

EXPLORING THE CAPABILITIES OF DEEP LEARNING MODELS FOR TRANSPORT AND HUMAN ACTIVITY RECOGNITION

GERARD CARAVACA IBÁÑEZ

Thesis supervisor: MÓNICA AGUILAR IGARTUA

Tutor: JAVIER BÉJAR ALONSO (Department of Computer Science)

Degree: Master's Degree in Artificial Intelligence

Master's thesis

School of Engineering
Universitat Rovira i Virgili (URV)

Faculty of Mathematics
Universitat de Barcelona (UB)

Barcelona School of Informatics (FIB)
Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

23/01/2024

Abstract

This thesis explores the transformative potential of deep learning smartphone-based transportation mode detection systems in enhancing urban planning in the city of Barcelona. The core of this thesis is the development and comparison of algorithms, coupled with extensive data analysis and preprocessing techniques, aimed at reliable transport mode detection. We delve into creating a real-time system capable of predicting transport usage patterns. For that, a dataset of smartphone sensor data has been created with examples of journeys using multiple modes of transportation in the metropolitan area of Barcelona. In the deep learning model, we have experimented with architectures combining convolutional networks and LSTMs to finally create a hierarchical model that combines the use of CNNs for feature extraction, with the ability to process time series from the LSTM layers using skip connections. For robust and battery-efficient detection, we have combined this model with statistical techniques, which allow us to detect at an early stage whether the user is moving, standing or walking. This allows not to make excessive use of the deep learning model, which can be costly in mobile devices. Following, an android application is presented which implements the mentioned techniques and presents a simple way to collect mobility data, which can be useful for future studies on urban mobility in the city. Finally, the various ethical, social and environmental issues that such systems may have are studied, describing the privacy and interpretability factors that this tools must comply with.

Index terms *- Activity recognition, deep learning, mobile sensors, transportation mode detection, urban mobility, MobilitApp.*

Acknowledgements

I would like to express my sincere gratitude to all those who have supported and guided me throughout the journey of this thesis.

First and foremost, my heartfelt thanks to my supervisor, Professor Mónica Aguilar, for giving me the opportunity to work on this project and for guiding me during these months. Additionally, I am profoundly thankful to the members of the SISCOM group and all the study volunteers. Their willingness to share their time and data was indispensable. This research simply would not have been achievable without their invaluable contributions.

I would also like to thank Professor Javier Béjar for his valuable feedback. His expertise greatly influenced my research. My gratitude extends to the faculty and staff at the *Facultat de Informàtica de Barcelona (FIB)*. Their support and the opportunities provided have been vital in enhancing my research experience, both in the bachelor's and master's degrees. The environment here has been both challenging and inspiring.

On a personal note, I extend my deepest gratitude to my family for their relentless love and support. They have been a constant source of inspiration and strength throughout this journey. Additionally, I am immensely thankful to my friends for their incredible support and encouragement.

Finally, I acknowledge *Autoritat del Transport Metropolità (ATM)* for their support and feedback, which was essential for completing this project

Contents

List of Figures

List of Tables

Acronyms

API Application Programming Interface. 35

ATM Autoritat del Transport Metropolità. 2

Bi-LSTM Bidirectional Long Short-Term Memory. iii, vi, 18, 19

CNN Convolutional Neural Networks. iii, 4, 14, 15

DAbase Data Augmentation base. 47, 48, 52

DAlarge Data Augmentation large. 47, 48

DCBL Deep Convolutional Bidirectional LSTM. 34

EMEF Enquesta de Mobilitat en dia Feiner. 5

FFT Fast Fourier Transform. 32

FN False Negative. 23

FP False Positive. 23

GAN Generative Adversarial Networks. 81

GDPR General Data Protection Regulation. 78

GELU Gaussian Error Linear Unit. 61

GPS Global Positioning System. 12, 27, 33, 34

GRU Gated Recurrent Unit. 33

HTC High Tech Computer Corporation. 34

IoT Internet of Things. 1

IQR Interquartile Range. 71

KNN K-Nearest Neighbors. iii, vi, 10, 11, 31, 53

LAC Localitation Area Code. 35

LSTM Long Short-Term Memory. iii, vi, vii, 17, 18, 21, 33, 43

MCC Mobile Country Code. 35

MCIN Ministry of Science and Innovation. 2

MLP Multilayer Perceptron. vii, 53, 55

MNS Mobile Network Code. 35

NLP Natural Language Processing. vi, 19

OF Overlapping factor. 46

ReLU Rectified Linear Unit. 61

RF Random Forest. iii, 10, 11, 53

RNN Recurrent Neural Networks. iii, vi, 4, 15–18

RSSI Received signal strength indicator. 35

SHL Sussex-Huawei Locomotion. iv, vi, 34–36

SISCOM Smart Services for Information Systems and Communication Networks. 42

SNR Signal to noise ratio. 35

SSID Service Set IDentifier. 35

SVM Support Vector Machines. iii, vi, 10, 31, 53

TMD Tranport Mode Detection. iv, vi, viii, 32, 34, 37, 38

TP True Positive. 23

UPC Universitat Politècnica de Catalunya. 2

WS Window Size. 46

Introduction

In an era defined by unprecedented technological advancements and the proliferation of smart devices, the interaction between humans and their environments has become an integral part of daily life. The advent of the **Internet of Things (IoT)** has marked an era where an abundance of data is generated, capturing intricate details about our surroundings and behaviors. This surge in data availability has opened up new possibilities for enhancing various aspects of our lives, including transportation and human activity recognition.

Understanding how and why people move in specific ways is not only a crucial activity but a necessity for modern urban planning, the preservation of human health, and the advancement of environmental science. Traditionally, to gain insights into the transportation behaviors of populations, it has been standard practice to conduct periodic travel studies and develop action plans based on the information acquired from these studies. However, these large-scale studies come at a considerable monetary cost and require substantial effort, often limiting their frequency and scope.

The last few years have seen a transformational change with the widespread adoption of smartphones. These everywhere devices have unlocked the potential to revolutionize the way we **monitor transportation in near-real-time**, offering an opportunity for faster and more adaptive responses from the perspectives of urban planning, human health, and environmental sustainability. Smartphones, equipped with an array of sensors and advanced computing capabilities, can serve as powerful tools for continuously capturing data about individuals' movements and transportation choices.

This thesis will embark on a journey to unlock the full potential of **smartphone-based transport mode detection** systems, encompassing algorithms development and comparison, data collection and analysis, preprocessing techniques, privacy considerations, and the practical implementation of such systems. By addressing these challenges, we aspire to contribute to the transformation of how we understand and manage transportation, ultimately fostering more sustainable, healthier, and friendly urban environments, specifically in my city, Barcelona.

In order to carry out this work, the document has been divided into the following chapters. In the first chapter (chapter 1), the thesis sets the stage by outlining its main objectives and research methodology. It then delves into the foundational aspects, including the current state of mobility in Barcelona, the challenges in transport recognition, prevalent deep learning architectures, and various data collection methods (see chapter 2). The third chapter (chapter 3) is dedicated to discussing relevant literature. Following this, the fourth chapter (chapter 4) presents the dataset specifically gathered for this project, comprising mobility data from Barcelona and its vicinity, sourced from smartphone sensors. Chapter five involves a thorough exploration of diverse preprocessing techniques and model architectures aimed at surpassing baseline results (see chapter 5). Chapter six marks the development of a comprehensive real-time system for predicting transport usage (see chapter 6). The thesis concludes by reflecting on the ethical considerations and environmental implications associated with such kind of projects (see chapter 7).

Chapter 1

Preliminary

This section has been carefully constructed to establish a solid foundation for the comprehensive analysis, ensuring both clarity and guidance throughout the study. It aims to present an overview of the topics to be discussed in the subsequent chapters, thus preparing the reader for a detailed investigation into the realms of urban mobility and deep learning. To achieve this, the context of the project, its motivations, and the methodology employed are described below.

1.1 Context

The research conducted in this master's thesis is situated within the framework of a collaborative agreement established between the *Autoritat del Transport Metropolità (ATM)* [1] and the *Universitat Politècnica de Catalunya (UPC)*. This collaboration agreement, effective from February 7, 2021, to February 7, 2024, outlines the mutual commitment of ATM and UPC to collaborate on various research projects. Notably, this master's thesis is an integral part of Project A-01358, which focuses on predicting the transportation modes utilized by citizens through the analysis of smartphone sensor data while upholding the paramount concern of privacy protection.

Furthermore, this research has received valuable support from the Spanish Government through the research project "Anonymization technology for AI-based analytics of mobility data (MOBILYTICS)," with reference TED2021-129782B-I00 [3]. This project is financially backed by the Ministry of Science and Innovation (MCIN), a key entity in Spain's research and development landscape (see Figure 1.1).

The collaborative agreement with ATM and the support from the Spanish Government underscore the significance and relevance of this master's thesis in addressing contemporary challenges related to transportation mode prediction and privacy preservation. This context establishes the foundation for the research undertaken in this thesis and highlights its real-world applications and contributions to the field of transportation and data privacy.

FIGURE 1.1: National and international organizations that support the project.

1.2 Motivation and research objectives

Picture a busy urban street during rush hour, where pedestrians and different modes of transportation come together. In this situation, the ability to accurately identify pedestrians, cyclists, and vehicles is more than just convenient; it is essential for safety, efficiency, and urban planning. This scene represents the complex challenges that our rapidly growing cities face, emphasizing the increasing need for precise recognition of both transportation methods and human activities.

In this thesis, we delve into the exploration of deep learning models, their capabilities, and their potential to revolutionize how we perceive and interact with our surroundings in an urban context. The expected outcome of this research is a complete analysis of deep learning-based transport activity recognition systems that can accurately identify different transport modes based on sensor data. The findings of this study will contribute to the growing field of activity recognition and have implications for transportation planning, urban mobility management, and the development of context-aware applications and services. With this in mind, the aim of the work is not only to obtain an accurate model for activity recognition but also to explore and compare the various capabilities of different machine learning based approaches applied in this domain. Finally, we aim to be able to give a global vision to this task having into account not only the technical and theoretical part but also the privacy and interpretability problems that the system might have.

1.3 Methodology overview

In order to obtain results in accordance with the objectives explained in the previous section, the thesis will be divided into the following parts explained on this section.

First, the theoretical foundations, necessary to understand the context of the work, will be established. For this purpose, the technical concepts, that will be put into practice later on, will be clearly defined. Next, a review of the main contributions of the literature on the subject will be carried out.

Once the background has been defined, the next step is to study the availability of the data and the possibility of developing a customized dataset. A diverse dataset will be collected, consisting of sensor data recorded from individualsusing various transportation modes and other human activities as walking. The development of a smartphone application to this purpose will be presented in this step. This will involve the study of public datasets and the development of data preprocessing techniques such as: data augmentation, outliers detection, features extraction...

The next step is the experimentation step, in which both traditional machine learning models and deep learning models, such as convolutional neural networks (CNN), recurrent neural networks (RNN) and Transformers, will be compared and adapted to the task of transport activity recognition. Following this, we will present the implementation of the proposed system after the study carried out.

Finally, the report will conclude with the discussion on the impact that the implementation of this type of technology can have on today's society and the future work proposed.

Chapter 2

Fundamental knowledge

This chapter sets up the theoretical groundwork on which this thesis relies and offers readers the essential theoretical information to comprehend the following chapters. The first part of the chapter delves into the topic of Mobility in the Barcelona metropolitan area, providing a comprehensive overview of the region's transportation dynamics. Subsequently, it introduces the Transport Mode Recognition task, describing its significance in the context of mobility analysis. In the pursuit of effective recognition, the chapter explores various machine learning techniques, with a special focus on deep learning methods. Additionally, it covers the critical aspect of evaluation, which ensures the reliability and accuracy of the developed models. Furthermore, the chapter explain the diverse Data Acquisition Modalities utilized in collecting the necessary data for this task.

2.1 Urban mobility in Barcelona

During a typical workday in 2019, there were 5.7 million trips made in Barcelona, averaging 3.6 trips per person. A slight decline of 0.3% in total trips was observed in 2021 (last official data available) when excluding professionals like taxi drivers. This data was collected from the annual Mobility Survey on Workdays (EMEF), conducted collaboratively by various local and metropolitan transportation authorities [26].

Public transport accounts for approximately 17% of all journeys made in Barcelona. Various modes of public transport serve every district of Barcelona, with buses, commuter rail, and light rail - encompassing both subway and tram - being the primary options [16]. Figure 2.1 illustrates the quarterly passenger count trends for each of these transport modes from 2016 to 2021.

As can be seen in the graph, since 2016 the use of the various modes of public transport in Barcelona has remained fairly stable with some fluctuations. However, as expected, during 2020 a decrease in all types of transport has been observed due to the Covid-19 season. During 2021 a rebound in data is observed without reaching pre-Covid levels. The rebounds in 2021 suggest a recovery phase and possibly a return of confidence in using public transportation. Although last year's data have not yet been published, it is expected that public transport use has returned to the levels of the pre-Covid years.

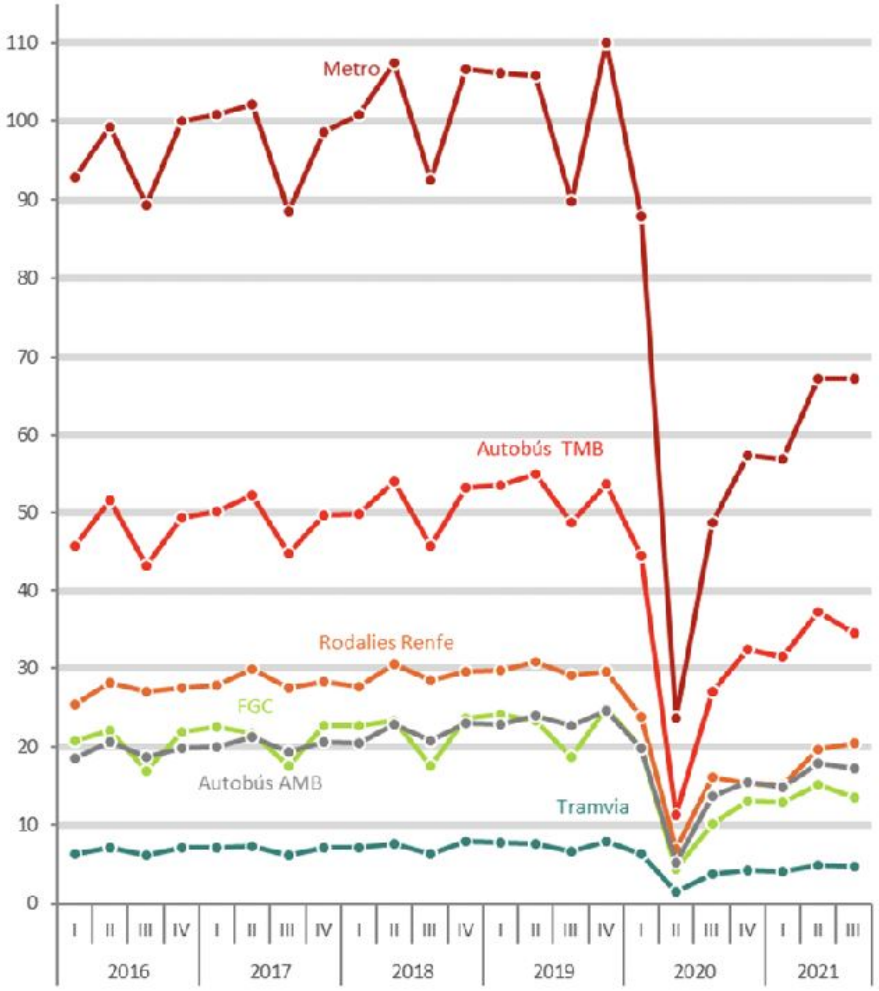


FIGURE 2.1: Number of users (millions) for each public transport per quarter in Barcelona (2016- 2021). [16]

On the opposite side, in terms of **private transport**, almost half the trips, specifically 39%, are made using personal cars, in Barcelona [16]. The recent pandemic has engendered a discernible shift in the populace's transportation preferences, primarily driven by health and safety concerns associated with public transit systems. The aversion to shared spaces, particularly in the context of public transportation, has prompted an augmented reliance on personal vehicles.

This emergent trend has intensified traffic congestion within the city. Empirical data suggests a consequential 29% augmentation in travel durations purely attributable to traffic congestion in Barcelona. In fact, Barcelona is now the city in Spain with the greatest traffic problems according to data from *TomTom traffic index* [58] (see Table 2.1). This substantial increment is unparalleled in the Spanish context; for instance, Madrid, another significant urban conglomerate, witnesses a comparatively lower 23% increase in travel time due to vehicular congestion [43]. This divergence in urban mobility patterns between the two cities underscores the imperative for tailored transportation strategies, particularly in the post-pandemic era.

TABLE 2.1: The 10 Spanish cities with the most traffic jams in 2022. [43]

City	Avg. 10 km time (min)	Hours in traffic per year	Avg. Speed (km/h)
Barcelona	18.3	161	29
Madrid	18	159	29
Valencia	16.3	141	33
Gijón	15.6	127	36
Sevilla	14.8	136	34
Vitoria	14.8	122	38
Málaga	14.6	126	36
Zaragoza	14.5	121	36
Granada	14.3	129	36
Palma de Mallorca	14.1	129	36

2.1.1 Future of Mobility in Barcelona

The Mobility Plan in Barcelona [17] showcases a forward-thinking approach towards urban sustainability and livability. By reducing private vehicle use and enhancing public transport efficiency, the plan addresses both environmental concerns and modern urban living standards. The expansion of pedestrian streets and bike lanes, along with the promotion of shared vehicle services, promises a more active, healthy lifestyle among residents. Furthermore, the "Superilla" project and Via Laietana's transformation contribute to creating communal and green spaces, which are vital for social interactions and mental well-being. However, residents might face a transitional phase, adapting to new mobility patterns and potentially altered traffic conditions. The ambitious goal of shifting the majority of commutes to walking, cycling, and public transport by 2024 means a substantial change in daily routines. Yet, if executed effectively, these alterations could lead to a more accessible and environmentally friendly urban landscape, enhancing the overall living experience in Barcelona.

It should be noted that this plan corresponds to Barcelona City Council's 2023 bid. It may be subject to change with the new candidature.

2.1.2 Our role in urban mobility of Barcelona

As can be seen from the data shown in the previous sections, most of the studies on mobility in Barcelona are based on data obtained from citizen surveys. This is why updated data for the last few years has not yet been obtained. This complicates the analysis of the impact of the agreed mobility plan to improve mobility in the city since data collection by conventional methods is slow and complicated, often requiring significant human resources and financial investment.

Relying solely on citizen surveys presents a series of limitations. Firstly, these surveys are subject to biases, as people might not accurately recall or honestly report their transportation habits. Secondly, the periodic nature of these surveys means that real-time or frequent data updates are virtually non-existent, making it challenging to monitor rapid changes in mobility patterns or to assess the immediate impacts of newly implemented policies.

Transitioning to a transport mode detection system using smartphone motion sensors offers a compelling alternative. Such a system ensures near real-time, objective data collection, eliminating biases inherent in personal recall or reporting. Additionally, this digital approach is cost-effective, scalable, and adaptable to rapidly changing scenarios. Leveraging this technology-driven method will greatly enhance Barcelona's ability to effectively analyze and manage urban mobility, ensuring more informed decision-making in its pursuit of improved transportation dynamics.

2.2 Transport mode recognition task

Transport mode recognition is a sub-field of the human activity recognition task that involves identifying and categorizing the mode of transportation used by an individual or an object. This process has various applications and, still today, suppose a number of challenges.

2.2.1 Applications of transport mode recognition task

In the realm of **urban planning and traffic management**, transportation mode recognition provides crucial data to optimize city infrastructure and improve traffic flow. By understanding how people move about the city, authorities can make informed decisions about public transit expansion, bike lane development, and road maintenance. At this point, the task of transport recognition plays a decisive role. This is because it allows the collection of data for the subsequent analysis of urban mobility on a large scale.

In the field of **healthcare**, this technology can contribute to tracking and encouraging physical activity, helping individuals lead healthier lives. Smart wearables and health apps can use transportation mode recognition to monitor users' activity levels and offer personalized fitness recommendations.

Moreover, transportation mode recognition has a profound impact on **environmental sustainability**. By promoting eco-friendly modes of transport like cycling or walking and discouraging the excessive use of personal cars, we can collectively reduce carbon emissions and alleviate the burden on our planet.

2.2.2 Challenges of transport mode recognition task

As summarized in 3.1, much research has been carried out in transport mode recognition. However, this field still faces many technical challenges. Some of the difficulties are shared by other pattern recognition domains, such as computer vision and natural language processing, while others are specific to sensor-based activity detection and need specialized algorithms for real-world applications. In the following, we present a set of categories outlining the challenges. A visual representation of this taxonomy can be found in Figure 2.2.

- **Data acquisition:** Training and evaluating deep learning models require large annotated data. In this context, it is particularly expensive and time-consuming to collect sensory activity data (see section 2.6).
- **Feature extraction:** This challenge is usually shared with other classification problems. For sensor-based transport recognition, it is even a more difficult task because

there is inter-activity similarity. This means that different transports may have similar characteristics (e.g., walking and running). Therefore, it is difficult to produce distinguishable features to represent activities uniquely.

- **Data distribution:** In this task, the dataset may be unbalanced for three reasons. The first one is class imbalance. This is an important challenge because it is difficult to find large amounts of data on less common transports, such as e-scooter. Apart from that, some transport patterns are user-dependent, which means that different users may have diverse activity styles. Finally, the position or different configuration of the sensors may influence the simulated data.
- **Computational cost:** This task is intended to be used on portable devices such as a smartphone. This type of device has limited computational resources. For this reason, lightweight and easily optimizable models must be generated for this type of device.
- **Concurrent transports:** Ideally, when performing classification tasks, it is taken into account that each sample belongs to only one possible class. However, in this case in today's public transport it is common for people to walk or take other types of transport with them. This should be treated differently depending on the final application of the system.
- **Privacy:** As the recognition system could potentially record users' lives continuously, there are risks of personal information disclosure, which make the privacy issue determinant to be analyzed before deploying the system.
- **Interpretability:** Sensory data cannot be read like images or sentences. Furthermore, due to the inherent flaws in sensors, sensory data invariably contains a lot of noise information. Therefore, trustworthy recognition solutions must be able to analyze sensory input and know which aspects of data help with recognition.

FIGURE 2.2: Categories of deep learning in sensor based human activity recognition challenges, also applicable to transport mode recognition. [14]

2.3 Traditional machine learning techniques

In the context of transport mode detection using smartphone motion sensors, while deep learning models offer advanced capabilities and intricate pattern recognition, traditional machine learning techniques remain essential to the foundation of predictive analytics. It is for this reason that this type of algorithm will be used as **baseline** for further experiments. This means that the results of these algorithms should be used to ensure that the accuracy of the complex models is not lower than that achieved with the simple models.

This section delves into three of the most pertinent traditional techniques for our context: RandomForest (RF), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM).

2.3.1 Support Vector Machines (SVM)

SVM [28] is a non-parametric supervised learning algorithm that operates by finding the hyperplane that best divides the data into classes (in classification tasks), ensuring that the margin between the classes is maximized. For data that is not linearly separable, SVM utilizes the **kernel trick**, mapping the data into higher dimensions where a separating hyperplane can be found. Figure 2.3 shows the decision function for a linearly separable problem, with three samples on the margin boundaries, called "support vectors".

The hyperparameters used to tune this algorithm are:

- **C (regularization parameter):** Controls the trade-off between maximizing the margin and minimizing classification errors. A smaller value of C creates a wider margin but may misclassify more data points.
- **Kernel type:** Specifies the kernel function to be used (e.g., linear, polynomial, Radial basis function).
- **Gamma:** Determines the shape of the decision boundary. A low gamma value will produce a more flexible curve, while a high value will create a more rigid confined shape.
- **Degree:** Degree of the polynomial kernel function.

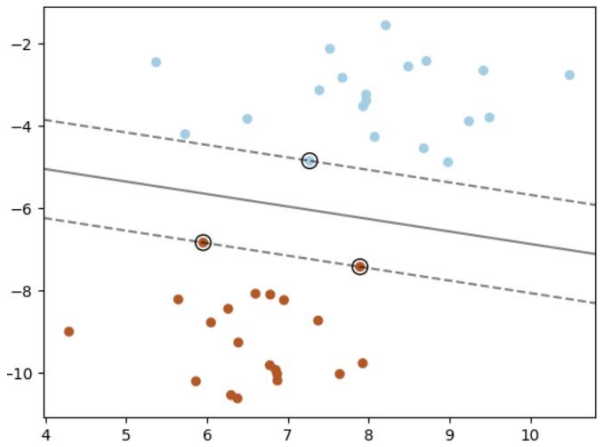


FIGURE 2.3: Example of a linearly separable problem solved by a SVM. [49]

2.3.2 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) [66] is one of the simplest yet surprisingly effective clustering-based supervised machine learning algorithms. Its fundamental premise is that data points that are close in feature space have similar output values, or class labels. For classification tasks, KNN simply stores instances of the training data and works by determining the 'k' training samples closest in distance to a new point and returning the most common output value among them by majority vote [66]. Figure 2.4 shows a visualization of this clustering algorithm.

The main hyperparameters used to tune this algorithm are:

- **Number of Neighbors (k):** The number of neighbors to consider when making classifications.
- **Distance Metric:** The method of calculating distance between data points, e.g. Euclidean, Manhattan, Minkowski, etc.
- **Weighting:** Decides if all neighbors have equal vote or if closer neighbors have a stronger influence on the prediction.

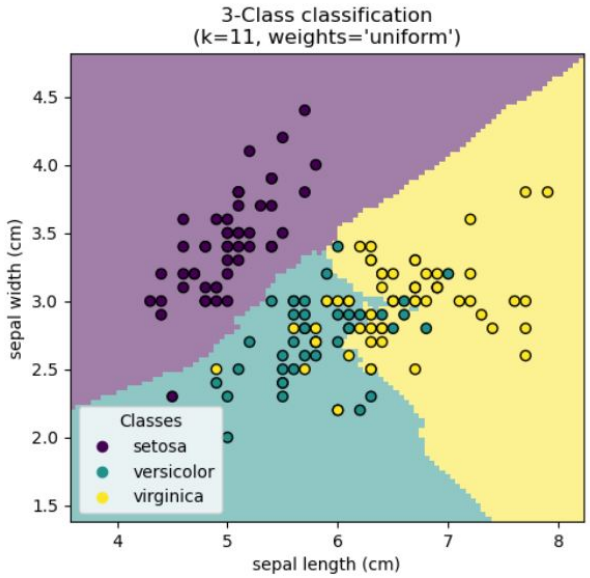


FIGURE 2.4: Visualization of the KNN algorithm applied to the Iris dataset. [49]

2.3.3 Random Forest (RF)

Random Forest (RF) [12] is an ensemble learning method that builds upon the foundational Decision Tree algorithm. It creates a "forest" of decision trees during training, each constructed using a random subset of the training data and a random subset of the features. When making a prediction, each tree in the forest casts a vote, and the Random Forest aggregates these votes to determine the final output. This ensemble approach both diversifies and stabilizes the Decision Tree model's predictions.

The main hyperparameters used to tune this algorithm are:

- **Number of Trees:** Specifies how many decision trees should be built in the forest.
- **Max Features:** The maximum number of features to consider when looking for the best split.
- **Max Depth:** The maximum depth of the tree.
- **Min Samples Split:** The minimum number of samples required to split an internal node.
- **Min Samples Leaf:** The minimum number of samples required to be at a leaf node.
- **Bootstrap:** Whether bootstrap samples are used when building trees.

2.4 Deep learning techniques

At its core, sensor based transport mode recognition involves information extraction of sensor's data embedded in smartphones. These sensors, such as GPS, accelerometers, gyroscopes and magnetometers, collect vast amounts of data about human movements and surroundings.

By definition, **deep learning** is a subset of machine learning, which is basically a neural network with three or more layers. These neural networks attempt to emulate the behaviour of the human brain-though far from matching its capabilities, but allow it to "learn" from large amounts of data. Although a neural network with a single layer can already make approximate predictions, additional hidden layers help to optimize and refine accuracy [31].

One of the remarkable aspects of deep learning in this context is its ability to detect patterns and features within sensor data that would be nearly impossible for humans to detect with other traditional methods. This is why, utilizing a neural network to extract temporal features becomes advantageous when building an end-to-end deep learning model [37]. This end-to-end learning approach streamlines the training process and improve mutual enhancement between feature learning and recognition processes. Numerous deep learning techniques have been employed for the extraction of temporal information, the most common ones for the task of this thesis will be explained below.

2.4.1 Multilayer perceptron

The **multilayer perceptron**, as described in reference [22], forms the foundation of feedforward networks. It consists of a system of simple interconnected neurons, or nodes, as illustrated in Figure 2.5, which is a model representing a nonlinear mapping between an input vector and an output vector. The nodes are connected by weights and output signals which are a function of the sum of the inputs to the node modified by a simple nonlinear transfer, or activation, function. It is the superposition of many simple nonlinear transfer functions that enables the multilayer perceptron to approximate extremely non-linear functions.

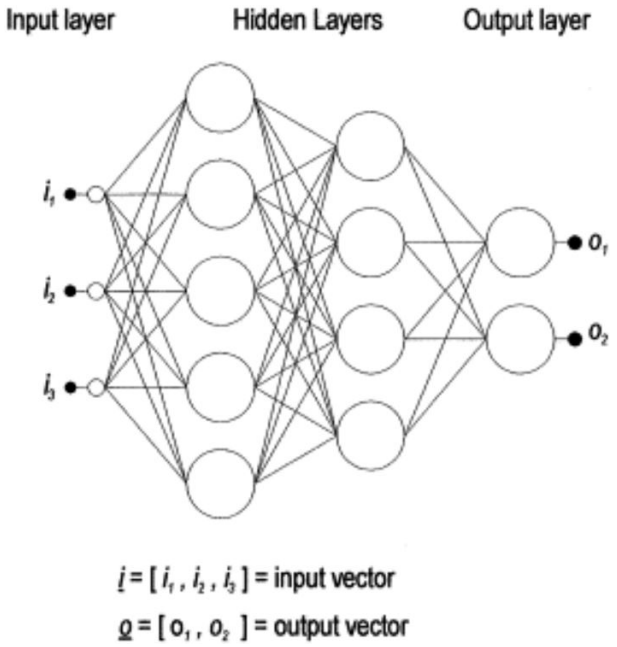


FIGURE 2.5: Representation of a multilayer perceptron with two hidden layers. [22]

The multilayer perceptron operates through a structured sequence of layers, each with a specific role in information processing:

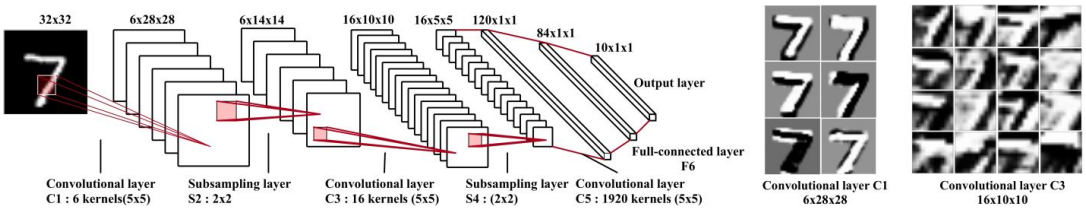
- **Input Layer:** This is the initial layer of the network, composed of artificial input neurons. These neurons hold the original data representing external inputs or features.
- **Hidden Layers:** Positioned between the input and output layers, the hidden layers play a pivotal role. They apply transformations to the input data using activation functions and then pass these transformed values to the output layer. Within these hidden layers, the neural network defines its weights, which signify the strength of connections between individual nodes. Weight updates, a critical component, represent the learning phase during neural network training. Adjusting weights helps the network fine-tune its ability to recognize patterns and make accurate predictions.
- **Output Layer:** This final layer of the network provides the algorithm's output, transmitting the results of the computations performed by the preceding layers.

Finally, the learning of the neural network is due to the **backpropagation** algorithm [62]. This algorithm involves two main steps: the forward pass, where input data is processed through the network to produce an output, and the backward pass, where the error between the predicted output and the actual target is propagated backward through the layers. During the backward pass, weights are updated to minimize this error by moving in the direction that reduces it, with the learning rate controlling the step size. This process is repeated for multiple iterations, gradually improving the network's ability to make accurate predictions on training data.

2.4.2 Convolutional Neural Networks (CNN)

One of the most popular deep neural network architectures is the **Convolutional Neural Network (CNN)**. It takes its name from the mathematical linear operation between arrays called convolution. What defines this type of neural network is the convolutional layer. It is designed specifically for processing grid-like data such as images. In this layer, a set of learnable filters (also called kernels) slide across the input data, performing convolutions. Each filter detects specific features or patterns within the input, such as edges, corners, or textures. Figure 2.6 shows an example of a convolutional neural network applied to images.

In order to generate a single value in the output feature map, the convolution procedure entails multiplying the filter's values element-by-element with a low percentage of the input data, then summarizing the results. This process is repeated for each filter and at multiple positions across the input, allowing the layer to capture different features and their spatial relationships.



(a) LeNet-5 network (b) Learned features

FIGURE 2.6: (a) The architecture of the LeNet-5 network. (b) Visualization of features in the LeNet-5 network. Each layer's feature maps are displayed in a different block. [8]

This type of neural networks can be particularly interesting for transport mode recognition from motion sensors in smartphones for some reasons. First, the convolution operation is advantageous because it excels at extracting local patterns and features within the data, making it capable of discerning unique movement patterns associated with various modes of transportation. Additionally, its shift-invariant property ensures that it can identify these patterns regardless of slight variations in sensor positioning or user movements. Moreover, CNNs are robust to noise, which is common in motion sensor data, as they automatically filter out irrelevant information.

Finally, CNNs are designed to capture spatial relationships within data. In the case of motion sensor data, the sequential nature of sensor readings can be treated as a spatial sequence, and CNNs can be adapted to learn meaningful temporal patterns from this sequential data. Some works employed **one-dimensional (1D) convolutions** on the individual univariate time series signals for temporal feature extraction. When there were multiple sensors or multiple axes, multivariate time series would be yielded, thus requiring the 1D convolutions to be applied separately as is shown in Figure 2.7.

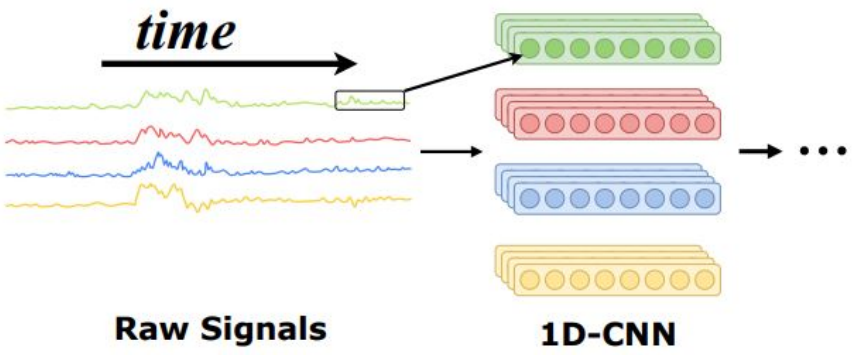


FIGURE 2.7: Example structure for temporal feature extraction applying 1D convolutional layers. [14]

2.4.3 Dilated CNN

Dilated convolutions introduced in [64], in contrast to regular convolutions, employ dilated convolution kernels to increase the convolutional receptive field (i.e., time length), which essentially means extending the area over which information is considered without sacrificing resolution. This expansion is achieved by introducing empty spaces (known as dilation) between the elements of the conventional convolutional kernel, as shown in Figure 2.8.

The key advantage of dilated convolutions is that they allow a CNN to capture a broader context without a significant increase in computational cost. This is because the dilation only adds "gaps" between kernel elements, rather than requiring additional calculations for each new position in the input data.

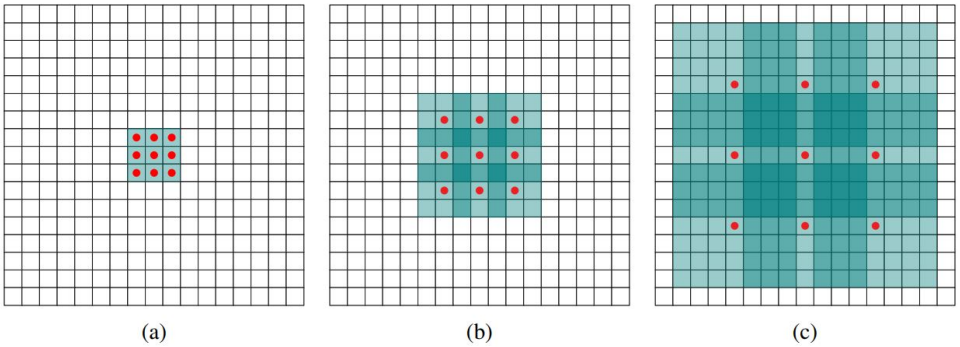


FIGURE 2.8: (a) 1-dilated convolution; each element has a receptive field of 3×3. (b) 2-dilated convolution; each element has a receptive field of 7×7. (c) 4-dilated convolution; each element has a receptive field of 15×15. [64]

2.4.4 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) are named "recurrent" due to their ability to execute a consistent operation for every element within a sequence. Their outputs are influenced

by the results of prior computations, essentially creating a sense of continuity. Another perspective on RNNs is that they possess a form of "memory" that retains information about the calculations conducted earlier in the sequence.

On the left side of the diagram provided in Figure 2.9, there is a representation of an RNN, while the right side illustrates the unrolling of an RNN into a complete network. In this context, unrolling refers to the process of reproducing the entire network structure for a given sequence. For instance, when dealing with a sentence composed of three timesteps, the unrolled network would manifest as a 3-layer neural network, with each layer corresponding to one of the timesteps in the sequence.

In order to model this functioning, RNNs have the following components:

- **Input:** *x(t)* is taken as the input to the network at time step *t.* For example, *x1*,could be a one-hot vector corresponding to a sensor sample of a sequence.
- **Hidden state***: h(t)* represents a hidden state at time t and acts as "memory" of the network. *h(t)* is calculated based on the current input and the previous time step's hidden state: **h(t)** = f(*U* **x(t)** + *W* **h(t-1)**). The function f is taken to be a non-linear transformation such as *tanh*, *ReLU.*
- **Weights**: The RNN has input to hidden connections parameterized by a weight matrix U, hidden-to-hidden recurrent connections parameterized by a weight matrix W, and hidden-to-output connections parameterized by a weight matrix V, and all these weights (*U*,*V*,*W)* are shared across time.
- **Output**: *o(t)* illustrates the output of the network.

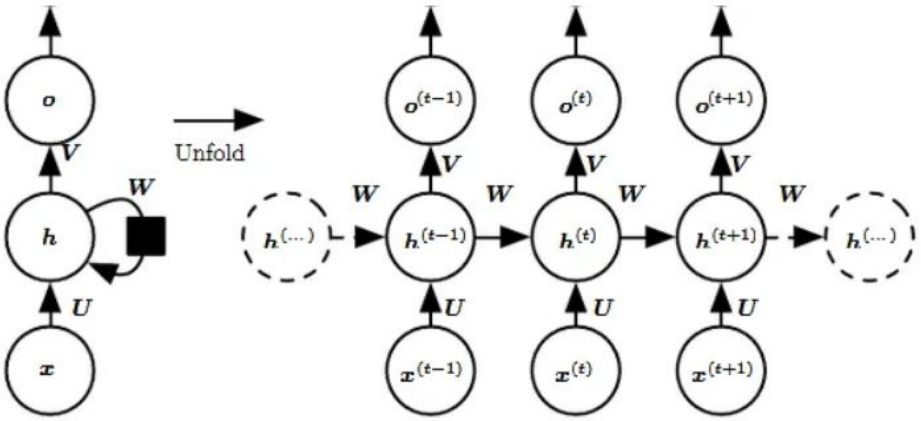


FIGURE 2.9: Diagram of a 3-layer recurrent neural network. [27]

In the context of time-series classification tasks, RNNs offer some advantages over CNNs. RNNs excel at handling sequential data, accommodating variable-length sequences, and capturing temporal dependencies, making them suitable for tasks where order and historical context matter. They can automatically extract relevant features from complex time-series patterns and are well-suited for real-time processing scenarios. Additionally, transfer learning with pretrained RNN models is more common in time-series tasks.

However, traditional RNN cells suffer from vanishing/exploding gradients problems, which limits the application on large samples.

2.4.5 Long short-term memory (LSTM)

The **Long Short-Term Memory (LSTM)** units have overcome the issue of vanishing/exploding gradients [24]. They do so by introducing key elements such as the cell state, which corresponds to the *C* layer in Figure 2.11. This cell state allows information to flow through the network with minimal alteration. It serves as the long-term memory of the LSTM. A comparison of the structure of traditional RNNs and the LSTM can be found at Figure 2.10

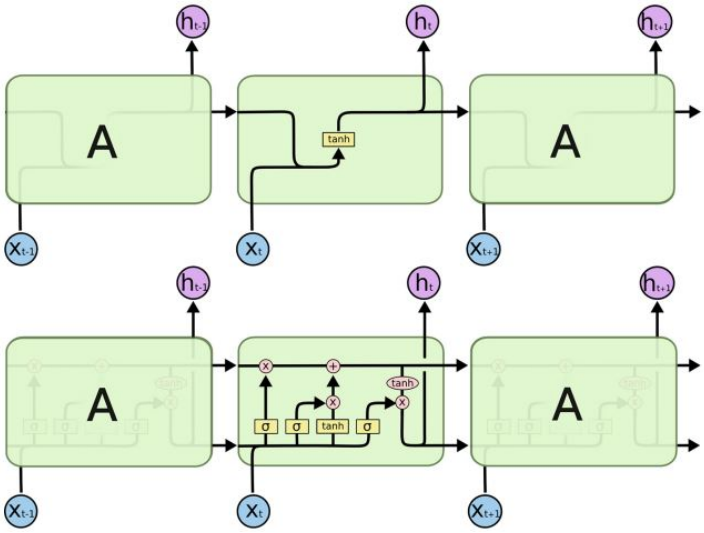


FIGURE 2.10: Comparison of a RNN cell (above) and an LSTM cell (below). [55]

Another important concept introduced by the LSTM cell are gates. They serve as a mechanism for selectively allowing or filtering information flow. A gate consists of two main components: a sigmoid neural network layer and a pointwise multiplication operation. The sigmoid layer produces values in the range of zero to one, representing the extent to which each component of the information should be permitted to pass through. In essence, gates play a pivotal role in regulating the flow of information by determining what gets through and what does not. LSTMs use three types of gates to control the flow of information within the network (see Figure 2.11):

- **Forget Gate:** This gate decides what information from the cell state should be discarded or kept.
- **Input Gate:** The input gate determines what new information should be added to the cell state and computes a candidate cell state. This candidate state is then combined with the output from the forget gate to update the cell state.
- **Output Gate:** The output gate controls what information from the cell state should be exposed to the network's output. It looks at the current input and previous hidden state, squashes the values, and produces the output.

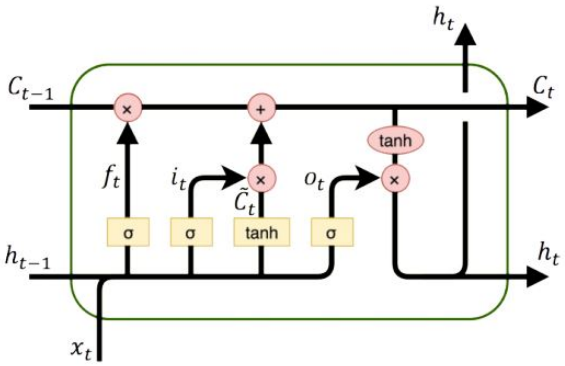


FIGURE 2.11: LSTM cell structure. [55]

In practice, to effectively process sequential data, it is advisable to have a minimum depth of two in an LSTM-based Recurrent Neural Network (RNN), as highlighted in the reference [35]. Given that sensor signals often constitute a continuous stream of data, a common approach involves employing a sliding window to partition the raw data into discrete segments, each serving as an input to an individual RNN cell. Figure 2.12 illustrates a typical LSTM-based architecture designed for extracting temporal features.

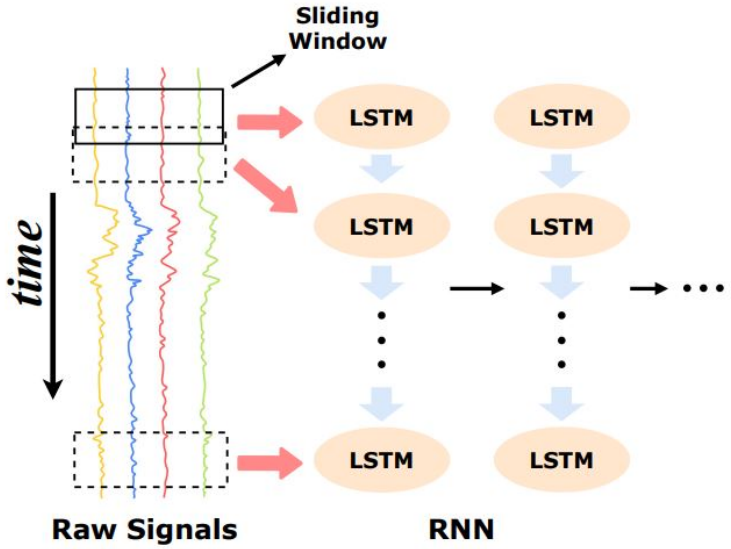


FIGURE 2.12: LSTM-based architecture example for extracting temporal features from sensor signals. [14]

2.4.6 Bidirectional LSTM (Bi-LSTM)

A **Bidirectional Long Short-Term Memory (Bi-LSTM)** is a sequence processing model that utilizes two Long Short-Term Memory networks (LSTMs) working in tandem: one processes the input data in a forward direction, while the other processes it in a backward direction. This bidirectional approach significantly enhances the network's ability

to capture and understand temporal patterns in the data. A diagram of this operation is shown in Figure 2.13.

By analyzing the data both forwards and backwards, Bi-LSTMs effectively double the amount of information available to the model [52]. This enhanced context enables the algorithm to gain a deeper understanding of the sensor data by knowing what data points immediately follow and precede a given point in the time series. This capability is particularly valuable when dealing with time-dependent phenomena, as it allows the model to capture intricate dependencies and relationships within the sequential sensor measurements, making it a valuable tool for tasks like prediction, anomaly detection, and feature extraction in time series analysis.

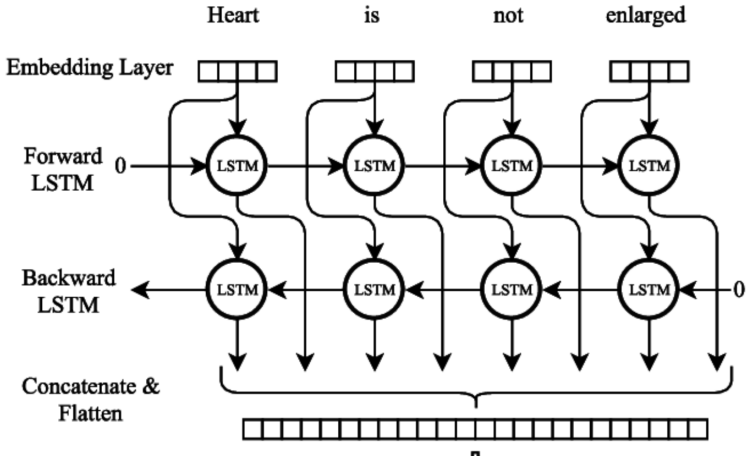


FIGURE 2.13: Bi-LSTM structure applied to NLP [7].

2.4.7 Attention mechanism

The **attention mechanism** is a fundamental component in deep learning models, particularly in the fields of natural language processing (NLP) and computer vision, but it has applications in various other domains as well. At its core, the attention mechanism is a mechanism that allows a model to focus on specific parts of input data while processing it, rather than treating all parts equally.

The attention mechanism is often used with sequences of data, such as words in a sentence, pixels in an image, or time steps in a time series. For each element in the input sequence, the attention mechanism calculates a weight or score that reflects the importance of that element in relation to the others. These weights are often referred to as attention scores. The sum of these scores is used to create a context vector, which is a weighted sum of the input elements. This context vector represents the "attention" the model pays to each input element. The context vector is then used to weight the processing of the input sequence. Elements that receive higher attention scores have a greater influence on the model's computations, while those with lower scores have less influence.

In practice, an attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each

value is computed by a compatibility function of the query with the corresponding key [59]. The most commonly used attention function is **Scaled Dot-Product Attention** (see Figure 2.14). In this variant, the attention function is computed on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. The matrix output is as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

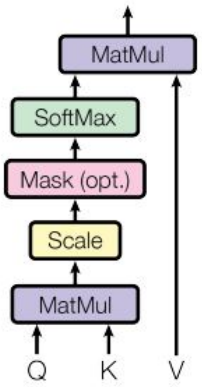


FIGURE 2.14: Scaled Dot-Product Attention mechanism [59].

The Scaled Dot-Product Attention mechanism, while effective, has its limitations when dealing with complex data structures. To address this, the concept of **Multihead Attention** was introduced [59]. The primary idea behind Multihead Attention is to allow the model to focus on different parts of the input simultaneously, capturing various types of relationships and dependencies. Instead of using a single set of attention weights, the multihead attention block employs multiple sets, often referred to as "heads". Each head computes its own attention weights and produces its own output vector. These outputs are then concatenated and linearly transformed to produce the final output of the multihead attention block. This process is represented by the following equation:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_n)W^O \tag{2.2}$$

Figure 2.15 represents the multihead attention block, taking h as the number of heads employed. As can be observed in the figure, after computing the attention scores and obtaining the weighted sum of the Value vectors, the results (for multiple heads in multihead attention) are concatenated. Finally, the output is typically passed through another linear layer to produce the final output.

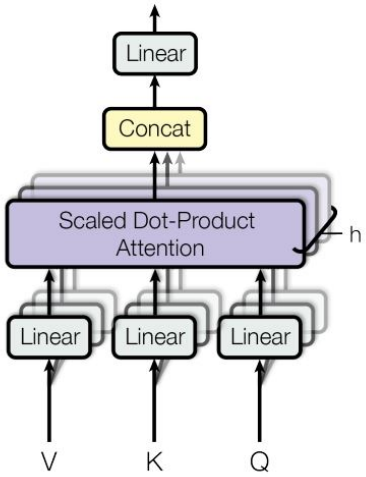


FIGURE 2.15: Multihead attention block [59].

2.4.8 Transformers

Transformers [59], employ an attention mechanism that systematically assesses an input sequence's elements, assigning varying degrees of importance to each element at every step. This novel approach has revolutionized sequential data analysis, progressively surpassing LSTM, which was the previous state-of-the-art model in this domain.

To achieve this, the Transformers follow a very characteristic architecture, shown in Figure 2.16. The left part of the architecture corresponds to the **Encoder** block. The encoder is composed of a stack of **N_x** identical layers. Every layer consists of two sub-layers. The initial one is a multi-head self-attention mechanism, while the subsequent one is a straightforward, position-wise fully connected feed-forward network. A residual connection around both sub-layers, complemented by layer normalization, is applied.

On the other hand, the right part of the architecture corresponds to the **Decoder** block. The decoder is also composed of **N_x** identical layers. Besides the two sub-layers in every encoder layer, the decoder incorporates a third sub-layer that executes multi-head attention on the encoder stack's output. Changes were made to the self-attention section in the decoder to prevent it from looking ahead. By shifting the output by one position, it's ensured that predictions at any given position are based solely on outputs from preceding positions.

Finally, two important components of the transformer architecture are embeddings and positional encodings. **Embeddings** are a way to convert discrete variables, like words or tokens, into continuous vector representations. **Positional Encoding** is added to the embeddings before the data is fed into the encoder or decoder. This ensures that the transformer has both the semantic information from the embeddings and the positional information from the positional encoding. This solves an inherent limitation of the standard transformer model, since originally it did not take into account the order of the input. This is because it does not inherently process the data in sequence, like recurrent neural networks (RNNs).

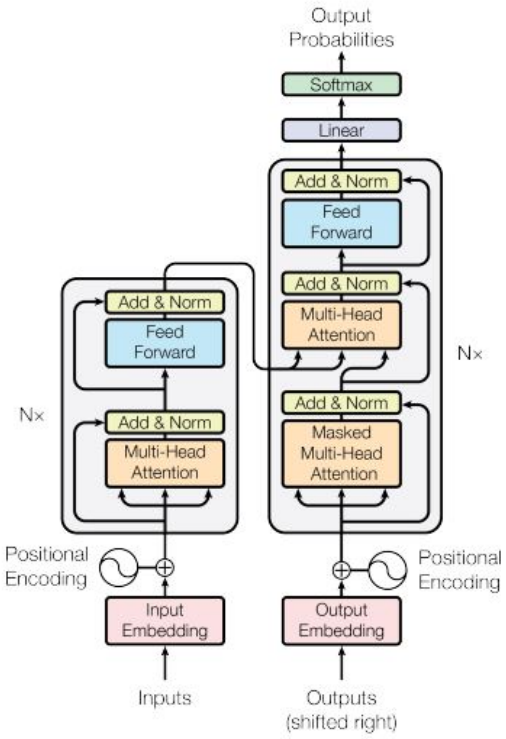


FIGURE 2.16: The Transformer - model architecture. [59]

2.4.9 Transformers in Transport mode recognition

Transformers, with their proficiency in handling sequential data, can be pivotal in transport mode detection using smartphone motion sensors. Their self-attention mechanism allows them to discern specific motion patterns that can characterize different transport modes, like distinguishing the rhythmic movement of walking from the smoother motion of vehicular travel. Moreover, the ability of transformers to integrate data from multiple sensors, such as accelerometers and gyroscopes, further improves their efficacy in this application.

However, these advantages come with certain trade-offs. For one, the computational demands of Transformer models are high, potentially posing challenges for real-time processing on resource-constrained smartphones. This complexity also translates to increased power consumption, which is a concern for battery-dependent devices. Additionally, while they excel in rich data environments, transformers can be prone to overfitting when trained on limited datasets. This means that despite their impressive performance during training, they might not generalize well to real-world, unseen data. Lastly, these models require substantial labeled data for effective training, and acquiring such data for every possible transport mode in varying conditions can be a huge challenge.

2.5 Performance evaluation

The effectiveness of a classifier is heavily influenced by the attributes of the data it needs to classify. To evaluate its performance in various contexts, a combination of metrics and

visual methods are employed. Here is a detailed explanation of the evaluation metrics used in this work.

2.5.1 Performance Metrics

Performance metrics are an integral component of every machine learning pipeline, serving as vital indicators of progress and success. These metrics provide quantitative measures that help in assessing the effectiveness of a model, offering clear, numerical insights into its performance. To this end, we delve into four key metrics: precision, recall, F1-score, and accuracy. Each of these metrics provides unique insights into the performance of the classifier.

- **Precision**: This metric indicates the proportion of correctly predicted positive observations to the total predicted positives.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{2.3}$$

- **Recall**: Recall calculates the proportion of actual positives that were correctly identified.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \tag{2.4}$$

- **F1-Score**: The F1-Score is the weighted average of precision and recall.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.5}$$

- **Accuracy**: It measures the proportion of true results (both true positives and true negatives) among the total number of cases examined.

Furthermore, while these metrics are intuitive, they may not always be the best metric for imbalanced classes, which is common in transport mode detection. Two variants of the above metrics are used in this case for a more informative performance analysis:

- **Macro Average:** This averages the metric independently for each class and then takes the average (hence treating all classes equally).
- **Weighted Average:** This accounts for class imbalance by weighting the average of the metric in favour of the most abundant class. It is calculated for each class label, and the average is weighted by the number of true instances for each label.

2.5.2 Confusion Matrix

A **confusion matrix** is a vital tool in the evaluation of classifiers. It offers a visual representation of a classifier's performance. In this matrix, each row represents the instances in an actual class, and each column corresponds to the instances in a predicted class (see Figure 2.17). The matrix aids in understanding the types of errors made by the classifier, for example, mistaking walking for cycling or driving for public transit.

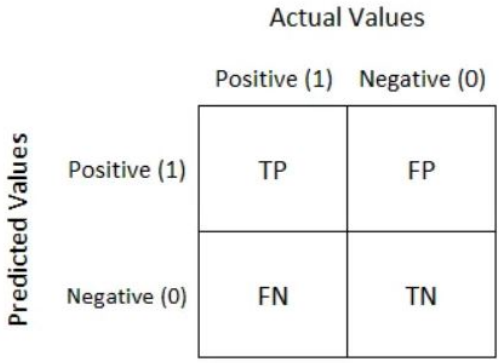


FIGURE 2.17: Confusion matrix's simple example [23].

2.5.3 Learning Curves

Training curves are graphical representations that show the evolution of the model's performance over time during the training process. These curves typically include loss curve and accuracy curve. Monitoring these curves helps in detecting issues such as overfitting or underfitting, bias and variance allowing for necessary adjustments to the model's training process (see Figure 2.18).

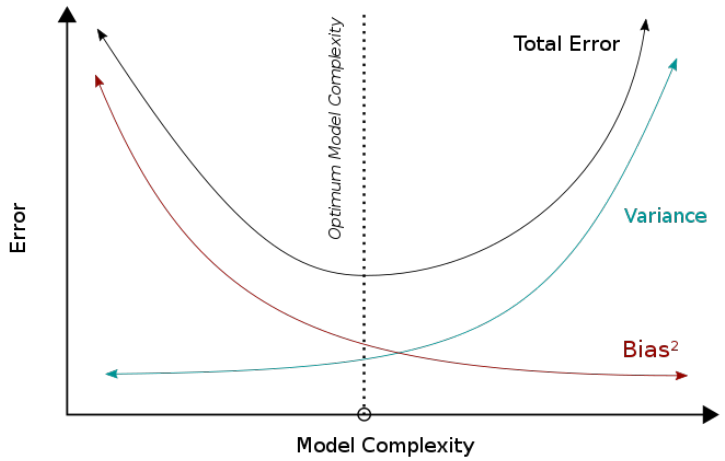


FIGURE 2.18: Bias and variance represented in an error's learning curve [23].

2.5.4 Group k-fold cross-validation

Group k-Fold Cross-Validation was utilized as the primary model evaluation technique in this study, tailored for datasets with distinct groupings. This method is especially appropriate for data segregated by entities such as users, which is the case, ensuring no overlap between training and testing sets. The dataset is divided into 'k' distinct groups with different users on each group, with each one successively used as a test set while the others form the training set. This strategy guarantees comprehensive utilization of data for both training and validation purposes and ensures that each group is entirely

excluded from the training data during its turn as the validation set, thus bolstering the evaluation's robustness and relevance.

Following the Group k-Fold Cross-Validation, an **Mann-Whitney U test** [45] was conducted to statistically compare models. This statistical test, also known as the Wilcoxon rank-sum test, is a non-parametric statistical test used to compare two independent samples to determine whether there is a difference in their distribution. It is particularly useful for comparing models in cases where the data does not follow a normal distribution, which is a common assumption for many parametric tests. By applying Mann-Whitney U test to the results obtained from each fold of the cross-validation, the study could determine if there is a significant difference in the performance of the two models.

2.6 Data acquisition modalities

In this section, we will explore various methods and technologies for gathering data related to human activity and movement. The section will delve into three primary modalities: motion, location, and ambient sensing. For each modality, the advantages and disadvantages will be examined, offering an objective overview of their strengths and limitations in data acquisition for diverse purposes.

2.6.1 Motion modality for transport mode recognition

The performance of an activity recognition system depends crucially on the sensor modality used. There are diverse types of sensors, such as wearable sensors, ambient sensors, and location sensors. However, in this work, we are going to focus on the most commonly used sensors available in smartphones, wearable sensors [20].

- **Accelerometers** are devices used to measure acceleration, specifically the rate of change in an object's velocity. They are typically measured in metres per second squared (m/s^2) or G-forces and operate at sampling frequencies ranging from tens to hundreds of Hz. They provide a tri-variate time series due to their three axes (see Figure 2.19).
- **Gyroscopes** measure orientation and angular velocity, with the unit of angular velocity being radiants per second (rad/s). Like accelerometers, they also operate at sampling rates ranging from tens to hundreds of Hz. Gyroscopes are often integrated with accelerometers, and also provide three axes of data (see Figure 2.20).
- **Magnetometers**, on the other hand, are commonly used wearable sensors and are usually combined with accelerometers and gyroscopes into an inertial unit. They measure changes in the magnetic field at a specific location, using Tesla (T) as the measurement unit and having sampling rates in the tens to hundreds of Hz. Magnetometers, like the others, also have three axes. They can be used to estimate the three-dimensional orientation of the device relative to the Earth's magnetic north. Figure 2.21 depicts the functioning of a magnetometer.

Advantages of motion modality for transport mode recognition

Using a combination of accelerometer, gyroscope, and magnetometer sensors is advantageous for activity recognition because it provides a comprehensive and robust dataset. This combination offers the following benefits:

- **Comprehensive Data:** A wide variety of information is provided by these three sensors when taken together, including linear acceleration (from the accelerometer), angular velocity (from the gyroscope), and orientation with regard to the Earth's magnetic field (from the magnetometer). This extensive dataset records many facets of motion and orientation, enabling a more complete comprehension of the user's motions.
- **Orientation Awareness:** The magnetometer offers important details regarding the user's orientation with respect to the Earth's magnetic field. This information is particularly useful for classifying modes like walking and bicycling, where changes in direction play a big role.
- **Real-Time Capability:** The data from these sensors can be processed in real-time, allowing for instantaneous mode recognition.
- **Privacy Considerations:** Unlike GPS, which can be highly intrusive in terms of user privacy, accelerometer, gyroscope, and magnetometer data can be processed without revealing the user's exact location.
- **Widespread availability:** All modern smartphones come equipped with these sensors as standard hardware components. This availability makes it exceptionally convenient and cost-effective to implement transport mode recognition on a large scale, as users do not need to invest in additional hardware or devices.
- **Battery consumption:** The battery consumption of the smartphone sensors is much lower than that caused by other sources of information. This can be seen in detail in Table 2.2.

Drawbacks of motion modality for transport mode recognition

On the other hand, this method of data collection also has some drawbacks, which are explained below:

- **Data Noise:** Sensor data can be noisy, especially in real-world scenarios. Vibrations, shocks, and external interference can introduce errors into the data. This noise can affect the accuracy of transport mode recognition algorithms, leading to incorrect results.
- **Dependency on Device Placement:** The placement of sensors within a device can affect their performance. Different smartphones and wearables may have sensors located in slightly different positions, leading to variations in data collection and recognition accuracy.
- **Integration Challenges:** Integrating sensor-based recognition into apps or devices can be technically challenging and may require specialized knowledge in signal processing, machine learning, and software development.

- **Calibration and Sensor Drift:** Over time, sensor values can drift due to temperature changes or wear and tear. Maintaining accurate and calibrated sensors can be a challenge.

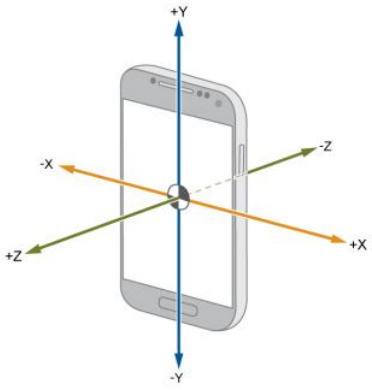


FIGURE 2.19: Axis directions for the accelerometer of the smartphones. [9]

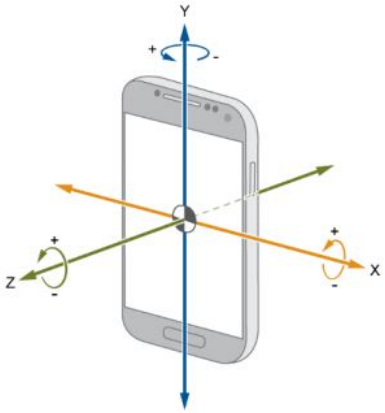


FIGURE 2.20: Axis directions for the gyroscope of the smartphones. [9]

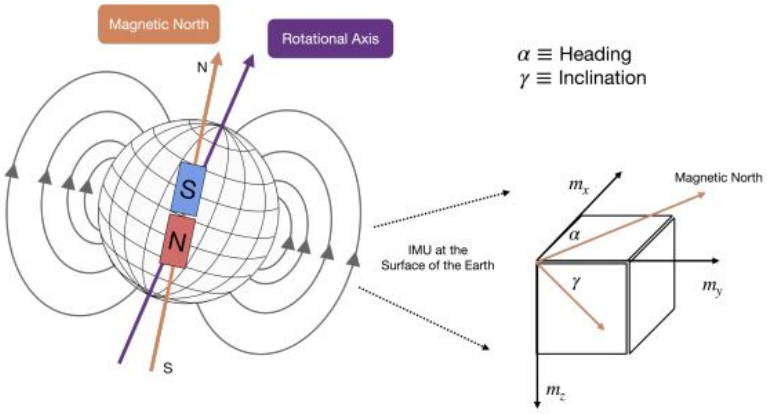


FIGURE 2.21: Functioning of the smartphone magnetometer. [9]

TABLE 2.2: Average battery consumption of the considered sensors. Information taken from many examples found in the literature and some tests made for diverse smartphones.

GPS (update 15sec)	WiFi (update 15sec)	ACC	MAG	GYR
250 mA	125 mA	0.23 mA	6.8 mA	6.1 mA

2.6.2 Location modality for transport mode recognition

Location-based information can also be used to identify a user's mode of transportation or activity. This method uses information from the **Global Positioning System (GPS)** or other location services to obtain a person's current position and rate of movement. The following are some benefits and drawbacks of utilizing location modality for identifying transport modes:

Advantages of location modality for transport mode recognition

- **High-Level Information:** Location data can provide high-level information about a user's activity, such as whether they are indoors, outdoors, in a car, on foot, or using public transportation. This can be valuable for recognizing transport modes.
- **Easy Integration:** GPS libraries are usually integrated into smartphone frameworks. This makes it easier to work with this type of data.
- **Contextual information:** location data provides contextual information about the trip.

Drawbacks of location modality for transport mode recognition

- **Indoor Limitations:** Location data can be less accurate indoors or in areas with poor GPS signal reception. Recognizing transport modes or activities indoors can be challenging.
- **Lack of Fine-Grained Information:** Location data may not provide fine-grained information about specific transport modes, such as distinguishing between different types of vehicles (e.g., car, bus, train).
- **Battery Consumption:** Continuous GPS usage can consume a significant amount of power, potentially affecting device battery life.
- **Privacy Concerns:** Gathering location data raises privacy concerns, as it can reveal a user's whereabouts. App developers must handle location data responsibly and transparently to address user privacy concerns.
- **Dependency on Location Services:** Transport mode recognition based on location data relies on the availability and accuracy of location services on the user's device. Any issues with these services can affect the reliability of the recognition system.

2.6.3 Ambient modality for transport mode recognition

Ambient sensors, such as Wi-Fi, RFID (Radio-Frequency Identification), and radar, can be used for transport mode recognition as well.

- **Wi-Fi** is a local-area wireless network connection technology that uses a transmitter to send signals to a receiver. The basis of WiFi-based human activity recognition is that human's movements and locations interfere with the signals' propagation path from the transmitter to the receiver, including both the direct propagation path and the reflecting propagation path.

- **RFID** uses electromagnetic fields to automatically identify and track the tags attached to objects, which contain electronically stored information. RSS is the most widely adopted tool for RFID-based activity recognition; an example is shown in Figure 2.22. The working mechanism is that human's movements would change the single strength received by the RFID reader [61].
- **Radars**. Unlike WiFi and RFID whose transmitters and receivers are placed on opposite sides, radar transmitters and antennas are mounted on the same side of users. The Doppler effect is the basis of the radar-based system [39].

Advantages of ambient modality for transport mode recognition

- **Multi-occupant detection:** Each person's presence and movements can influence the Wi-Fi signals differently, making it possible to recognize and distinguish multiple occupants in the monitored area. This capability is useful for applications like occupancy sensing in smart homes or tracking the number of people in a public space.
- **Indoor localizing:** Wi-Fi can work well indoors, where GPS signals may be weak or unavailable.
- **Network-based data:** Wi-Fi can provide information about the availability of Wi-Fi networks, which can be used as a context clue for transport mode recognition.

Drawbacks of ambient modality for transport mode recognition

- **Infrastructure deployment:** Deploying an RFID or Radar infrastructure can be costly and time-consuming, making it less practical for wide-scale use.
- **Battery consumption:** Depending on the radar system's design, it may consume a significant amount of power, which can be a drawback for battery-powered devices. In addition, having the phone's Wi-Fi activated also consumes a significant amount of energy.
- **Limited Outdoor Accuracy:** Ambient based recognition may be less accurate for outdoor activities, where GPS is more reliable.

FIGURE 2.22: Example of the use of RSS technology in indoor spaces [39].

Chapter 3

Background

This chapter serves as an overview for understanding the scope of the study. It begins with the *Key Contributions* section, which catalogues the significant advancements that have fundamentally shaped the current state of the field. Following, the *Data Availability* section examines the availability of datasets for experimentation in the field. This section highlights the main characteristics of the datasets available in the literature, as well as their advantages and disadvantages.

3.1 Key contributions

In the following section, we will delve into the key contributions, from traditional ones to more recent ones, that have shaped the landscape of the subject matter. These pivotal contributions represent the foundation upon which my understanding and progress in this field have been built.

3.1.1 Traditional proposals

Initially, the task of transport mode classification focused on manual feature extraction and training of machine learning models using traditional algorithms such as SVM, KNN or Decision trees.

Hemminki et Al.

An example of that is the *Hemminki et Al.* [29] proposal. This proposal is based on a hierarchical feature-based system composed of three stages. The hierarchy begins with a *Hemminki et Al.* [29] proposalr, which initially distinguishes between pedestrian motion and other types of motion at a coarse level. If the kinematic motion classifier fails to detect significant physical movement, such as walking, the process advances to a stationary classifier. This classifier then assesses whether the user is either stationary or on some form of motorized transportation. When motorized transportation is identified, the classification process moves on to a

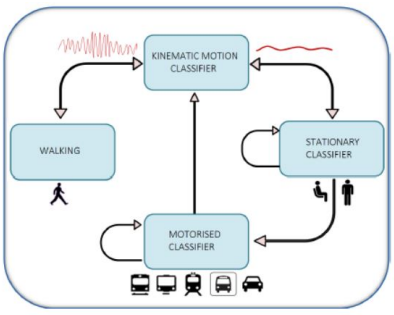


FIGURE 3.1: Architecture proposed by Hemminki et Al. [29].

motorized classifier. This classifier is responsible
for categorizing the ongoing activity into one of five modalities: bus, train, metro, tram, or car, and it employs Adaptive Boosting for enhanced accuracy. The diagram shown in Figure 3.1 summarizes that architecture.

A thing to improve in this system is the fact that the complexity of the classification framework and the number of parameters involved in the feature extraction make results interpretation much more difficult and enhancements to the classification logic hard to demonstrate and test.

Manzoni et Al

Another interesting accelerometer-based approach in the literature is the work of *Manzoni et Al.* [40]. In this work, a decision tree classifies a set of features including 32 FFT coefficients and the signal variance. The resultant accuracy is slightly lower than the ones presented by Hemminki et Al. However, the number of transportation modes taken into account is higher (eight instead of seven).

Nham et Al.

Following a similar path, *Nham et Al.* [46], proposed an offline classification of transportation modes by training a Support Vector Machine with 253 features (250 FFT (Fast Fourier Transforms) coefficients, signal energy, mean and variance), obtaining accuracy over 90%. An interesting approach that introduced this study is the computation of the total vector of acceleration magnitudes in order to eliminate the dependency of the signal processing from the phone orientation. The fact that they trained the model with data from each individual separately rather than training it on one group of subjects and evaluating it on another is a drawback of this research.

Brezmes et Al.

Brezmes et Al.'s approach, as described in their study [11], entails the real-time classification of frequency-based features using a server-based application. This method serves a broader purpose of activity detection, encompassing activities such as standing, walking, running, ascending stairs, descending steps, and more. It is worth noting that certain activity recognition systems considered in this evaluation may be relevant, even if the categorized activities differ from those directly related to TMD. This is because the K-nearest neighbours algorithm is used assuming no prior knowledge of the phone orientation, still achieving very decent results.

Ravi et Al. and Kwapisz et Al.

All previous studies have considered that the device collecting sensor data does not have a fixed position. Differing from this branch of research, examples of activity recognition approaches using fixed phone positions are those of *Ravi et Al.* [50] and of *Kwapisz et Al.* [36]. These present similar solutions but with higher accuracy, since the problem is simplified.

Rosenberg Randleff et Al.

There are also other studies that, in addition to using sensor data, use GPS data to improve classification. An example of this is the approach proposed by *Rosenberg Randleff et Al.* [51]. The plan was to primarily rely on accelerometer data unless it was not clear what mode of transportation was being used. This way, they save energy and keep the algorithm simple by using more power-hungry sensors and data processing only when necessary. It is important to note that the algorithm was tested on a small set of data, and users will need to put in some effort to label their trips with device orientation and transportation mode during the training phase. Still, the concept of using extra sensors when needed to supplement accelerometer data is interesting.

3.1.2 Recent contributions

In the context of machine learning and artificial intelligence, the importance of feature selection has been a long-standing challenge. Traditionally, the process of identifying and selecting relevant features from raw data has been a crucial step in building effective predictive models. However, modern deep learning techniques have revolutionized this landscape by enabling us to bypass the feature selection process and directly ingest raw data for tasks such as activity recognition and transportation mode prediction.

Gjoreski et Al.

One of the pioneering techniques in this regard is the use of **Convolutional Neural Networks** (CNNs) [25]. CNNs have shown remarkable capabilities in understanding spatial relationships within data, making them highly effective in tasks like image recognition. When applied to activity recognition, CNNs can directly process raw sensor data, allowing for the automatic extraction of relevant features from the input.

Murad et Al.

Deep Recurrent Neural Networks [44] have also played a significant role in eliminating the need for feature selection. These networks, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), excel in modeling sequential data. They can capture temporal dependencies in sensor data, making them ideal for tasks where the order of observations matters, like in the case of Time Series.

Okita et Al.

In the pursuit of more sophisticated models, **ConvLSTM** [48] emerged as a fusion of CNNs and LSTMs. This architecture combines the spatial awareness of CNNs with the temporal modeling abilities of LSTMs, making it a potent choice for tasks like activity recognition. By processing raw data directly, ConvLSTM effectively overcomes the challenges posed by traditional feature selection approaches.

Song et Al.

Building upon these advancements, **Song et al.** introduced **DeepTransport** [53]. This innovative approach applies deep learning techniques to model human mobility and

transportation modes by utilizing GPS (Global Positioning System) traces. By directly processing GPS data, DeepTransport sidesteps the need for handcrafted features or preprocessing steps, resulting in more accurate and adaptable transportation mode predictions.

Jeyakumar et Al.

Inspired by the transformative success of deep learning in activity recognition and transportation mode prediction, the Vahan team has proposed a novel deep learning model known as the **Deep Convolutional Bidirectional LSTM (DCBL)** [34]. DCBL takes raw sensor data as input, combining the power of convolutional layers to capture spatial patterns and bidirectional LSTMs to model temporal dependencies. This approach enables precise predictions of transportation modes without the complexity of feature engineering or selection.

3.2 Data availability

In this section, We explore several prominent public datasets featured in the literature, focusing on their sources, features, and relevance to transportation mode detection research. My emphasis will be on datasets that encompass sensor data from smartphones. Specifically, we will examine the SHL dataset [60], TMD dataset [13], and Collecty dataset [19]. The discussion will begin with a detailed description and analysis of the data each dataset offers, followed by a reflection on their respective strengths and weaknesses.

It should be noted that a well-known dataset that has been extensively studied in the literature is the HTC dataset. The problem with this dataset is that it is not public, therefore it is necessary to reach an economic agreement with the distributor to have this data available. For this reason, it has been decided not to take it into account.

3.2.1 Sussex-Huawei Locomotion Dataset (SHL)

The **Sussex-Huawei Locomotion Dataset** (SHL) [60] is a research dataset primarily focused on human locomotion, aiming to facilitate advancements in wearable sensing, particularly for recognizing human activities and modes of transportation.

The dataset was produced as a collaboration between **the University of Sussex** and **Huawei**, the global telecommunications and consumer electronics manufacturer. This dataset involves multi-sensor data collected from smartphones and smartwatches, worn by participants during various activities. Specifically, this dataset offers the following transports: Still, Walk, Run, Bike, Car, Bus, Train, Subway. The distribution of the dataset is shown in Figure 3.2

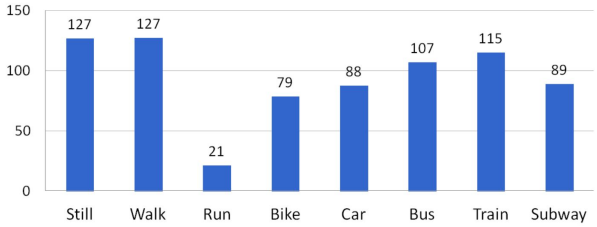


FIGURE 3.2: Cumulative duration in hours of each activity in the SHL dataset. [60]

This data was collected from three participants performed full-time data collection in realistic scenarios. The used device was a Huawei Mate 9 smartphone placed on different positions described in Figure 3.3. From there, a large number of sensors and other types of data are collected in this dataset. The complete list of these is as follows:

- **Accelerometer**: x, y, z in $\mathrm{m/s^2}$
- **Gyroscope**: x, y, z in $\mathrm{rad/s}$
- **Magnetometer**: x, y, z in $\mu\mathrm{T}$
- **Orientation**: quaternions in the form of w, x, y, z vector
- **Gravity**: x, y, z in $\mathrm{m/s^2}$
- **Linear acceleration**: x, y, z in $\mathrm{m/s^2}$
- **Ambient pressure** in hPa
- **Google's activity recognition API output**: 0-100% of confidence for each class ("in vehicle", "on bicycle", "on foot", "running", "still", "tilting", "unknown", "walking")
- **Ambient light** in lx
- **Battery level** (0-100%) and temperature (in °C)
- **Satellite reception**: ID, SNR, azimuth and elevation of each visible satellite
- **Wifi reception** including SSID, RSSI, frequency and capabilities (i.e. encryption type)
- **Mobile phone cell reception** including network type, location area code (LAC), mobile country code (MCC), mobile network code (MNS) and signal strength
- **Location** obtained from satellites (latitude, longitude, altitude, accuracy)
- **Audio**

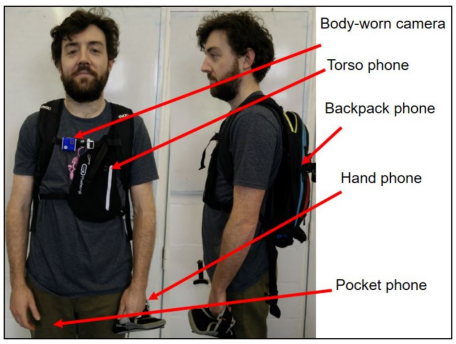


FIGURE 3.3: Positioning of the device in the data collection process. [60]

As with any dataset, the SHL presents a unique set of advantages and disadvantages that researchers and analysts should consider. Below is an examination of its strengths and potential limitations.

Advantages of the SHL dataset

- **Sensor variability:** With data points ranging from basic motion sensors like accelerometer, gyroscope, and magnetometer to ambient light and audio, the dataset provides a multi-dimensional view of the surroundings and actions.
- **Integrated Third-Party Data:** The integration of Google's activity recognition API output gives a benchmark to compare with any custom activity recognition models.
- **Battery and Connectivity Data:** Information about battery level and temperature, WiFi and mobile network details provide insights into device status, which can be crucial for real-world, continuous monitoring applications.
- **Positioning information:** The dataset gives information about the exact positioning of the device at each moment of the data collection. This can help to detect if the generated models fail at any particular position.

Limitations of the SHL dataset

- **Overwhelming Complexity:** The sheer breadth and granularity of data might be overkill for simple applications. Handling such vast data requires more processing power and advanced algorithms.
- **Potential Redundancies:** Some data types might overlap in the information they provide. For instance, accelerometer data combined with gyroscope and orientation might lead to redundant information in some scenarios.
- **Hardware Dependency:** the dataset only comes from a specific device, which can make the accuracy of the developed models very dependent on the device in use.
- **Limited Diversity in Participants:** The dataset comes from only 3 members, which significantly limits the diversity of the data. This might not represent a broader population, affecting the generalizability of any models trained on this dataset.

3.2.2 Transport Mode Detection Dataset (TMD)

The **TMD dataset** [13] was assembled by researchers at the **University of Bologna** who gathered sensor data from thirteen volunteer subjects, comprising ten males and three females (see Table 3.1). The primary aim of this dataset is to classify a range of activities, which include walking, driving a car, standing still, being on a train, and riding a bus. Altogether, the dataset consists of 226 labeled files. These files represent over 31 hours of data with breakdowns as follows: 26% of the data is designated as walking, 25% as driving a car, 24% as standing still, 20% as being on a train, and 5% as being on a bus (see Table 3.2).

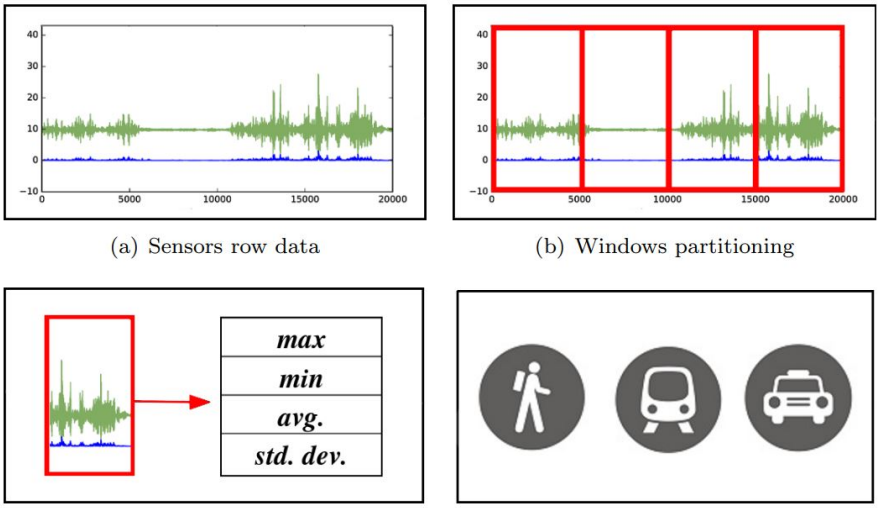
TABLE 3.1: Summary of User Data in the TMD dataset. [13]

ID	Sex	Age	Occupation	Device	Android Version
U1	M	30	student	LG G2	5.0.2
U2	F	27	student	Sony XPERIA Z3 Compact D5803	6.0.1
U3	M	30	student	Nexus 5	7.0
U4	M	36	office worker	Huawei Honor 5X	6.0.1
U5	M	36	stage director	Huawei P8 Lite	6.0.1
U6	M	27	researcher	Samsung galaxy s3 neo	4.4.2
U7	M	32	cameramen	Samsung S7	6.0.1
U8	F	32	bartender	Huawei Tag-l01	5.1
U9	F	24	student	Motorola Moto G	5.1
U10	M	22	student	Huawei P9	7.0
U11	F	31	office worker	Nexus 5	7.0
U12	M	31	researcher	Samsung Galaxy S6	6.0.1
U13	M	60	retired	Nexus 5	7.0

TABLE 3.2: Time durations for various activities in the TMD dataset. [13]

Bus	Car	Still	Train	Walking	Total
01:44:35	07:53:50	07:29:35	06:20:25	08:20:25	31:48:50

In the initial data **preprocessing** phase, the researchers undertook a series of data cleaning operations. These included the removal of measurements from non-pertinent sensors and ensuring the positivity of values from the sound and speed sensors, among other adjustments. It is noteworthy that some sensors, especially the ambient ones like sound, light, and pressure, as well as the proximity sensors, produced a single data value. This data was directly incorporated into the dataset. On the other hand, other sensors yielded multiple values because they were associated with a coordinate system, implying that their outputs were heavily influenced by orientation. For most of these, the team adopted an orientation-independent metric, termed magnitude. Following the data cleaning process, the dataset was segmented into time windows, each of either 5 seconds or half a second's duration. Subsequent to this division, four distinct features (min, max, dev.std, and mean) were extracted from every sensor. Figure 3.4 serves as a summary of the preprocessing steps.



(a) Sensors row data (b) Windows partitioning

(c) Features extraction (d) New data predictions

FIGURE 3.4: Preprocessing steps on the TMD dataset. [13]

Advantages of the TMD dataset

- **Diverse Data Collection:** The dataset includes sensor data from thirteen volunteer subjects, enhancing its representativeness. This diversity accounts for ten males and three females, providing a balanced gender distribution.
- **Thorough Preprocessing:** An initial data cleaning phase ensures the removal of unnecessary measures, the correction of sensor values, and the adaptation of certain sensor outputs. This results in a polished and easily usable dataset.
- **Accurate User Information**: the dataset explanatory paper provides precise information on the users and the devices used to collect the data. This can help to see if the models produced are able to generalize to different smartphone models.

Limitations of the TMD dataset

- **Fixed Time Window:** The data is segmented into specific time windows (either 5 seconds or half a second). This fixed windowing might not capture all nuances of certain activities or might oversimplify others.
- **Restricted Number of Features**: The dataset confines its scope to only four features extracted from each sensor. This limitation might omit potentially valuable information or insights that other features could provide.
- **Constrained Data Collection Duration:** While over 31 hours of data seems extensive, in the context of capturing diverse human activities and behaviors, this duration might be considered limited. It might not encompass the full spectrum of variabilities and patterns inherent to each activity.
- **Limited number of classes:** It would have been interesting if the dataset had included other transportation such as: run, scooter, bike, subway, etc.

3.2.3 Collecty Dataset

The **Collecty dataset** [19] offers a unique perspective into the transportation habits within Croatia, specifically in the City of Zagreb. The data has been accumulated using the mobile application "Collecty" on Android devices. Key sensors, including the accelerometer, gyroscope, and magnetometer, have been instrumental in data collection. Ensuring data privacy, the format remains raw and anonymized. The dataset boasts contributions from 15 participants across various age groups, spanning a data collection period of 5 months. These participants were tasked to activate the mobile application, which then recorded sensor data according to their mode of transportation. To ensure accuracy, upon reaching their destinations, participants validated their routes via the app's displayed digital map. Table 3.3 shows the trips made by each user present in the database.

TABLE 3.3: Distribution of data by transport mode per user expressed in hours for the Collecty dataset. [19]

User ID	Walk	Run	Bike	Car	Bus	Train	Tram	E-scooter	Total
17	7.92	0.00	0.00	6.24	0.00	2.84	0.00	0.14	17.13
23	10.76	0.06	0.18	0.54	22.73	0.35	5.05	0.00	39.68
24	0.29	0.00	0.00	1.78	0.00	0.00	0.00	0.00	2.06
25	0.71	0.00	0.00	9.47	0.00	0.00	0.75	0.00	10.92
29	29.88	0.00	0.00	4.60	0.92	39.32	0.00	0.18	74.9
37	0.00	0.00	0.00	28.13	0.00	0.00	0.00	0.00	28.13
39	1.27	0.00	0.00	1.47	0.82	3.78	0.00	0.00	7.35
40	6.22	0.02	0.00	0.27	5.66	0.00	2.27	0.00	14.43
18	6.59	3.31	9.41	15.8	0.00	0.00	0.00	5.98	41.09
20	0.02	0.00	0.00	0.00	0.00	0.00	0.14	0.00	0.15
26	0.03	0.00	0.00	0.12	0.00	0.00	0.00	0.00	0.14
27	1.87	0.00	0.00	0.83	0.00	0.78	0.00	0.00	3.47
28	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.04
31	0.16	0.00	0.00	0.29	0.39	0.00	0.00	0.00	0.85
35	0.09	0.00	0.00	2.07	0.00	0.00	0.00	0.00	2.16
Total	65.85	3.39	9.59	71.61	30.52	47.07	8.21	6.3	242.54

This dataset provide a a wide range of transports described in Figure 3.5. Tramway sees the least amount of use, tallying just a bit over 10 hours. In contrast, Walking and Car are the predominant methods of transport, with both nearing 70 hours. Bus travel is also a notable mode, accounting for approximately 30 hours. While Bike and E-scooter have diminished numbers, with Bike hours slightly surpassing 10 and E-scooter just falling short of that mark. Train travel, on the other hand, is quite prominent, registering a little more than 50 hours. The least frequent activity is Running, which is nearly absent on the representation.

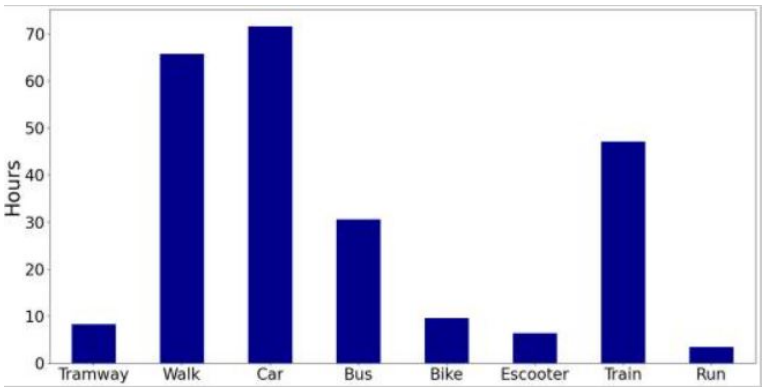


FIGURE 3.5: Collecty dataset distribution. [19]

Advantages of the Collecty dataset

- **Wide Range of Transports**: The dataset covers various modes of transportation, including tramways, walking, cars, buses, bikes, e-scooters, trains, and running. This offers a holistic view of transportation habits.
- **Detailed Duration Data:** The dataset provides exact hours of usage for each user and mode of transport, allowing for quantitative analysis and comparisons.
- **Anonymized Data:** Ensures privacy and confidentiality of participants while still providing valuable insights.

Limitations of the Collecty dataset

- **Geographical Limitation:** The data pertains only to transportation networks within Croatia, specifically the City of Zagreb, which could limit its generalizability.
- **Application Not Public:** The Collecty mobile application used for data collection is not publicly available, which could hinder replication or validation of the study by other researchers.
- **No Information on Devices:** The dataset does not provide information regarding the specific devices used for data collection, which could impact the accuracy and consistency of the sensor data.

In conclusion, the current landscape lacks a comprehensive public dataset capable of effectively training machine learning models for predicting transportation modes in urban settings. An ideal dataset for this purpose would encompass data from a broad spectrum of urban residents, encompassing variou transportation methods. Additionally, it is crucial to ensure meticulous control over data generation processes and other relevant parameters. Within the framework of the MobilitApp [2] and Mobilytics [3] projects, conducted in partnership with the ATM [1], there is a concerted effort to create and provide such a robust, public dataset dedicated to urban mobility data, serving both research and urban planning objectives.

Chapter 4

Our urban mobility dataset

This chapter is dedicated to describing the methodology behind the creation and refinement of the dataset that is central to this study. It starts with the deatiled exposition of the sensors employed. Then the data collection process is detailed. Subsequently, the preprocessing techniques are described. Furthermore, some data augmentation techniques are explained, with the intention of being applied to improve the dataset. Then *Feature Extraction* follows, detailing the transformation of preprocessed data into a structured feature set, including the *Outlier Detection* process. Finally, the chapter culminates with *Resultant Dataset* section, where the characteristics of the final dataset —ready for exploratory and confirmatory data analysis— are thoroughly described, setting the stage for the research covered by this project.

The subsequent content of this chapter is governed by a confidentiality agreement.

Chapter 5

Model experimentation and refinement

The *Experimentation* chapter serves as a detailed investigative work with the aim of exploring and analyze the performance of various computational models in the field. It begins with the definition of the framework, documenting the experimentation with different framework configurations, including variations in prepocessing. Once the basis of the experiments has been established, the next step in this chapter is to evaluate the performance of conventional machine learning algorithms, which is used as a baseline for further experiments. Then, different deep learning architectures are explored in conjunction with the attempted finetuning of a pretrained mdoel, trying to push the boundaries of classifying performance. Finally, the *Results Discussion* section synthesizes the experimental findings, providing a comprehensive overview of the outcomes and main conclusions of the experimentation.

5.1 Framework definition

In the context of this work, the framework refers to the set of methods, techniques, and algorithms used to carry out the research experiments. Therefore, in this section we will explain the framework taken as a template and present some experiments on variations of the preprocessing steps to determine the best configuration for the following experiments.

5.1.1 Starting point

This project does not start from scratch; in this case, we will start from a baseline developed by researchers and previous students of the SISCOM group [6][33]. This section will describe this basis, which will serve as a foundation for further experiments.

The main things to comment on about the framework used so far are the configuration of the sliding window, the overlapping factor, the separation between training and validation sets, the architecture of the model used, and some other preprocessing steps.

Initial preprocessing

The preprocessing steps carried out in the pre-project phase were the same as those described in **??**. However, it is important to emphasize the critical role of specific parameters in this process, particularly the size of the sliding window and the overlapping factor, both of which significantly influence the system's performance.

The baseline configuration for these parameters is detailed as follows:

- **Window size:** The window size is established at **200 timesteps**. This dimension dictates the volume of data the model processes at any given time, influencing the system's ability to interpret and learn from temporal patterns within the data.
- **Overlapping factor:** An overlapping factor of **50%** has been selected. This indicates that each data window shares half of its content with the succeeding window.

Initial sets separation

In any project involving machine learning models, it is imperative to first segregate the dataset into training, validation, and test sets before proceeding to model evaluation. This segmentation is fundamental to accurately evaluate the model's performance in real-world applications and to conduct a thorough analysis of potential overfitting.

For the baseline, the dataset was divided primarily through **random partitioning**. This approach was executed to ensure that the distribution of classes and users remained consistent across all three datasets - training, validation, and testing.

Initial model architecture

The architecture previously used up to the time of this project was an LSTM model. The particularities of these architectures are explained in subsection 2.4.5. In this case, the particular architecture consists of two LSTM (Long Short-Term Memory) layers interspersed with dropout layers, followed by a dense layer at the end. The **LSTM layers**, vary in their complexity and output dimensions, with the first layer preserving the sequence's length, using 200 units, and the second condensing it into a single vector, using 512 units. Dropout layers are employed to prevent overfitting, using a **20% dropout rate**, randomly deactivating a fraction of the neurons during training [54]. The final dense layer serves as the output layer, applying the **Softmax** activation. The diagram in Figure 5.1 shows the global architecture of the model.

This architecture will be trained using the following parameters for the baseline experiments:

- **Adam optimizer**

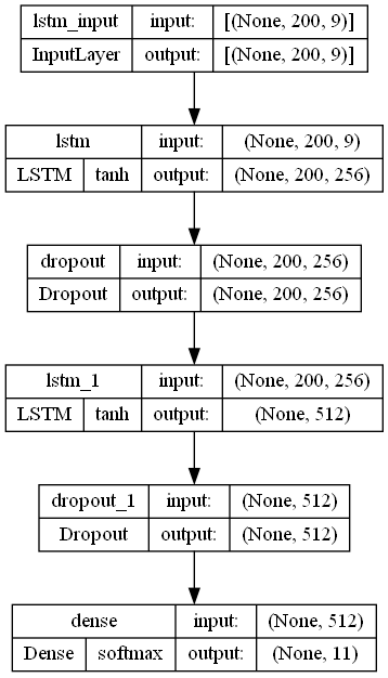


FIGURE 5.1: LSTM baseline arquitecture.

- **Learning rate: 0.001**
- **Batch size: 64**
- **Num. epochs: 70**
- **Loss function: Categorical crossentropy**
- **Random seed: 42**

The evaluation of each experiment was carried out by running the experiment in a 5-group fold and obtaining the average performance. The experiments were developed in a PC equipped with: at the core a 13th Generation Intel Core i9-13900F processor, complemented by 64 GB of physical memory, the graphics processing is managed by an Nvidia Geforce RTX 3060 Lite with 12GB of video memory and a 512 GB solid-state drive.

5.1.2 Effects of sets separation algorithm

In this first experiment, the effects of two types of dataset separation algorithms will be studied. The objective is to compare how the performance of the model varies using the classical separation algorithm, explained in section 5.1.1, and using a slightly more complex one.

The standard algorithm allocates data into training, validation, and test sets randomly, ensuring the distribution of classes and users remains consistent. The newer algorithm, however, places emphasis on populating the validation and test sets with users who are not present in the training set. This strategy aims to better simulate real-world scenarios where the model encounters users it has not previously seen.

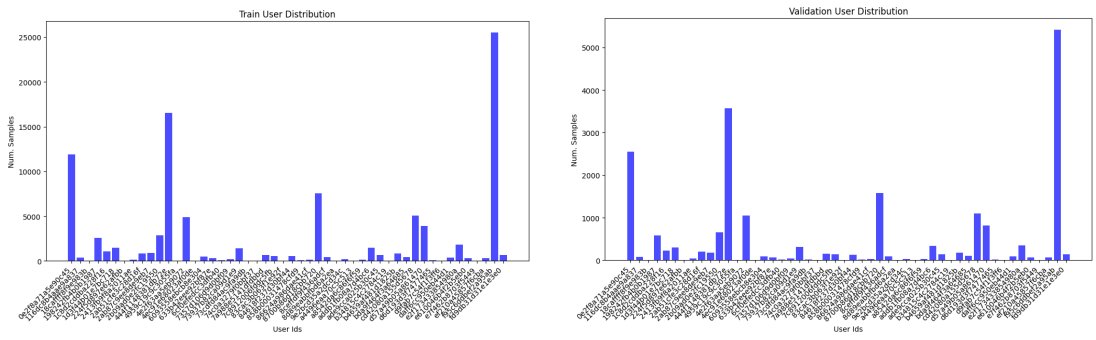


FIGURE 5.2: Training and validation users distribution after applying the random split configuration.

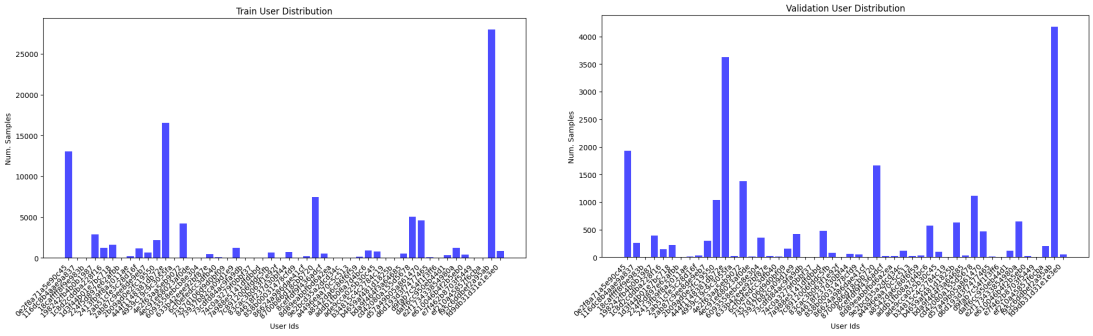


FIGURE 5.3: Training and validation user distribution after applying the different users split configuration.

In the comparative analysis between Figure 5.2 and Figure 5.3, a uniform distribution of users is observed across the training and validation sets in the first configuration, while the second configuration exhibits a distinct user distribution in the validation set. Furthermore, Table 5.1 elucidates the number of novel users by transportation mode. Despite efforts in dataset partitioning to achieve diversity, a perfectly balanced representation across all classes remains challenging due to the variability in user numbers per category. Nonetheless, this dataset splitting strategy more closely mirrors real-world conditions, potentially enhancing the robustness and generalizability of machine learning models.

For a comprehensive comparison, the configurations and model parameters outlined in previous sections are consistently applied to both algorithms. Consequently, Table 5.2 presents the average performance metrics derived from a 5-fold cross-validation for each configuration. Initially, the random separation approach may appear superior due to its significantly higher performance metrics. However, this enhanced performance is attributed to the model's overfitting to data from the same users. In contrast, when the model trained with this approach is evaluated on unseen data, there is a notable decrease in its f1-score, which drops to **74%**. Therefore, for this study, the more suitable algorithm is identified as the one used in the second configuration, known as the **different users separation algorithm**, due to its similarity to the real environment.

TABLE 5.1: Number of unseen users by transport in the validation set after applying the different users split configuration.

Transport	Unseen users
Bike	4
Bus	3
Car	3
Subway	2
Motorbike	1
Run	1
Stationary	7
Train	3
Tram	1
Walk	10
e-Scooter	2

TABLE 5.2: Average performance of the group 5-fold with each of the two separation algorithms. (M.avg. means Macro Average and W.avg. means Weighted Average)

Split conf.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
random	93	91	89	90	93	93	93
dif. users	79	71	67	67	79	79	78

5.1.3 Effects of window size and overlapping factor

After analyzing differences in the set separation method, the attention turns to analyzing the effects of various sliding window algorithm parameters. In order to determine how different **window sizes** and **overlapping factors** affect the deep learning model's performance, this section of the study specifically looks into these topics.

To carry out this comparison, the configuration already described in subsection 5.1.1 and the set separation algorithm stipulated in subsection 5.1.2 will be used. On the other hand, the parameters selected for evaluation in this experiment are as follows:

- Window size: 256, 512, 1024.
- Overlapping factor: 0.15, 0.3, 0.5, 0.7.

TABLE 5.3: Average performance of the group 5-fold with each of the configurations. (M.avg. means Macro Average and W.avg. means Weighted Average) (WS means Window Size and OF means Overlapping Factor.

WS - OF	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
256 - 0.15	77	67	64	64	77	77	76
256 - 0.3	80	70	69	68	81	80	79
256 - 0.5	79	71	67	67	79	79	78
256 - 0.7	79	68	67	66	81	79	78
512 - 0.15	70	66	67	66	66	67	69
512 - 0.3	72	68	66	66	73	72	72
512 0.5	**79**	**72**	**71**	**71**	**79**	**79**	**79**
512 - 0.7	78	71	71	71	78	78	78
1024 - 0.15	71	62	60	60	71	71	70
1024 - 0.3	71	63	62	62	71	71	72
1024 - 0.5	76	69	69	68	75	76	75
1024 - 0.7	75	70	70	67	75	76	75

Table 5.3 displays the average performance obtained with each of the configurations. The experiment reveals that window sizes exceeding 512 are not only unnecessary but also result in a lower F1-score across all tests. This phenomenon can be attributed to two primary reasons. First, a larger window size reduces the number of available samples, which can negatively impact the learning process. Second, to achieve satisfactory results with larger window sizes, there is a need for increasingly complex models, which may not be feasible or efficient in all scenarios.

Another key insight from this experiment is the benefit of a higher overlapping ratio in enhancing the model's ability to generalize. This improvement is likely because overlapping samples provide more contextual information, allowing the model to learn more robust features from the data. When samples overlap, the model gets exposed to more varied combinations of input data, which can aid in understanding the nuances and patterns that may not be evident in non-overlapping or less overlapping datasets.

In conclusion, the findings suggest that for optimal performance, it is crucial to strike a balance between window size and overlapping ratio. This balance ensures that the model has enough, but not excessive, data to learn from, and the data is varied enough to enable the model to generalize well to new, unseen examples. For these reasons, it has been considered that the best configuration for these data is a **window size of 512** and

an **overlap factor of 50%**. It is also the one that gives the best overall results. Therefore these values will be fixed for the following experiments.

5.1.4 Effects of data augmentation

In the present section, as outlined in **??**, an exploration is conducted to assess the potential of synthesizing data from the original dataset to enhance the model's generalization capabilities. This task involves the implementation of two distinct data augmentation algorithms, each differing in their level of intensity. Both variations employ the techniques of rotation and obfuscation, as previously delineated, with a consistent obfuscation rate of 25%. The key difference lies in the quantity of data generated through rotation, which varies to produce either a higher or lower sample count per class. A comparative analysis of these two data augmentation strategies, namely **DAbase** and **DAlarge**, alongside the distribution of the original dataset, is presented in Figure 5.5 and contrasted against the original distribution illustrated in Figure 5.4. Notice that these figures show the distribution in number of instances per class, after preprocessing the data using the sliding window technique and the overlap factor discussed in the previous section.

In more detail, following the aforementioned designed data augmentation strategy, we have performed two data augmentation options, namely DAbase and DAlarge, which are depicted in Figure 5.5. DAbase increases only the transport modes with least samples, while DAlarge increases all the transport modes to get aproximately the same amount of samples. The original distribution of samples for each transport mode is illustrated in Figure 5.4.

The primary objective of the **DAbase** method is to incrementally bolster underrepresented classes while preserving the integrity of the original dataset. In contrast, the **DAlarge** approach aims for a more radical balancing of the dataset by significantly augmenting underrepresented classes, resulting in a database predominantly composed of synthetic examples.

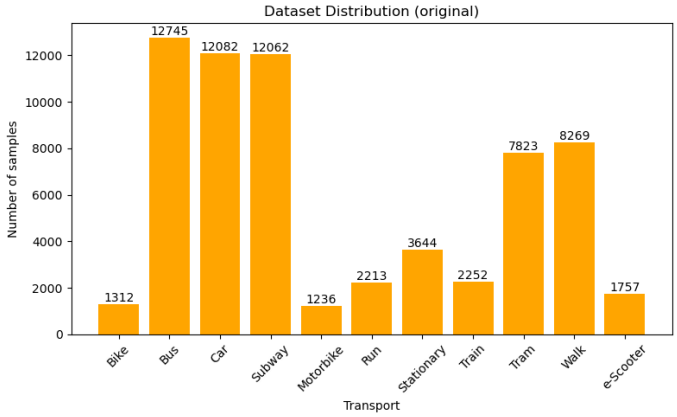


FIGURE 5.4: Original dataset distribution, without data augmentation.

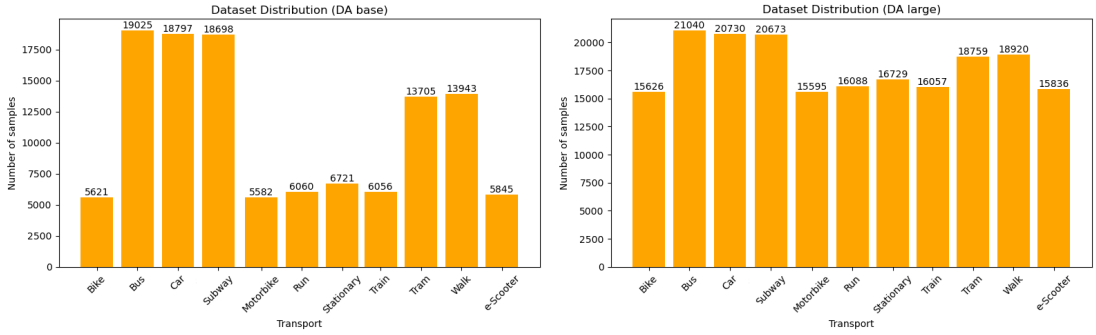


FIGURE 5.5: Class distribution in the dataset augmented using two variations of data augmentation techniques.

Experimental outcomes reveal that the more intensive **DAlarge** approach does not necessarily contribute to enhanced model performance; in fact, it often yields the contrary. This could be attributed to a potential reduction of data quality due to the inclusion of an extensive volume of synthetic data. Conversely, the **DAbase** configuration demonstrates results that are generally on par with those observed in the original setup, as depicted in Table 5.4. While this similarity offers no substantial insight at a glance, a more detailed examination of the model's performance across individual transportation modes, as elucidated in Table 5.5, indicates marginally improved outcomes for less represented classes such as bicycles and e-scooters. Given the critical role these transport modes play in evaluating the sustainability of urban mobility, the **base data augmentation** strategy emerges as the most suitable choice for this project.

TABLE 5.4: Average performance with each of the data augmentation versions. (M.avg. means Macro Average and W.avg. means Weighted Average)

Configuration	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
Original	79	72	71	71	79	79	79
DAbase	**81**	**71**	**69**	**70**	**80**	**81**	**80**
DAlarge	77	68	70	69	77	77	77

TABLE 5.5: Classification Reports using Original configuration (left) vs data augmentation base configuration (right).

Transport	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bike	0.35	0.37	0.36	191	0.38	0.37	0.38	191
Bus	0.79	0.78	0.78	1898	0.85	0.79	0.82	1898
Car	0.90	0.82	0.86	1808	0.85	0.95	0.90	1808
Motorbike	0.71	0.89	0.79	181	0.57	0.48	0.52	181
Run	0.79	0.98	0.87	337	0.99	0.93	0.96	337
Stationary	0.77	0.53	0.63	537	0.76	0.64	0.70	537
Subway	0.72	0.89	0.80	1849	0.80	0.81	0.80	1849
Train	0.53	0.29	0.37	332	0.25	0.11	0.16	332
Tram	0.83	0.90	0.87	1172	0.83	0.90	0.86	1172
Walk	0.93	0.81	0.87	1237	0.86	0.92	0.89	1237
e-Scooter	0.61	0.55	0.58	264	0.63	0.72	0.67	264
Accuracy			0.79	9806			0.81	9806
Macro avg	0.72	0.71	0.71	9806	0.71	0.69	0.70	9806
Weighted avg	0.79	0.79	0.79	9806	0.80	0.81	0.80	9806

5.1.5 Effects of smoothing technique

Other techniques that can significantly enhance system performance are smoothing methods. These methods are particularly effective in reducing noise when working with time-series data. Smoothing entails averaging data points with their neighboring values to diminish noise in such datasets. There are various noise-mitigation techniques, including kernel smoothing, weighted moving average, and simple moving average. In this section, we will delve into the comparative analysis of model results with and without the application of Gaussian smoothing.

The fundamental principle of **Gaussian smoothing** lies in its capability to filter out high-frequency components from sensor signals. These high-frequency components typically represent non-essential variations or environmental noise, as discussed in Friston (2000) [21]. Gaussian smoothing focuses on the more consistent, lower-frequency components of the data. This focus is instrumental in uncovering the underlying patterns crucial for accurately detecting transportation modes. Figure 5.6 illustrates an example of applying Gaussian smoothing to our dataset.

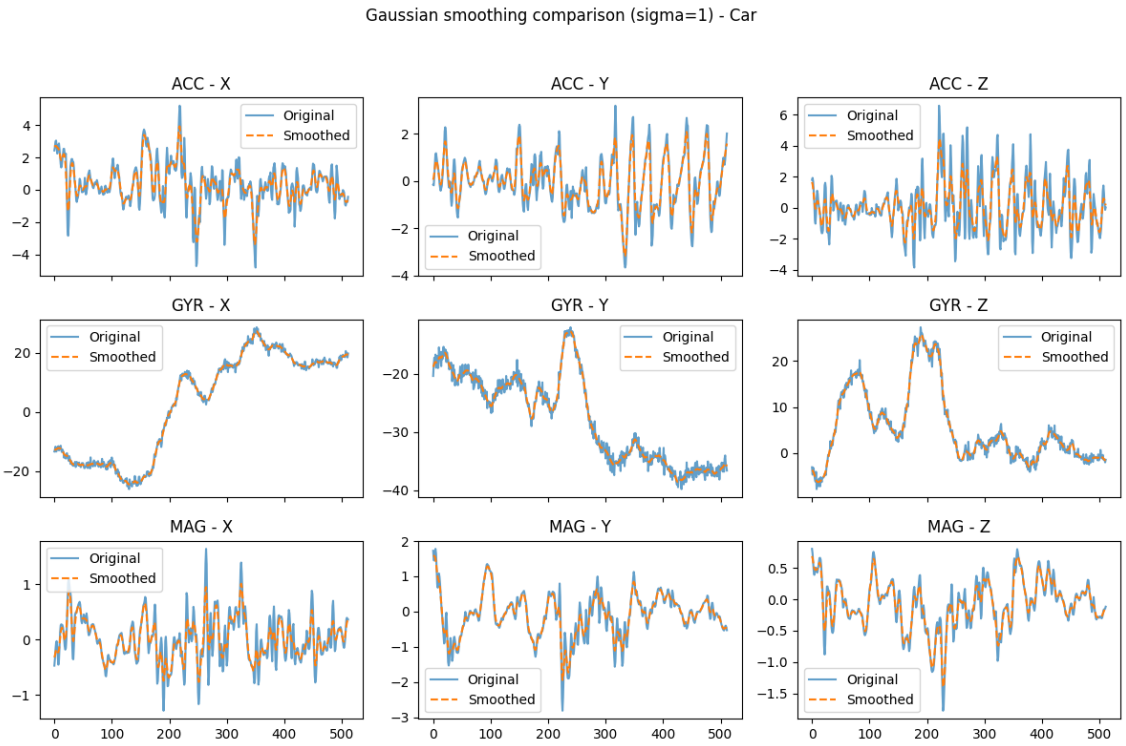


FIGURE 5.6: Visualization of the effects of Gaussian Smoothing technique with sigma=1, in a Car sample.

The parameter σ **(sigma)** in Gaussian smoothing is pivotal in determining the extent and effectiveness of the smoothing process. Sigma represents the standard deviation of the Gaussian kernel, which essentially controls the width of the Gaussian bell curve used in the smoothing process. This means that a small sigma value means less smoothing but might not effectively filter out noise, while a too-large sigma can overly smooth the data, potentially erasing important features or patterns in the data. For this reason, in this experiment, we compare the results with different sigma values, including 0.5, 1, 2.

TABLE 5.6: Average performance using different Gaussian smoothing parameters. (M.avg. means Macro Average and W.avg. means Weighted Average) (The first entry in the table shows the results without smoothing)

σ (sigma)	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
-	**81**	**71**	**69**	**70**	**80**	**81**	**80**
0.5	79	68	69	69	78	79	78
1	76	65	66	66	75	76	76
2	73	63	62	62	73	73	73

The results of these experiments shown in Table 5.6 clearly indicate that the applied technique does not help the model to obtain a more accurate classification. A notable observation is that an increase in the degree of smoothing correlates with a decline in performance. This pattern implies that smoothing the sensor data potentially impedes the model's ability to capture certain transport-specific characteristics inherent in the raw

data. Consequently, it has been resolved to exclude this technique from the final configuration.

5.1.6 Effects of outlier elimination

In this experiment, the outlier detection methodology outlined in **??** is employed. This approach involves using the **Mahalanobis distance** [42] in conjunction with a chi-square value, set at a 95% confidence interval, to identify outliers within the dataset. The application of this method has established a decision **threshold at 53.38**. Examples that exhibit a distance exceeding this threshold are categorized as outliers. As a result, **128 instances** within the dataset have been identified as outliers.

To validate the effectiveness of this outlier detection technique, an analysis of these outliers is conducted. Table 5.7 is compiled, detailing the user and label of each identified outlier, along with the total number of instances in the dataset that share the same user and label. This approach is helpful for verifying that the technique is precisely identifying specific anomalous instances, rather than erroneously classifying entire user groups as outliers.

TABLE 5.7: Outlier Analysis by User ID and Label

User ID	Label	Outliers	Total Samples
f95c6ddbef2a95ab	Run	9	1130
444f1148381db72e	Car	2	1749
444f1148381db72e	Run	10	1244
a8542da6c86f6f59	Subway	7	171
ade9cca025b30c45	Bike	29	3170
633b42eee92bf87e	e-Scooter	21	461
d57a49b05a861470	Walk	2	2079
b4659af481d1825b	Walk	9	712
198242fb4b0b1987	Bus	30	1133
224f5d897e62afbb	Bike	9	158

After confirming the correctness of the outlier detection process, the subsequent step involves evaluating its impact on the performance of the deep learning model. The provided Table 5.8 delineates the model's performance metrics after the application of outlier removal. Given the relatively small number of samples excluded, a minimal variance in performance between the original (Table 5.5) and revised models is anticipated. However, the data reveals a notable improvement in the model's performance, particularly for transport modes like bike and train. Conversely, the results for e-scooter and motorbike remain consistent, highlighting these as areas with potential for significant enhancement, e.g. starting by increasing the number of samples and users since those transport modes have a poor number of samples in the current dataset (see Figure 5.4). Consequently, the inclusion of this outlier detection technique in the preprocessing pipeline of the system has been considered advantageous.

TABLE 5.8: Classification Reports showing average performance of LSTM model trained after outliers detection

Class	Precision	Recall	F1-score	Support
Bike	0.44	0.46	0.45	195
Bus	0.80	0.79	0.80	1957
Car	0.85	0.92	0.88	1805
Motorbike	0.47	0.45	0.46	171
Run	0.98	0.92	0.95	357
Stationary	0.75	0.67	0.71	495
Subway	0.78	0.81	0.80	1781
Train	0.44	0.18	0.25	349
Tram	0.84	0.86	0.85	1152
Walk	0.87	0.91	0.89	1265
e-Scooter	0.66	0.68	0.67	276
Accuracy			0.80	9803
Macro avg	0.72	0.70	0.70	9803
Weighted avg	0.79	0.80	0.80	9803

5.1.7 Final configuration of the baseline model

As a summary of the previous sections, different configurations of the preprocessing steps have been tested. As a result, it has been decided to establish the following configuration as the basis of the project:

- **Separation algorithm:** different users separation.
- **Windows size:** 512
- **Overlap factor:** 50%
- **Data augmentation:** DAbase
- **Smoothing techniques:** no smoothing
- **Outliers detection technique:** Mahalanobis Distance

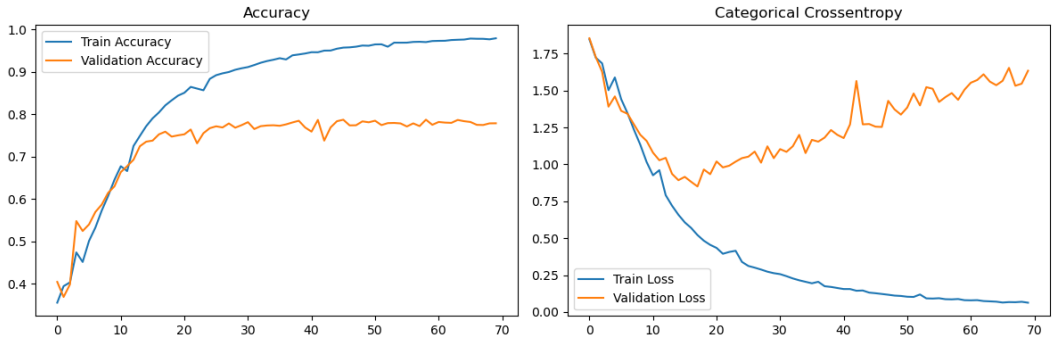


FIGURE 5.7: Learning curves throughout the different epochs of the model trained using the final configuration.

In this first experiment, the implemented configuration yielded an F-score of approximately **80%** and a macro average F1-score of **70%** . Detailed insights into these results are elaborated upon in preceding sections. Nonetheless, there exists substantial scope for improvement. The confusion matrix, as illustrated in Figure 5.8, reveals certain misclassifications, notably between transport modes like train and subway. An improvement strategy may involve alterations to the model architecture. For instance, employing residual networks could facilitate the extraction of features across varied scales. Additionally, alternative architectures such as Convolutional Neural Networks (CNNs) might prove more efficient in feature extraction. A further branch for improvement lies in model training. As depicted in Figure 5.7, the model exhibits early signs of overfitting. This suggests that the optimization parameters currently in use are suboptimal, indicating another potential area for refinement. The subsequent experiments will delve into these aspects, using the parameters established in the current study as foundational benchmarks.

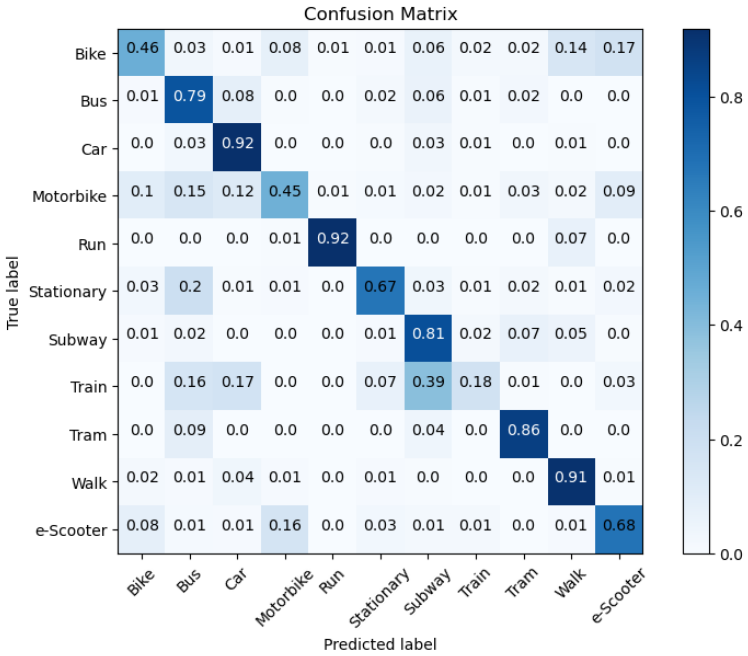


FIGURE 5.8: Confusion matrix of the model trained using the final configuration.

5.2 Traditional machine learning baselines

Once the experiment setup is set, the first study to be carried out is to test the effectiveness of machine learning models trained with the manually extracted features. For this, the process described in detail in **??** will be carried out. With the configuration of the dataset set in the previous section it has been obtained that the PCA manages to reduce the number of features from **99 to 38** making the number of features feasible for training using traditional architectures. On this basis, four different models have been tested: **RF**, **SVM**, **KNN** and **MLP**. For all of them, different combinations of hyper-parameters have been tested using **Random Search** [10], in order to obtain the maximum effectiveness.

All tested combinations are shown in Table 5.9. In addition, the parameters that result in the best performance for each model are also highlighted.

TABLE 5.9: Parameter settings for SVM, RF, KNN, and MLP random searches. Bold values indicate that this is the final value chosen for each parameter.

Model	**Parameter**	**Values**
SVM	C	0.1, 1, **10**
	kernel	**rbf**, poly
	degree	2, **4**
	gamma	**scale**, auto, 0.1, 1
RF	n_estimators	50, **100**, 200, 500
	max_features	auto, **sqrt**, log2
	max_depth	**None**, 15, 30, 50
	min_samples_split	2, **5**, 10
	min_samples_leaf	1, **2**, 3, 4
	bootstrap	True, **False**
KNN	n_neighbors	1, 2, 3, 4, **5**...31
	weights	**uniform**, distance
	metric	euclidean, **manhattan**, minkowski
MLP	n_neurons	[128, 64], [256, 128], **[512, 256]**, [512, 512]
	learning_rate	**0.001**, 0.01, 0.1

The data presented in Table 5.10 clearly demonstrates the efficacy of manually extracted features in transport prediction for this dataset. Notably, the computational resources required for training these models are significantly lower compared to the LSTM approach, making it a viable alternative for systems constrained by limited resources. However, as anticipated, the peak F1-score achieved with these models, at 76%, does not match the 80% attained by the previously discussed LSTM model. Despite this, it remains a valuable approach to assess the efforts invested in prior experiments. Moreover, these results serve as a useful baseline for comparative analysis. For additional information on the accuracy of the model in the different classes, the confusion matrix is shown in Figure 5.9.

TABLE 5.10: Average performance using traditional machine learning architectures. (M.avg. means Macro Average and W.avg. means Weighted Average)

Model	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
SVM	75	62	61	60	73	75	74
RF	76	64	62	61	75	76	75
KNN	74	61	60	60	73	74	73
MLP	**76**	**64**	**65**	**64**	**77**	**76**	**76**

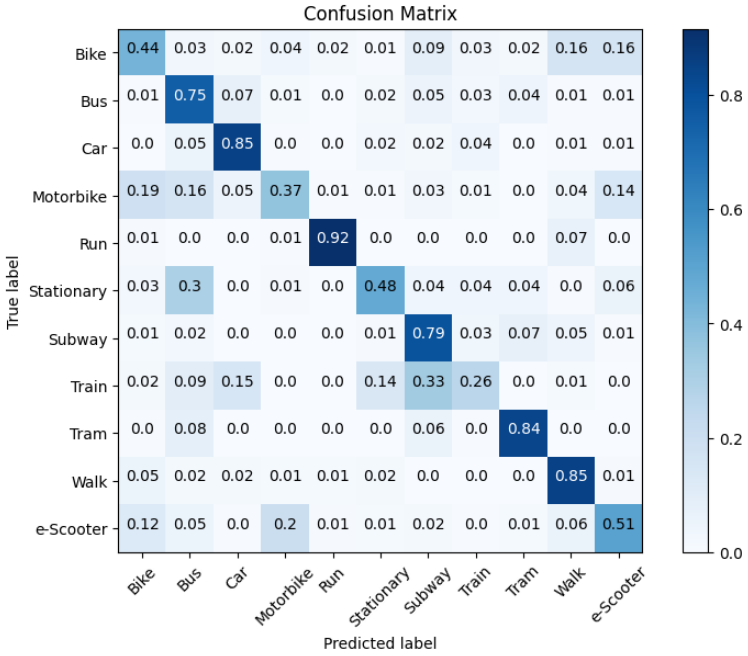


FIGURE 5.9: Confusion matrix of the presented MLP model.

5.3 BiLSTM

In this experiment, an enhancement of the LSTM model's performance is tried through the implementation of a Bidirectional LSTM (BiLSTM) architecture. As detailed in subsection 2.4.6, the BiLSTM, unlike its LSTM counterpart, processes data sequences bidirectionally, potentially offering a more comprehensive analysis of temporal data patterns. This research involves a meticulous examination of various training parameters and architectural modifications to optimize the model's efficacy.

The exploration of training methodologies includes the evaluation of different optimization algorithms, namely **RMSprop** and **SGD** [18]. Additionally, the investigation extends to assessing the impact of varying learning rate values on the model's training stability. Further, the study incorporates the **weight decay** approach and an adaptive **momentum** technique, though these modifications initially led to training destabilization. A notable breakthrough was achieved with the adaptation of the loss function, incorporating **class-weighting** for less-represented classes in the training set. This adjustment aims to enhance the model's sensitivity to these classes. Moreover, a **smoothing parameter** is integrated into the loss function to mitigate overfitting. The best results were obtained with the following configuration:

- **Optimizer:** Adam
- **Learning rate:** 0.001
- **Batch size:** 64
- **Num. epochs:** 70
- **Loss function:** Weighted Categorical Crossentropy

- **Smoothing parameter:** 0.1

Figure 5.10 shows the training curves throughout the different epoafter training the BiLSTM model with the specified parameters. As can be seen, compared to the curve shown in Figure 5.7, the new loss function is better adapted to the training data and therefore the training is better guided by it, showing a more stable training.

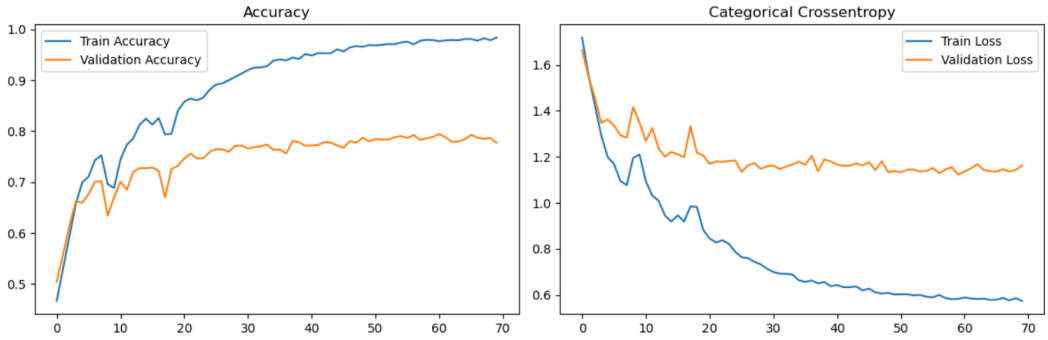


FIGURE 5.10: Learning curves throughout the different epochs of the BiLSTM model trained using the tuned optimization parameters.

On the model side, the experiment replicates the structure of the initial LSTM model but substitutes LSTM layers with BiLSTM layers. Furthermore, the study delves into determining the optimal size for the BiLSTM layers, tailored to the specifics of the dataset in use, thus seeking to maximize the model's performance. For this purpose, three sizing configurations for the BiLSTM layers have been tested: [128, 128], [128, 256], [256, 256]. The results of the experiments with the different sizes are shown in Table 5.11. Despite the efforts exerted in testing various hyper-parameter configurations, the best results of the experiments performed with this architecture do not surpass the previous best results. However, they are relatively similar in the case of the second configuration by using a model with **700k less parameters**.

TABLE 5.11: Average performance using different sizes for the BiLSTM layers. (M.avg. means Macro Average and W.avg. means Weighted Average)

Size	Params.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
[128, 128]	0.5M	78	67	70	68	79	78	78
[128, 256]	**1.1M**	**79**	**69**	**70**	**69**	**79**	**79**	**79**
[256, 256]	2.1M	78	68	69	68	77	78	77

5.4 Mixed model

This section explores a **mixed architecture model**, combining convolutional and recurrent layers, following the initial testing of models solely based on recurrent networks. For this purpose, we builded a base model inspired by the architecture proposed by Tang et Al. in [57]. Consequently, several experiments with variations of this architecture and its hyper-parameters will be performed.

The designed architecture is shown in Figure 5.11. The architecture depicted takes input from the three sensors. Each sensor feed passes through a series of **convolutional blocks** (Conv Block), which are likely composed of convolutional layers with batch normalization, dropout, and max pooling as indicated in the highlighted key for one Conv Block. The *N* parameter corresponds to the number of filters of the convolution and the *K* parameter corresponds to the kernel size. After feature extraction, the outputs of the convolutional blocks from each sensor channel are concatenated and fed into a LSTM layer. Finally there are a dense layer (also known as a fully connected layer), which processes the features learned by the **LSTM**, and a dense output layer with a **softmax** activation function which outputs the probabilities for each transport.

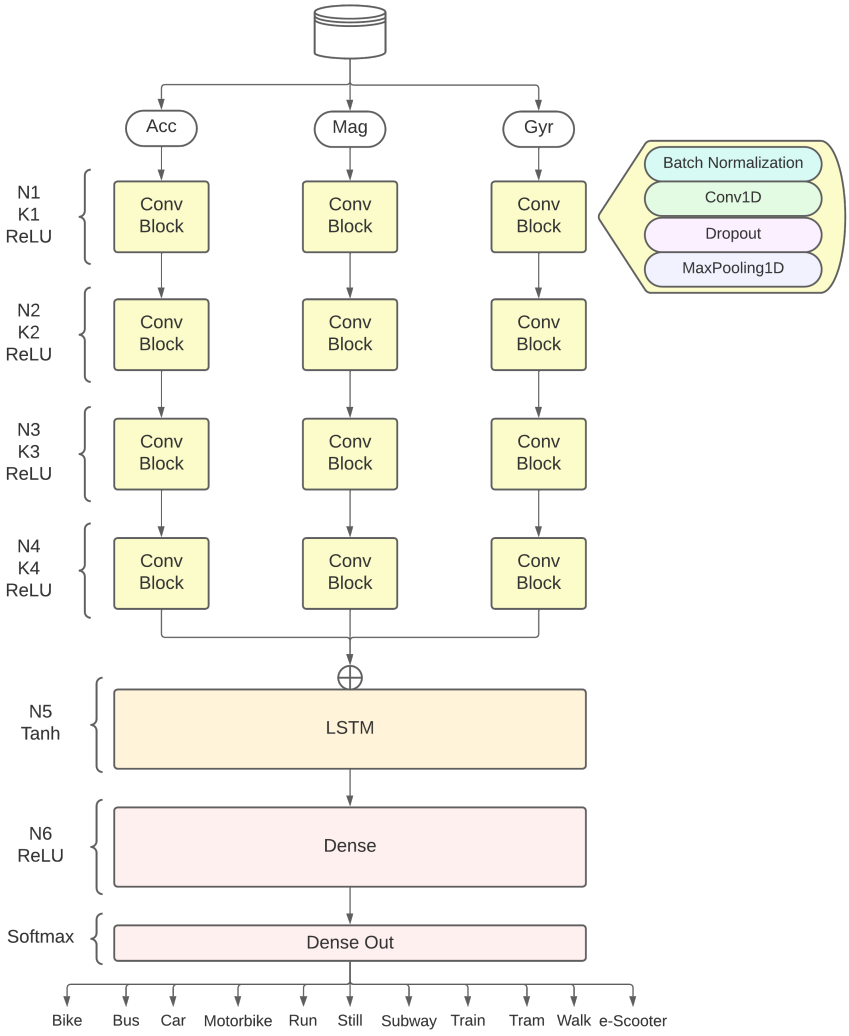


FIGURE 5.11: Base architecture diagram for the mixed model combining LSTM and CNNs.

The primary goal of this mixed architecture is to synergize the feature extraction capabilities of convolutional networks with the sequence processing strengths of recurrent networks. The training parameters selected for this study are based on those used in the

previous section, with modifications to optimize training for deeper models. Specifically, the learning rate has been reduced, and the number of epochs has been increased. The **training parameters** for the experiments in this section are outlined below:

- **Optimizer:** Adam
- **Learning Rate:** 1×10^{-4}
- **Batch Size:** 64
- **Number of Epochs:** 100
- **Loss Function:** Weighted Categorical Crossentropy
- **Smoothing Parameter:** 0.1

To develop the most effective classification model for this dataset, we conducted several experiments in the following sequence:

1. **Regularization:** Testing various configurations to prevent overfitting, including different Dropout rates, learning rates, and other regularization techniques.
2. **Convolutional Block Variations:** Experimenting with different layer configurations in the convolutional block.
3. **Activation Function Tuning:** Evaluating various activation functions, including ReLU, Leaky ReLU, and GeLU.
4. **Recurrent Block Variations:** Assessing different approaches in the recurrent layer, such as LSTM, BiLSTM, and a lightweight model without a recurrent layer.

5.4.1 Regularization

Regularization in Deep Learning is known as a technique used to prevent overfitting. Regularization works by adding additional information or constraints to the model to simplify it, making it less likely to capture the noise in the training data. The two techniques covered by this experiment are Dropout and L2 regularization. **Dropout** [54] randomly skip neurons during training, which helps in preventing over-reliance on any one node and encourages a distributed representation of features. A variation of the classic dropout technique is the **Spatial dropout** [38]. It addresses the structure aspects of CNNs by dropping out entire feature maps from the convolutional layers instead of individual neurons. This forces the network to maintain robustness not just at the neuron level but also at a higher level of abstraction On the other hand, **L2 regularization** [15] adds the squared value of the weights to the loss function. It encourages the model weights to be small, but not necessarily zero.

The experimental framework of this study involves varying dropout ratios and assessing the potential synergy of combining both regularization techniques. The conducted experiments include:

- Classical Dropout at dropout rates of 20% and 10%.
- Spatial Dropout at dropout rates of 10% and 5%.
- Combination of Spatial Dropout and L2 regularization.

Results, as detailed in Table 5.12, suggest that both Spatial Dropout and L2 regularization contribute to improved model generalization. Notably, dropout rates exceeding 20% not allows the model to learn fine grain data features. An analysis of the learning curves further elucidates these findings. The comparison of the learning curves reveals that a 20% dropout rate leads to stagnant loss oscillations, impeding learning. In contrast, lower spatial dropout rates exhibit more dynamic training progress and enhanced learning capabilities.

TABLE 5.12: Average performance with each of the regularization methods in the mixed model. (M.avg. means Macro Average and W.avg. means Weighted Average) (Dp refers to the dropout rate, SDp refers to the Spatial dropout rate and L2 refers to the L2 regularization parameter)

Configuration	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
Dp: 0	74	66	67	67	74	74	74
Dp: 0.2	70	60	63	60	71	70	69
Dp: 0.1	74	64	66	65	75	74	74
SDp: 0.1	75	66	66	66	75	75	75
SDp: 0.05	75	67	68	67	76	75	75
SDp: 0.05, L2: 0.001	**76**	**69**	**69**	**68**	**76**	**76**	**76**

A detailed assessment of the regularization techniques' effectiveness is facilitated through an examination of the learning curves. Comparative analysis of Figure 5.12 and Figure 5.13 reveals distinct outcomes for different dropout rates. Specifically, at a 20% dropout rate (Figure 5.12), the loss exhibits persistent oscillation within a narrow range, indicative of the model's inability to effectively learn from the training data. Conversely, Figure 5.13, representing a scenario with a reduced spatial dropout rate, demonstrates a more dynamic training process. Despite some instability, this lower dropout rate is associated with enhanced learning outcomes. The fluctuation in loss, although present, does not hinder the model's learning capacity to the same extent as observed in the higher dropout rate scenario. With all this in mind, the configuration chosen to continue with the following experiments is **5% spatial dropout**, combined with **L2 regularization in the fully connected layer**.

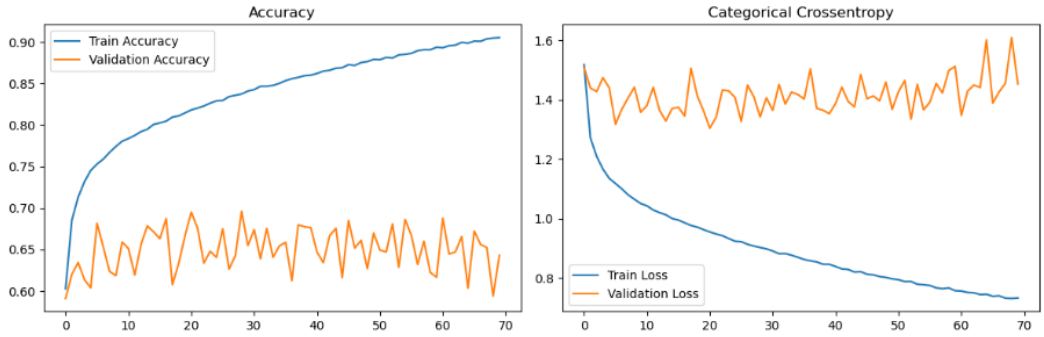


FIGURE 5.12: Learning curves throughout the different epochs of the mixed model trained using using a dropout rate of 20%.

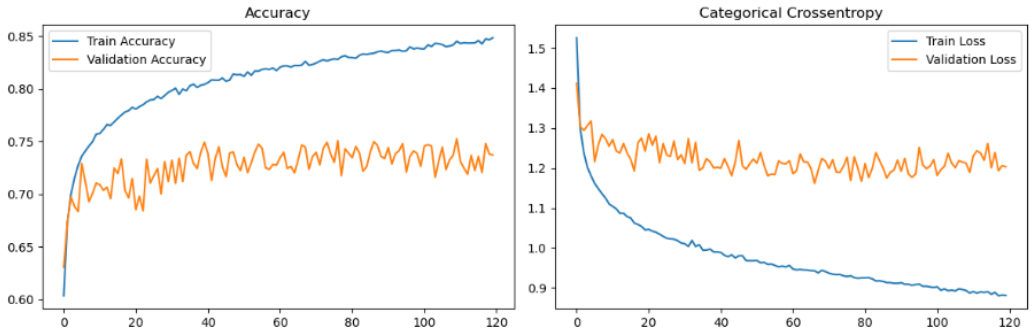


FIGURE 5.13: Learning curves throughout the different epochs of the model mixed trained using using a spatial dropout rate of 5%.

5.4.2 Convolutional Block Variations

The **convolutional block** is the most important part of the model for feature extraction. The achievement of these blocks should allow the model to extract the features from the data that allow the rest of the model to differentiate between the various modes of transport.

This section details the exploration of different configurations for convolutional blocks, focusing on layer arrangements and convolution layer parameters. The conducted experiments, aimed at optimizing feature extraction capabilities, are summarized as follows:

- A comparison of classical convolutions and **dilated convolutions**, with an increasing dilation rate correlating to block depth: 2, 4, 8, 8. Details on dilated convolutions are provided in subsection 2.4.3.
- Variations in **kernel size** for convolutions: [10, 7, 5, 5, 5] compared to [7, 5, 3, 3].
- Implementing **multiple convolutions** within each convolutional block.
- Investigating the optimal placement of the **batch normalization** layer, whether before or after the convolution process.

These experiments have been run sequentially, so that for each experiment the parameters of the previous best experiment are retained. As can be seen in the results table (see Table 5.13), the dilated convolutions are a great improvement in the capabilities of the model to extract features from a large stream of temporal data. In addition, the increased dilation ratio helps the model to extract features at different levels in each block of convolutions. In contrast, it has been observed that a higher number of convolutions staked in the same block does not help to improve the results. On the other hand, it has been detected that the best position for the batch normalization layer is before the convolution for this particular architecture.

TABLE 5.13: Average performance with each of the convolutional block configurations in the mixed model. (M.avg. means Macro Average and W.avg. means Weighted Average) (The difference between the second and the third experiment is that in the third experiment an extra convolution per block is added)

Configuration	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
dilations [10, 7, 5, 5]	78	67	68	67	78	78	78
dilations [7, 5, 3, 3]	**80**	**70**	**69**	**69**	**80**	**80**	**80**
dilations [7, 5, 3, 3] x2	80	68	67	67	78	80	79
dilations [7, 5, 3, 3], BN after conv	80	70	68	68	79	80	79

As a result, the best configuration extracted from this part of the study is the one using dilated convolutions with kernel sizes of 7, 5 and 3, and with the initial block's structure.

5.4.3 Activation Function Tuning

The **activation function** in an artificial neural network node is a mathematical function that determines the output of that node given a set of inputs and their corresponding weights. This function plays a crucial role in the network's ability to capture and represent complex patterns and relationships in the data. It introduces non-linearity into the model, enabling the network to learn and perform more than just simple linear transformations.

This is why this section has been devoted to testing variations of the activation function of the convolution layers and the fully connected layer. In this case it has been decided to experiment with ReLU, Leaky ReLU [63] and GeLU [30] as activation functions.

- **ReLU (Rectified Linear Unit)**: Defined as $f(x) = \max(0, x)$, ReLU is efficient, setting negative inputs to zero and keeping positive values unchanged. It aids in alleviating the vanishing gradient problem but can cause inactive neurons due to its zero output for negative values.
- **Leaky ReLU**: It modifies ReLU by allowing a small gradient when inactive, defined as $f(x) = x$ for $x > 0$ and $f(x) = \alpha x$ for $x \leq 0$, where α is a small constant. This prevents neurons from becoming inactive.
- **GELU (Gaussian Error Linear Unit)**: A smoother function defined as $f(x) = x\Phi(x)$, with $\Phi(x)$ being the Gaussian distribution's cumulative distribution function. GELU allows probabilistic gating of inputs, offering nuanced activation behavior and demonstrating efficacy especially in natural language processing.

TABLE 5.14: Average performance with each of the activation functions in the mixed model. (M.avg. means Macro Average and W.avg. means Weighted Average)

Act. Fun.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
ReLU	**80**	**70**	**69**	**69**	**80**	**80**	**80**
Leaky ReLU	79	69	69	68	79	79	79
GELU	78	67	68	67	78	78	78

As indicated in the results table (Table 5.14), the experiments did not yield enhanced performance beyond that achieved using **ReLU** as the activation function.

5.4.4 Recurrent Block Variations

In the concluding experiment of this study, variations for the model's final segment are explored. Specifically, the focus is on evaluating whether replacing the **LSTM** layer with a **BiLSTM** layer enhances the model's performance. Additionally, a more parameter-efficient variant of the model is proposed as a lightweight alternative. This variant substitutes the LSTM layer and the fully connected layer with a **Global Average Pooling** layer, thereby creating a purely CNN based architecture.

For this study, the number of neurons employed in the LSTM layer was set to 512, whereas in the BiLSTM layer, it was reduced to 256. This adjustment aimed to maintain the overall complexity of the model. As observed in Table 5.15, the performance of the lightweight model significantly trails that of the other models. Nevertheless, it stands as a viable option in scenarios where fine-grained transport detection is not critical, such as distinguishing only among vehicle, railway, and pedestrian modes. Additionally, the comparative analysis reveals negligible differences between models utilizing LSTM and BiLSTM. Consequently, due to the higher computational cost associated with training BiLSTM models, the **LSTM-based architecture has been selected as the preferred model**.

TABLE 5.15: Average performance with each of the different configurations in the top of the mixed model's architecture. (M.avg. means Macro Average and W.avg. means Weighted Average) (GAP means Global Average Pooling)

Configuration	Params.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
LSTM	**2.4M**	**80**	**70**	**69**	**69**	**80**	**80**	**80**
BiLSTM	2.2M	80	70	69	69	80	80	79
GAP	0.26M	76	65	67	65	78	76	77

As a conclusion of this experiment and all previously performed experiments, the final model is summarized below:

- **Number of filters**: N = [32, 64, 128, 128, 512, 256]
- **Kernel size**: K = [7, 5, 3, 3]
- **Activation function**: ReLU
- **Regularization**: Spatial dropout of 5% and L2 regularizer of 0.001
- **Architecture** shown in Figure 5.11
- **Dilated convolutions**

More details on the model evaluation are shown in Table 5.16, Figure 5.14 and Figure 5.15.

TABLE 5.16: Classification Report showing average performance of the mixed model

Class	Precision	Recall	F1-score	Support
Bike	0.38	0.46	0.41	195
Bus	0.78	0.86	0.82	1957
Car	0.84	0.91	0.88	1805
Motorbike	0.43	0.45	0.44	171
Run	0.93	0.92	0.93	357
Stationary	0.67	0.64	0.66	495
Subway	0.92	0.75	0.82	1781
Train	0.37	0.21	0.27	349
Tram	0.80	0.85	0.83	1152
Walk	0.86	0.87	0.87	1265
e-Scooter	0.67	0.71	0.69	276
Accuracy			0.80	9803
Macro avg	0.70	0.69	0.69	9803
Weighted avg	0.80	0.80	0.80	9803

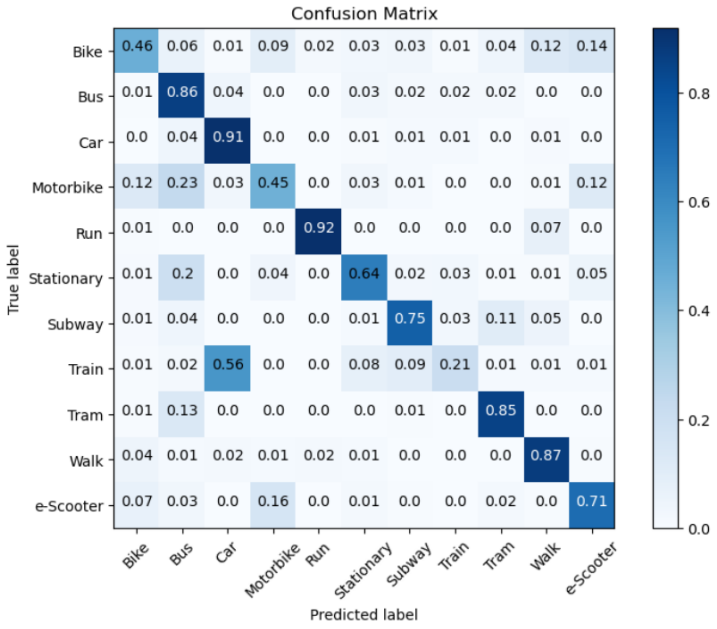


FIGURE 5.14: Confusion matrix of the presented mixed model.

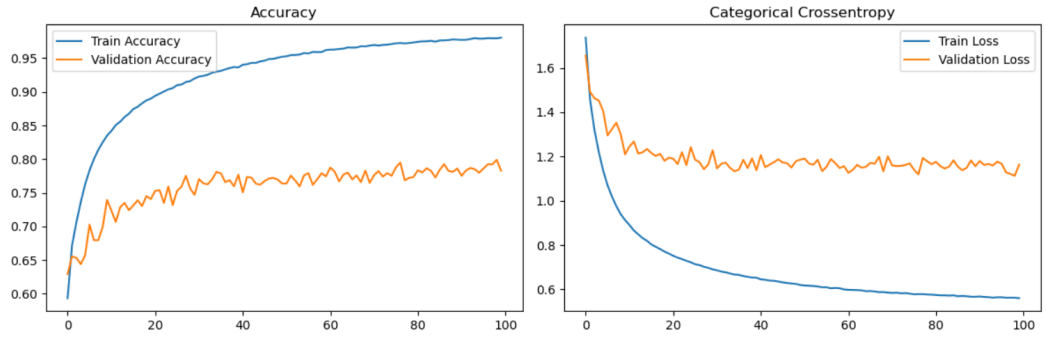


FIGURE 5.15: Learning curves throughout the different epochs of the mixed model trained using the final configuration.

5.5 Hierarchical model

The final iteration of the model architecture takes inspiration from the efficacy of the mixed model and advances it by introducing a **hierarchical design** as proposed in [56]. Retaining the convolutional framework of its predecessor, this model integrates **skip connections** that channel the feature maps directly into dedicated LSTM layers. Concretely, the outputs from the first, second, third, and fourth max-pooling stages are each fed into separate LSTM units. Such a design enables individual LSTMs to capture and analyze temporal patterns at various levels of feature abstraction, enriching the model's capacity to discern complex temporal relationships within the data. This architecture can be seen in a simpler form in the diagram in Figure 5.16.

Most of the variations approved in this architecture have already been evaluated in previous sections, so in this experiment we will simply test different layers and parameters in the recurring blocks:

- Apply **LSTM** with number of neurons 128 and 256 in each block
- Apply **BiLSTM** with number of neurons 128 and 256 in each block

Table 5.17 presents the average outcomes across four experimental variations, which yielded remarkably similar performance metrics. The experiments suggest that the incorporation of BiLSTM layers does not result in significant enhancements to the model's capabilities. Moreover, models with an excess of 5 million parameters do not surpass the performance of their less complex counterparts. When comparing the two LSTM-utilizing models, negligible differences are observed, although the larger model demonstrates a slight edge in classifying certain modes of transport, such as Cars and Trains. On the other hand, the smaller model exhibits superior generalization across various other transport categories, albeit with a noted deficiency in recognizing Trains. A more granular comparison of these two models, broken down by individual transport mode, is detailed in Table 5.18.

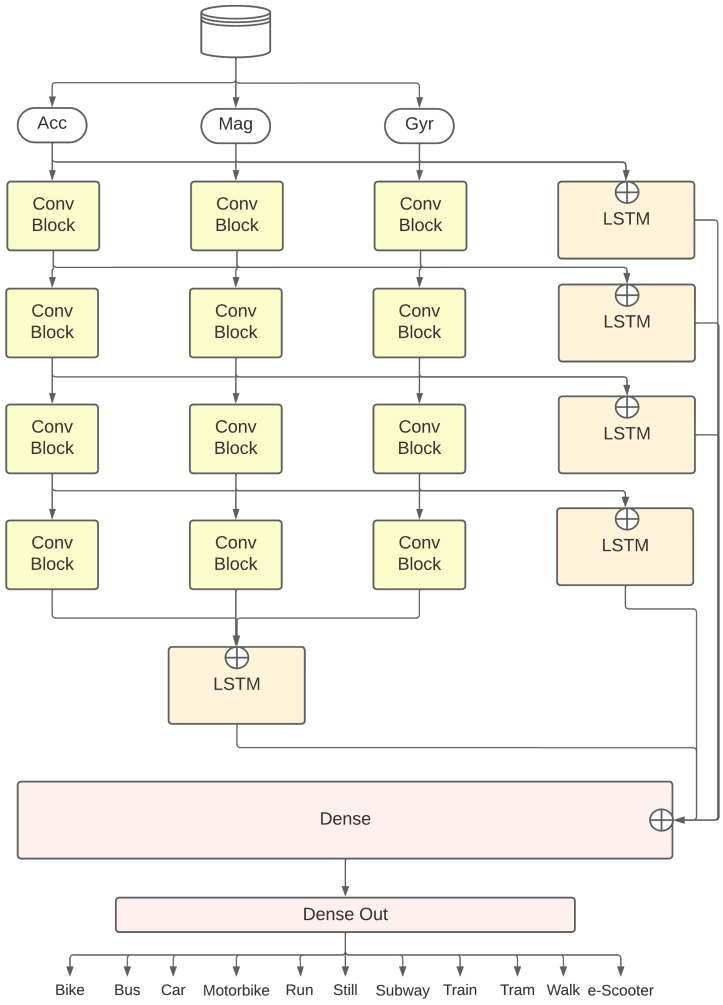


FIGURE 5.16: Base architecture diagram for the hierarchical model.

TABLE 5.17: Average performance with each of the different configurations in the recurrent blocks of the hierarchical model. (M.avg. means Macro Average and W.avg. means Weighted Average)

Configuration	Params.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
LSTM 128	1.3M	82	72	72	71	81	82	81
LSTM 256	**2.9M**	**82**	**72**	**72**	**72**	**82**	**82**	**82**
BiLSTM 128	2.3M	82	72	71	70	81	82	81
BiLSTM 256	5.9M	81	71	70	70	80	81	81

TABLE 5.18: Classification Reports using LSTM 256 (left) vs LSTM 128 (right) in the hierarchical model. The transports in which each model stands out are highlighted in bold.

Transport	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Bike	0.46	0.43	0.44	195	0.48	0.47	**0.48**	195
Bus	0.87	0.85	**0.86**	1957	0.84	0.85	0.84	1957
Car	0.88	0.95	**0.91**	1805	0.85	0.90	0.88	1805
Motorbike	0.49	0.50	**0.50**	171	0.52	0.47	0.49	171
Run	0.99	0.93	0.96	357	1.00	0.94	**0.97**	357
Stationary	0.68	0.66	**0.67**	495	0.60	0.74	0.66	495
Subway	0.86	0.76	0.81	1781	0.89	0.78	**0.83**	1781
Train	0.48	0.31	**0.38**	349	0.30	0.14	0.19	349
Tram	0.82	0.91	0.86	1152	0.86	0.89	**0.87**	1152
Walk	0.87	0.90	0.88	1265	0.86	0.93	**0.90**	1265
e-Scooter	0.56	0.74	0.64	276	0.67	0.78	**0.72**	276
Accuracy			0.82	9803			0.82	9803
Macro avg	0.72	0.72	0.72	9803	0.72	0.72	0.71	9803
Weighted avg	0.82	0.82	0.82	9803	0.81	0.82	0.81	9803

The selection of the optimal model from the final two candidates is contingent upon the specific application and the relative significance of each mode of transport within that context. For this study, the **LSTM model with 128 neurons** has been designated as the most suitable, primarily because it utilizes a significantly reduced number of parameters. This reduction in complexity renders the model more suitable for deployment on smartphones, aligning with one of the pivotal goals of the research. Consequently, the balance between model performance and computational efficiency guided the decision in favor of the more streamlined LSTM configuration. The full details on the evaluation and training of this model are shown in Figure 5.17 and Figure 5.18.

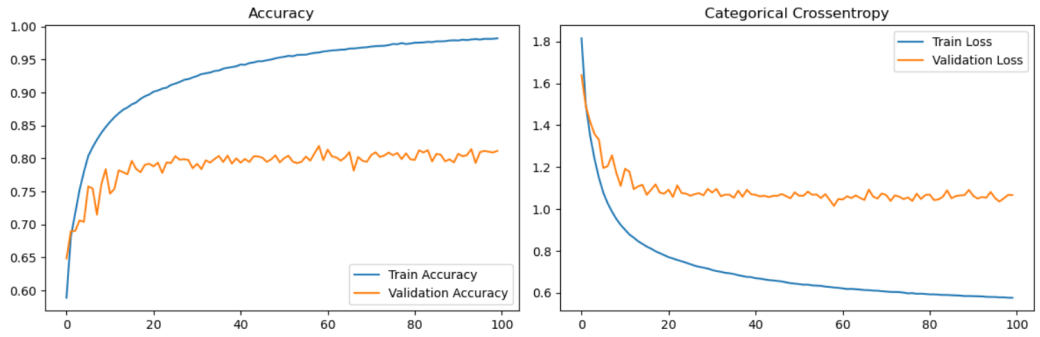


FIGURE 5.17: Learning curves throughout the different epochs of the hierarchical model trained using the final configuration.

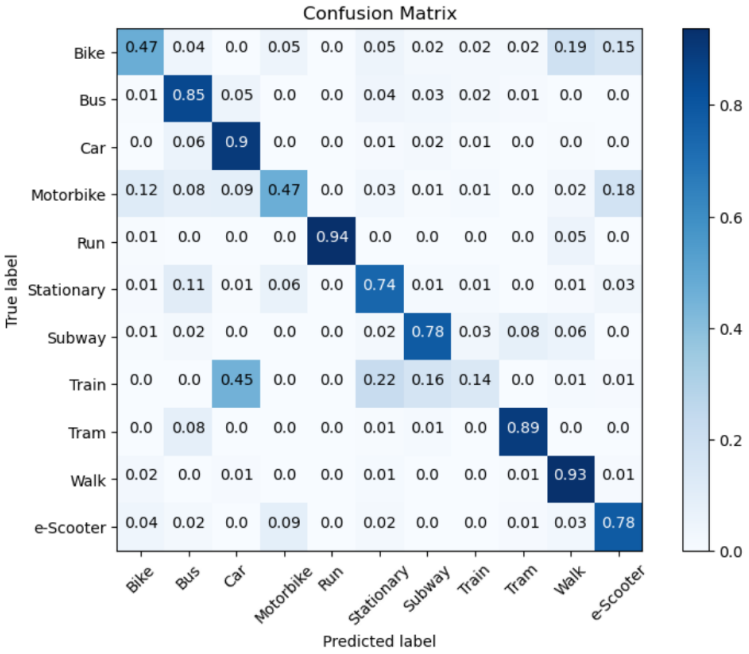


FIGURE 5.18: Confusion matrix of the presented hierarchical model.

5.6 Transfer learning

In the preceding sections, efforts were concentrated on enhancing the model's performance through refinements in its structure and hyperparameters. These modifications yielded notable improvements, elevating the F1-score from an initial 77% to an impressive 82%. Nevertheless, it has become evident that the margin for further improvement is diminishing, as demonstrated in the recent experiments. This plateau suggests that the model's performance is now primarily constrained by the quality of the data, particularly the under-representation of certain classes in the dataset. Consequently, this section will explore the application of **transfer learning** as a strategic approach to overcome these data-related limitations and potentially boost the model's performance.

In this context, **transfer learning** involves using a pre-trained model, which has been developed for one task, as a starting point for a similar task. This approach leverages the model's learned features, providing a foundation for the new task. **Fine-tuning** follows transfer learning, where the pre-trained model is further trained on the new task's data. This process involves making small, incremental adjustments to the model's parameters, allowing it to adapt more specifically to the new task [67].

In this research domain, unlike more established fields like computer vision, it is challenging to find models pre-trained for specific tasks. While some studies have explored this, publicly available models trained on relevant sensor data are rare. Moreover, comprehensive datasets for these tasks are generally not open to the public, though subsets of preliminary data are occasionally released. These circumstances have led to the decision to develop a model from scratch using the **SHL preview** dataset, as detailed in subsection 3.2.1. This model will then be fine-tuned using the mobilitapp dataset. Notice that the SHL preview dataset comprises data collected only by three users, all of them using

the same Huawei Mate 9 smartphone. Therefore, this would not be a good represetative dataset to be used as a starting dataset for a trandfer learning process followed by a fine tuning hase with the own dataset. This is precisely one of our aims in the MobilitApp [2] project in collaboration with the ATM, we aim to generate a large representative dataset of urban mobility data collected from a huge number of citizens in Barcelona. Thus, our MobilitApp public dataset could be used by other researchers and urban planners as a starting point to develope their particular predicte machine learning based models. Returning to the SHL dataset, the data distribution, depicted in Figure 5.19, reveals that the preview dataset, despite being preliminary, encompasses more hours of data for most transport modes compared to other datasets. However, it covers a smaller range of transport types.

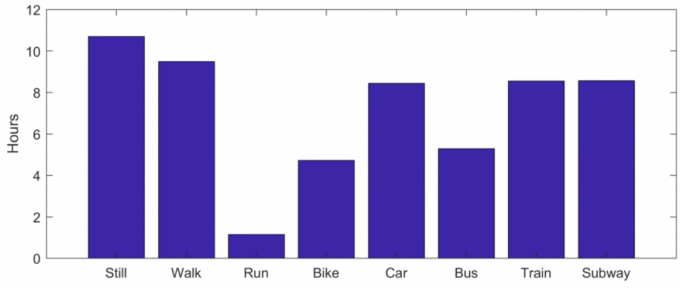


FIGURE 5.19: Distributions of the activities in the SHL Preview dataset. [60]

The methodology employed involved utilizing the model configuration established in the preceding section, the hierarchical model in Figure 5.16, maintaining identical parameters and training algorithms. For this specific iteration, a training duration of 25 epochs proved sufficient to yield notable results on the SHL preview dataset. The effectiveness of this approach is evident in the learning curves depicted in Figure 5.20 and the confusion matrix presented in Figure 5.21, where the model achieved an F1-score of **92%**. This outcome underscores the model's proficiency in adapting to and performing well on the dataset in question. Note, however, that this dataset is composed of data generated by only 3 users and all of them use the same smartphone model. Therefore, its representativeness and usefulness is limited.

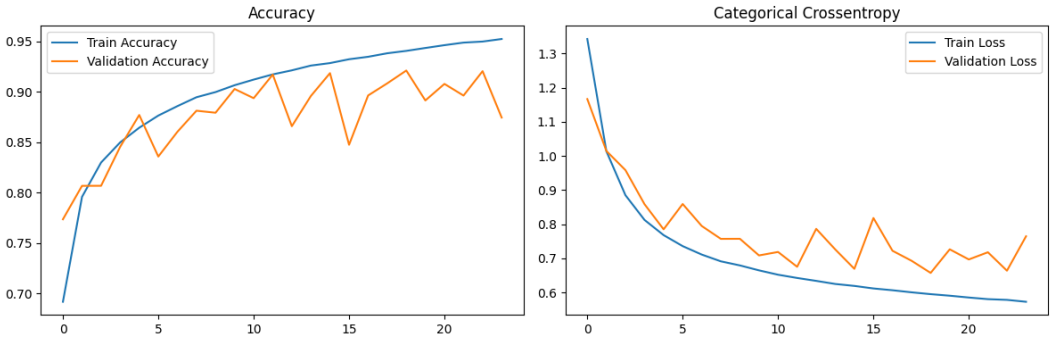


FIGURE 5.20: Learning curves throughout the different epochs of the hierarchical model trained in the SHL preview dataset.

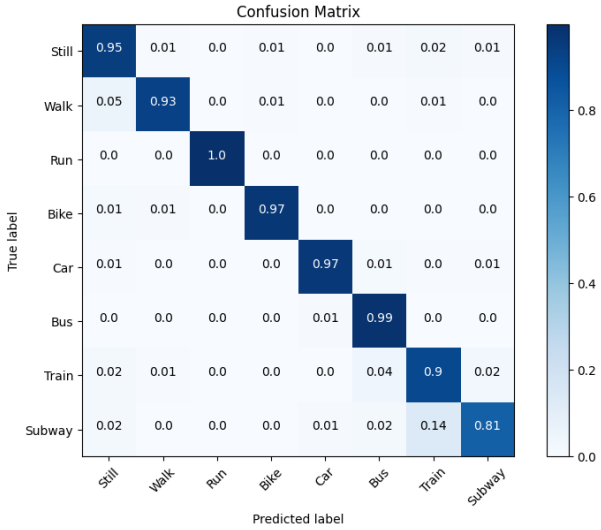


FIGURE 5.21: Confusion matrix of the hierarchical model trained in the SHL preview dataset.

Once the base model was obtained, it was decided to perform finetuning by **freezing the convolutional blocks** and reducing the learning rate to 0.5e-4. This is a very common practice in finetuning. This approach solidifies the feature extraction mechanism and eases the training process by maintaining certain initial weights of the model. However, no better results than those obtained previously have been achieved, with the F1-score stopped at **76%**. This may be due to the fact that the task carried out with the preview dataset is clearly easier to complete than the one proposed in this project, since it has fewer modes of transport, a reduced number of users and a single mobile device. Further details of the results of this experiment are shown in Figure 5.22 and Figure 5.23.

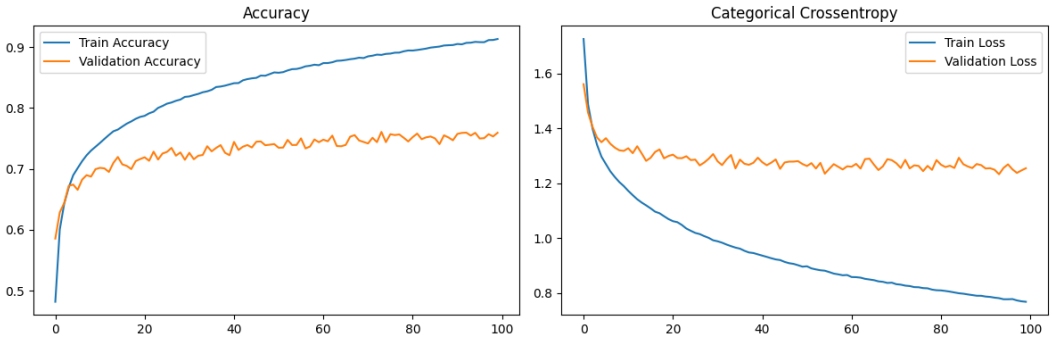


FIGURE 5.22: Learning curves throughout the different epochs of the hierarchical model trained in the SHL preview dataset and finetuned.

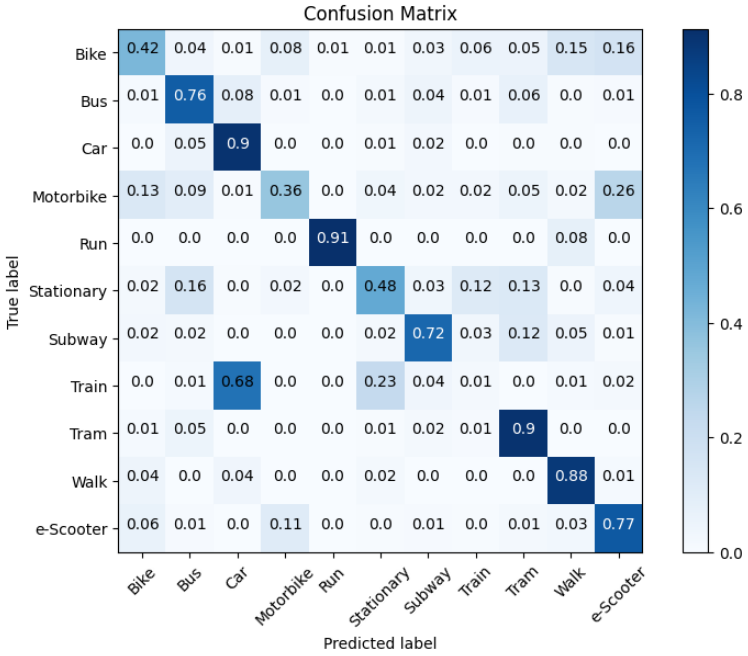


FIGURE 5.23: Confusion matrix of the hierarchical model trained in the SHL preview dataset and finetuned.

5.7 Results discussion

This section delves into a comparative analysis of four distinct models, evaluated through a robust methodological framework. The approach adopted for this comparison involved group-fold cross-validation with five folds, executed across three different random seeds. This procedure culminated in a total of 15 runs per model, ensuring a comprehensive assessment of each model's performance consistency and resilience to varying data splits. In particular, the models compared in this section are the two best models of the previous sections and the two baseline models: the **LSTM baseline** from subsection 5.1.7, the **MLP baseline** from section 5.2, the **mixed model** from subsection 5.4.4 and the **hierarchical model** from section 5.5.

To ascertain statistically significant differences in performance among the models, the **Mann-Whitney U test** [45] was utilized, as explained in subsection 2.5.4. This non-parametric test is particularly adequate for this analysis, given its suitability for comparing distributions without the need for normality assumptions. Each model's set of F1-scores across the 15 runs was compared against those of the other models, using the Mann-Whitney U test to determine if any observed differences in performance were statistically significant or merely the result of random variation.

To begin with the analysis, as a reminder of the previous sections, Table 5.19 shows a summary of the results obtained with the models mentioned above. However, as already mentioned, no empirical conclusions can be drawn from metrics alone. In contrast, Table 5.20 shows the statistical comparison between the hierarchical model, which seems to be the best according to the metrics, and all the other models. In this context, the **p-value**

corresponds to the probability that the observed results would occur under the null hypothesis, which in this case is the assumption that there is no difference between the two models. To interpret this value, **0.05 is normally used as a threshold**. Consequently, a value below this threshold indicates that the observed data is very unlikely under the null hypothesis. This leads to the rejection of the null hypothesis, suggesting that there is a statistically significant difference between the two models. As can be seen in the table all p-values are below this threshold which indicates that the difference in **the results of the hierarchical model is significant enough to confirm that this model offers better results overall for these data in this task**.

TABLE 5.19: Summary of the average performance with each of the LSTM baseline from subsection 5.1.7, the MLP baseline from section 5.2, the mixed model from subsection 5.4.4 and the hierarchical model from section 5.5. Results obtained from previous sections. (M.avg. means Macro Average and W.avg. means Weighted Average)

Model	Params.	F1-score	M.avg. P	M.avg. R	M.avg. F1	W.avg. P	W.avg. R	W.avg. F1
LSTM baseline	1.8M	80	72	70	70	79	80	80
MLP baseline	0.01M	76	64	65	64	77	76	76
Mixed model	2.4M	80	70	69	69	80	80	80
Hierarchical model	**1.3M**	**82**	**72**	**72**	**72**	**71**	**81**	**82**

TABLE 5.20: P-values resulting from comparing the highlighted models with the hierarchical model from section 5.5 using the Mann-Whitney U test [45]. Note that a value below 0.05 indicates a significant difference in the results.

Models	p-value
LSTM baseline - Hierarchical model	3.57e−5
MLP baseline - Hierarchical model	3.39e−6
Mixed model - Hierarchical model	4.2e−3

On the other hand, for a more illustrative view of the comparative models, Figure 5.24 is shown. Each box in the plot delineates the interquartile range (IQR) of the F1-scores, with the median value conspicuously marked by a central horizontal line. The "whiskers" of the plot extend to cover the full spread of the data, providing insight into the variability and reliability of each model. This graphical representation is crucial for understanding the comparative strengths and potential limitations of each model, and in this one it can be seen that for most cases the hierarchical model is superior to the others, which indicates that it is the one that offers the best performance.

Finally, to conclude the study, we wanted to show the performance of the hierarchical model applied to each of the sensors separately, as shown in Figure 5.25. It can be seen that each of the sensors separately is not able to obtain good results, which indicates that each of them is useful for transport mode classification since the combination of them results in a much higher performance.

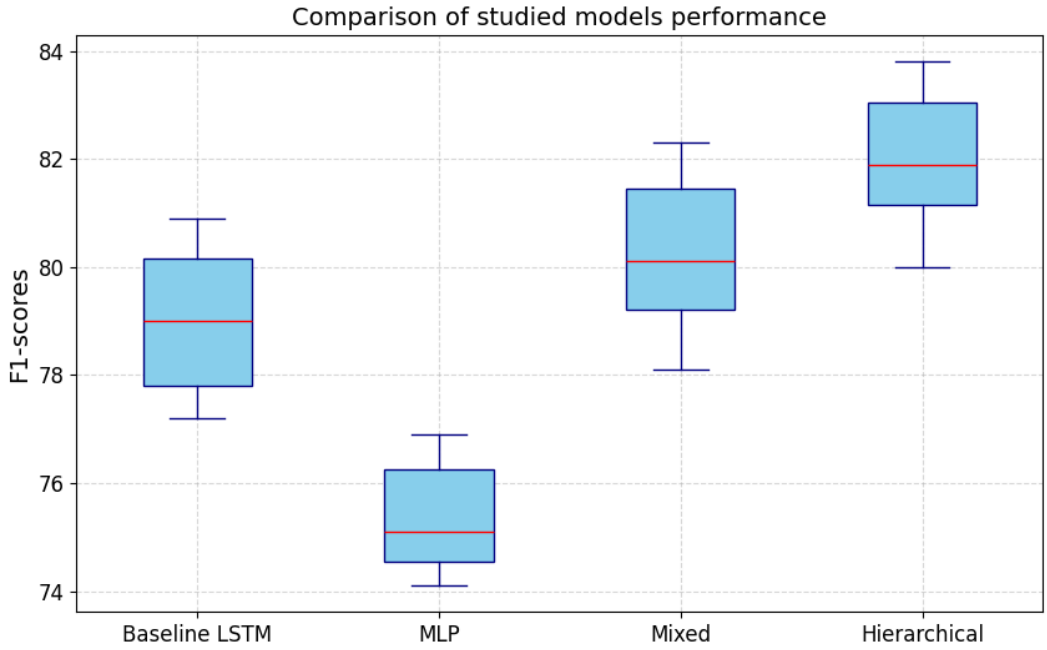


FIGURE 5.24: Box plot depicting the F1-scores of the LSTM baseline from subsection 5.1.7, the MLP baseline from section 5.2, the mixed model from subsection 5.4.4 and the hierarchical model from section 5.5. The central line in each box represents the median F1-score, the edges of the boxes indicate the interquartile range, and the whiskers extend to the full range of the data, excluding outliers.

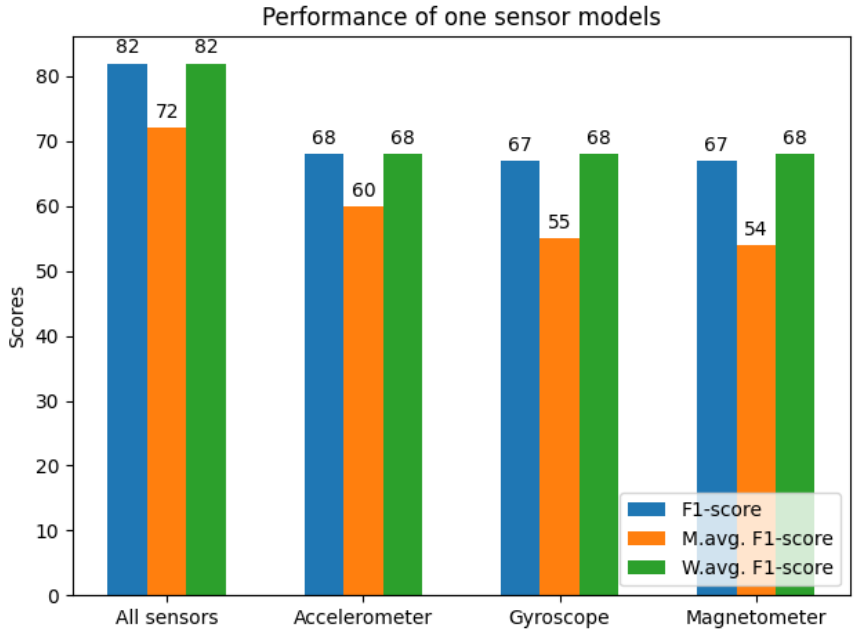


FIGURE 5.25: Bar chart comparing the performance of the hierarchical model from section 5.5 using only one sensor. (M.avg. means Macro Average and W.avg. means Weighted Average)

5.7.1 Final Hierarchical model evaluation

As a reminder from section 5.5, the hierarchical model combines convolutional networks with LSTM in a pyramidal structure that allows the processing of the extracted features at different levels using skip connections. The final configuration of the model is as follows:

- **LSTM size:** 128
- **Number of filters in CNN:** N = [32, 64, 128, 128, 512, 256]
- **Kernel size:** K = [7, 5, 3, 3]
- **Dilated convolutions**
- **Activation function:** ReLU
- **Regularization:** Spatial dropout of 5% and L2 regularizer of 0.001
- **Optimizer:** Adam with 1e-4 of learning rate
- **Loss Function:** Weighted Categorical Crossentropy with 0.1 of smoothing
- **Number of epochs:** 100
- **Batch size**: 64

After all the previous analysis the last thing to do is to evaluate the final model (hierarchical model) in the testing set. The obtained performance is depicted in Table 5.21 and Figure 5.26. As a summary, the difference between the validation and test results is small, indicating that the model has not been overfitted to the validation test.

TABLE 5.21: Classification Report of the hierarchical model from section 5.5 on the test set.

Class	Precision	Recall	F1-score	Support
Bike	0.59	0.40	0.48	198
Bus	0.84	0.80	0.82	1866
Car	0.84	0.91	0.87	1818
Motorbike	0.50	0.51	0.50	199
Run	0.98	0.93	0.96	305
Stationary	0.66	0.74	0.70	598
Subway	0.86	0.79	0.82	1835
Train	0.21	0.07	0.11	326
Tram	0.86	0.93	0.90	1194
Walk	0.86	0.94	0.90	1215
e-Scooter	0.57	0.80	0.67	251
Accuracy			0.81	9805
Macro avg	0.71	0.71	0.70	9805
Weighted avg	0.80	0.81	0.80	9805

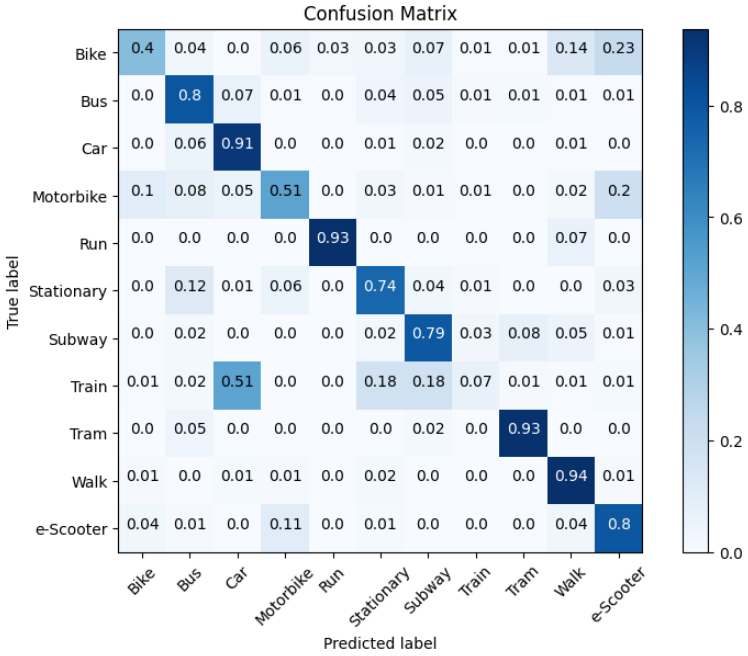


FIGURE 5.26: Confusion matrix of the hierarchical model from section 5.5 on the test set.

Chapter 6

MobilitApp tool for recognition of transportation modes

While the deep learning model constitutes the cornerstone for predicting transportation modes, its practical application necessitates integration within a comprehensive system. This system encompasses several critical components, including real-time data acquisition, preprocessing, the transport prediction, and feedback to the user. This chapter details the development and implementation of an Android application named **MobilitApp** [2], embodying an end-to-end system that seamlessly integrates data collection and model evaluation processes.

The subsequent content of this chapter is governed by a confidentiality agreement.

Chapter 7

Ethical and environmental concerns

In transport mode detection systems setting up a balance between technological advancement and ethical, environmental responsibility is a paramount concern. This chapter delves into the multifaceted ethical and environmental considerations surrounding the deployment and operation of these systems, establishing a guideline that ensure fair use, secure data storage, environmental stewardship, and sustainable practices.

7.1 Ethical considerations

This section examines the ethical considerations specific to the application of Artificial Intelligence (AI) in transport mode detection systems. Addressing these ethical aspects is crucial to ensure the responsible deployment and acceptance of such technologies in urban environments. The main aspects to take into account when talking about ethics applied to the study of artificial intelligence are: privacy, interpretability and security.

7.1.1 Privacy

As seen in the previous chapters, the main applications of transport mode detection systems involve monitoring the activities of humans during their trip. Since the manner of driving, walking, running, or even using a phone on public transport changes between users, it may be possible for an adversary to infer from triangulations between samples which sample belongs to which user.

Specifically, in the case of deep learning-based systems, the characteristics of these models pose a risk for the disclosure of sensitive user characteristics. Iwasawa et Al. in [32] investigated the privacy issue of using Convolutional Neural Networks as human activity recognition models. Their studies revealed that, although the CNN was initially trained using a cross-entropy loss focused solely on activity classification, the resulting CNN features unexpectedly demonstrated a significant capacity for user discrimination. Utilizing these CNN features, a straightforward logistic regressor was able to attain an impressive 84.7% accuracy in classifying users. This contrasts markedly with the meager 35.2% accuracy achieved by the same classifier when applied to raw sensor data. Consequently, it becomes crucial to consider and address the potential privacy risks inherent in deep learning models originally designed for this task.

There are multiple branches of study in the field of data privatization. And in this section, we will cover two of them: Transformations and Perturbation. In the case of **Transformation**. this technique involves using an adversarial loss function during the training process of a model to minimize the accuracy of identifying specific private information. This method was explored by researchers like Iwasawa et al. [32], who integrated adversarial loss with standard activity classification loss to reduce user identification accuracy. However, adversarial loss functions can complicate the end-to-end training process, often leading to unstable convergence.

On the other hand, **Perturbation** is an alternative strategy to data transformation for addressing privacy concerns in activity recognition. This method involves modifying data to balance privacy with recognition accuracy. Lyu et al. [41] introduced two data perturbation mechanisms: Random Projection and Repeated Gompertz, aiming to achieve an optimal balance between maintaining privacy and ensuring accurate recognition. **Random Projections** reduce data dimensionality by projecting it onto a lower-dimensional space using random matrices, maintaining distance relationships while enhancing privacy. In case of **Repeated Gompertz**, a mathematical model is applied, typically used for growth processes, to systematically alter data, obscuring specific features while preserving useful statistical properties.

7.1.2 Interpretability

Sensory data is unreadable for humans, specially in the case where more than one sensor is used in a time window. Given the differing importance of modalities and time intervals, interpreting neural networks is essential to understand the underlying factors influencing model decisions. For instance, in identifying a user's activity like walking, it is crucial to determine which specific modality and time interval are key determinants. As a result, enhancing the interpretability of deep learning methods is emerging as a significant trend within the transport mode recognition community. Some progress has been made in this field and there are several ways to represent model knowledge. We are presenting Feature Visualization and Attentive Selection.

Feature Visualization in the context of interpretable deep learning involves illustrating how neural networks prioritize and process different parts of input data. The core concept is to automatically discern the importance of each input segment, focusing on salient parts for improved accuracy while disregarding less important elements. Researchers have developed methods to visually represent these features, allowing for an understanding of how certain features correlate with specific activities. Additionally, techniques like those proposed by Nutter et al. [47] involve transforming sensory data into image formats, enabling the application of visualization tools for more straightforward interpretability of the data.

On the other hand, **Attentive Selection** applies neural attention mechanisms to deep learning models, enabling them to concentrate on a subset of inputs considered most relevant. Various studies have utilized the attention mechanism as a tool for interpreting the behaviors of deep models, providing insights into which aspects of the input data the models consider most critical for their decision-making processes [65].

7.1.3 Security

This section addresses the **ethical and responsible use** of AI-based transport mode detection systems, focusing on the importance of using these systems for **legitimate research** and urban planning purposes, rather than for invasive monitoring or tracking of individuals.

The primary use of transport mode detection systems should be for enhancing urban mobility, improving public transportation, and conducting transportation-related research. It is essential to clearly define and limit the purpose of these systems and communicate this to the public. Transparency in how data is used and for what purposes reinforces trust and mitigates concerns about surveillance or unauthorized tracking.

To ensure fair use, it is critical to collect only the data necessary for the specified purpose and to anonymize personal data. Adherence to legal and ethical standards is non-negotiable. This includes compliance with privacy laws and regulations, such as **GDPR** [5], which set strict guidelines on personal data usage. The system must also conform to ethical guidelines concerning digital surveillance and individual rights, ensuring that it does not infringe on personal freedoms.

In conclusion, ensuring the fair use of AI-based transport mode detection systems is a multifaceted task that requires a careful balance between technological capabilities and ethical considerations. By implementing these measures, the system can be used effectively for its intended research and urban planning purposes, while safeguarding against invasive monitoring and respecting individual privacy and rights.

7.2 Environmental impact

The integration of transportation mode detection systems into urban planning and mobility management can have significant environmental implications. This section explores the potential impacts and benefits of these systems.

Firstly, transportation mode detection systems can play a pivotal role in **encouraging sustainable transportation** choices among the populace. By providing detailed insights into individual and collective travel patterns, these systems enable users to assess their carbon footprint related to transportation. Such awareness can motivate individuals to opt for more eco-friendly modes of transport, such as cycling, walking, or public transit.

Furthermore, data collected through transportation mode detection can be instrumental in **designing more energy-efficient transportation systems**. By analyzing travel patterns and modal preferences, urban planners can optimize public transport routes, schedules, and resources, leading to a more efficient transport system.

Additionally, understanding human travel behavior and transportation choices is crucial in managing and **reducing urban traffic congestion**. Congestion not only has economic costs but also environmental impacts due to increased emissions from idling vehicles. Transportation mode detection systems can provide valuable insights for traffic prediction and scheduling, enabling more effective congestion management strategies. This, in turn, can lead to reduced greenhouse gas emissions and improved air quality in urban areas.

Finally, in the public transportation area, by identifying the most frequently used routes and modes of transport, these systems can help in **optimazing the utilization of public transport**.

Chapter 8

Conclusions and Future work

This thesis represents an in-depth exploration into the realm of transport mode and human activity recognition, offering a comprehensive overview of the urban mobility landscape in Barcelona. It traverses the spectrum from traditional methodologies to cutting-edge approaches, painting a vivid picture of the current state of the art. Throughout this journey, the capabilities of deep learning models in addressing these complex tasks were meticulously examined. This study not only delves into the practical applications of these models, but also critically assesses the challenges and ethical implications they present in the context of urban mobility and social and environmental impact. The investigation revealed several key findings that not only underscore the complexity of the task, but also chart a path forward for future research.

The research demonstrated that deep learning, offers a promising approach to accurately detect and differentiate between transport modes, particularly in handling temporal data from various sensors. The deployment of these techniques revealed the intricacies involved in managing and preprocessing sensory data, highlighting the necessity for **robust data handling protocols** to ensure quality results. Above all, it is necessary to take into account that the **distribution of users** between training and evaluation is different in order to evaluate the model in a similar situation to the one in which it is to be deployed. In addition, it has been shown that the combination of recurrent architectures such as **LSTM and convolutional** networks helps to obtain a good transport mode classification model using accelerometer, magnetometer and gyroscope as data sources. Finally, a more complex **hierarchical architecture** has been shown to help the model generalize and obtain better results.

A significant achievement of this study is the development of an Android application named **MobilitApp** [2] that combines deep learning with statistical methods, proving robustness to the transport mode recognition system. This tool, capable of generating multimodal trip summaries, represents a leap forward in understanding urban mobility patterns, offering valuable insights for urban planners, proving its utility in social science studies.

Notwithstanding these developments, the study encountered a number of obstacles proposed for future work. First, the requirement for comprehensive **testing** of the Android application and enhancement given the feedback of users. Secondly, the major problem encountered and the major limitation is the lack of **quality data** on some of the transports, i.e. Train, Motorbike, Bike and e-Bike. The team is therefore encouraged to collect more data on these transports. On the other hand, although promising, the investigation

into **transfer learning** was hampered by the absence of standardized, publicly accessible state-of-the-art datasets in this domain. Furthermore, significant thought must be given to the issues of **privacy** and the system's **interpretability**, particularly prior to the system and dataset being made publicly available. In addition, as regarding to scalability, consideration should be given to the possibility of developing a version of MobilitApp for iOS, as this is the second most widespread operating system for smartphones.

In reference to the study of deep learning architectures, the potential application of **Transformers**, and the innovative use of **Generative Adversarial Networks (GANs) for data augmentation** has opened new avenues for research. These strategies represent the ongoing development of deep learning in interpreting complicated transport data, as they seek to overcome existing limits and investigate novel ways.

In conclusion, this thesis adds to the body of knowledge already available in the area of transport mode recognition using deep learning and opens the door for future studies that will improve and expand upon these systems' capabilities. The potential impact of this research extends beyond the academic realm, offering practical solutions and insights that could shape the future of urban mobility in Barcelona. Showing this, the research of this thesis has opened a way to write a future paper together with ATM [1] and UPC Sostenible [4] explaining the MobilitApp [2] and analyzing the results of the multimodal card collection campaign in the UPC Community, during 2024.

To end with, the methodology developed in this thesis aims to help to contribute to have more sustainable urban mobility in Barcelona. Notice that this methodology could also be adapted for any other city. The **MobilitApp** [2] tool can help urban planners and public service providers to better know the urban mobility habits of citizens, to measure the impact of iniciatives to improve mobility and to improve the public transportation service.

Bibliography

[1] Collaboration agreement ATM - UPC (02/07/2021-02/07/2024): Cooperation between the "Autoritat del Transport Metropolità (ATM)" and the UPC to collaborate in several research projects. Project A-01358. Prediction of the transportation mode used by the citizens in their multimodal trips from the analysis of smartphone sensors, while ensuring privacy. 9 members. Prof. Mónica Aguilar Igartua (UPC) and Mr. Francesc Calvet (ATM). https://futur.upc.edu/35021523.

[2] MobilitApp: Tool to help analyze the mobility flow of citizens in the Metropolitan Area of Barcelona. https://mobilitapp.upc.edu.

[3] MOBILYTICS (01Dic2022 - 30Nov2024) project: Anonymization technology for AI-based analytics of mobility data. TED2021-129782B-I00. Proyectos Estratégicos Orientados a la Transición Ecológica y a la Transición Digital. Convocatoria 2021. Ministerio de Ciencia e Innovación. 211,945 euros, 12 members. Prof. Mónica Aguilar Igartua, Javier Parra Arnau. http://mobilytics.upc.edu.

[4] UPC Sostenible, Comunitat UPC sostenible. https://sostenible.upc.edu/ca.

[5] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016. Accessed: 2023-12-22.

[6] Adrian Catalin Diaconeasa. Treball de Final de Grau. Design of an algorithm for MObilitApp tool to automatically detect changes of activity of the citizens along the trip. https://upcommons.upc.edu/handle/2117/386155, 2022.

[7] META AI. Bidirectional long short-term memory (bilstm). https://paperswithcode.com/method/bilstm. Accessed on 08/10/2023.

[8] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[9] Abdulrahman Alruban, Hind Al-obaidi, Nathan Clarke, and Fudong Li. Physical activity recognition by utilising smartphone sensor signals. 02 2019.

[10] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, feb 2012.

[11] Jacopo Biancat, Chiara Brighenti, and Attilio Brighenti. Review of transportation mode detection techniques. *ICST Transactions on Ambient Systems*, 1:e7, 10 2014.

[12] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[13] Claudia Carpineti, Vincenzo Lomonaco, Luca Bedogni, Marco Felice, and Luciano Bononi. Custom dual transportation mode detection by smartphone devices exploiting sensor diversity. 10 2018.

[14] Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhiwen Yu, and Yunhao Liu. Deep learning for sensor-based human activity recognition: Overview, challenges and opportunities, 2021.

[15] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. L2 regularization for learning kernels, 2012.

[16] Ajuntament de Barcelona. Documentació i dades de mobilitat a barcelona. https://www.barcelona.cat/mobilitat/ca/actualitat-i-recursos/documentacio-i-dades, 2023. Accessed: 2023-10-19.

[17] Ayuntamiento de Barcelona. Las transformaciones que vivirá barcelona en 2023. https://www.barcelona.cat/mobilitat/es/actualidad-y-recursos/noticias/las-transformaciones-que-vivira-barcelona-en-2023-1242260, 2023. Accessed: 2023-10-20.

[18] Jarek Duda. SGD momentum optimizer with step estimation by online parabola model. *CoRR*, abs/1907.07063, 2019.

[19] Carić T Erdelić M, Erdelić T. Dataset for multimodal transport analytics of smartphone users - collecty. 2023.

[20] Shih-Hau Fang, Yu-Xaing Fei, Zhezhuang Xu, and Yu Tsao. Learning transportation modes from smartphone sensors based on deep neural network. *IEEE Sensors Journal*, PP:1–1, 08 2017.

[21] KJ Friston, O Josephs, E Zarahn, AP Holmes, S Rouquette, and J-B Poline. To smooth or not to smooth?: Bias and efficiency in fmri time-series analysis. *NeuroImage*, 12(2):196–208, 2000.

[22] Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[23] Aurelien Geron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 2nd edition, 2019.

[24] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.

[25] Hristijan Gjoreski, Jani Bizjak, Martin Gjoreski, and Matja Gams. Comparing deep and classical machine learning methods for human activity recognition using wrist accelerometer. 2016.

[26] Pablo González and Jesús Sancho. Radiografía de la movilidad en barcelona: así se desplazan los barceloneses. https://www.lavanguardia.com/local/barcelona/20180809/451250326790/radiografia-movilidad-barcelona-desplazamientos.html, 2018. Accessed: 2023-10-19.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[28] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998.

[29] Samuli Hemminki, Petteri Nurmi, and Sasu Tarkoma. Accelerometer-based transportation mode detection on smartphones. 11 2013.

[30] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016.

[31] IBM. Deep learning: what is and how it works, 2021. Accessed: 2023-10-04.

[32] Yusuke Iwasawa, Kotaro Nakayama, Ikuko Yairi, and Yutaka Matsuo. Privacy issues regarding the application of dnns to activity-recognition using wearables and its countermeasures by use of adversarial training. pages 1930–1936, 08 2017.

[33] Jaume Planas i Planas. Treball de Final de Màster. Improving the MobilitApp tool using deep learning models to automatically identify citizens activity including sustainable transport modes. https://upcommons.upc.edu/handle/2117/382385, 2021.

[34] J. Jeyakumar, Eun Sun Lee, Zhengxu Xia, Sandeep Singh Sandha, Nathan Tausik, and Mani B. Srivastava. Deep convolutional bidirectional lstm based transportation mode recognition. *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 2018.

[35] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *CoRR*, abs/1506.02078, 2015.

[36] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor.*, 12:74–82, 2011.

[37] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[38] Sanghun Lee and Chulhee Lee. Revisiting spatial dropout for regularizing convolutional neural networks. *Multimedia Tools and Applications*, 79:1–13, 12 2020.

[39] Xinyu Li, Yuan He, and Xiaojun Jing. A survey of deep learning-based human activity recognition in radar. *Remote. Sens.*, 11:1068, 2019.

[40] Huichao Liu, Ying Feng, and Liguo Zhang. Transportation mode identification based on smartphone. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 2015:5349–5354, 03 2015.

[41] Lingjuan Lyu, Xuanli He, Yee Wei Law, and Marimuthu Palaniswami. Privacy-preserving collaborative deep learning with application to human activity recognition. pages 1219–1228, 11 2017.

[42] G. Mclachlan. Mahalanobis distance. *Resonance*, 4:20–26, 06 1999.

[43] ABC Motor. Ciudades españolas con más atascos atascos, 2023. Accessed: 2023-10-19.

[44] Abdulmajid Murad and Jae-Young Pyun. Deep recurrent neural networks for human activity recognition. *Sensors*, 17:2556, 11 2017.

[45] Nadim Nachar. The Mann-Whitney U: A Test for Assessing Whether Two Independent Samples Come from the Same Distribution. *Tutorials in Quantitative Methods for Psychology*, 4, 03 2008.

[46] Ben Nham, Kanya Siangliulue, and Serena Yeung. Predicting mode of transport from iphone accelerometer data. 04 2012.

[47] Mark Nutter, Catherine H. Crawford, and Jorge Ortiz. Design of novel deep learning models for real-time human activity recognition with mobile phones. *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2018.

[48] Tsuyoshi Okita and Sozo Inoue. Recognition of multiple overlapping activities using compositional cnn-lstm model. pages 165–168, 09 2017.

[49] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E. *Scikit-learn: Machine Learning in Python*, 2011.

[50] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael Littman. Activity recognition from accelerometer data. volume 3, pages 1541–1546, 01 2005.

[51] Zabic M Rosenberg Randleff L., Bundgaard Wanscher J. Distributed travel mode estimation. 2012.

[52] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. The performance of lstm and bilstm in forecasting time series. In *2019 IEEE International conference on big data (Big Data)*, pages 3285–3292. IEEE, 2019.

[53] Xuan Song, Hiroshi Kanasugi, and Ryosuke Shibasaki. Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level. In *International Joint Conference on Artificial Intelligence*, 2016.

[54] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[55] Ralf Staudemeyer and Eric Morris. Understanding lstm – a tutorial into long short-term memory recurrent neural networks. 09 2019.

[56] Qinrui Tang and Hao Cheng. Feature pyramid bilstm: Using smartphone sensors for transportation mode detection, 2023.

[57] Qinrui Tang, Kanwal Jahan, and Michael Roth. Deep cnn-bilstm model for transportation mode detection using smartphone accelerometer and magnetometer. 06 2022.

[58] TomTom. Tomtom traffic index: Measuring congestion worldwide. https://www.tomtom.com/traffic-index/, 2023. Accessed: 2023-10-19.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[60] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Paula Lago, Kazuya Murao, Tsuyoshi Okita, and Daniel Roggen. Three-year review of the 2018–2020 SHL challenge on transportation and locomotion mode recognition from mobile sensors. 6 2020.

[61] Yanwen Wang and Yuanqing Zheng. Modeling rfid signal reflection for contact-free activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2:1 – 22, 2018.

[62] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[63] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical Evaluation of Rectified Activations in Convolutional Network, 2015.

[64] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions, 2016.

[65] Dalin Zhang, Lina Yao, Kaixuan Chen, Sen Wang, Pari Delir Haghighi, and Caley Sullivan. A graph-based hierarchical attention model for movement intention detection from eeg signals. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 27(11):2247–2253, 2019.

[66] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.

[67] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019.