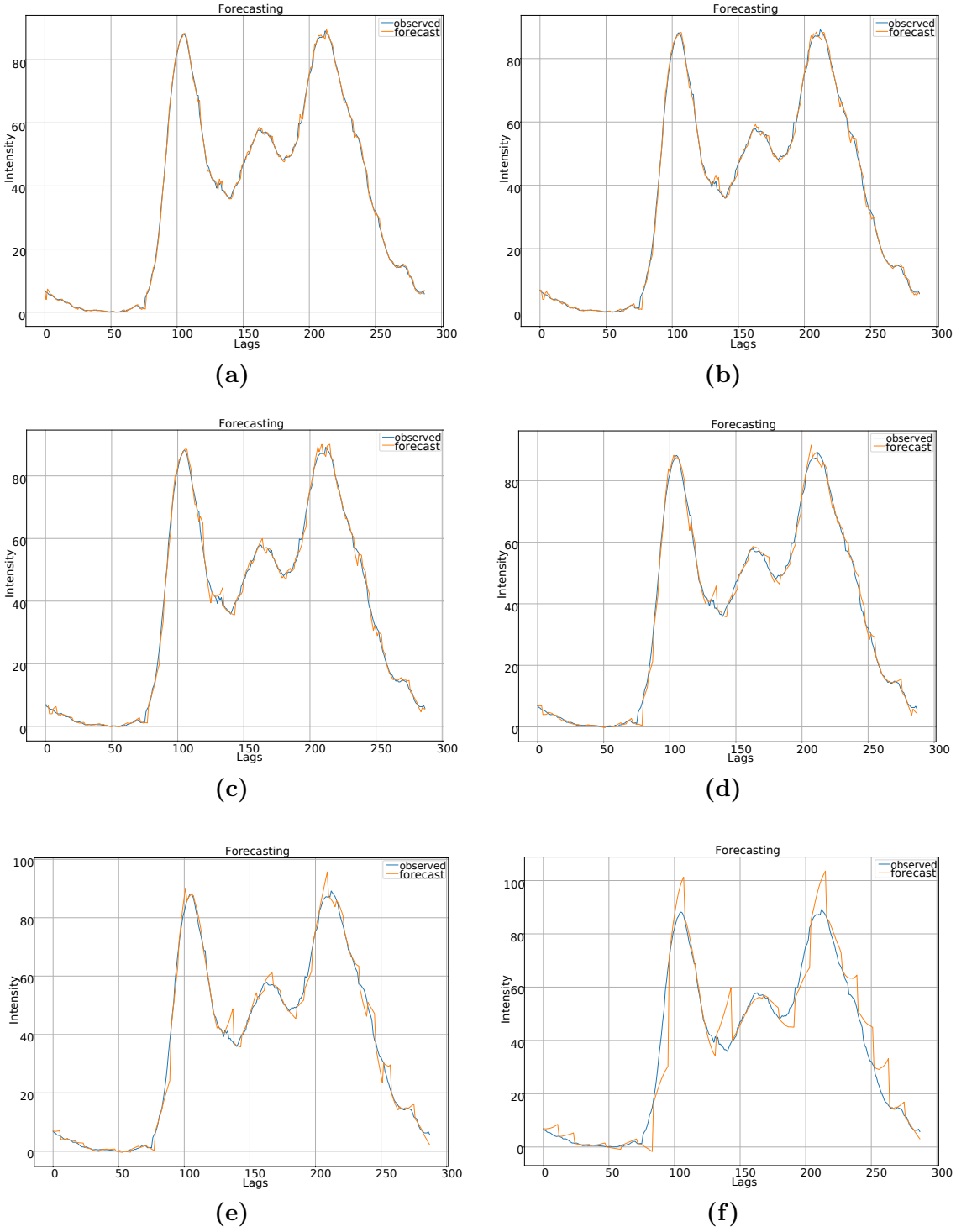


Figure 5.32: Forecasting traffic flow observed with SARIMA using different steps and BIC as an evaluation metric **(a)** 1 step **(b)** 2 steps **(c)** 3 steps **(d)** 4 steps **(e)** 6 steps **(f)** 12 steps.

Monday. The first one is the standardized residuals that indicate how different are the observed values from the true values over time. The histogram presented in the second subplot is very close to the normal, indicating the presence of some white noise. The Normal Q-Q plot, also known as the Quantile-Quantile plot, shows how closest is the data distribution from the Gaussian distribution that is represented by the red line. Since the blue dots are very close to the red line, then we can assume that there is a normal distribution of the data. The correlogram is the autocorrelation plot for the first 10 lags.

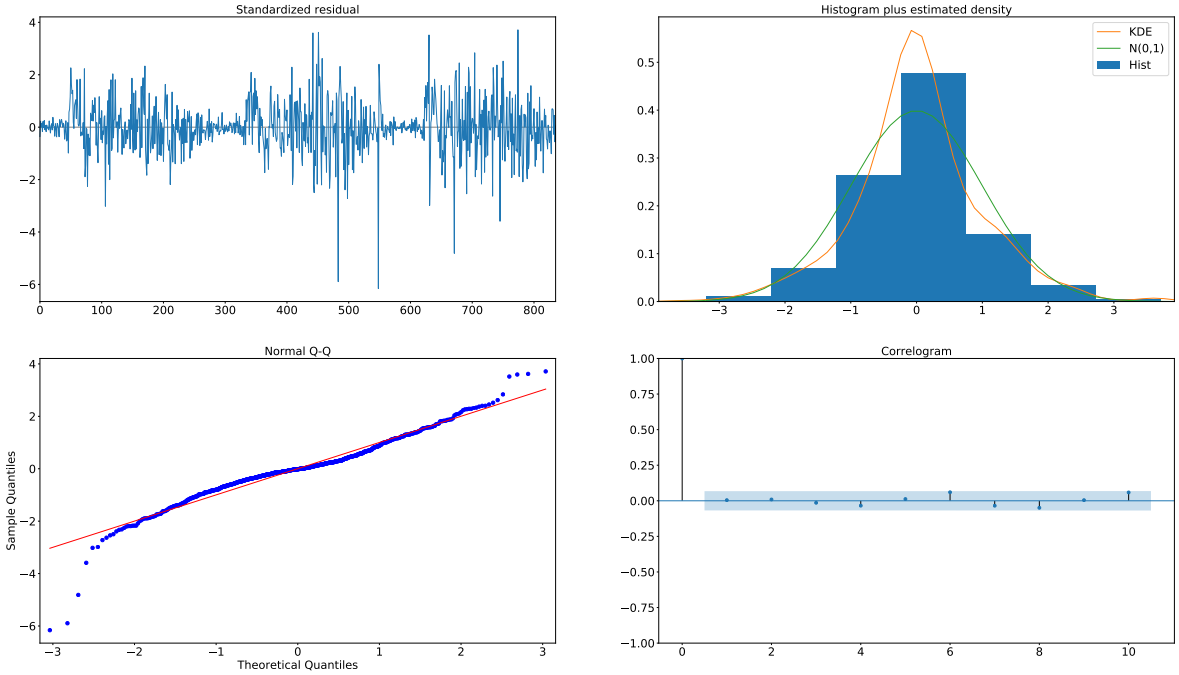
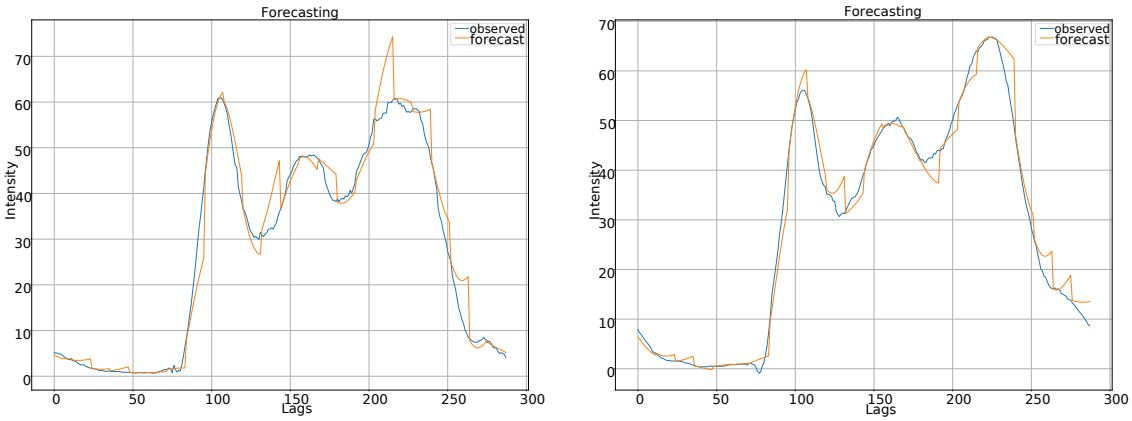


Figure 5.33: Plot Diagnostics of the traffic flow time-series

The presented work was specific for a traffic flow sensor. One of the goals was to forecast the values of other sensors without having to repeat all the work. A direct application of the model cannot be performed because we trained the model with the specific values of the traffic flow in study. To reuse the developed work, we will train new models with the same parameters but with the historic values of the traffic flow observed in study.

To evaluate if it was possible to apply models with the same parameters to other sensors and get a good performance, we tested the performance for the traffic flow sensor that presented high correlation values with sensor CT1Z8. The cross-correlation was previously performed and the results are presented in subsection 8.1.4.

Figure 5.34 shows the result for two different days. In general, the models presented a good performance, since the MSE is low and the curve is close to the real curve. By observation, it is possible to verify that the model does not overfit the data.



(a) **(b)**

Figure 5.34: Application of the SARIMA models to another traffic flow sensor (CT2Z8) that presented a high correlation with the sensor used to choose the model parameters **(a)** Monday **(b)** Wednesday.

5.5 Predicting the traffic flow with deep learning

Using deep learning methods to predict future traffic flow values allow us to predict, for example, one week of data. Deep learning methods are complex but achieve good results in long term forecasting. This section will present the process to choose the best lags to perform predictions, the results obtained using LSTMs and FFNN, and the impact of performing dropout.

Prediction methods

In the previous section we explained the network details of the several ANNs developed and tested. This section presents the results obtained by the LSTM networks and the several types of FFNN. It compares both of them and presents the predicted values.

Despite being different types of ANNs, LSTMs and FFNNs share the same types of parameters, as it happens with several types of ANNs. Table 5.14 contains the possible configurations to be tested.

Table 5.14: Configuration details of Artificial Neural Networks.

	Configurations	Values for LSTM	Values for FFNN
Model	Batch size	1	1
	Number of epochs	100	100
	Neurons	1, 2, 4, 8, 16, 32	1, 2, 4, 8
	Dropout	0, 0.1, 0.2	0, 0.1, 0.2
	Activation functions	sigmoid, tanh, relu, softmax	sigmoid, tanh, relu, softmax
Model Compile	Loss	MAE, MSE	MAE, MSE
	Optimizer	rmsprop, adam	rmsprop, adam

The first configuration is the batch size, that is the number of training samples that will go through the network in one forward and backward passage. The number of epochs is the number of times that the network will perform the learning part; for each one of them, the entire training dataset goes through the network. The neurons parameter is the number of neurons of the layers. The dropout is the percentage of dropout that is performed. The activation functions are the functions that are used in the neurons at each layer to process the information. The loss function is the function that the model will use to minimize the error. At last, the optimizer is an adaptative learning rate algorithm and the use of optimizers will help to reduce the losses. The evaluation metric chosen is the MSE.

Table 5.15 presents the number of possible configurations for LSTM, FFNN (with one, two, or three layers) and the total for all the ANNs.

Table 5.15: Number of combinations of the configurations of Artificial Neural Networks.

Model	Configuration Combinations
LSTM	288
FFNN	192 (1 layer) + 768 (2 layers) + 3072 (3 layers) = 4032
Total	4320

LSTM

LSTM neural networks are mostly used to predict sequential future values; however, they can be useful to predict other types of values. Table 5.16 presents the best configuration parameters for the LSTM neural network. The first column is an identification of the model (M stands for model) to compare the several best models with the results presented in table 5.17. It was chosen to verify the difference in the results when it was used 1, 2, 4, or 8 neurons. The second column is the number of neurons (N stands for neurons). After it was verified an improvement related to the number of neurons in the LSTM networks, it was also tested with 16 and 32 neurons.

Table 5.16: The best tested configurations for the LSTM neural network

M	N	Activation Function	Dropout	Loss	Optimizer
1	1	tanh	0.1	MAE	adam
2	2	tanh	0.1	MSE	adam
3	4	tanh	0.1	MSE	rmsprop
4	8	tanh	0.1	MSE	rmsprop
5	16	sigmoid	0.0	MAE	adam
6	32	sigmoid	0.0	MAE	adam

There are significant improvements when we increase the number of neurons that are used in the LSTM model. From 1 neuron to 2 neurons, the model decreases the MSE in 9,252. From 2 to 4 neurons, it presents an improvement of 5,753. From 4 to 8, it improves in 7,185. The increase of the number of neurons to 16 presents a very small improvement in the MSE, and the increase to 32 neurons increases a little the MSE. The MSE value is very similar to 8, 16, or 32 neurons. The same happens with the other evaluation metrics that present significant improvements until 8 neurons.

Table 5.17: Comparison of the values obtained by the evaluation metrics for the LSTM neural network

M	N	MSE	RMSE	MAE	Explained Variance	R^2-Score
1	1	63.173 ± 3.799	7.945 ± 0.241	6.390 ± 0.256	0.917 ± 0.004	0.916 ± 0.005
2	2	53.921 ± 1.832	7.342 ± 0.124	5.438 ± 0.132	0.929 ± 0.002	0.928 ± 0.002
3	4	48.168 ± 6.729	6.925 ± 0.490	5.005 ± 0.393	0.940 ± 0.006	0.936 ± 0.008
4	8	40.983 ± 3.151	6.397 ± 0.244	4.471 ± 0.291	0.947 ± 0.003	0.945 ± 0.004
5	16	39.174 ± 1.034	6.258 ± 0.082	4.367 ± 0.025	0.949 ± 0.001	0.948 ± 0.001
6	32	40.011 ± 1.084	6.324 ± 0.086	4.495 ± 0.104	0.948 ± 0.001	0.947 ± 0.001

The first four models present the same activation function and dropout value. The last two models present the same activation function, dropout value, loss parameter, and optimizer. Since dropout performs an important task, to ensure that it is not overfitting, the last two models present some disadvantages since the dropout value is 0.

We try to find the best models with dropout and 16 and 32 neurons. With 16 neurons, the second-best model used the *tanh* function as activation function and 0.1 as dropout, and presented a 40.790 ± 8.000 MSE value. With 32 neurons, in the second place, is the model with a $softmax$ activation function, a dropout of 0.2, and an MSE error of 40.812 ± 0.808. Both models present good results and show that it is possible to get good models with more than 8 neurons.

It is important to perform several simulations, in this case 6, because the same model can give different values. If we just perform one simulation and choose the best model, we can have the risk of choosing a biased model to the input data. For that reason it is very important that we look into the standard deviation. All models in table 5.17 present a small standard deviation value.

One of the simulations that allow us to obtain the best performance models is presented in figure 5.35. The model used the parameters of the best model with 8 neurons. The MSE has a value of 40.983 on average, the RMSE presents a value of 6.397 on average, and the MAE presents a value of 4.471 on average. All these metrics present low error values and the standard deviation is also low in all cases. This means that the behavior of the model is very close to the real behavior. The value of the variance is very close to 1, meaning that the model could learn the dataset and very little information was lost. The R^2-Score value is also very close to 1. The R^2-Score is very similar to the variance and refers to the part of the model in which the output variable can be explained by the input variable. The standard deviation is low in both cases.

It is possible to observe that the behavior of the predicted values is very close to the behavior of the true values. There are some local maximums and minimums that present the major differences between predicted and true values. Anyway, the model presents a good behavior since the predicted curve is very close to the true curve.

The training time is an important factor when we are choosing the model. If a model is very good but takes too much time to train, sometimes it can be better to use a model that presents a similar but worst performance and has a more acceptable training time. Table 5.18 presents the training time for the LSTM models. It was chosen to divide the results by the number of neurons and the dropout ratio.

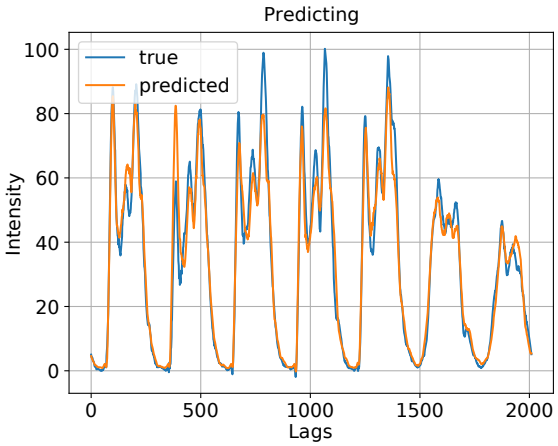


Figure 5.35: Predicting traffic flow observed by using an LSTM neural network.

Table 5.18: Training time (min:sec) for the LSTM models

	Dropout		
N	0	0.1	0.2
1	13:48.863	12:49.775	11:58.303
2	13:42.483	12:40.991	11:49.502
4	13:26.971	12:29.341	12:08.250
8	13:45.996	12:50.998	12:07.562
16	13:49.534	12:41.554	11:49.280
32	13:49.806	12:56.986	12:19.271

By observation, it is possible to conclude that the dropout ratio has an impact on the training time. Increasing the dropout leads to a decrease in the training time. This happens because, by performing dropout, we are discarding information, and that information will not be used in the following steps. The number of neurons does not affect the training time, given that the values in each column are very similar. This is due to the capability of the networks to use all available processing resources regardless of the network configuration.

To evaluate the capability to reuse the network configuration in other traffic flow sensors without having to choose the best models, it was trained a network for predicting the values for the sensor CT2Z8. Figure 5.36 is the result obtained after we create the model with the best parameters using 8 neurons.

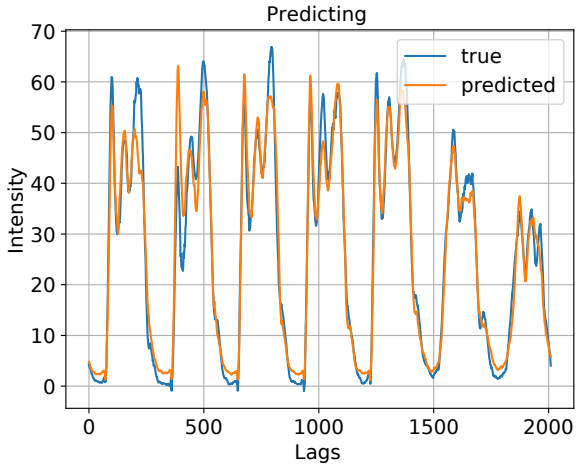


Figure 5.36: Reusing the best LSTM model configurations to predict another traffic flow sensor.

It is possible to observe that the predicted curve is close to the real curve. Once more there are some problems with the local maximums, but in general, the model presented a good performance, having a MSE of 17.892, a RMSE of 4.229, and MAE of 2.906. All error metrics have low values. Regarding the explained variance and R^2-Score, the model presented values of 0.957 and 0.957, which are both values close to 1. The model presented good results when applied to another sensor, and the evaluation metrics are even better than the ones obtained for the sensor CT1Z8. Note that the range of values is smaller for this sensor, and that can lead to smaller errors.

FFNN

FFNNs are the other type of ANN being tested. It was tested different FFNN topologies with networks of 1, 2, or 3 layers. To a better understanding of the network topologies, please check section 6.3.

The best results obtained by the application of FFNN are presented in tables 5.19 and 5.20. The first column of table 5.19 is the model number that corresponds to the model results in table 5.20. Once more, the second column is the number of neurons and the third column is the number of layers. This table presentes three types of different models, but they are all in the same table because they are all FFNNs.

In all the FFNN models the best result was obtained by having no dropout. This can reveal a big issue. Not having dropout means that the model is doing overfitting of the data. If the model does overfitting of the data, it will not have the capability to adapt over time.

Table 5.19: The best tested configurations for the feed-forward neural network.

M	Number of neurons	Number of layers	Activation Functions	Dropout	Loss	Optimizer
1	1	1	sigmoid	0.0	MAE	adam
2	1	2	sigmoid, sigmoid	0.0	MAE	adam
3	1	3	sigmoid, sigmoid, sigmoid	0.0	MAE	adam
4	2	1	relu	0.0	MAE	adam
5	2	2	relu, sigmoid	0.0	MAE	adam
6	2	3	relu, sigmoid, sigmoid	0.0	MAE	adam
7	4	1	sigmoid	0.0	MSE	adam
8	4	2	sigmoid, relu	0.0	MSE	rmsprop
9	4	3	sigmoid, relu, sigmoid	0.0	MSE	rmsprop
10	8	1	softmax	0.0	MAE	adam
11	8	2	sigmoid, relu	0.0	MSE	rmsprop
12	8	3	sigmoid, relu, sigmoid	0.0	MAE	rmsprop

Table 5.20: Comparison of the values obtained by the evaluation metrics

M	MSE	RMSE	MAE	Explained Variance	R2 Score
1	34.334 ± 0.159	5.859 ± 0.013	4.225 ± 0.030	0.954 ± 0.001	0.954 ± 0.002
2	37.317 ± 0.377	6.108 ± 0.030	4.410 ± 0.074	0.952 ± 0.005	0.950 ± 0.004
3	39.543 ± 0.616	6.288 ± 0.049	4.573 ± 0.055	0.950 ± 0.006	0.947 ± 0.008
4	39.748 ± 7.668	6.282 ± 0.608	4.256 ± 0.570	0.953 ± 0.005	0.947 ± 0.010
5	38.706 ± 2.276	6.219 ± 0.181	4.409 ± 0.261	0.951 ± 0.004	0.948 ± 0.003
6	36.862 ± 3.733	6.065 ± 0.301	4.251 ± 0.327	0.952 ± 0.003	0.951 ± 0.004
7	40.087 ± 4.362	6.324 ± 0.339	4.490 ± 0.433	0.948 ± 0.005	0.947 ± 0.005
8	36.469 ± 5.270	6.027 ± 0.437	4.033 ± 0.329	0.953 ± 0.005	0.951 ± 0.006
9	35.915 ± 2.672	5.989 ± 0.220	3.968 ± 0.197	0.954 ± 0.001	0.952 ± 0.003
10	41.524 ± 0.917	6.443 ± 0.071	4.472 ± 0.135	0.946 ± 0.001	0.945 ± 0.001
11	34.070 ± 1.741	5.835 ± 0.149	3.877 ± 0.211	0.955 ± 0.001	0.955 ± 0.002
12	36.416 ± 1.304	6.033 ± 0.108	4.206 ± 0.154	0.952 ± 0.001	0.951 ± 0.001

There is not a specific pattern relative to the activation function used; however, the *sigmoid* and the *relu* functions are relevant in the results. The evaluation metrics are all very similar for each model, being the best model, the number 11 with 8 neurons and 2 layers. The model presents low values in the error metrics (MSE, RMSE, and MAE), and values very close to 1 in the remaining ones. In a general way, the model presents a good performance.

Model 11 is the best model; however, model 1 presents a very close MSE value and presents a much lower standard deviation. This could indicate that model 1 is better than model 11.

However, if a model presents an extremely small value for standard deviation, that might mean that the model will react worst to changes because it has no adaptive capabilities. The differences are small between both models, but model 11 has better variance and R^2-Score values, meaning that model 11 describes better the data.

We try to find the best models that perform dropout (0.1 or 0.2). As expected, the error values increased, and in some cases, the error increases more than the double. In most of the cases, the best models with dropout appear after several models without dropout. FFNNs tend to be less flexible than LSTMs, because they perform dropout on the FFNNs, this means the loss of important information and that leads to increasing the error. The most common activation functions are *tanh* and *sigmoid*.

The best model with dropout has 8 neurons and 1 layer that uses the activation function *tanh*. The model presents low error values, having an MSE of 44.857 ± 0.752, and a variance and R^2-Score close to 1. Given that the increase in the error metrics is small, it is preferable to use the best model with dropout. Figure 5.37 shows the best model with dropout.

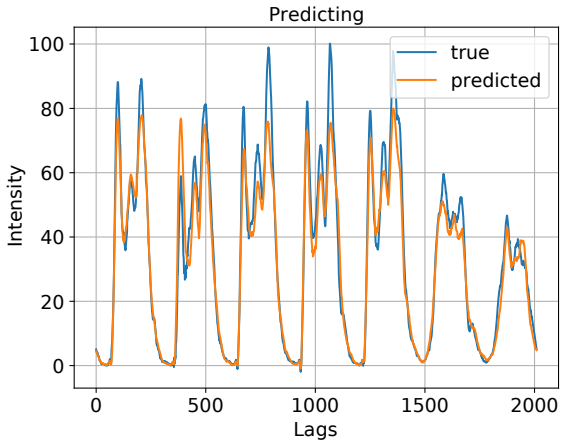


Figure 5.37: Predicting traffic flow observed with FFNN.

It was studied the training times of the FFNN. Table 5.21 contains the times organized by the number of layers, number of neurons, and dropout ratio. By observation, we can conclude that the training time depends on the three factors in which the table was divided. The training time increases with the number of layers, with the number of neurons, and increases with the presence of dropout. The columns with 0.1 of dropout versus 0.2 of dropout have very similar values.

It was tested, once more, the application of a model with the same configurations of the best dropout model obtained for the FFNN to another sensor. The results are presented in figure 5.38.

The model presented an excellent performance, being the error metrics 12.295 for the MSE, 3.506 for the RMSE, and 2.341 for the MAE. The variance and R^2-Score are both very close to 1, being both 0.970.

Table 5.21: Training time (min:sec) for the FFNN models

		Dropout		
L	N	0	0.1	0.2
1	1	3:22.167	3:30.213	3:31.124
1	2	3:23.487	3:34.512	3:34.182
1	4	4:09.369	4:42:110	4:33.129
1	8	4:04.125	4:36.567	4:35.333
2	1	4:40.384	4:59.249	4:58.340
2	2	4:45.527	5:01.003	5:01.492
2	4	5:58:021	6:31.470	6:45.299
2	8	6:18.452	6:47.339	6:48.286
3	1	5:33.266	5:55.329	5:54.485
3	2	5:40.099	6:00.016	6:00.338
3	4	7:15.199	8:00.442	8:18.020
3	8	7:44.264	8:22.588	8:21.597

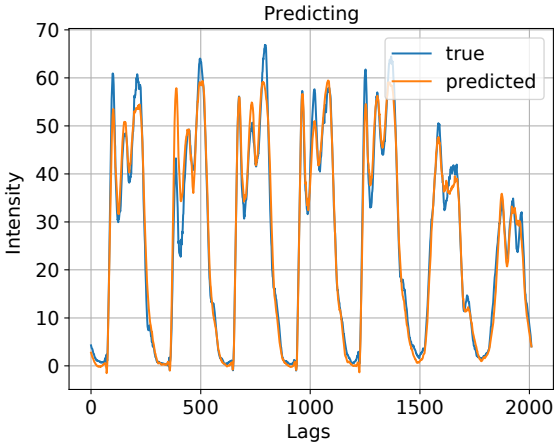


Figure 5.38: Reusing the best FFNN model configurations to predict another traffic flow observed sensor.

The impact of dropout

Performing dropout has its advantages and disadvantages, as it was being discussed over this dissertation. To understand better what means to perform dropout, it will be presented some images, in figure 5.39, that show the impact of dropout. It was chosen to use the configuration parameters of the best LSTM model. The only parameter that was changed was the dropout ratio.

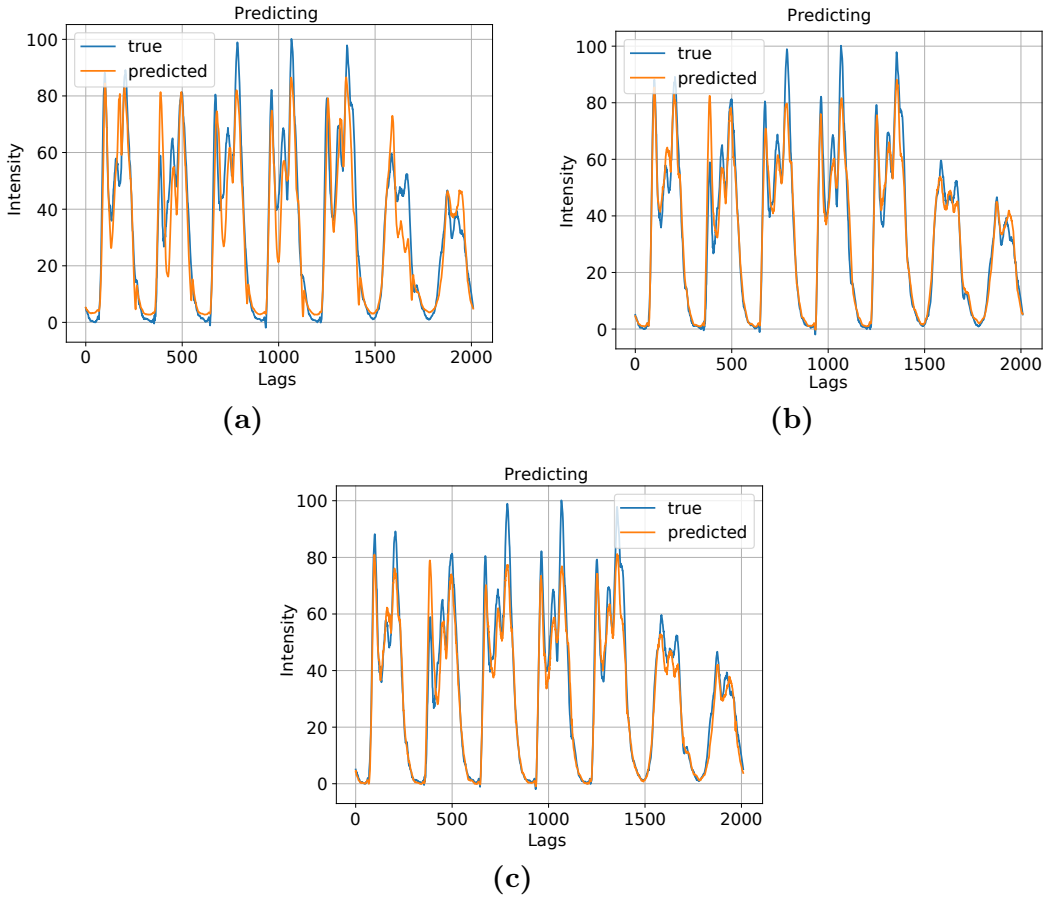


Figure 5.39: The impact of dropout in the prediction of future values, using LSTM models, with dropout value of **(a)** 0 **(b)** 0.1 **(c)** 0.2.

Not using dropout leads to some errors in the curve behavior, as it is possible to observe by the simulations of figure 5.39a. The simulation did overfitting of a pattern observed in the data. The images 5.39b and 5.39c are very similar and show one of the problems associated with performing dropout. There is a set of values that are discarded. In the images, it is possible to see that there are some local maximums and/or minimums that are never achieved.

Some of the best models were obtained without dropout, but because the model performed overfitting of the data, the same configurations could also lead to models with a really bad performance. This contributes to the creation of models with a huge standard deviation for the evaluation metrics.

Discussion

LSTM seems the most appropriate ANN to use to predict the traffic flow because it presents good results; the best models perform dropout and it is consistent in terms of parameter values. Some FFNN achieve better results but did not perform dropout, meaning that they are more sensitive to changes. The best LSTM parameters seem to be the best model with 8 neurons. SARIMA models presented good results, but they can not perform long-term forecasting as the ANN models.

LSTMs present bigger training times, but since we want to predict the values for one week, the training times are relatively small. The best models for each type of ANN can indicate the best configuration parameters for the construction of models for sensors that are very correlated with the sensor in the study.

5.6 Abnormal traffic behaviour detection

Anomaly detection allows us to recognize that the present or near future conditions of the city (its traffic) are uncommon, given the observed history. Anomaly detection is difficult to perform because it is, in most of the cases, unexpected. However, there are a few cases in which we could predict that it will happen something unusual. Since we expect to observe patterns in traffic as a result of calendar seasonality, we can also expect the existence of anomalies when there is something that is not dictated by the usual seasonality. The best example is when a holiday occurs.

November first is a religious holiday celebrated in Portugal. In 2019, this holiday happened on a Friday, being the perfect candidate for our study. Since this holiday happens one week later after the last week in the study, it was chosen to analyze the next week based on the models previously calculated to verify their behavior. Besides that, it is also performed some analysis that can help to detect some anomalies.

In figure 5.40, we observe that, in the week from October 27 of 2019 to November 3 of 2019, is it possible to detect the anomaly presented in the holiday. The traffic values are much lower than on the Fridays of the other weeks. The holiday even presents lower values than Saturday and Sunday.

After it was performed the smoothing of the time-series, it was done the seasonal decomposition from October 7 of 2019 to November 3 of 2019. By looking into figure 5.41, we can observe the presence of the anomaly. The observed component is the time-series after it was performed the smoothing step, and, once more, we can visualize a decrease in the values corresponding to the holiday. In the residual component, there is a significant decrease in the holiday, highlighting the anomaly.

It is also possible to visualize a decrease in the trend component, even before the holiday. This indicate that, the existence of a holiday in a Friday allows that some people take some additional vacation days.

The only component in which the anomaly is not visible is the seasonal component. This happens because the seasonal component only retains the patterns, the seasonal and cyclic patterns. Since every year, the weekday in with the holiday happens changes, it would be very difficult to have a pattern with just one or two years of information. It would be necessary much more information to detect and study the pattern presented in the holiday.

The residual component is the component that contains the anomaly. To detect and isolate the anomaly, we should focus on what happens in this component. In this case, the anomaly

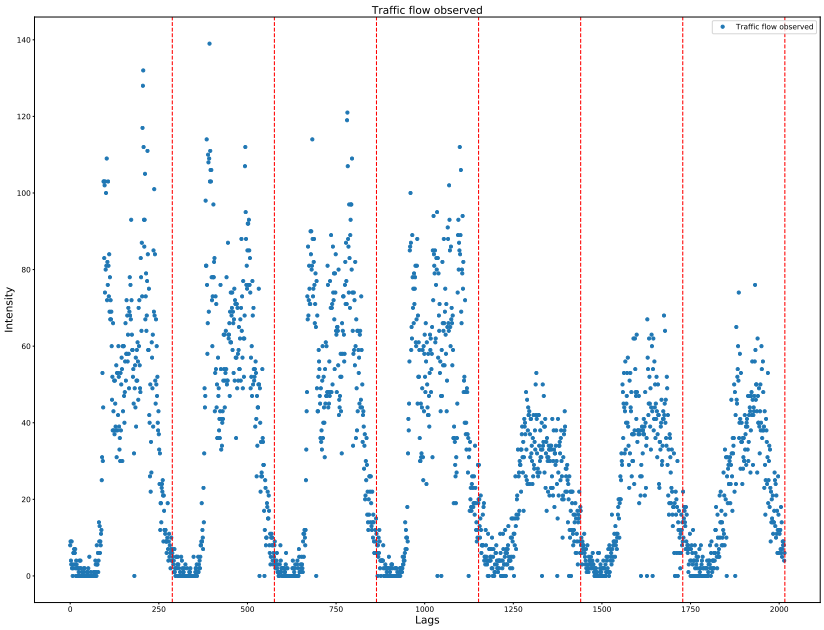


Figure 5.40: Traffic flow observed from October 27 of 2019 to November 3 of 2019.

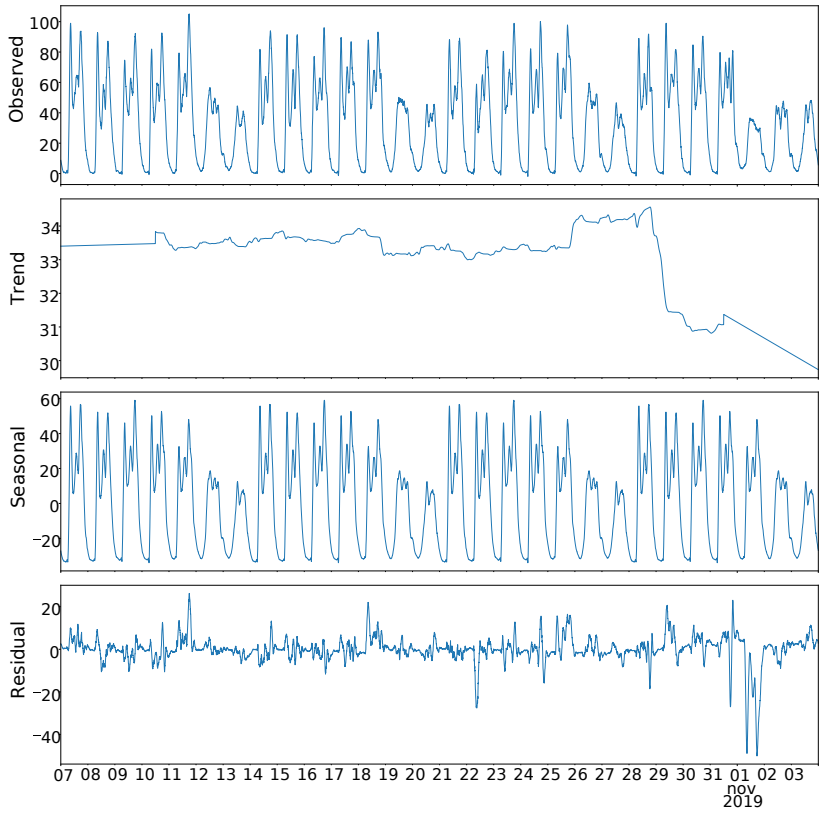


Figure 5.41: Time-series additive decomposition (frequency = 2016) with an anomaly

generates a decrease in traffic. In figure 5.42c, a marked pattern is found corresponding to the traffic anomaly; it corresponds to the analysis after performing a day resample. In contrast, other time frames do not accomplish the goal, as exemplified on figures 5.42a, and 5.42b.

Resample data by day is useful if we expect anomalies for an entire day, but for detecting more isolated anomalies, like accidents, it might be more useful not to perform the resample or just resample in one hour interval. If the anomalies can affect an entire week, like the academic week, we might even have to perform the resample by week.

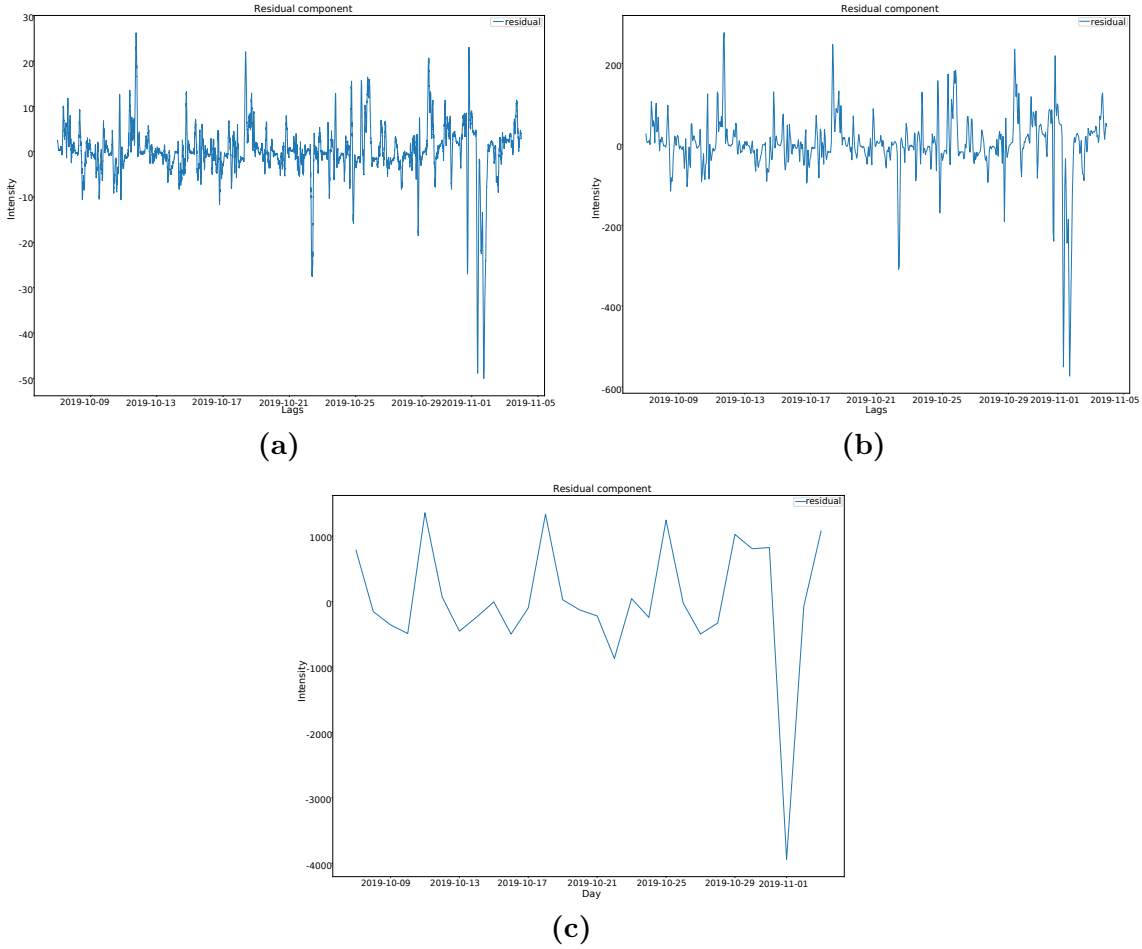


Figure 5.42: Time-series additive decomposition - Residual component **(a)** Original **(b)** 1 hour frequency **(c)** 1 day frequency.

We already have a way to determine the anomaly; the real question is if it is possible to forecast the traffic flow even when there is an anomaly. To answer to this question, we first started with SARIMA model. Figure 5.43 presents the forecasting done with SARIMA. The only disadvantage of using SARIMA is that we have to retrain the model at each hour. However, the model presents good performance, proving once more that the model is generalized. The same did not happen when we try to apply the LSTM neural networks, as can be observed in figure 5.44.

Since LSTM was trained to perform long term forecasts, the LSTM model did not contain

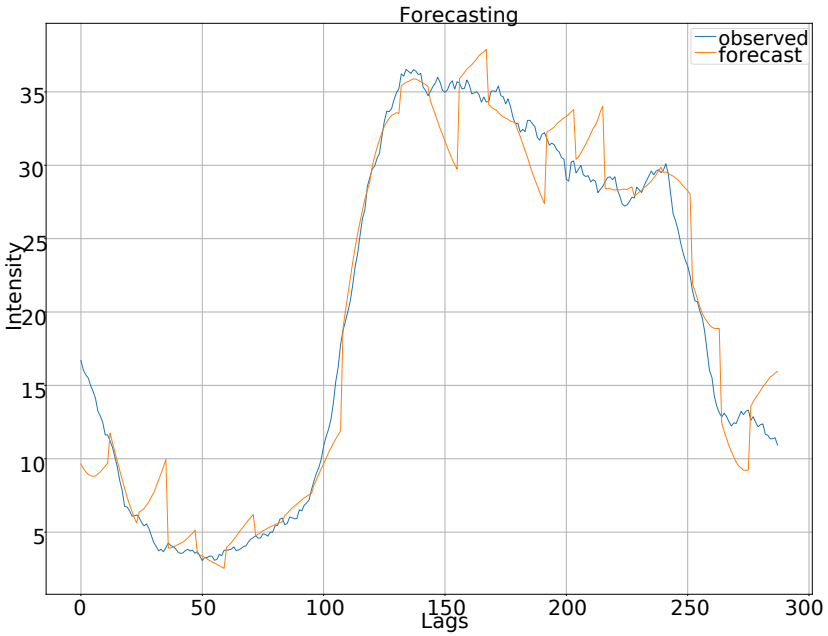


Figure 5.43: Forecasting anomalous traffic flow observed with SARIMA (12 steps)

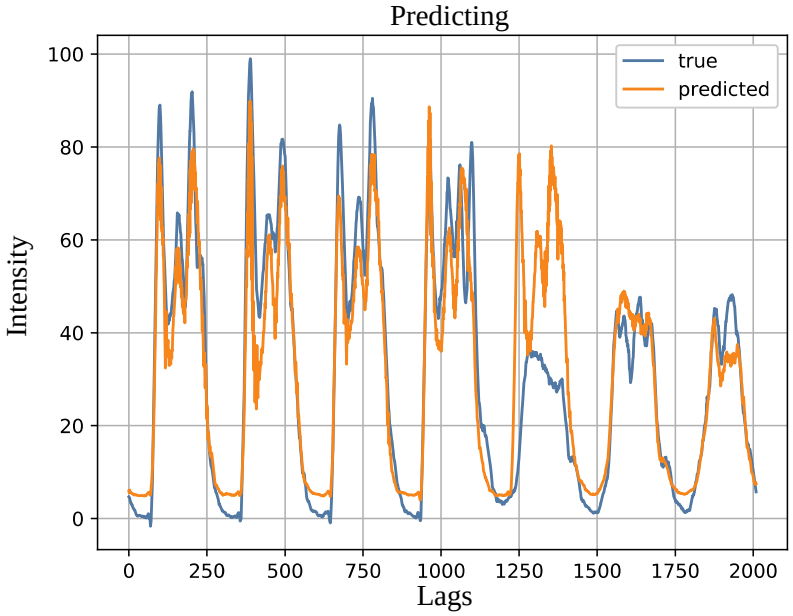


Figure 5.44: Predicting anomalous traffic flow with LSTM.

any information that the anomaly occurred. As such, it was not able to get good results, and there is a huge difference between the predicted values and the true values for the November first.

Even when the previous hour was added to the features group, in order to verify the impact in the predicted values, there was not a significant improvement, as can be observed in figure 5.45. Both LSTM models are very similar and the error obtained for this Friday is large, as it is represented in figure 5.46.

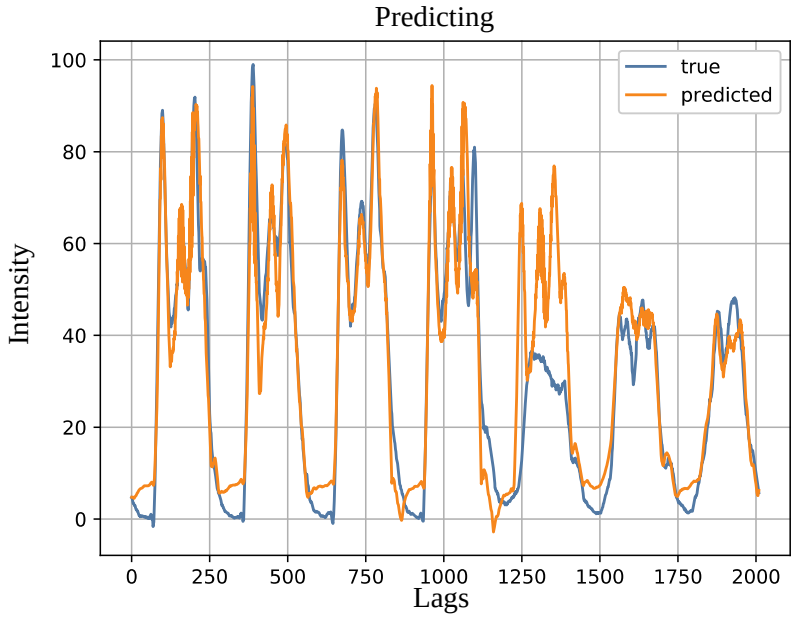


Figure 5.45: Predicting anomalous traffic flow with LSTM using the previous hour.

The error obtained by the predicted value and the true value shows that there is a bigger error when the holiday occurs, as can be observed in figure 5.46, because it is something unexpected, and the model can not predict. Once more, we can detect the anomaly after it happened.

Note that, all lags present a strong correlation with lag 0, meaning that the use of lag 12 begun not important, and that might be the reason why the model does not adapt.

One of the reasons why we get better results with SARIMA than with LSTM is due to the capability of SARIMA to look into data as a sequence. However, LSTM is a type of model for sequential data and can not adapt. This might be due to ANNs being formatted to receive input data and output data, and not pure sequential data.

By analyzing the residual component or the error component, it is possible to detect the anomalies; however, the detection just happens after they occur. Is it also possible to detect them when they occur, if we use SARIMA, but it is not possible to detect them far in advance.

5.7 System implementation

Figure 5.47, representes the architecture developed to process data from Porto. Note that part of the infrastructure already existed, as is the case of Veniam database. Besides that,

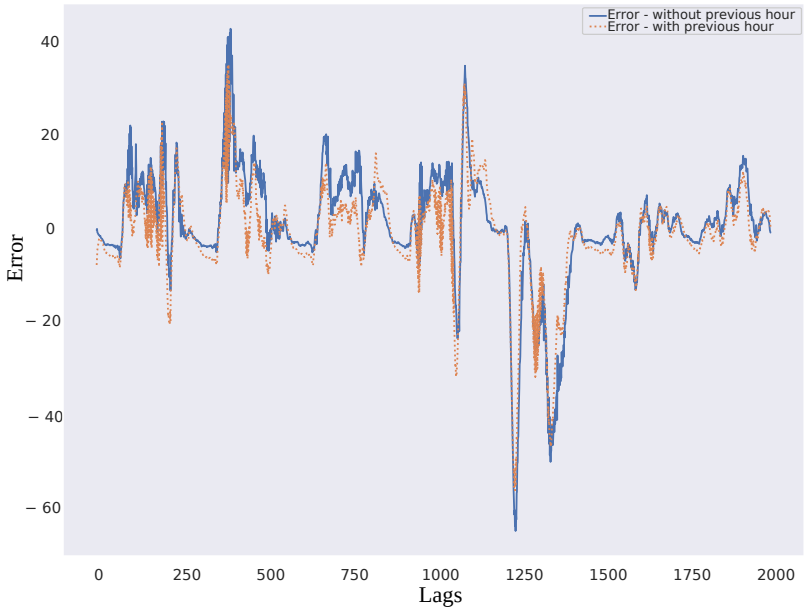


Figure 5.46: Predicting anomalous traffic flow - LSTM errors.

---

the components were distributed by several machines. The database from Veniam was in one machine, the infrastructure responsible for persist the PortoDigital data was in another machine and all the other components were in a third machine.

The schema relative to the Veniam data source and database is simplified because it already existed. However, there is a Webhook Infrastructure responsible to persist the data from the WebSocket endpoint into the database.

It was necessary to build an infrastructure to gather and manage the data from PortoDigital. Fiware [52] is a standardized open-source platform that allows us to achieve our goals and, since PortoDigital provides us an endpoint using Fiware Orion Context Broker Generic Enabler (also known as Orion), it seemed the obvious choice. Fiware offers different components (Generic Enablers), being Orion the only that is mandatory. Orion provides a Restful API designated by Fiware NGSIv2 API. The Cygnus module allows the persistence of data in the databases [52].

Both components, Orion and Cygnus, allow interaction with other components through the creation of subscriptions. Note that, the MongoDB database was just used as an auxiliary database to the Fiware infrastructure. MongoDB persisted data related to subscriptions and data about the entities, but just for a short period. The database responsible to persist the information at a long term was the PortoDigital database in PostgreSQL.

The main reason to use GTFS files were to associate segments with bus location or traffic flow locations. The GTFS files have one file that allows us to get all the road segments where buses go through, called *shapes.txt*. The *shapes.txt* file was converted into a shapefile, and after that, it was created a PostGIS database and all the segments were inserted. The GTFS files also contained information about the buses schedules. This information was useful to validate the information from Veniam.

The cache files associated with the preprocessing models were created because the infor-

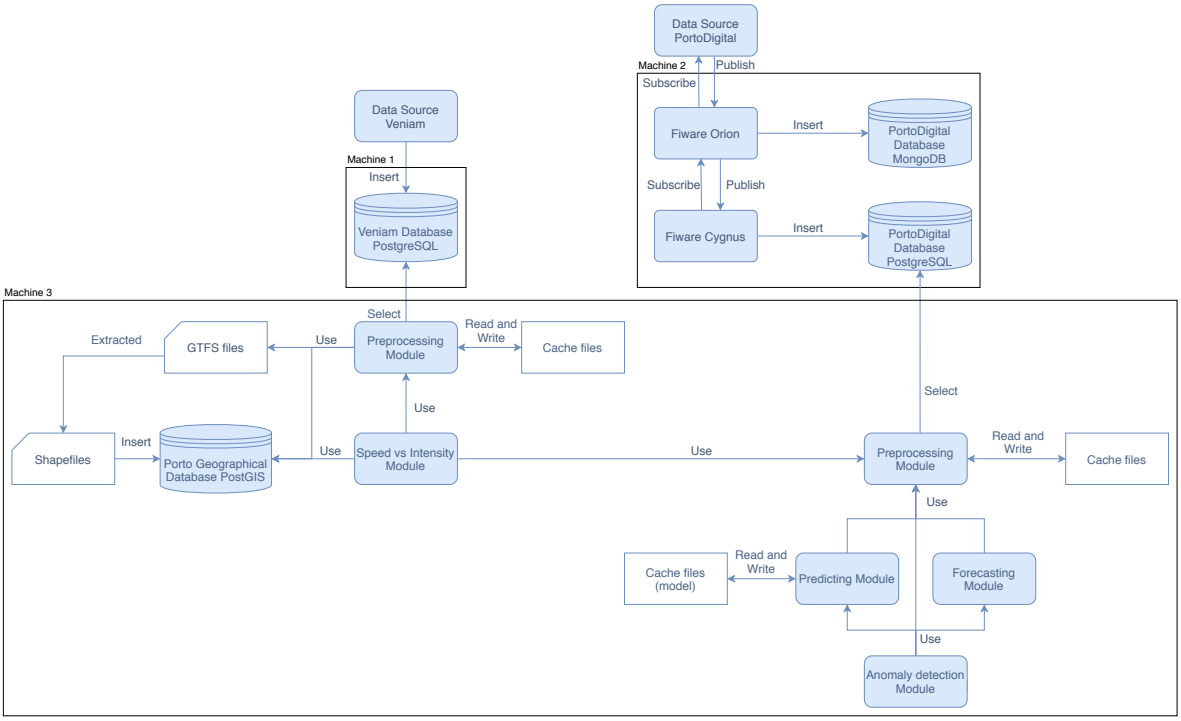


Figure 5.47: The system architecture of data from Porto

---

mation that was used was in most cases the same. Besides that, the third machine would have to be connected to IT *Virtual Private Network* (VPN), since the other two machines, the ones with Veniam and PortoDigital data, were just accessible inside the IT network, and the communication cost was significant. Another reason for the creation of cache files was that, for some of the use cases, it was useful knowing the segment of the bus, and the process to calculate was computationally expensive.

The cache files associated with the predicting model were used to save the machine learning models when they were created. Those files have an extension $h5$ and can be loaded to make predictions, retrain the model, etc.

The preprocessing module connected to the Veniam database is responsible for preprocessing data from Veniam, check the quality of the data, and associate the GPS position to bus segments. The preprocessing module connected to the PortoDigital database has a similar role but for the data from PortoDigital. Connected to this module are the modules to perform the forecasting and the prediction of the traffic flow data. The anomaly module uses the predicting module and the forecasted module to predict anomalous data. The speed vs intensity module tries to find a relationship between traffic flow intensity and the bus data.

All the databases and the Fiware components are deployed in Docker containers. The Veniam database uses Docker Swarm. The cache files, GTFS files, and Shapefiles are kept in the file system. The models that realize the preprocessing and the analysis are developed in Python 3.

5.8 Summary

In this chapter, it was explained the methodology adopted for forecasting the traffic flow. The dataset exploration at the early stages of the work reveals several problems associated with the data. The major problems were related to missing data, the existence of larger values than expected, imprecise GPS coordinates, etc. These problems were solved by applying smoothing techniques, working with road segments, etc. In general, the difficulties were overcome.

It was observed that, we can not describe a relationship between speed and traffic flow, however; when one of these variables presents large values, the other will present small values. Since we can't find the relationship, the speed values were not used in the predicting module. Besides that, the speed values had an enormous quantity of NaN values when associated with a traffic flow sensor because, even in a city like Porto, there are a lot of intervals in a location, which does not go any bus for a time interval.

An analysis of the data reveals patterns and seasonality. The data is stationary, meaning that it is not necessary to apply any differential techniques. Smoothing data prof effective and solve the major problems associated with the data.

The forecasting techniques presented good results; however, SARIMA can only make forecasting for a very limited period of time (1 hour), while the deep learning models can make long-term predictions (1 week). However, SARIMA can adapt better in anomalous conditions. This happens because of the nature of the models. SARIMA sees the data as sequential, knowing the logical order, while ANNs despite knowing the sequence, do not perceive the order in the same way. Note that, LSTM can save information, and even that is not enough.

Chapter 6

Driving Behavior

Driving behavior is a complex field of study, being the driver the central element. There are several aspects that can influence the way how driving is performed. Having a safe driving behavior is much more than respecting the speed limits and the road rules. The driver should have a defensive driving, should be observant and careful with what is going on around, and should adapt his/her behavior when it is necessary.

Section 1 introduces the 3 datasets used in this chapter. We had to prepare the datasets before performing any analysis. In section 2, we explain the method created to classify driving behavior. Section 3 presents the web application created to support the city manager to make decisions. Section 4 contains the system implementation developed for this chapter.

6.1 Data set preparation

The data from Aveiro also came from different sources, correspond to different types of information, and had different formats. Once more it was necessary different pipelines for preprocessing the data. Since the data from OSM came from sources without any kind of verification, it was necessary to verify the quality of that data. Table 6.1 contains a resume of the data types, sources, and location for Aveiro.

Table 6.1: Data sources

Data source	Data type	Data collection location
AveiroBus / NAP	Bus data	Aveiro
OSM	Road data	Aveiro
OSM	*Shapefiles* data	Aveiro

The data from the buses from Aveiro, with the OSM data, will be used to study driving behavior.

6.1.1 AveiroBus data

Currently, it is being prepared the integration of the data from sensors installed on AveiroBus buses with the existing infrastructure. At the time of this work was being developed, the integration did not exist; for that reason raw data files collected directly from the OBUs in the buses were used. Later, we made the necessary changes.

Members of our group installed 10 DCUs and 10 OBUs in the buses to receive information about the speed of the buses, the heading, the GPS coordinates, and the environment sensors. The data was collected every second. Figure 6.1 depicts one of the OBUs that was installed.

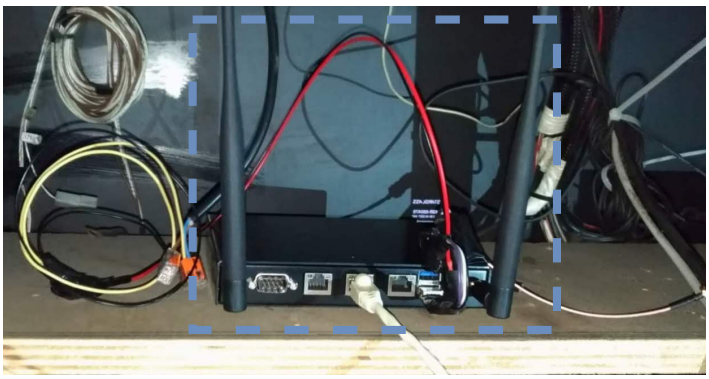


Figure 6.1: Example of an OBU installed on a bus.

AveiroBus data had some problems, particularly some timestamps had negative values, there were special characters, and some speed and heading values were NaN. There were also some outliers because the values were too high.

Note that, we are using anonymised data, and it is not possible to tag the driver, as long as multiple buses are considered in the analysis.

6.1.2 Shapefiles Aveiro

It was not possible to obtain for Aveiro the same type of GTFS files that we have for Porto. However, by using available OSM API's, it was possible to get some information about the bus network of Aveiro. This information was not verified by any entity, and it can have some faults and be incomplete. However, it was useful to understand the bus stop localizations. Figure 6.2 contains the information given by the files.

6.1.3 Roads Aveiro

The roads' information was also obtained by using an OSM library. Similar to the previous case, this information was not verified by any entity. However, it was useful to associate GPS positions from buses with the respective road segments. Besides that, it contained some useful information about the maximum speed for cars. Since we are working with data from buses, it was necessary to correct some of the data to get the maximum speed for buses.

For some of the road segments, there is not a value of maximum speed. The maximum bus speed was determined using the type of road associated with the segment. In some cases, even the type of road was missing. In that case, it was considered that the type of road was residential, and the maximum bus speed was 50km/h.

Table 6.2 [40] resumes the maximum speed for cars versus the maximum speed for buses. All the corrections were made taking that information into consideration. It is also important to remember that there can be some specific restrictions in some segments. If the maximum speed was lower than the expected speed, then it was not changed.

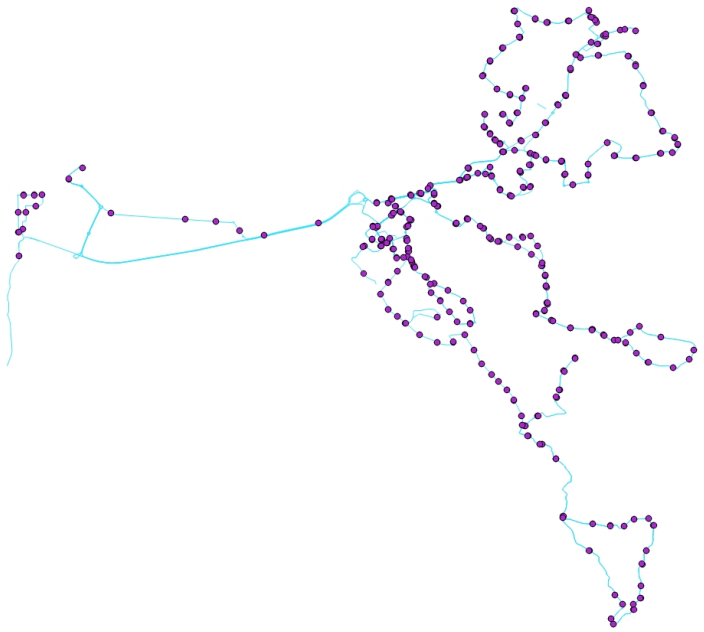


Figure 6.2: Visualization of the Aveiro bus network, defined by the GTFS dataset.

Table 6.2: Maximum speed corrections

Type of road	Maximum speed (cars)	Maximum speed (buses)
Coexistence areas	20 Km/h	20 Km/h
Residential	50 Km/h	50 Km/h
Remaining ways on public streets	90 Km/h	80 Km/h
Reserved lanes	100 Km/h	90 Km/h
Motorways	120 Km/h	100 Km/h

Note that, the OSM terminology has several names for the types of streets when compared with Portuguese terminology. Some of the names were: residential, secondary, tertiary, trunk, unclassified, tertiary link, primary, trunk link, secondary link, primary link, living street, motorway link, motorway, etc.

Figure 6.3 contains the road segments with the maximum bus speed associated. This image was obtained after the several corrections mentioned above were done. Note that, there can be some errors associated with our assumptions.

6.2 Classifying driving behavior

The mobility data from Aveiro offers an increased frequency (1 sample per second) than the data from Porto (each minute), thus enabling to study acceleration as a dimension to understand driving behavior. The main attributes are GPS location, speed, and timestamp, it was necessary to choose a model based on these type of features, or features that could be calculated like acceleration, or traveled distance.

To analyze driving behavior, we created a model inspired by the model proposed in [46].

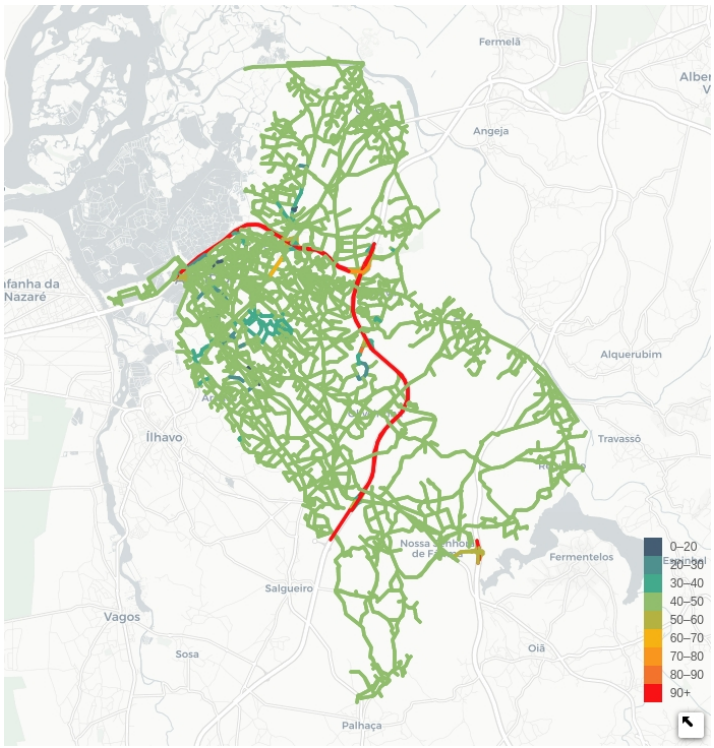


Figure 6.3: Roads Aveiro.

This model is based on the relationship between speed and acceleration, and the relationship between speed and road friction to determine if the driving behavior is safe or unsafe. This model was described in section 3.3 with more details.

The relationship between speed and acceleration, and the relationship between speed and friction are simplified since the road material present in Aveiro is mostly the same and the month when the data was collected had little precipitation. For this reason, the friction coefficient used is constant and we get an equation as the one presented in 3.5.

It was necessary to adapt the model to the reality of Aveiro. In Aveiro, even inside the city, there can be several maximum bus speeds. It is necessary to evaluate if the maximum speed that the bus can achieve is respected. The created model evaluates if the driving is safe and has some conditions associated based on the maximum bus speed beyond those described in the paper.

Working with GPS coordinates can be difficult; for that reason, it was necessary to associate the GPS position to a specific road segment. To get the road segments for Aveiro we used the road's information obtained by using an OSM library. This information had to be preprocessed to get the maximum bus speed, as it was described in section 6.1.3.

Besides that, we calculated the traveled distance and the acceleration of the bus. The traveled distance is the geographical distance (in meters) between two geographical points using Vincenty's formula. The acceleration was calculated as described in equation 6.1, being Δv the difference between speeds, and Δt the difference between time. Note that, if the bus was stopped and there was no data for more than one second, the acceleration would be

considered NaN. After this, the data was ready to be studied.

$$a = \frac{\Delta v}{\Delta t} \tag{6.1}$$

We added an intermediate state between safe condition and unsafe condition based on the fact that in Portugal if the drivers speed is bigger than the maximum speed, but the difference is less then 10km/h, it is not considered an infraction [40].

Note that, the evaluation between safe driving behavior versus unsafe driving behavior is subjective and depends on the criteria chosen for the evaluation. Anyway, since the concepts of safe and unsafe driving behavior were very limited, we created six categories to evaluate driving behavior.

Table 6.3 contains different categories regarding driving behavior. For each one of the categories, we assigned a color in the representation of the driving behavior classification associated. The colors chosen for encoding this information on the map are based on the colors used in traffic signs. The green represents that everything is ok; the yellow represents that the driver should be careful; and the red indicates danger. The darker green, the orange, and the burgundy have the same connotations associated but represent an additional danger because it is being performed significant variations in speed.

Table 6.3: Classification of the driving behavior.

Speed	Safety Domain	
	Within	Outside
speed $\leqslant$ max. speed		
(speed $>$ max. speed) and (speed $\leqslant$ max. speed + tolerance const.)		
speed $>$ max. speed + tolerance const.		

The first line corresponds to safe speeds, this means that the speed of the driver is smaller or equal than the maximum bus speed. The second line corresponds to the speeds that are in the threshold area between safe and unsafe. Those are the speeds that are bigger than the maximum bus speed, but are smaller or equal to the maximum bus speed plus a tolerance constant. The tolerance is 10Km/h, by default. All the other speeds belong to the unsafe driving behavior. The first column corresponds to drivers that do not make hard braking or hard accelerations, and the second column corresponds to those that make hard braking or hard accelerations.

Classifying driving behavior can be a difficult task. Speed and acceleration can be useful. The GPS position allows us to perform a more complete classification, because we can use the GPS position to associate the bus with road segments and we can obtain the maximum speed for those segments. This section details how the classification of driving behavior have been performed and presents a web application that is created to allow a more dynamic study.

Figure 6.4 contains the driving behavior for the first day of study for one of the buses. The blue lines are the limits that separate a safe driving behavior from a non-safe driving behavior. If the driver performs a safe driving, then all points or the vast majority of these belong inside the limits. In the figure, we observe that very few points are outside the limits.

When the acceleration is larger than 0, it means that the bus driver is accelerating. If the acceleration is positive and is outside the bounds, then the driver performed a sudden acceleration. When the acceleration is smaller than 0, it means that the bus driver is decelerating

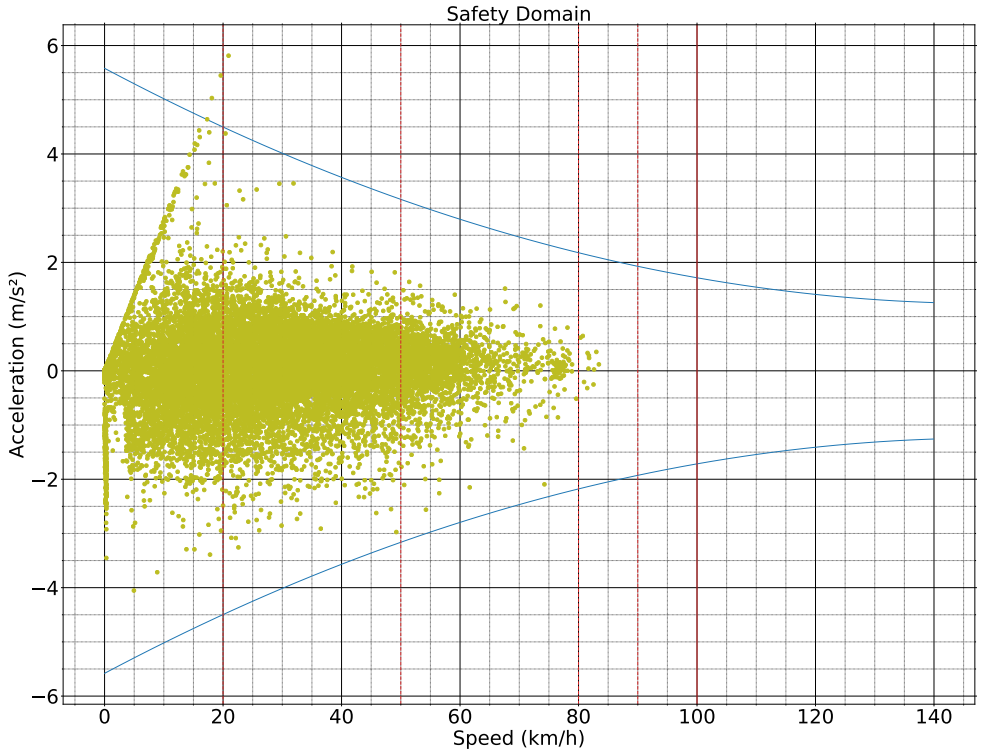


Figure 6.4: Driving behavior - Relationship between speed and acceleration.

and the bus driver can even be performing a braking. A negative acceleration outside the limits means that the driver performed a sudden braking.

If we look carefully at the data, it is possible to observe that the majority of the points are centered in the graphic, but there are two distinct lines formed by the points. One of those lines has a positive trend, and the other is very close to the straight segment where the speed is 0. The line with the positive trend corresponds to the initial start when the driver turns on the bus. The other line corresponds when the driver is stopping the bus.

The red lines symbolize the several limits of speed that the bus driver can find in the route. The first three are red dashed lines because they can only be applied if the bus is on a certain road. The last red line is the maximum limit of the maximum speed that the bus can achieve. There can be other limits for specific roads.

Considering that the bus has a speed of 60km/h, but it is on a road with a limit of 50km/h. The bus driver has an excessive speed and, despite having an acceleration that is inside the limits, the driver is performing an non-safe driving. To perform a more accurate classification, it is necessary to know the maximum speed limit where the bus is.

In order to achieve a more accurate classification, it was initially planned to create a code color for the points, but we could have an overlap of points and that could lead to wrong interpretations. With this in mind, it was planned the creation of a web application as a tool to study driving behavior.

6.3 Traffic behavior web dashboard

We should take into consideration that, even if a driver does a movement that is not considered safe, that does not mean that his/ her behavior is unsafe. We must always look for what happened through time. For that reason, it was created a web application for a better understanding of the driving behavior.

The goal of the web application was not to study the behavior of a specific driver, but the behavior of all the drivers. Thus, it was possible to identify what are the situations that most contribute to more dangerous behavior. Some of the main goals in the study of driving behavior were the ability to:

- Visualization of the driving behavior of bus drivers,
- Compare different temporal snapshots,
- Compare different days of the week, different weeks, etc,
- Compare time periods (for example: the morning period, afternoon period, evening, etc),
- Focus the study in one street or region.

To achieve the desired goal, one of the possibilities was the creation of a tool to simplify user interaction and allow a more dynamic study. It was planned for the creation of a web application with the capabilities previously mentioned. Table 6.4 contains the period in which the analysis can be done.

Initially, it was planned some use cases for this application, that are discussed in section 4.2.2. Briefly, the application should allow the comparison of temporal snapshots to highlight problems associated with roads, epochs of the day, etc.

Table 6.4: Calendar for studying driving behavior.

March 2020							
Week	Mon.	Tue.	Wed.	Thur.	Fri.	Sat.	Sun.
1	2	3	4	5	6	7	8
2	9	10	11	12	13	14	15
3	16	17	18	19	20	21	22
4	23	24	25	26	27	28	29

As the goal was to perform comparisons, the web application is divided into two sides, each one of the sides contains a map. Figure 6.5 presents one of the sides of the web application. Note that both maps are synchronized, and performing zoom in one, will affect the other. The same happens when we change the visualization area of the map.

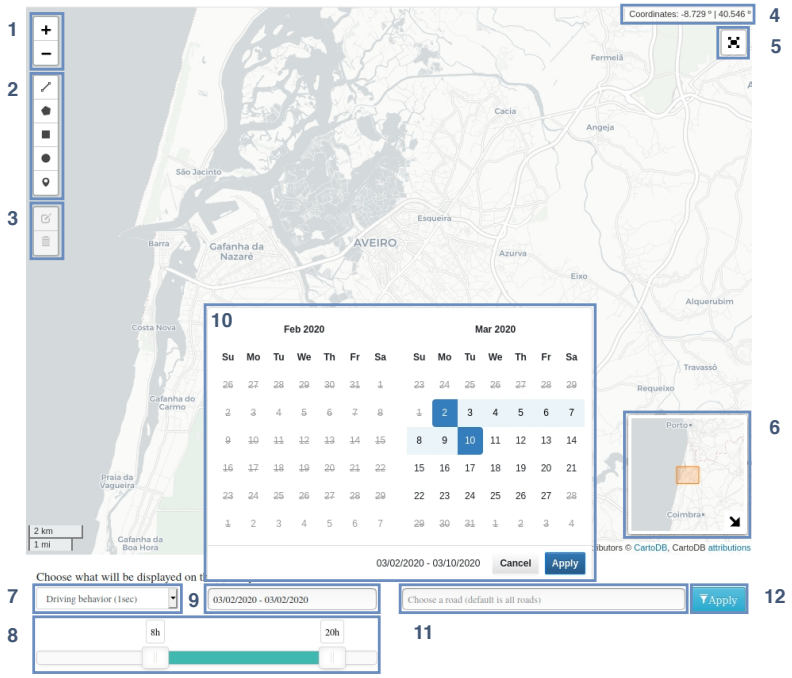


Figure 6.5: Dashboard elements **(1)** Zoom in and zoom out **(2)** Add line segments, polygons, and markers **(3)** Edit and delete line segments, polygons, and markers **(4)** Mouse GPS position **(5)** Fullscreen **(6)** Minimap (it can be minimized) **(7)** Select information **(8)** Choose an hour interval **(9)** Pop-ups a calendar **(10)** Choose a day or an interval of days **(11)** Choose a road (with autocomplete functionality) **(12)** Apply the changes.

In the type of information, the user can choose to analyze the driving behavior, the maximum bus speed, or the number of buses, the average speed, and the average acceleration. For the driving behavior there are two options, the user can choose the periodicity of the dataset of 1 second or 1 minute. These two options are given because, while the dataset from Aveiro presents information from 1 in 1 second, the dataset from Porto (Veniam dataset) presents information with 1 minute interval. Thus, we can compare the impact of increasing

dataset periodicity. With that in mind, it was created the same functionality for all the other options except the maximum bus speed. It was also necessary to recalculate the metrics as traveled distance, acceleration, and if the behavior is safe or unsafe.

Figure 6.6 contains the comparison done for driving behavior with 1 second period between the same days interval in different hours intervals. It is possible to observe some differences and some patterns in the images. For example, there is one road segment that is red in both cases. Besides that, it is perceptible that the buses performed different paths in the morning shift versus in the afternoon shift, because there are some differences between the colored lines in both maps.

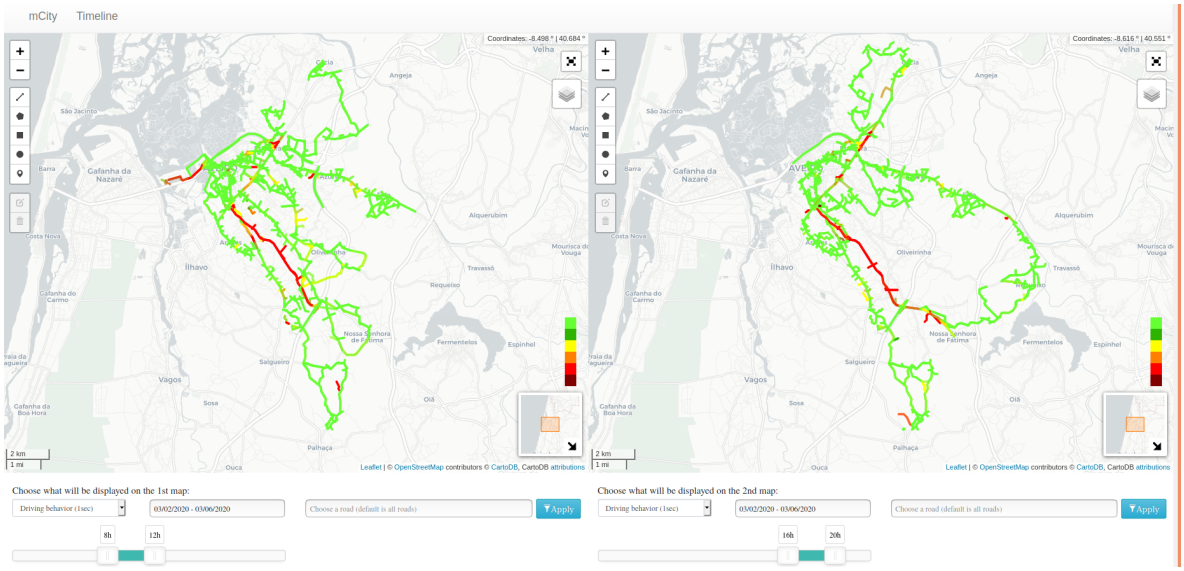


Figure 6.6: Comparison of different periods of driving behavior.

With the mouse over the box under the fullscreen button, a menu is opened to select the information that is being displayed. If it is being studied driving behavior, then it is opened the menu present in figure 6.7. This menu is built based on table 6.3. For example, if we want to see the difference in terms of maximum speed being exceeded by more than 10km/h, we just select the third and fourth elements. By doing that, we will get as a result the information on figure 6.8.

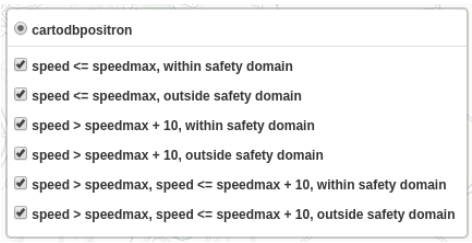


Figure 6.7: Driving quality behavior menu.

Through observation, it is possible to detect some roads in both maps in which the speed

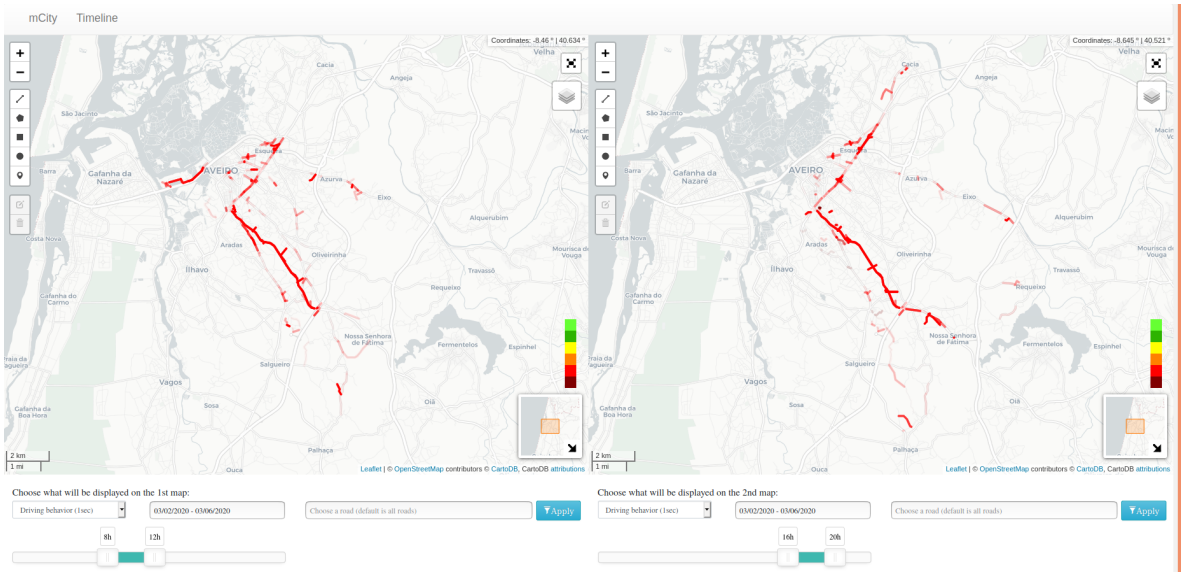


Figure 6.8: Side-by-side comparison of different periods of driving behavior, maximum speed being exceeded by more than 10km/h.

---

is exceeded by more than 10km/h. If we want a more detailed analysis, we can focus the map in one specific area, as can be observed in figure 6.9. Some of the lines are more transparent than others, as it is possible to observe. This is due to the percentage of unsafe driving behavior being different. For each segment, it was calculated the percentage of the several driving classification labels. A more opacity line means that the drivers performed more unsafe driving behavior.

It is also possible to compare, for example, the driving behavior with the maximum bus speed, as can be observed in image 6.10. With the mouse over the box under the fullscreen button, on the second map, it is possible to select the information by roads that have a maximum bus speed of 20km/h, 30km/h, etc.

Impact of data frequency in driving behaviour analysis

The major difference between the datasets from the buses from Aveiro versus the buses from Porto, besides the network size, was the periodicity. One of the big questions was about the impact of having information every second (Aveiro) versus every minute (Porto). Figure 6.11 presents the driving behavior comparison between 1 second and 1 minute. Figure 6.12 presents, for the same interval, the bus count information that contributes to the formation of figure 6.11.

As can be observed, there is a loss of information that results in some differences in figure 6.11. The difference between the information that contributes to the creation of the maps is big. This happens because, instead of 60 points, we will have just 1. Figure 6.13 presents a more simplified explanation with just 5 points instead of 60. In the figure on the left, $t0$ represents when we start to measure the values and $t5$ happens after we measure the other 4 values. If we ignore those, we will only have $t0$ and $t5$, or, as it is represented in the figure on the right $t'0$ and $t'1$. This will alter the traveled distance, acceleration, and driving behavior

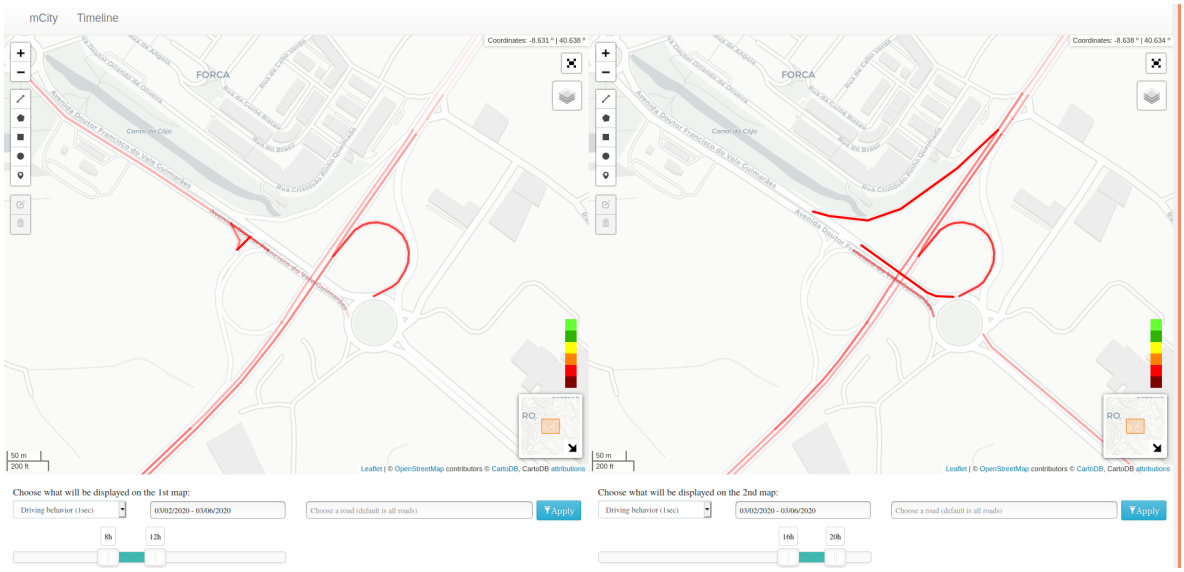


Figure 6.9: Side-by-side comparison of different periods of driving behavior, maximum speed being exceeded by more than 10km/h, zoom in with a focus on a specific area.

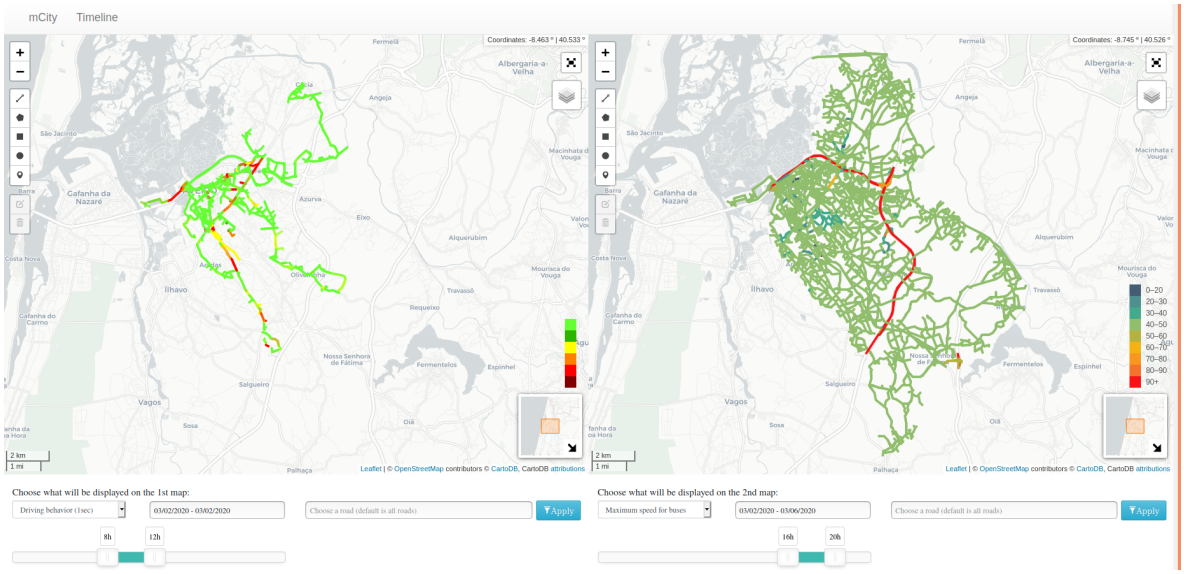


Figure 6.10: Side-by-side comparison of driving behavior with the maximum bus speed.

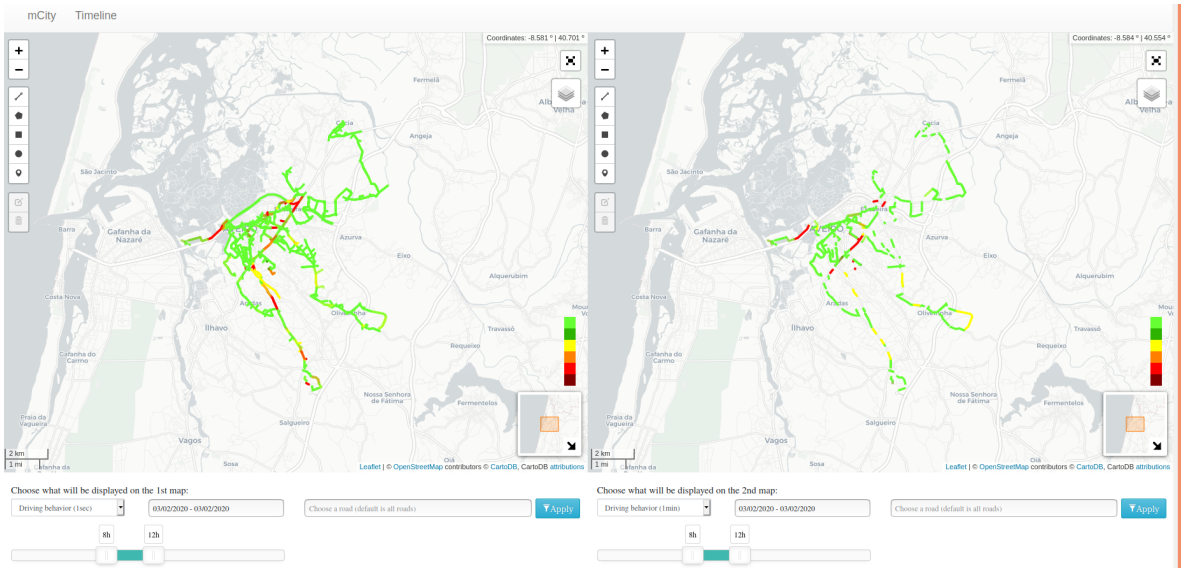


Figure 6.11: Side-by-side comparison of driving behavior using information with a period of 1 second versus 1 minute.

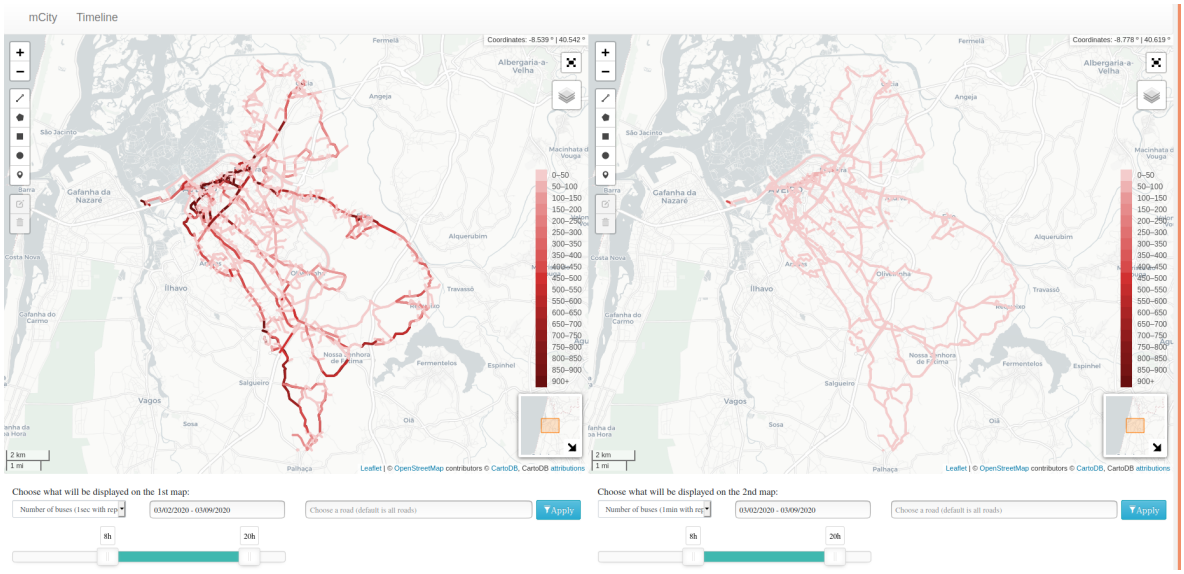


Figure 6.12: Side-by-side comparison of the number of buses using information with a period of 1 second versus 1 minute.

classification.

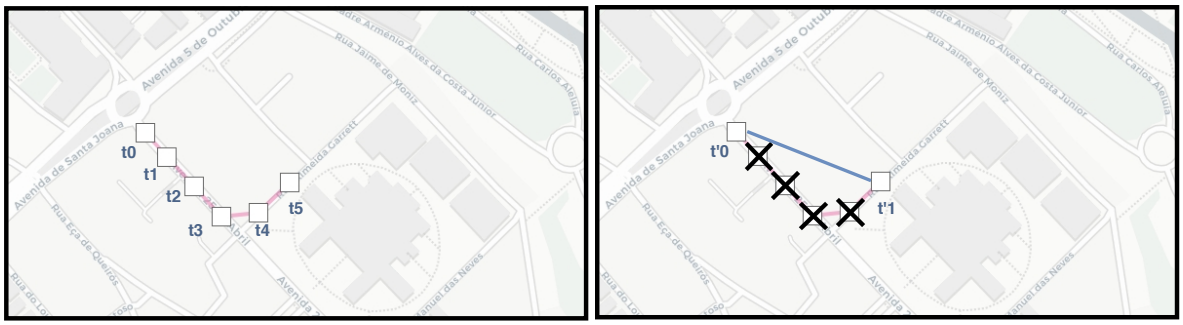


Figure 6.13: The effects of periodicity in data.

The impact of increasing periodicity can be one of the reasons why it is difficult to establish a relationship between traffic flow intensity and bus speed.

Figure 6.14 presents the comparison between the average speed profile using the information from 1 in 1 second versus using the information from 1 in 1 minute. Since the map with information from 1 in 1 minute contains less information, as expected, this map presents a larger variation of the values.

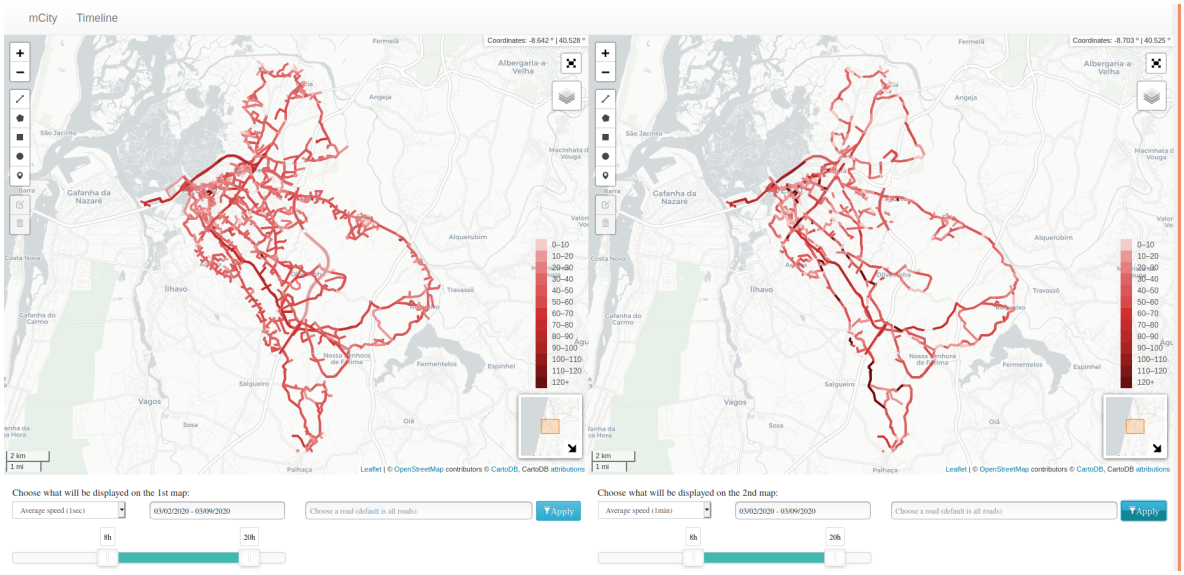


Figure 6.14: Speed profile.

The same was observed for the average acceleration profile, presented in image 6.15.

Visualizing the city pulse with time-lapse approach

To visualize the evolution of the different metrics through a period, it was created the possibility for the user to visualize a timelapse of the metric timeline chosen. Figure 6.16 contains the interface presented, and figure 6.17 contains some of the transitions observed.

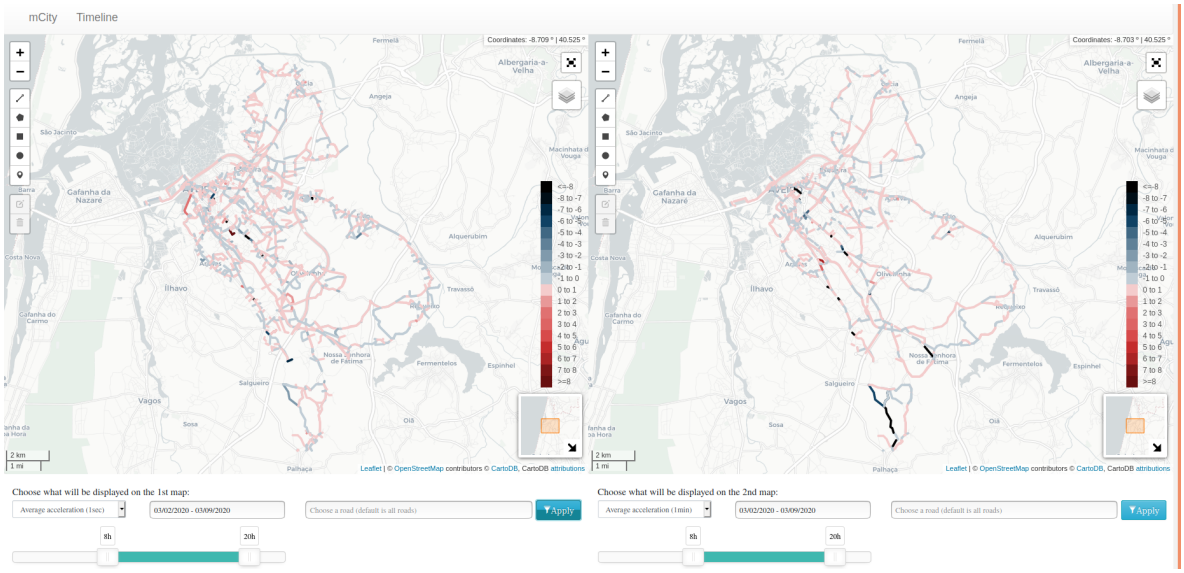


Figure 6.15: Acceleration profile.

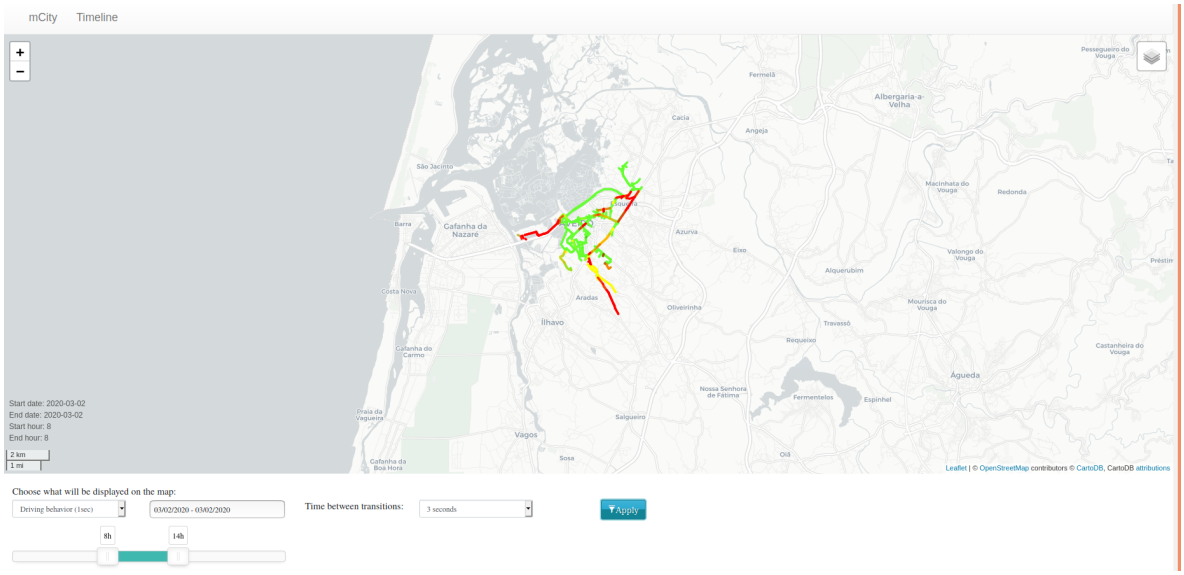


Figure 6.16: Timelapse interface.

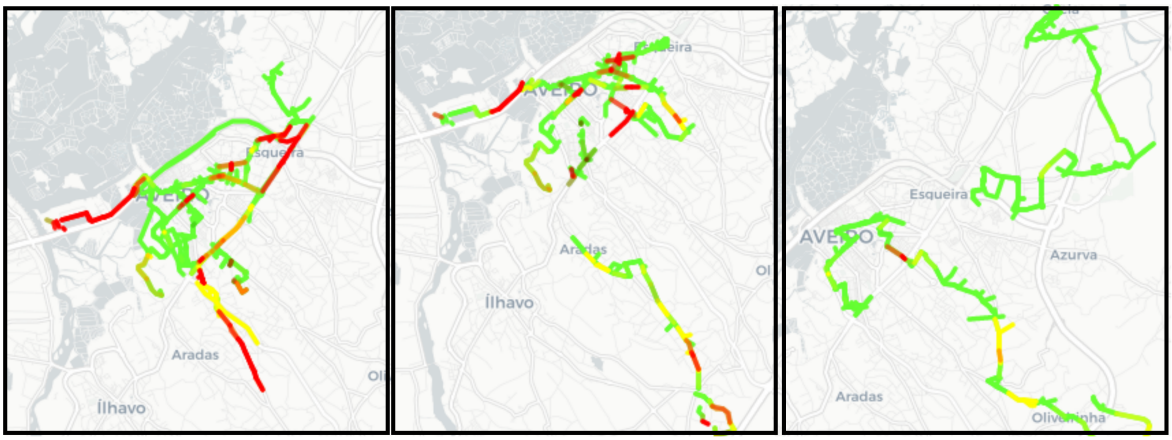


Figure 6.17: Timelapse transitions.

6.4 System implementation

The data used to study driving behavior was the data obtained by the deployed infrastructure with communication and sensing information in the vehicles, and the data obtained through OSM. The system architecture is described in figure 6.18.

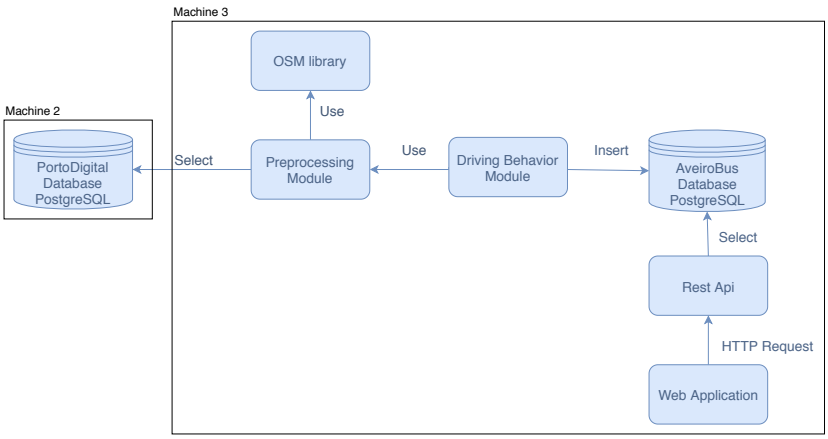


Figure 6.18: System architecture of the driving behavior module

The preprocessing module is responsible to process the collected data from the buses, and associating the buses GPS positions to road segments by using a library developed to get information from OSM. The information from buses was added to the Fiware infrastructure mention in the previous section. The Fiware infrastructure is responsible to persist the data in the database presented in the figure.

The driving behavior module will calculate the metrics as traveled distance, acceleration, and if the driving behavior is safe or not. Then, it will save the information in a database. Note that any process related to the association of GPS coordinates with road segments is computationally expensive.

The Rest API allows us to perform requests and get information from the database to the web application. The web application is the tool that was created to analyze the driving behavior.

The modules were developed in Python 3, and the web application was built using HTML5, CSS, JavaScript, Bootstrap 4, and the Leaflet JavaScript library.

6.5 Summary

As was discussed in section 6.1, the data from buses from Aveiro have some problems and needed to be processed. Besides that, the information from the infrastructure had the maximum speed for cars. Since we were working with buses, some corrections needed to be made. Because of that, it may exist some mistakes associated.

For the driving profile, we created 6 distinct categories, based on the relationship between speed and acceleration, and in the relationship between speed and the maximum speed allowed for buses.

To allow a dynamic study of driving behavior, we created a web application. The created tool enables a more complete study of driving behavior. Thus, even a person without knowing how to program can analyze the driving behavior in a city. The applications enable the user to perform several actions. The user can compare information, and he/she can visualize a time-lapse from metrics related to driving behavior. The user can even study the impact of different frequencies in the data.

Chapter 7

Results

Chapters 5 and 6 contain a section with a detailed explanation of the obtained results, and mention the multiple steps that contribute to the final result. This chapter is focused on the final result of the several stages of this work.

7.1 Results from the traffic flow analysis and forecasting

7.1.1 Mobility dataset aspects

Preprocessing the data from buses and sensors has a key role in this work. The data presents several problems like null values, abnormal values, missing data, and GPS imprecision. The data faults could be solved by applying different techniques as data cleaning, smoothing, associating GPS positions to road segments, etc.

The data from infrastructure prove to be a good allied to make a more complete analysis. Without the infrastructure data, we could not make GPS positions associations to road segments. Both infrastructure datasets had to be processed. The GTFS dataset from Porto had to suffer some alterations in order to get smaller segments. As a result, it was created a database with the smaller road segments. The OSM dataset from Aveiro contains information about road segments and it's maximum allowed speed. That information was altered in order to get the maximum allowed bus speed.

7.1.2 Traffic flow informed by deployed traffic counters

There are several traffic counters in Porto. Those traffic counters allow to have a perception of the evolution of traffic over time in the city. Because of that, they should exhibit traffic patterns.

Time-series decomposition performed for 4 weeks of data reveals some of the existing patterns. Figure 7.1 contains the additive decomposition. Note that, the weekends are highlighted by a red dashed box, and there are some peaks highlighted with a green circle in the seasonal component.

By analyzing the seasonal component, we can observe a clear distinction between the weekdays versus the weekend. The curve is similar between each one of the weekdays, and the curve is similar between each one of the days of the weekend. The traffic is higher on the weekdays. The first three green circles belong to the first Monday. Each one of the green circles is localized in one of the three peaks in the curve that corresponds to the hours

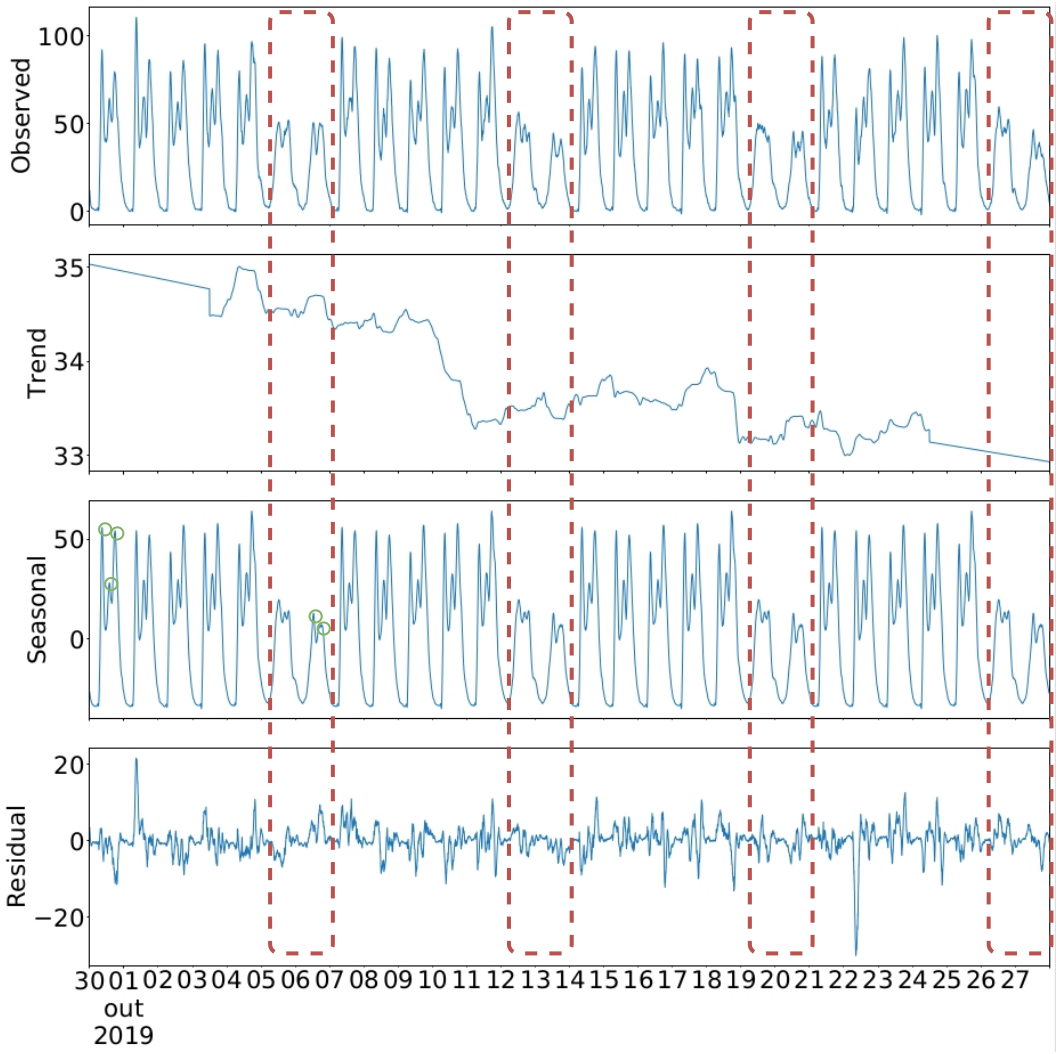


Figure 7.1: Time-series additive decomposition, frequency = 2016.

when is supposed to be more traffic. The major peeks happen at the more busy morning and afternoon period, and the other peek happens in the lunch hour. The differences in traffic at the weekend are less significant that at the weekdays.

In the observed part of the graphic, the curve is similar between different weeks. That's why, it is possible to observe the pattern in the seasonal component, and the remaining values at the residual component are very low.

Autocorrelation also exposed some of the patterns. that the days are similar, being more similar the weekdays with other weekdays. The same happens for the days of the weekend. It was possible to identify similar traffic flow sensors by using cross-correlation.

Some of the traffic counters present anomalous values due the existence of malfunctions. We should always try to understand if the observed values make sense. The observed values present in figure 7.1 seem right since we observe expected patterns. This type of study should have in consideration the location of the sensors because if we have a sensor in a place where it is rare to pass vehicles, we should not expect to observe the same type of patterns, there ca even not exist any patterns.

7.1.3 Forecasting the traffic flow

Predict traffic flow can help city managers identifying and solving traffic related problems, like frequent traffic jams locals, etc. We have available information about traffic counters and buses speed. Since buses can not represent urban mobility, we should verify if we can use or not the buses speed.

We were not able to find any relationship between speed and traffic flow intensity. Despite not establishing a relationship, the obtained results agree with what we can observe in a city. When there is much traffic, the speed of the vehicles will be low. One good example of this is traffic jams, and rush hour. When the speed of the vehicles is very high, there can't be much traffic, so the traffic flow observed is low. In this case, we can have an example of non-rush hour.

In most of cases, we can not have a prediction of a value by using the other, since we could not find any relationship between bus speed and intensity. To forecast traffic flow it was used just traffic data. Models like SARIMA can not use any additional type of information, but deep learning models could have benefited from the speed data.

The best model for forecasting traffic flow is LSTM neural networks. The MSE obtained for one-week prediction is low (40.983 ± 3.151) and the variance is very close to 1, meaning that the model could learn the data. The training time is low (12 minutes and 29.341 seconds) given that we are predicting an entire week.

The other models also obtained good values, but the SARIMA only can make 1-hour forecasting and the FFNN model has a bigger tendency to make overfitting of the data.

The deep learning models were designed to make long term forecasting. For that reason, they could not make good predictions of anomalous days. Even if we added the last hour, it was not enough. Since the SARIMA model is retrained each hour, and SARIMA sees the data has a sequence, the SARIMA model could adapt.

7.1.4 Results from the driving behavior analysis

We choose to use data from Aveiro to analyze driving behavior because the dataset from Aveiro was sampled with a bigger frequency. One of the big advantages of using data from

Aveiro is that in Aveiro there aren't any bus tracks. Even though buses do not represent general traffic because they have pre-establish routes, trips, and stops, in this case, is eliminated one of the problems that existed in the dataset from Porto. Thus, we have a more approximated representation of the general traffic.

We aim to characterize driving behavior as safe or non-safe and find patterns in time and space that leads to safe or non-safe driving. Data from multiple trips and buses are analyzed together, and the system does not offer specific support to filter for individual drivers/buses.

The method created to analyze driving behavior has two components. The first component relates speed and intensity. The second component relates speed and maximum bus speed. It was observed that in most of the cases, the drivers do not perform sudden acceleration or braking. The major problem is in the speed.

In the graphics obtained that relate speed and acceleration, it was visible a distinction between start driving, driving, and the immobilization of the vehicle. For the first one and last one, there was observed a distinct line for each one. The other values were more distributed between the limit lines for safe driving.

7.2 Software prototypes

7.2.1 Web dashboard for traffic behavior visualization

The user application created allow a more broad analysis of the driving behavior. There are two major functionalities. The first one allows the user to compare driving related metrics. The second one allows the user to visualize a time-lapses of one of the metrics.

The web application is useful for identifying problematic road segments, identifying problematic times of the day, compare zones, etc.

For the same day and time period, figure 7.2 contains a side-by-side comparison of 2 different metrics. On the left side, we can visualize the driving profile when the speed is bigger than the maximum bus speed plus the tolerance constant (10km/h). We used one of the other available options (maximum speed for buses) to find the maximum bus speed for the road that is colored in red and we verify that was 50km/h. On the right side is presented the average speed, after it was filtered for average speeds bigger than 60 km/h.

From the left side, we can conclude that a big portion of the buses that went through the road at that time presented a non-safe driving behavior. From the right side, we can see that the average speed presented values between 60km/h and 80km/h. This analysis allowed the identification of this road as potentially dangerous and can potentiate a more complete study. For instance, we should verify if this happens more times, if it is associated with certain times of the day, or specific days, etc.

7.2.2 Container-based processing pipelines

For each one of the main use cases, it was created configurable pipelines (a pipeline for predicting future values and a pipeline for classifying driving behavior) that are able to perform the desired tasks autonomously by running scripts. Some of the configurations are the period in the study and the sensor id.

There are several auxiliaries pipelines that support the main pipelines, like the ones for preprocessing data, performing statistical analysis, performing graphical analysis, performing

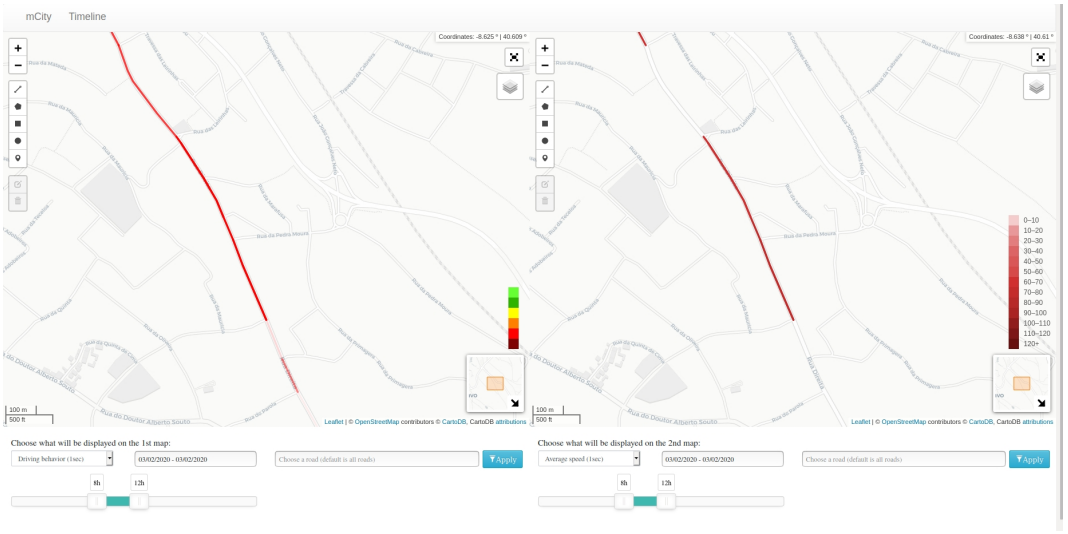


Figure 7.2: Side-by-side comparation of non-safe driving behavior with average maximum speed bigger than 60km/h.

---

smoothing, correlate sensors, decompose time-series, etc. Some of the tasks are optional and can be excluded from the main task.

For predicting future values there are several tasks that have to be performed. Choosing the best parameters for predicting traffic flow assumes the training and testing of models with different parameters and the comparison of the obtained predictions with a metric error to choose the best model. After that, we can train models and made predictions.

The analysis presented for the forecasting of the traffic flow was done using just one sensor. We showed that we could use the same metrics, the best metrics that were calculated for the sensor, to predict the values of another sensor that presented a strong correlation with the sensor in the study. If the sensor does not present a strong correlation, we can calculate the best parameters for that sensor.

The pipeline created for classifying driving behavior receives the information from the buses and calculates for every bus the necessary metrics (classification of the driving behavior, acceleration, traveled distance, and road segment).

7.3 Relationship between the results for the two cities

The two case studies were developed in isolation since we have different data for Porto and Aveiro. We try to take advantage of the best features of each one of the datasets.

The component that proves to have a major impact on data is the periodicity of the data collected by the sensors. There are some major differences in having data collected from 1 in 1 second versus data collected from 1 in 1 minute. Despite not having the data from Veniam from 1 in 1 second, we can perform the driving behavior, but we have to take into consideration that the results are less reliable.

The methods developed for Aveiro can be used in Porto, but the analysis will have to take into consideration the limitations of not having the same frequency. The methods used for the datasets from Porto could be also used in Aveiro. Study the relationship between speed

and intensity could achieve better results since we would have a more reliable bus dataset. We would only need to be installed the traffic counters sensors in the city.

The developed work will be integrated into the S2MovingCity smart city services platform by providing several services through an API and the merge of the web application with existing ones.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

Improving smart urban mobility can have several positive impacts on the city and its citizens. For example, the identification of problematic traffic zones can lead the responsible authorities to implement measures to combat the problems.

Since we had different data from Aveiro and Porto, it was possible to focus on different analyses for each one of the datasets. The data from sensors had to pass through a process of data cleaning, reduction, and transformation. The data from the infrastructure also needed preprocessing to match our requirements.

Time-series analysis was a good starting point because it helped in the identification of patterns and problems. Smoothing data solved some of the problems associated with the data like outliers, missing data, noise, etc.

To forecast the traffic flow observed using SARIMA, we have chosen the model based on the performance for the first forecasted hour. Since we could just forecast one hour (due to limitations of the model), a model was developed for each weekday. The metric that was selected to choose the model was BIC because this metric does not perform overfitting of the data, as it was observed. Thus, it was possible to forecast an entire day, by retraining the model at every hour.

To use deep learning methods, we performed feature selection of time lags. The results obtained previously by autocorrelation functions helped in the choice of feature lags to compare. All the feature lags chosen had a strong correlation.

It was possible to forecast one week of the traffic flow by using deep learning methods; however, we found some limitations. While SARIMA models could adapt and make good predictions even in the presence of anomalies, the machine learning methods were not able to do the same. One of the reasons is that they were designed to make long term forecasting.

The methods used to forecast and predict the traffic flow can be used by other sensors, besides the one in which the study was based if the sensors have a strong cross-correlation. The parameters used are the same, and it is just necessary to train the models. These were tested to apply the models (Seasonal AutoRegressive Integrated Moving Average, Long Short-Term Memory and FeedForward Neural Network) with the same parameters to different sensors. The results obtained were very close to the real values in both cases, proving the advantages of performing cross-correlation.

Using the data, we created a method to classify driving behavior based on an existing

method that used the relationship between speed and acceleration. Because that relationship is not enough, the method proposed has into consideration that, on different road segments, the speed limits can be different. We verified that most of the time the bus drivers do not perform hard accelerations or brakings. The part of the proposed method that has a major impact is the speed limits. This might happen because buses are obligated to stick to schedules. For a more dynamic study of the driving behavior, we developed a web application that can help in the identification of traffic problems. The web application allows the identification of problematic roads and times of the day. Besides that, it also allows the visualization of the driving behavior evolution (and other metrics) and comparison in time and space.

On a final note, we conclude that it is difficult to study urban mobility if there is only information about buses. In Porto, buses travel through bus lanes, and even without the lanes, buses have pre-established routes, trips, and stops, which makes it more difficult to analyse mobility. However, the mobility can be inferred with this information, and then be improved with the information of different types of vehicles.

8.2 Future Work

The study of smart urban mobility is far from being completely explored. Regarding the developed work, there are some elements that can be improved or developed. Noteworthy:

- One of the ways to improve results and understand if the traffic speed is related to the traffic intensity would be using sensors to measure the average speed in a given interval or the instantaneous speed of the vehicles; however, they have to be placed in strategic places due to the existence of bus lanes in Porto;

- Predict traffic flow together with speed, creating models that predict speed using traffic flow and models that predict traffic flow using the speed. In this way, we could predict the traffic with a full picture of the main aspects: speed and intensity;

- Transfer methods between cities: at least in terms of the bus data forecasting, this can be performed both in Porto and in Aveiro;

- Understand the causes that lead to speeding, for instance, if it happens more frequently when the bus is delayed. It could be used part of the developed work, by other members of the group, to associate buses with routes and trips;

- Test new algorithms for anomaly detection (road accidents, holidays, constructions, etc.) to detect anomalies that happen during short or long-term periods. It could be done a hybrid method composed by Seasonal AutoRegressive Integrated Moving Average and Long Short-Term Memory;

- Adaptive methods (methods that can recalculate the parameters used if the error has a significant increase) to predict traffic flow even if something unexpected happens;

- Study the opportunity to create a recommendation system for the bus drivers, for example, it could be suggested for the bus drivers to decrease speed, or that they have to be careful due to the existence of something unexpected;

- Analyze driving behavior trends (in space and time), for example, analyze the evolution of driving behavior through the day in a zone using statistical or deep learning methods. This way, city managers could try to change some aspects (like traffic signals, speed limit) to improve driving behavior;
- Analysis of the traffic jams progression through the day in the affected areas, and the flow of the affected areas;
- Include other aspects in the driving behavior analysis like the disrespect for traffic signals, road marks, u-turns in prohibited roads, etc. However, this may be difficult due to the lack of information.

Though the computational methods are implemented and operational, there are a few opportunities to enhance the system deployment:

- Integrate the methods into production, considering, in particular, the context of the projects S2MovingCity and Aveiro Steam City:
  - Deploy into production the prediction module, and provide updated predictions to users;
  - Deploy into production the driving behavior module and allow real-time analysis and rich visualizations.
- Validate the tools with end-users (usability tests).

References

[1] "Aveiro STEAM City - IT/Ubiwhere sensors," www.google.com/maps/d/u/0/viewer?ll=40.639146280449886%2C-8.646452703674314&z=15&mid=1BoIXFVvDkbC1yIk2Ypvb028cxbt9sRuM, accessed: 2020-07-15.

[2] J. Pereira, L. Ricardo, M. Luís, C. R. Senna, and S. Sargento, "Assessing the reliability of fog computing for smart mobility applications in vanets," *Future Gener. Comput. Syst.*, vol. 94, pp. 317–332, 2019.

[3] L. Ricardo, S. Sargento, and I. Oliveira, "An information system for bus travelling and performance evaluation," 01 2018, pp. 395–402.

[4] A. F. Tavares, I. Oliveira, S. Brás, and S. Sargento, "Estimation of buses arrival time combining historic and live mobility data." 2019, p. 302–313.

[5] D. Liu, L. Tang, G. Shen, and X. Han, "Traffic speed prediction: An attention-based method," *Sensors*, vol. 19, p. 3836, 09 2019.

[6] B. Williams and L. Hoel, "Modeling and forecasting vehicular traffic flow as a Seasonal ARIMA process: Theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, pp. 664–672, 11 2003.

[7] Q. Cui, Y. Wang, K. Chen, W. Ni, I. Lin, X. Tao, and P. Zhang, "Big data analytics and network calculus enabling intelligent management of autonomous vehicles in a smart city," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2021–2034, April 2019.

[8] "General Transit Feed Specification," https://gtfs.org/, accessed: 2019-11-03.

[9] R. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 2nd ed. Australia: OTexts, 2018.

[10] Z. Bartlett, L. Han, T. T. Nguyen, and P. Johnson, "A novel online dynamic temporal context neural network framework for the prediction of road traffic flow," *IEEE Access*, pp. 1–1, 2019.

[11] Xing, Ban, Liu, and Shen, "Large-scale traffic congestion prediction based on the symmetric extreme learning machine cluster fast learning method," *Symmetry*, vol. 11, p. 730, 05 2019.

[12] "Geographic Information System: What is gis," https://www.esri.com/en-us/what-is-gis/overview, accessed: 2019-11-03.

[13] "ArcGIS," https://www.arcgis.com/index.html, accessed: 2019-11-03.

[14] "QGIS," https://docs.qgis.org, accessed: 2019-11-03.

[15] "Postgis," https://postgis.net/, accessed: 2019-11-03.

[16] "OpenStreetMap," https://www.openstreetmap.org, accessed: 2020-05-21.

[17] A. Pal and P. Prakash, *Practical Time Series Analysis: Master Time Series Data Processing, Visualization, and Modeling Using Python.* Packt Publishing, 2017, ISBN: 9781523116744.

[18] "SPSS Tutorials: Pearson Correlation," https://libguides.library.kent.edu/SPSS/PearsonCorr, accessed: 2019-12-21.

[19] P. Brockwell and R. Davis, *Introduction to Time Series and Forecasting*, ser. Springer Texts in Statistics. Springer International Publishing, 2016, ISBN: 9783319298542.

[20] J. Brownlee, *Introduction to Time Series Forecasting With Python: How to Prepare Data and Develop Models to Predict the Future.* Machine Learning Mastery, 2017. [Online]. Available: https://books.google.pt/books?id=-AiqDwAAQBAJ

[21] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures." *Analytical Chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964. [Online]. Available: https://doi.org/10.1021/ac60214a047

[22] R. M. Heiberger and B. Holland, *Statistical Analysis and Data Display: An Intermediate Course with Examples in R*, 2nd ed. Springer-Verlag, New York, 2015. [Online]. Available: http://www.springer.com/us/book/9781493921218

[23] S. Raschka and V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2, 3rd Edition.* Packt Publishing, 2019. [Online]. Available: https://books.google.pt/books?id=sKXIDwAAQBAJ

[24] F. Chollet, *Deep Learning with Python.* Manning Publishing, Nov. 2017.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* The MIT Press, 2016.

[26] D. Osinga, *Deep Learning Cookbook: Practical Recipes to Get Started Quickly.* O'Reilly Media, 2018. [Online]. Available: https://books.google.pt/books?id=TMFeDwAAQBAJ

[27] "MIT 6.S191 - Introduction to Deep Learning," http://introtodeeplearning.com/, accessed: 2019-12-03.

[28] "How to select the Right Evaluation Metric for Machine Learning Models: Parte 1 Regression Models," https://medium.com/@george.drakos62/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0/, accessed: 2020-04-16.

[29] J. Brownlee, *Probability for Machine Learning: Discover How To Harness Uncertainty With Python.* Machine Learning Mastery, 2019. [Online]. Available: https://books.google.pt/books?id=uU2xDwAAQBAJ

[30] "ITS Research Fact Sheets - Benefits of Intelligent Transportation Systems," https://www.its.dot.gov/factsheets/benefits_factsheet.htm, accessed: 2019-12-05.

[31] X. Liu, A. Heller, and P. S. Nielsen, "Citiesdata: A smart city data management framework," *Knowl. Inf. Syst.*, vol. 53, no. 3, p. 699–722, Dec. 2017. [Online]. Available: https://doi.org/10.1007/s10115-017-1051-3

[32] K. Vidovic, S. Mandžuka, and D. Brčić, "Estimation of urban mobility using public mobile network," 09 2017, pp. 21–24.

[33] A. Pagani, F. Bruschi, and V. Rana, "Knowledge discovery from car sharing data for traffic flows estimation," 05 2017, pp. 1–6.

[34] A. I. Dontu, L. Gaiginschi, and P. D. Barsanescu, "Reducing the urban pollution by integrating weigh-in-motion sensors into intelligent transportation systems. state of the art and future trends," *IOP Conference Series: Materials Science and Engineering*, vol. 591, p. 012087, aug 2019. [Online]. Available: https://doi.org/10.1088%2F1757-899x%2F591%2F1%2F012087

[35] Y. Semet, B. Berthelot, T. Glais, C. Isbéerei, and A. Varest, "Expert competitive traffic light optimization with evolutionary algorithms," in *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS,*, INSTICC. SciTePress, 2019, pp. 199–210.

[36] I. Okutani and Y. Stephanedes, "Dynamic prediction of traffic volume through kalman filtering theory," *Transportation Research Part B: Methodological*, vol. 18, pp. 1–11, 02 1984.

[37] W. Yuankai, H. Tan, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, 05 2018.

[38] G. Dai, C. Ma, and X. Xu, "Short-term traffic flow prediction method for urban road sections based on space-time analysis and gru," *IEEE Access*, vol. PP, pp. 1–1, 09 2019.

[39] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187 – 197, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X15000935

[40] C. P. d. Costa, António Alves; Silva, *Código da estrada*. Edições Alves Costa, 2018.

[41] T. Liu, Y. Yang, G. Huang, Y. K. Yeo, and Z. Lin, "Driver distraction detection using semi-supervised machine learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1108–1120, 2016.

[42] O. Derbel and R. J. Landry], "Driver behavior assessment based on the g-g diagram in the dve system," *IFAC-PapersOnLine*, vol. 49, no. 11, pp. 89 – 94, 2016, 8th IFAC Symposium on Advances in Automotive Control AAC 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896316313350

[43] A. Dempster, "A generalization of bayesian inference," *Journal of the Royal Statistical Society, Series B 30*, p. 205–247, 1968. [Online]. Available: https://doi.org/10.1111/j.2517-6161.1968.tb00722.x

[44] A.-O. Boudraa, L. Bentabet, and F. Salzenstein, "Dempster-shafer's basic probability assignment based on fuzzy membership functions," *ELCVIA : Electronic Letters on Computer Vision and Image Analysis; Vol.: 4 Núm.: 1*, vol. 4, 01 2004.

[45] A. Jeffrey, D. Zwillinger, I. Gradshteyn, and I. Ryzhik, "15 - norms," in *Table of Integrals, Series, and Products (Seventh Edition)*, 7th ed. Boston: Academic Press, 2007, pp. 1081 – 1091. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780080471112500224

[46] L. Eboli, G. Mazzulla, and G. Pungillo, "Combining speed and acceleration to define car users' safe or unsafe driving behaviour," *Transportation Research Part C: Emerging Technologies*, vol. 68, pp. 113–125, 07 2016.

[47] M. Karlaftis and E. Vlahogianni, "Statistical methods versus neural networks in transportation research: Differences, similarities and some insights," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 3, pp. 387 – 399, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X10001610

[48] P. M. Santos, J. G. P. Rodrigues, S. B. Cruz, T. Lourenço, P. M. d'Orey, Y. Luis, C. Rocha, S. Sousa, S. Crisóstomo, C. Queirós, S. Sargento, A. Aguiar, and J. Barros, "Portolivinglab: An iot-based sensing platform for smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 523–532, 2018.

[49] G. P. Pessoa, M. Luís, L. Guardalben, C. Senna, and S. Sargento, "Evaluation of content dissemination strategies in urban vehicular networks," *Information (Switzerland)*, vol. 11, no. 3, pp. 163–163, March 2020.

[50] "Aveiro - STEAM CITY," www.aveirotechcity.pt/en/Projects/AVEIRO-STEAM-CITY, accessed: 2020-07-15.

[51] R.-C. Chen, E. Yulianti, M. Sanderson, and W. B. Croft, "On the benefit of incorporating external features in a neural architecture for answer sentence selection," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1017–1020. [Online]. Available: https://doi.org/10.1145/3077136.3080705

[52] "Fiware developers," https://www.fiware.org/developers, accessed: 2019-10-17.