POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

A Hybrid Model for Pedestrian Motion Prediction

Tesi di Laurea Magistrale in
Computer Science and Engineering

Author: **Enrico Dalboni**

Student ID: 968182
Advisor: Prof. Paolo Cremonesi
Co-advisors: Prof. dr. Mykola Pechenizkiy (TU/e), Manuel Muñoz Sánchez (TU/e), Dr. ir. Emilia Silvas (TNO), Robin Smit (TNO)
Academic Year: 2021-22

Abstract

Research on autonomous vehicles (AVs) is one of the most active in the automotive industry. To achieve greater levels of automation, AVs need to be able to safely operate in any conditions. Urban roads are challenging domains to model due to the high number of moving entities and the difficulty in predicting pedestrian motion. Different kinds of pedestrian prediction models exist, but it is unclear whether they have the potential to be used for real-time prediction in an AV.

A common practice for pedestrian trajectory prediction models is to evaluate the accuracy of the predictions offline on a standard sample of benchmark datasets, which data is collected on sidewalks and urban squares, and hardly represents their behavior in environments shared by pedestrians and vehicles. In this study, the types of pedestrian trajectories recorded on traditional datasets are compared to the ones recorded on more recent AV motion datasets. This analysis highlights the differences between the two kinds of datasets, especially in terms of the complexity of the trajectories, given by greater deviation from linear motion and speed variations.

Hybrid models for motion prediction are systems that combine models of different natures and they have recently been proposed as a potentially better alternative to methods of one kind. A hybrid model for robust pedestrian motion prediction is presented. By exploiting the benefits of the constant velocity model and a neural network, the hybrid model is able to predict trajectories in situations of partially occluded view or sensor failures.

The performance of the models on realistic, non-ideal data is often left behind. For example, on late detection or tracking failure. When using prediction models in real-time AV applications, their performance in non-ideal conditions is critical to ensure the safety of the AV and other traffic participants at all times. This is assessed by performing an offline robustness evaluation on different types of artificially altered data, simulating realistic prediction tasks.

Finally, the deployment of prediction models on a vehicle and their usage in real-time have many added challenges. To understand the potential suitability of these models for real-time prediction, a framework for real-time in-vehicle prediction is presented and deployed on a test vehicle.

Keywords: av motion datasets, pedestrian trajectory prediction, hybrid model, real-time prediction, robust prediction

Abstract in lingua italiana

La ricerca sui veicoli autonomi (AV) è una delle più attive nel settore dell'automotive. Per raggiungere più alti livelli di automazione, i veicoli autonomi devono essere in grado di operare in sicurezza in qualsiasi condizione. Le strade urbane sono domini difficili da modellare a causa dell'elevato numero di entità in movimento e della difficoltà di prevedere il movimento dei pedoni. Esistono diversi tipi di modelli di previsione, ma non è chiaro se abbiano il potenziale per essere utilizzati in tempo reale in un AV.

Una pratica comune per i modelli di previsione delle traiettorie dei pedoni è quella di valutare l'accuratezza delle previsioni in maniera offline su un campione standard di dataset, i cui dati sono raccolti su marciapiedi e piazze urbane, difficilmente rappresentanti il comportamento dei pedoni in ambienti condivisi con veicoli. In questo studio, i tipi di traiettorie dei pedoni presenti sui dataset tradizionali sono confrontati con quelli presenti sui più recenti AV motion dataset. L'analisi evidenzia le differenze tra i due tipi di dataset, soprattutto in termini di complessità delle traiettorie, data da una maggiore deviazione dal moto lineare e dalle variazioni di velocità.

I modelli ibridi per la previsione delle traiettorie sono sistemi che combinano modelli di natura diversa e sono stati recentemente proposti come alternativa potenzialmente migliore ai metodi di un solo tipo. Viene presentato un modello ibrido per la previsione robusta delle traiettorie dei pedoni. Sfruttando i vantaggi del modello a velocità costante e di una rete neurale, il modello ibrido è in grado di prevedere le traiettorie in situazioni di vista parzialmente occlusa o di guasti ai sensori.

Le prestazioni dei modelli su dati realistici e non ideali vengono spesso trascurate. Ad esempio, in caso di rilevamento tardivo o di errore di tracciamento. Quando si utilizzano modelli di previsione in applicazioni AV in tempo reale, le loro prestazioni in condizioni non ideali sono fondamentali per garantire la sicurezza dell'AV e degli altri utenti della strada in ogni momento. Questo aspetto viene valutato eseguendo una valutazione della robustezza offline su diversi tipi di dati artificialmente alterati, simulando situazioni real-

istiche.

Infine, l'implementazione di modelli di previsione su un veicolo e il loro utilizzo in tempo reale comportano numerose sfide aggiuntive. Per comprendere la potenziale idoneità di questi modelli per la previsione in tempo reale, viene presentato un framework per la previsione in tempo reale a bordo di un veicolo e viene implementato su un veicolo di test.

Parole chiave: av motion dataset, previsione delle traiettorie dei pedoni, modello ibrido, previsione in tempo reale, previsione robusta

Contents

1 | Introduction

1.1. Motivation

Attention to safety in the automotive industry is extremely important so that fewer accidents occur and those that do occur are of low severity. Over the past few decades, many systems have been implemented in vehicles and infrastructures to increase their safety levels. Crash tests have become the norm in new car design [14], and airbags have made crashes less dangerous [28]. Innovations in the development of safety systems have come with the arrival of electronics and control systems. Nowadays, the latest frontier in road safety is advanced driver-assistance systems (ADAS) [33]. All these innovations have succeeded over the years in constantly lowering the number of deaths in road accidents. However, as the European Commission confirms, the curve of fatal accidents per year has stabilized since 2012 to around 25,000 deaths per year [11]. This means that the sophisticated safety devices we have at our disposal today can only be effective up to a certain point. Human error remains present and is the main cause of accidents [40]. Autonomous vehicles (AVs) have the potential to remove this factor from the accident equation and lower the number of road casualties.

SAE International is a professional association and standards developing organization for engineering professionals [43]. In 2014 it published a classification system for AVs with six levels of automation, ranging from 0 to 5 [12] and in 2021 the first level 3 car was launched on the market by Honda [1]. However, to reach levels 4 and 5, cars need to be able to handle any situation that can be managed by a human driver without any input from the passengers. For this, they need to behave with a high degree of reliability and make the right decisions also on previously unseen data, or in possibly dangerous situations.

Predicting pedestrian trajectories is a task of extreme importance. In 2019 alone, around 4,700 pedestrians were killed in road accidents in the EU only [2]. AVs have the potential of reducing this number, as well as decrease the number of road accidents in general. To

make the transition to AVs faster and safer, reliable and accurate trajectory prediction systems are required.

The trajectory prediction approaches available today are of different kinds and have different advantages and disadvantages. Physics-based models were the first to be developed, widely supported in the literature, and can generally deliver robust and reliable predictions, while their performance peaks in the short-term horizon. Recent progress in the research of machine learning, especially in artificial neural networks, made pattern-based models increasingly powerful and able to generalize to complex scenarios. However, their performance is often bounded by the training data and their black-box structure does not allow for explaining their behavior. These limitations are particularly problematic because models need to be flexible to safely operate in any situation. Finally, goal-based models can reason on the optimal trajectories to reach a goal and can be even more precise than pattern-based models. However, they require a map of the area, which may not be available, and their complexity scales with the size of the environment.

Hybrid models are systems that combine models of different natures. This solution has great potential because it can take advantage of the benefits of the various models and mitigate their shortcomings. In the case of pedestrian motion prediction, they have recently been proposed as a potentially better alternative to conservative methods.

To study pedestrian motion and validate the suitability of prediction models, different types of datasets have been collected during the last decades. Pedestrian motion datasets contain trajectories in pedestrian areas recorded using a fixed camera. Despite their small size and simplicity of their trajectories, they are still used as reference by the research community. Next, pedestrian intention datasets contain trajectories labeled with the action performed by the pedestrians. All these features can be used to create complex and powerful models using intentions as features. Finally, AV motion datasets are specifically meant to develop motion prediction models for AVs and usually contain trajectories of multiple categories of road users, as well as contextual information such as the road structure and the states of traffic lights. However, they are still a new approach and still disregarded by research on pedestrian motion.

1.2. Thesis objectives

The purpose of this thesis is to design a pedestrian trajectory prediction model suitable for in-vehicle, real-time prediction. In order to achieve this result, several tasks should be

completed:

1. The difference of the trajectories found on pedestrian datasets and AV motion datasets should be understood. The first have a greater support on literature, but they may not be representative of the entire population of the road users because of the method used for their collection. AV motion datasets seem to be a better choice, but the benefits of their use are not clear. A comparison between the trajectories contained in these two classes of datasets will determine which type of dataset is the most suitable for the selected task.

2. Physics-based and pattern-based models for pedestrian motion prediction have many differences. Their performance on clean data is evaluated to understand their advantages and disadvantages on ideal conditions. A hybrid model will be considered if the individual models do not deliver adequate performance in terms of accuracy.

3. Realistic trajectories have many added challenges that derive from hardware failures and/or an occluded field of view. Studying the robustness of the prediction models to realistic trajectories will determine their suitability for prediction in real life. Once again, a hybrid model will be considered if the individual models do not deliver adequate performance in terms of robustness.

4. A framework for real-time prediction will be designed. Its deployment and testing on a vehicle will provide insights on its suitability for real-life scenarios.

1.3. Major contributions

The first contribution of this research is a study of the trajectories found on a sample of AV motion datasets and a comparison with the ones found on pedestrian datasets. This analysis highlights the differences between the two kinds of datasets, especially in terms of the complexity of the trajectories, given by greater deviation from linear motion, speed variations and lower predictability. The higher complexity of the trajectories implies that the prediction task using this kind of data is more challenging, and that AV motion datasets are more suitable than pedestrian datasets for motion prediction in urban scenarios.

The second contribution of this research is an analysis of physics-based and pattern-based pedestrian trajectory prediction models. Under ideal conditions, all the neural networks examined outperformed the constant velocity baseline in short-term and long-term prediction. Among the neural networks studied, the Conv-LSTM had the best accuracy. At

a prediction horizon of 4.8s, the mean of the ADE and the FDE of the Conv-LSTM were respectively 20% and 14% lower than the ones of the CV.

Next, a hybrid model for the prediction of realistic trajectories (HC-LSTM) is presented. The accuracy of predictions on realistic data was measured including the most severe cases, on highly degraded data. Under these conditions, the mean of the ADE and the FDE of the HC-LSTM at 4.8s were respectively 33% and 25% lower than the ones of the CV. Moreover, comparing the performance degradation from clean to realistic trajectories, for the CV the mean of the ADE and FDE at 4.8s degrade respectively by 25% and 19%, while for the HC-LSTM the same metrics degrade by 13% and 9%.

Finally, a novel real-time motion prediction architecture is proposed, to allow AVs to predict realistic trajectories using the proposed hybrid model. This framework was successfully integrated in the pipeline of a test vehicle and shown to be functional.

1.4. Thesis outline

In the next sections, Chapter 2 will provide the operational context of autonomous driving, as well as defining the trajectory prediction problem considered in this study. Chapter 3 will provide an overview of the state of the art for pedestrian motion prediction. After that, Chapter 5 will include a quantitative analysis of the trajectories contained in the AV motion datasets and compare them with the ones found in pedestrian datasets. Next, in Chapter 6 the prediction models are presented and evaluated on ideal trajectories. Chapter 7 will discuss the challenges given by realistic trajectories and present a hybrid model designed for their prediction. Furthermore, Chapter 8 will introduce an algorithm for real-time prediction and show its implementation on a vehicle. Finally, Chapter 9 will discuss the implications of this work and directions for future works.

2 | Preliminaries

This chapter provides relevant background information to introduce the context of this study. First, an overview of autonomous driving is given, and finally, trajectories and their prediction are defined and explained.

2.1. Autonomous driving

An AV is a complex system composed of hardware and software elements. Describing the components and the flow of information in the process of autonomous driving (AD) is important to understand the role and the context of trajectory prediction. Fig. 2.1, shows the building blocks that constitute the AD pipeline. The process is divided into two main sections: *i) Scene Understanding and ii) Decision making & Planning.* In the first stage, the AV collects information about its surroundings using its sensors and analyzes the scene to create an abstraction of the scene, identifying the road structure, object locations, and predicting vehicle and pedestrian behaviors. In the last stage, this information is processed to plan and actuate the AV's route, according to a policy that is set to obey the traffic laws and avoid collisions.

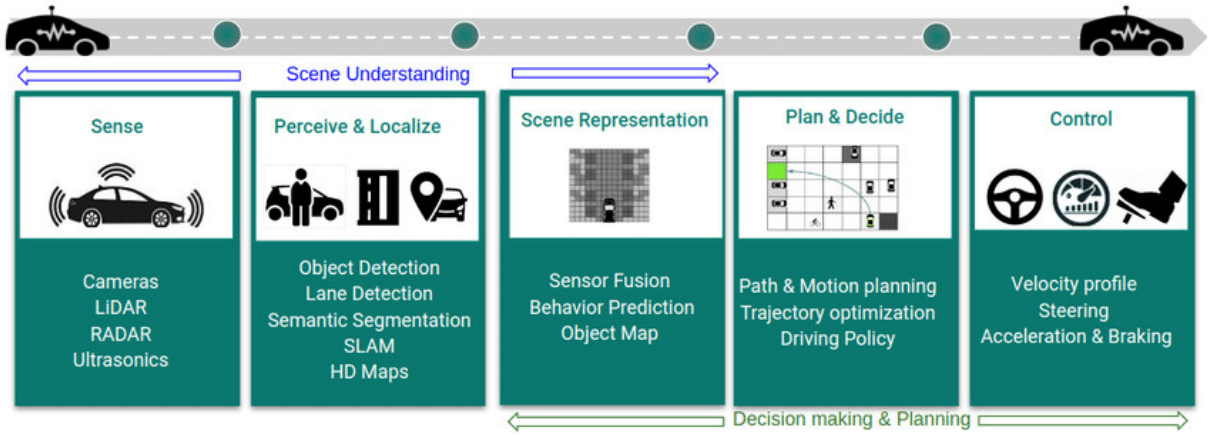

Figure 2.1: The autonomous driving pipeline [22]

1. **Sense**: Information is collected using the onboard sensors with a defined update

rate. Modern AVs are usually equipped with a sophisticated sensor suite, including cameras, LiDAR, RADAR, and ultrasonics.

2. **Perceive & Localize**: The raw data collected in the first phase is initially processed to detect the relevant objects in the scene. For instance, videos can be analyzed with semantic segmentation algorithms to locate and classify different types of agents (vehicles, public transportation, cyclists, pedestrians, etc.) and the structure of the road (lane boundaries, sidewalks, pedestrian crossings, traffic lights, etc.).

3. **Scene Representation**: The segmented data is aggregated to create an abstracted representation of the scene, that includes for each object their position and state during the observed time. Prediction algorithms are used to forecast the future states of each agent given the information available.

4. **Plan & Decide**: In this phase, the AV is fully aware of the surrounding environment and a motion planning algorithm evaluates and plans the future states of the AV in order to drive towards its final destination while obeying to the driving policy, defined over the traffic laws, the avoidance of collisions and the maintenance of an adequate comfort level.

5. **Control**: A controller defines the acceleration and steering angle necessary to actuate the trajectory defined by the planner. In practice, these control methods are founded in control theory and can be often stated as a minimization of a cost function defined over a set of states and control actions.

This process is executed iteratively for the entire operation of the AV until the final destination is reached. As the environment constantly changes, the trajectory of the AV needs to be updated continuously. We can see that every block depends on inputs from its predecessors. This means that an error at the beginning of the process propagates to the rest of the chain, and may be the cause of an unpleasant driving experience, or in the worst case an accident.

2.2. Trajectory prediction

As seen in the previous section, trajectory prediction takes place in the *Perceive & Localize* phase and it is important to ensure a safe driving experience, avoiding collisions with moving entities, such as vehicles and pedestrians. In this section, a trajectory is formally defined.

Let the state of a pedestrian over time, sampled with a sampling frequency f, be repre-

sented by the following function:

$$s(t) = (x, y, v_x, v_y, \theta), \tag{2.1}$$

where x and y are the coordinates in a Euclidean space, v_x and v_y are the velocities along the axis, and θ is the heading angle. Let also $S(t_n, t_m)$ be the collection of states between time t_n and t_m, defined as following:

$$S(t_n, t_m) = \{s(t_n), s(t_{n+1}), ..., s(t_m)\}. \tag{2.2}$$

Fig. 2.2 describes a pedestrian walking in an open environment.

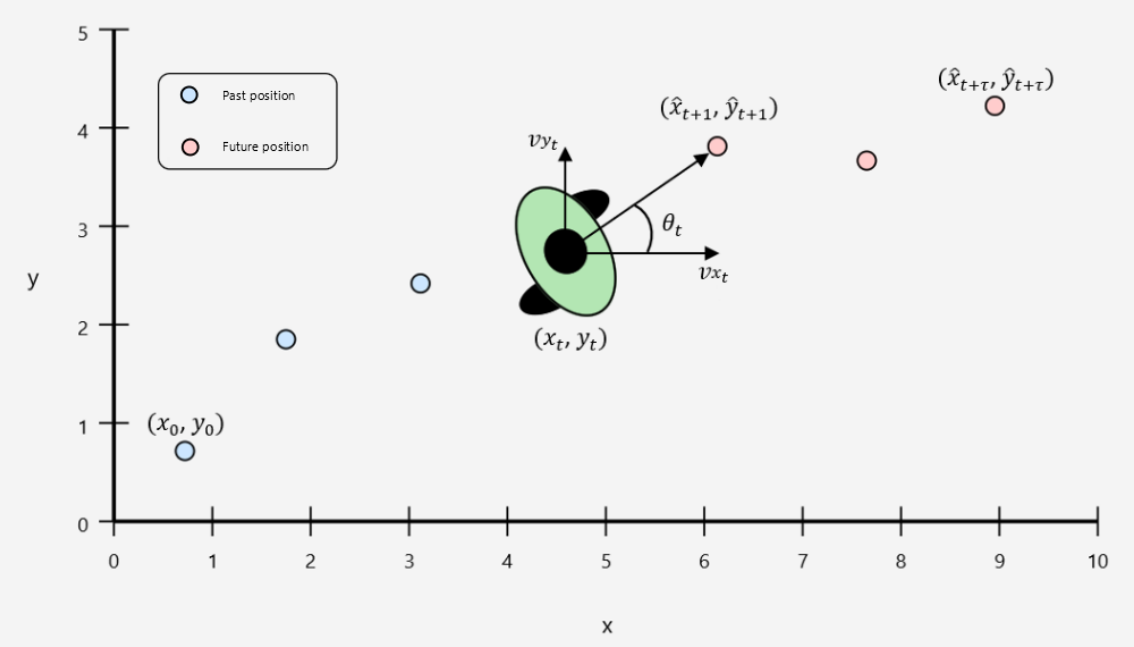

Figure 2.2: Definition of trajectory

The pedestrian has been walking starting from the initial time 0 until the current time t. Assuming that the history $S(0, t)$ has been recorded by the sensors of an AV, the prediction module is executed at time t, and it returns a new collection of states $\hat{S}(t+1, t+\tau)$, which forecasts the future states from the next future time-step $t+1$ until the prediction horizon τ, given the history. As time passes, the real trajectory can be recorded and compared to the prediction to measure its error. These metrics will be introduced in the following chapters. The goal of the prediction module designed in this study is to forecast the most probable future trajectory, and therefore the one which has the highest chance of being as close as possible to the real future trajectory (the ground truth). However, the

problem could be modeled using approaches, including prediction of multiple trajectories and intention prediction.

3 | Related Works

In this chapter, previous research on the topic of pedestrian motion prediction is introduced. In the first section, the three main categories of prediction models are presented, and then the focus is shifted to the datasets that have been collected for pedestrian motion analysis.

3.1. Pedestrian Motion Prediction

Over the past several decades, the automotive industry has been moving toward greater degrees of automation [30]. AVs are able to monitor their state and their environment using a wide range of sensors, and they may also communicate with each other and the traffic infrastructure using wireless vehicle-to-everything (V2X) communication [32]. Based on the information collected they can make decisions, plan their motion, and follow the plan utilizing a set of sophisticated controllers. However, when it comes to pedestrian motion prediction, AVs face several challenges due to the complexity of human behavior and the variety of its internal and external stimuli. The literature shows different approaches to solve this problem. Rudenko et al. [42] presented a classification framework for human motion prediction and classified the models that have been studied during the last decades in three categories: *i) Physics-based / knowledge-based models, ii) Pattern-based / data-based models, iii) Planning based / goal-based models.* In the following sections, these categories of models are presented, together with relevant previous works.

3.1.1. Physics-based models

Physics-based (PB) models use a set of explicitly defined dynamics equations, that follow a physics-inspired model, following the *Sense - Predict* paradigm [42]. They have been developed to describe the motion of dynamic objects in a different variety of applications and are typically used as building blocks for complex systems.

Single-model approaches are the simplest way to represent human motion. Popular examples include the *constant velocity* (CV) model that assumes piecewise constant velocity.

Schöller et al. [44] remark on the relevance of the CV showing that it can outperform neural models and suggest that, due to its strong performance on commonly accepted benchmarks and its simplicity, the CV should in the future be included as a standard baseline for pedestrian motion prediction. Adding to the category of simple PB models, the *constant acceleration* (CA) model assumes piecewise constant acceleration, and the *coordinated turn* model assumes constant turn rate and speed. Møgelmose et al. [31] presented a motion prediction framework with integrated map-based hazard inference and Elnagar used CA for Kalman filter-based (KF) prediction of dynamic obstacles [13].

PB models can be extended using information extracted from a map, or sensed in the environment, to make better predictions considering semantic constraints. These constraints can be static (sidewalks, crosswalks, etc.) or dynamic (other pedestrians, vehicles, etc.). Batkovic et al. predict pedestrian motion based on a simple graph representation of the road geometry [8] and Helbing and Molnar presented the social force (SF) model, which attributes a potential field to each object in the environment, and models human behavior under the assumption that these forces may be repulsive or attractive [18].

While physics-based models excel in flexibility and explainability, their explicit set of rules may be a biased generalization of human behavior, which may depend on many other factors that cannot be translated into equations, or that were not considered.

3.1.2. Pattern-based models

Pattern-based, or data-based (DB) models learn human motion dynamics from the data, following the *Sense – Learn – Predict* paradigm [42]. These methods make extensive use of data mining and machine learning practices, and recent developments in artificial neural networks have resulted in these algorithms becoming increasingly powerful. Rudenko et al. [42] classify these models as *sequential* and *non-sequential*, depending on whether the prediction models use information from an history of past states or just the current state.

In the context of autonomous driving, the interest is to make predictions in outdoor spaces, so models that learn location-independent patterns are relevant. On the other hand, location-dependent models are constrained by a determined map of the environment and are more suitable for indoor prediction. Several recent works assume that future states depend on very early states (higher order Markov property), and neural networks with recurrent layers (RNNs) [27] exploit the dependency from the past. Alahi et al.

proposed a model based on the Long Short-Term Memory (LSTM) [19], Social LSTM [4], to forecast pedestrian trajectories in crowded spaces, and Bartoli et al. extended it to consider static objects in the surroundings [7]. These methods make predictions in a top-down projection of the scene. However, this kind of data could be difficult to retrieve by autonomous vehicles, which usually have onboard sensors that offer an *egocentric view*. M2p3, a work by Poibrenski et al. [36], combines a conditional variational autoencoder with a recurrent neural network encoder-decoder architecture to predict a set of possible future locations of each pedestrian in a traffic scene. Other models use contextual information such as actions performed and body features to estimate goals and trajectories. Yao et al. [50] use estimated intent and action to predict crossing behavior. Malla et al. [29] created a dataset and developed an RNN-based model that makes predictions based on the pedestrian's actions. Similar approaches include [38], [49] and [17].

These methods can achieve very good results on the commonly accepted accuracy metrics. However, they are not perfect. Neural networks require a vast amount of data to train, they often lack interpretability, and if trained inappropriately they may overfit the training data. A successful application of these models in real-time depends on many factors which have not yet been studied properly. The computation capabilities of the AV are particularly important and should be enough to avoid delays in the predictions.

3.1.3. Goal-based models

These models follow the *Sense - Reason - Act* paradigm [42], solving a sequential decision-making problem to predict an agent's behavior. In the context of human motion prediction, the agent is the pedestrian, and models could consider one or multiple agents to represent complex environments. Rudenko et al. [42] classify these methods based on the definition of the cost function used to train the model: *Forward planning-based approaches* use a pre-defined one. For example, Rudenko et al. [41] use a Markov Decision Process (MDP) approach to make environment-aware joint predictions of multiple agents. On the other hand, *inverse planning-based approaches* learn the cost function by observing the data (Pfeiffer et al. [35]).

Goal-based approaches work well if the pedestrian's goals can be explicitly defined and a map of the environment is available. In these cases, the planning-based approaches can generate better long-term predictions than the physics-based techniques and generalize to new environments better than the pattern-based approaches. However, the run-time of planning-based approaches scales exponentially with the number of agents, the size of the

environment, and the prediction horizon [42]. For these reasons, their implementation in pedestrian-rich scenarios such as urban areas or large outdoor spaces may be challenging [42].

3.1.4. Hybrid models

Recent work from Korbmacher and Tordeux [23] compares PB models with deep learning (DL) approaches and concludes that nowadays it is questionable whether PB models are still relevant to predicting pedestrian trajectories because state-of-the-art DB models have surpassed them in terms of accuracy. On the other hand, they might still play an important role to overcome the disadvantages of neural networks due to their flexibility. They believe that the hybrid approach could have great potential in future works because it could leverage the benefits of both models, mitigating their individual weaknesses. Silvestri et al. [46] showed that domain knowledge can be injected into a deep neural network at training time and its robustness with a small amount of data increases significantly. Following this trend, Antonucci et al. [6] presented a hybrid framework that embeds the SF into a neural network, by modeling the network's structure so that its connections reflect the dynamics of the SF. Using this prior knowledge, the complexity of the model is reduced and its predictions become physically explainable. Another relevant work by Kothari et al. [24] builds a hybrid model that leverages the theory of discrete choice models that can output interpretable intents and accurate predictions.

3.2. Datasets

To evaluate the performance of a prediction model, and, if necessary, to train one, many different datasets have been collected, recording pedestrian trajectories and behavioral features in different environments. Depending on the type of data collected and on their focus, these datasets can be grouped into three categories: *i)Pedestrian motion datasets, ii) Pedestrian intention datasets, iii) AV motion datasets.*

3.2.1. Pedestrian Motion Datasets

By far, the most common datasets that have been used in literature are the ETH BIWI Walking Pedestrians Dataset (ETH) [34] and UCY [26]. Each of them contains videos and pedestrian annotations recorded from a top-down view of a selection of sidewalks and urban squares with little to no interaction between pedestrians and vehicles. Despite their small size and low complexity, these datasets have become the standard for pedestrian

motion prediction, and even today's state-of-the-art models are still using them to benchmark their performance. Fig. 3.1 shows one frame extracted from the ETH. As visible, the camera is on an elevated position and records pedestrian walking in a controlled space.

Figure 3.1: A frame from the ETH Dataset

3.2.2. Pedestrian Intention Datasets

Joint Attention in Autonomous Driving (JAAD) [25] and Pedestrian Intention Estimation (PIE) [37] are relatively new datasets, which include videos, pedestrian attributes, and spatial annotations, with a focus on behavioral and intention annotations, which were manually labeled by the researchers. All these features can be used to create complex and powerful models using intentions as features. Models based on these datasets can also be trained to perform a classification task, to predict the intentions at the very beginning of the task. For example, a model could first analyze the scene to determine if a pedestrian is going to cross, and then use this information to predict their trajectory. This strategy has a good potential to make more accurate predictions, but a failed intention classification has the potential of conditioning the predicted trajectory in a negative way, other than adding an overhead to the prediction time, which in a real-time application should be kept to a minimum. Fig. 3.2 represents an annotated frame from PIE. As visible, each pedestrian was labeled with a bounding box and the action they are performing. For example, the pedestrian on the right is labeled as *looking* and*standing*.

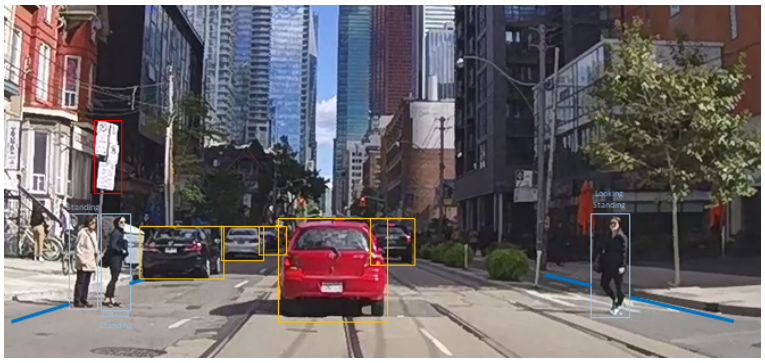

Figure 3.2: An annotated frame from the PIE Dataset

3.2.3. AV Motion Datasets

In recent years, many other datasets were collected, including scenes from the egocentric view and data from the AV's onboard sensors (radar, LiDAR, etc.). These datasets are specifically meant to develop motion prediction models and usually contain trajectories of multiple categories of road users, as well as contextual information such as the road structure and the states of traffic lights. KITTI Vision Benchmark Suite (KITTI) [16] is one of the first attempts to create a dataset for use on an AV, which contains scenes taken on German streets and pedestrian annotations. Most recent works include Argoverse 2 [48], Waymo Motion [47], nuScenes [9] and Apolloscape [20]. These last datasets were collected in many cities around the world and contain millions of different trajectories of vehicles and pedestrians, together with raw sensor data and high-definition maps. Since these models have been collected from the AV's point of view, they should be more suitable for the development of models specifically designed for an AV. Moreover, DB models trained on a higher number of data points have the potential to generalize to new scenarios with better accuracy. As visible in Fig. 3.3, this class of datasets does not usually contain a video of the scene, but a virtual representation of the scene can be constructed using the annotations of the road geometry and the positions of the agents.

3.3. Conclusions

Most of the previous work had the goal of improving the accuracy of the predictions of trajectories sampled from pedestrian datasets (most commonly on ETH and UCY). The main observed trend is about the focus of the models, which is to predict trajectories when walking in pedestrian-rich scenarios. This trend is mainly given by the fact that pedestrians are usually the only road users included in the most common datasets.

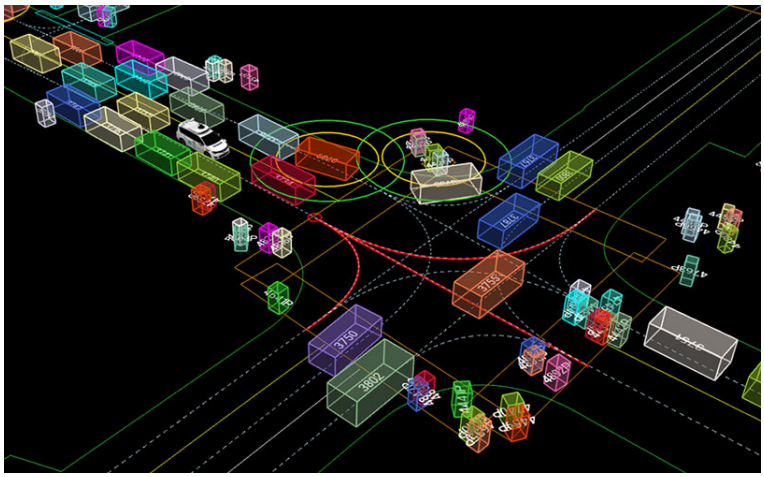

Figure 3.3: A virtual scene representation from the Waymo Dataset

Another trend among previous works is that the performance of prediction models is generally tested offline, on ideal conditions, and with the same reference parameters of 8 frames ($3.2s$ sampled $2.5Hz$) as observation time and 12 frames ($4.8s$ sampled at $2.5Hz$) as prediction horizon. Little to no information is known about the robustness of the models in adverse conditions, such as late detection or missing data.

Overall, previous research often disregarded some aspects of realistic pedestrian motion prediction for AV applications. In the next chapter, the problem will be stated and research questions will be formulated.

4 | Problem Formulation

Based on the analysis of the current state of the art of pedestrian motion prediction, in this chapter, the problems identified are discussed, the research questions are presented, and finally a research framework is designed to answer them.

4.1. Problem statement

As seen in the previous chapter, pedestrian datasets are the reference for the design and the evaluation of prediction models. The problem with these datasets is that their data is collected in pedestrian areas, usually with a camera mounted on a fixed elevated position, not capturing the behavior of pedestrians interacting with vehicles. New datasets collected using sensors mounted on AVs have been released, but for ease of comparison with previous works, research on pedestrian trajectory prediction models has not fully adopted them yet. These datasets have the potential of consistently improving the accuracy of prediction models on realistic data when used in AV applications. While pedestrian datasets have been studied deeply and the characteristics of their trajectories are known, no exploratory study has been conducted on AV datasets to understand the characteristics of the pedestrian trajectories they contain and their difference from the ones from pedestrian datasets.

As previously discussed, both PB and DB models have strengths and weaknesses. Hybrid models are cited in recent literature as an approach to creating powerful models that can exploit the synergy of different architectures. Further research is needed to understand what are the contexts in which each type of model can deliver the best performance, and how and to what extent a hybrid model could further improve the predictions.

Another issue with previous works is in the evaluation of the performance of the models. The standard approach of evaluating the metrics at a prediction horizon of $4.8s$ enables a model to be conveniently benchmarked with the state of the art but lacks flexibility. Moreover, the reported performance is mostly computed on clean trajectories, not considering the corner cases such as incomplete or skewed observations, which are instances

that may happen frequently in real life and should be assessed to understand the expected performance drop, and the possible directions to design models robust to non-ideal data.

Finally, the deployment of models on vehicles for real-time prediction has many added challenges that may lead to a degradation in the quality of the predictions. These challenges include generalizability to unseen environments, robustness to sensor failures, and delays in the output of trajectories. These aspects are very important for the deployment of prediction models in AVs, but they are usually disregarded by research. Therefore, there is not a clear understanding of how good pedestrian motion prediction models work on autonomous vehicles (AVs), how well they can generalize to real-world data, and what are the conditions enabling them to predict trajectories accurately. Analyzing the performance of a model in a real-life setting would enable us to compare the accuracy of the predictions with the ones calculated on public datasets. However, before a quantitative study of this kind can be carried out, it is required to understand how a robust prediction model can run on an AV in real time.

4.2. Research questions

RQ1a: What are the quantitative motion properties of the trajectories contained in autonomous driving datasets?
RQ1b: How do they differ from the ones of the trajectories contained in pedestrian datasets?

RQ2a: What are the strengths and weaknesses of physics-based and pattern-based models for pedestrian motion prediction?
RQ2b: What are the potential benefits of hybrid models over physics-based and pattern-based models for pedestrian motion prediction?

RQ3a: What are the challenges associated with pedestrian motion prediction on realistic data?
RQ3b: What is the performance degradation with respect to the predictions on clean data?

RQ4: How can a prediction model run in real-time on an autonomous vehicle?

4.3. Methodology

In this section it is discussed how each of the RQs is answered with an experimental analysis.

4.3.1. RQ1 - AV dataset analysis

Waymo and Argoverse 2 were chosen as samples for autonomous driving (AV) datasets because their data is recorded with on-vehicle sensors and contain a variety of different trajectories collected in different scenarios. In the comparison with pedestrian datasets, particular focus is given to ETH [34] and UCY [26] as they are the standard benchmark used in literature. The analysis is conducted by creating a sample of pedestrian trajectories from AV and pedestrian datasets and analyzing them with OpenTraj [5], an open-source framework for pedestrian trajectory analysis.

4.3.2. RQ2 - Prediction on ideal data

The selection of the DB model will follow after an analysis of different neural network architectures with recurrent layers. As seen in the previous chapter, these kinds of recurrent networks have become the state of the art in trajectory prediction because of their capability to leverage the relationships specific to sequential data. The constant velocity model was chosen as a sample of physics-based models because it is assumed that pedestrians move at a velocity that can be approximated to constant. Moreover, it has one of the lowest computation times, and despite its simplicity can still deliver good performance. Different types of trajectories are sampled from the Waymo Motion Dataset, and the performance of both PB and DB models is evaluated using average displacement error (ADE) and final displacement error (FDE) as metrics (6.2), because they have been adopted by the vast majority of previous works, and are useful to understand the difference in displacement between the real and predicted trajectories. The metrics are calculated at multiple prediction horizons to evaluate their prediction power in the short and long term, using ideal trajectories. If the individual models do not deliver adequate performance in terms of accuracy, the design and implementation of a hybrid model will be considered.

4.3.3. RQ3 - Prediction on realistic data

To answer this question, the trajectories extracted from the Waymo Dataset will be altered, removing positions to resemble realistic conditions, where pedestrians may be hid-

den behind objects or sensors may have failed in correctly tracking them. The performance of the models will be determined using the same methodology adopted for **RQ2**, but in this case, the goal is to measure the robustness of the models when approaching realistic data. Therefore, the performance will be assessed at multiple prediction horizons and with multiple types of alterations. In this case, the implementation of a hybrid model will be considered if the models do not deliver adequate performance in terms of robustness.

4.3.4. RQ4 - Real-time prediction

After the implementation and the performance analysis of the models will be concluded, a framework for real-time prediction will be designed. Its deployment and testing on a vehicle will provide insights on its suitability for real-life scenarios.

5 | Analysis of AV Datasets

As discussed in the previous chapter, previous research on pedestrian trajectory prediction has adopted a selection of datasets as standard to train and evaluate prediction models. Although this choice allows the performance of the models to be easily compared with previous works, it may be a limit to their performance in the real world. Pedestrian datasets such as ETH [34] and UCY [26], which are the most popular datasets in research, were collected in sidewalks and urban squared using a fixed camera. Therefore, they may represent a biased representation of pedestrian behavior. Depending on the type of AV, pedestrian motion in these contexts may be relevant or not. However, a prediction model should be capable of predicting accurate trajectories independently of the selected context. AV motion datasets are new types of datasets that were collected from the point of view of the vehicles in many different cities and contain millions of trajectories. Therefore they should offer a less biased representation of the real world compared to the traditional pedestrian datasets. In this chapter, the trajectories found in a sample of AV motion datasets are analyzed and compared with the ones that can be found in a sample of pedestrian datasets with the goal of understanding their difference and similarities. This analysis will start with a comparison of the fundamental motion properties of the trajectories. Next, the deviation from linear motion and the efficiency of the trajectories will be studied, by the mean of visual inspection of trajectories and analysis of their distribution. These experiments have the goal of understanding the regularity, the shape and the complexity of the trajectories. Finally, the predictability of the trajectories will be assessed by analyzing the entropy and applying clustering techniques. The experiments will be performed using the OpenTraj framework [5].

5.1. OpenTraj Framework

OpenTraj is an open-source framework to visualize, analyze and compare pedestrian datasets. Several datasets of different kinds are included, and it can be extended to be compatible with data in different formats. In the paper introducing the framework [5], there is a comparison of a selection of pedestrian datasets. This work will extend that

comparison by adding a selection of AV datasets. The sample of AV motion dataset is composed by 10000 pedestrian trajectories randomly selected from Waymo Open Motion and Argoverse 2. On the other hand, the sample of pedestrian datasets is composed by ETH [34], UCY [26], SDD [39] and Wildtrack [10]. Even if not relevant for pedestrian motion outdoors, two datasets that report motion in bottlenecks are included (BN-1d-w180 and BN-2d-w160 [45]), which show the motion of people walking through doors or narrow corridors. To allow for a homogeneous comparison, the trajectories from each dataset are processed to extract fragments of a fixed length ($4.8s$). These subsections will be referred to as *trajlets*.

5.2. Motion properties

The first properties analyzed are the motion properties. The top two rows of Fig. 5.1 show the distributions of the average speed and the speed variation, calculated for each trajectory, in m/s. In the bottom two rows, the distributions of the average accelerations and maximum accelerations are reported, using the same method as above, in m/s^2. Since every point in the distribution represents a trajectory, wider distributions mean that many points fall into a specific value. A research by the TCRP-NCHRP [15] shows that people aged 14-64 have an average walking speed of $1.46m/s$, while for older people this value drops to $1.20m/s$. The values depicted in the figure show that pedestrians rarely walk faster than the average, and many of them were recorded while not moving. Interesting values can be seen for the accelerations of the trajectories in Waymo, which can be much higher than the rest of the sample, possibly because of noise in the data.

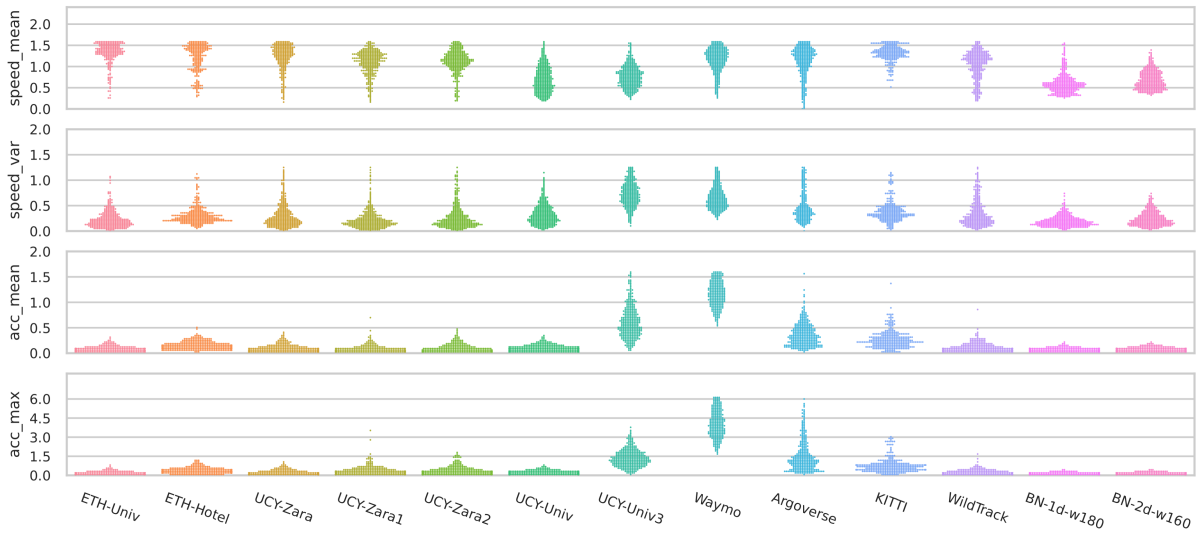

Figure 5.1: Motion properties

5.3. Deviation from linear motion

In this section, the deviation with respect to linear motion is reported. The deviation was measured at $t = 2.4s$ and $t = 4.8s$, subtracting the final heading angle to the one at $t = 0s$. As depicted in Fig. 5.2, despite not having the highest values for this metric, the deviation of Waymo and Argoverse 2 is almost double the one of ETH and UCY, both . This means that a change of direction in the trajectories of the ETH and UCY datasets is less frequent. Fig. 5.3 and Fig. 5.4, report plots of the trajectories for ETH

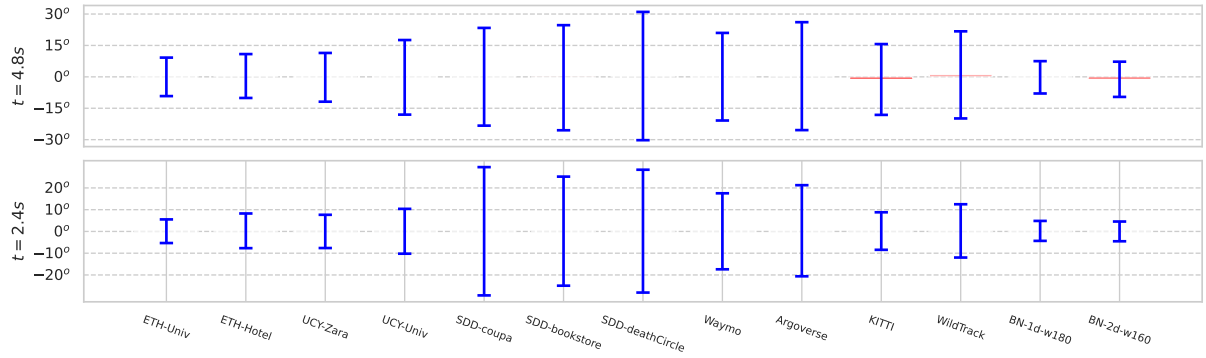

Figure 5.2: Trajectory deviation

and Waymo, which have been normalized to have the same initial position $(0, 0)$ and heading (front). In the figures, the red dots represent the average position at every other timestep, while the arches represent the average displacement. The higher density of dots in Waymo shows the higher sampling rate used to record its trajectories. These figures show that the trajectories contained in the Waymo dataset are much more diverse than the ones found in ETH. Therefore, the choice of Waymo as reference implies the use of more complex trajectory prediction models. However, its higher sampling rate determines a higher definition of the trajectories.

5.4. Efficiency

Another important metric to assess the complexity of a trajectory is its efficiency. This is defined as the ratio of the distance between the trajectory endpoints over the trajectory length. A trajectory achieves its maximum efficiency when a pedestrian reaches their destination walking in a straight line. In Fig. 5.5 we see that, while the majority of the trajectories has an efficiency near to 100%, the tails of the distributions extend close to 90%. This means that, overall, pedestrians are expected to walk on nearly straight paths.

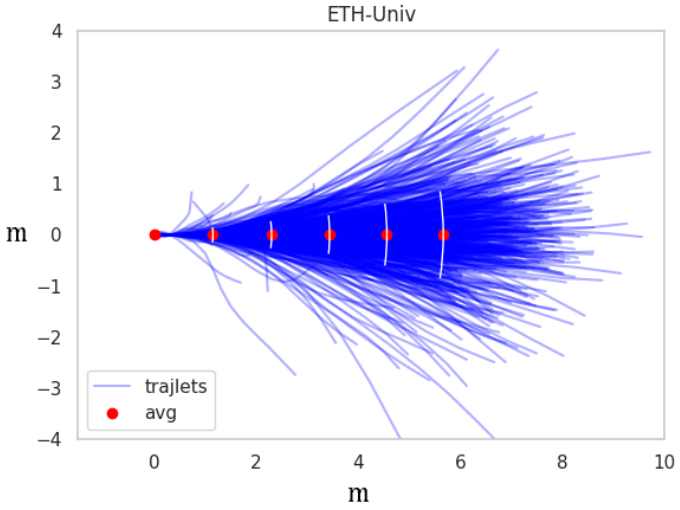

Figure 5.3: Deviation of ETH.

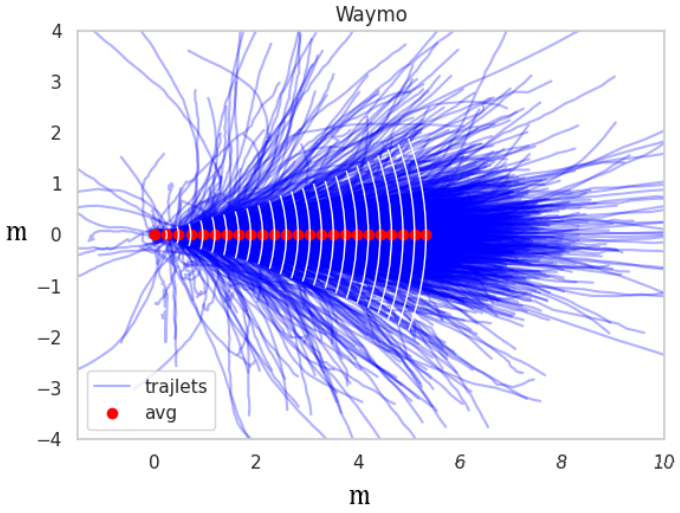

Figure 5.4: Deviation of Waymo.

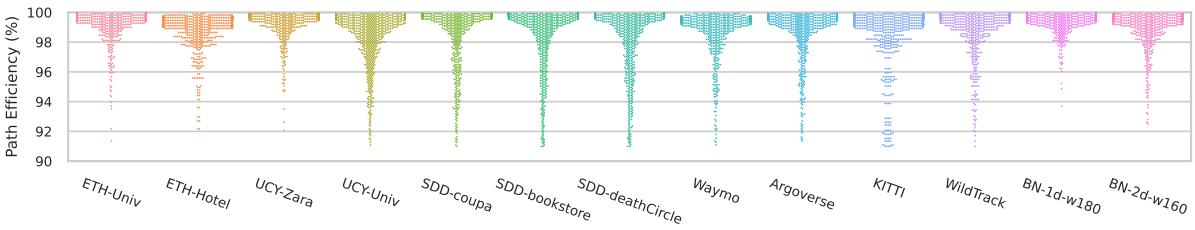

Figure 5.5: Path efficiency

5.5. Entropy

In the top part of Figure 5.6, a Gaussian Mixture Model was fit to the samples, using Expectation Maximization to select the parameters, and applying the Bayesian Information Criterion to select the number of clusters [5]. In the bottom part of the figure, a kernel density estimation was used to estimate the entropy. High entropy means that many data points do not occur frequently, while low entropy means that most data points are easy to be predicted. Similarly, a large number of clusters would require a more complex predictive model.

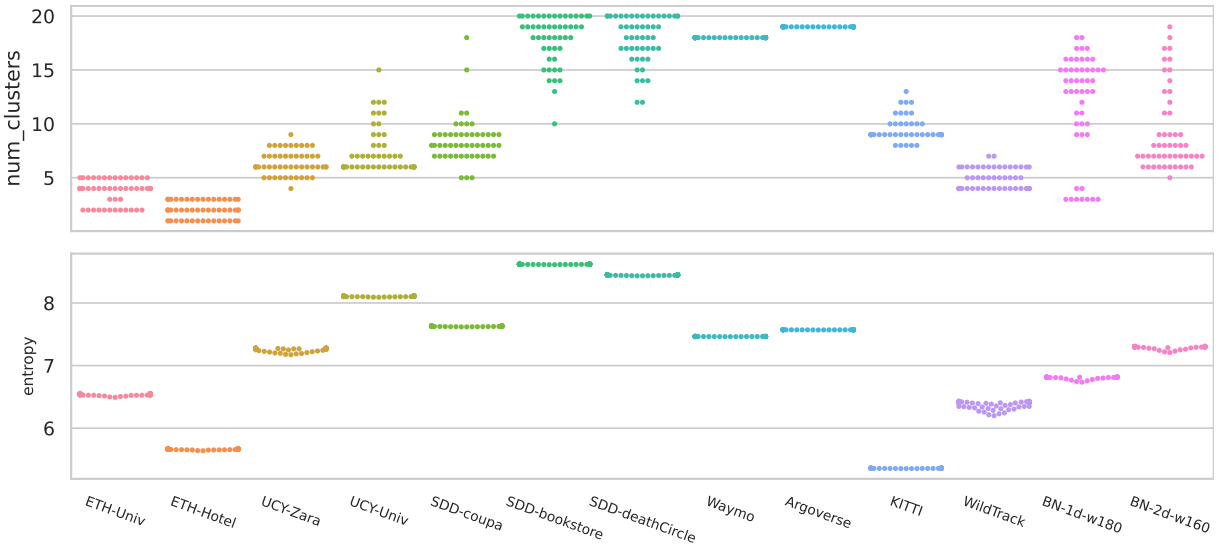

Figure 5.6: Overall entropy

Fig. 5.7 depicts the conditional entropy, representing the entropy of a trajectory given an history of $3.2s$. A Gaussian kernel was defined over the sum of Euclidean distances between the consecutive points along the observed and future parts of a trajectory [5]. Also in this case, higher conditional entropy means that a trajectory is more difficult to predict.

5.6. Conclusion

In this chapter, a sample of AV motion datasets was analyzed and compared to a sample of pedestrian motion datasets. This analysis showed the differences in terms of motion properties and predictability of the trajectories contained in the two kinds of datasets. Pedestrian motion depicted in AV motion datasets has higher deviation from linear motion and higher entropy. Because of its complexity, and expected greater capability of representing pedestrian motion in urban scenarios, Waymo was chosen as reference for

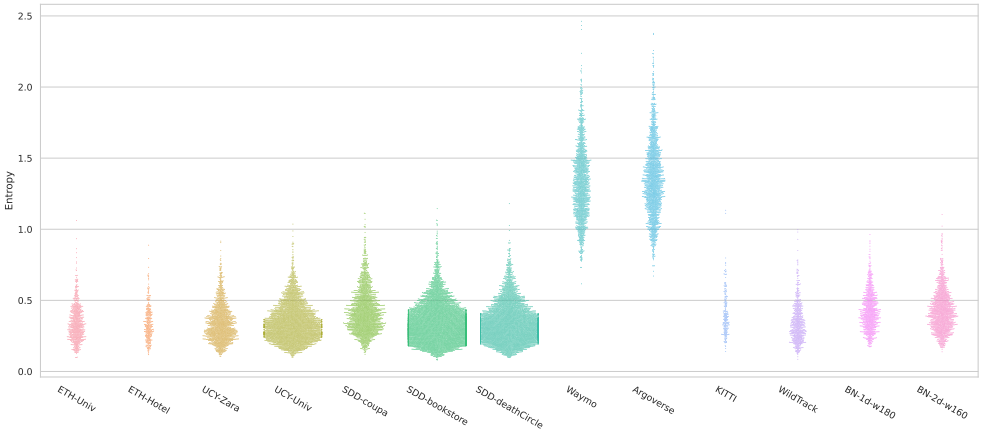

Figure 5.7: Conditional entropy

further analysis.

6 | Prediction on clean data

In this section, two prediction models of different kinds are presented and compared on ideal conditions, where the agent was correctly tracked for the entire observation window. Next, a performance comparison is made based on the predictions using a clean dataset. The first model is based on constant velocity (CV) and is presented in Section 6.3. The second is a pattern-based model using a neural network and is presented in Section 6.4. Both models are *deterministic* rather than *probabilistic*, since their output is a single trajectory that predicts the future positions, given an observation.

6.1. Data Exploration and Pre-processing

As discussed in the conclusions of the analysis performed in the previous chapter, Waymo Open Motion Dataset was selected as a reference for the development of prediction models. The full trajectories recorded on the Waymo dataset are 9.1s long and contain 91 data points since they were sampled at $10Hz$. For the purpose of developing prediction models, each trajectory was purposely split into its *observed* and *future* part. The first $2.0s$ were selected as observation, while the last $7.1s$ were selected as future. This selection allowed the models to be evaluated on both short-horizon and long-horizon predictions.

Feature selection

The Waymo dataset contains a wide range of features to describe the state of an agent at a given time. Referring to the definition given in 2.2, the selected set of features, for each trajectory, corresponded to $S(t_0, t_{19})$. Therefore, each trajectory contained 20 observed positions sampled at $10Hz$. Furthermore, each data point in the Waymo dataset contains an indicator stating if the position is valid or not, because the sensors may have failed in recording the position at that time, or the observed pedestrian may be hidden behind an object. This indicator was extracted among the other features. For each trajectory and for each timestep, the following features were selected:

- The observation is valid - $observed \in \{0, 1\}$

- x-axis position - x $[m]$
- y-axis position - y $[m]$
- Heading angle - θ $[rad]$
- x-axis velocity - v_x $[m/s]$
- y-axis velocity - v_y $[m/s]$

The data was then stored in a tensor of the following dimensions: $[\#trajectory] \times [\#timestep] \times [\#feature]$.

Pre-processing

The data in the Waymo datasets is recorded using a representation referred to as *global coordinates*: virtual coordinates that represent a point in the space of each of the cities where the data was collected. For example, in a city that has a square area where each edge has a length of 2km, its global coordinates would be in the range $[0, 2000]$ $[m]$. While this type of coordinates allows for convenient visualization using the built-in maps, it may not be the best choice for the trajectory prediction task using machine learning models. Since the space of the coordinates in its raw format is in the order of kilometers, two similar trajectories which were recorded in distant locations in space would have a very different representation. This would inevitably make the learning process slower and less accurate, because large error gradients would have a negative effect during the training phase, making it very unstable. For these reasons, the data needs to be pre-processed.

The trajectories are pre-processed to transform them to *front-headed*, *location-independent*. This transformation facilitates the learning phase of the neural network, since the trajectories all have the same direction (left to right), and the pedestrian's position at prediction time is centered in the origin $(0, 0)$. However, the trajectories can always be reverted to their original coordinates, applying the inverse of the pre-processing functions in reversed order.

The following steps were executed considering an observation time of $2.0s$, instead of the commonly used value of $3.2s$. This was made possible by the higher sampling rate of Waymo, which is four times the one of most pedestrian datasets [34] [26] and allowed to reduce the observation time without sacrificing the number of collected data points.

The data is pre-processed using the following process:

1. Translate the trajectory to the origin at prediction time, which is the last position in the observation
2. Rotate the positions and velocities with respect to the heading angle at the prediction time
3. Scale the data in the interval $[0, 1]$ using *MinMax Normalization*, dividing each datapoint by the maximum value in its scale (only needed when the data is input to the neural network)

The Python code for the pre-processing is depicted in Appendix A.

6.2. Metrics

To measure the difference between two trajectories, and therefore to assess the quality of a prediction model, the most common approach is to evaluate the average displacement error (ADE) and the final displacement error (FDE). These metrics compare the predictions and the ground truth and return a value in meters.

These metrics are defined as follows:

$$ADE = \frac{1}{n}\sum_{i=0}^{n}\left(\sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}\right), \tag{6.1}$$

$$FDE = \sqrt{(x_n - \hat{x}_n)^2 + (y_n - \hat{y}_n)^2}, \tag{6.2}$$

where n is the number of points in the future trajectory, x_i and y_i are the ground truth coordinates at the timestep i, and $\hat{x}_i$ and $\hat{y}_i$ are the predicted coordinates at the same instant. Fig. 6.1 shows a visualization of the ADE computed on a sample trajectory, while Fig. 6.2 shows the same visualization for the FDE.

Their average over the test dataset, measured at a prediction horizon of 4.8s, is most frequently the value reported in research papers. In this study, these metrics will be always reported for multiple timesteps because it is insightful to show the performance of the models for the entire length of the available prediction horizon ($7.1s$). However, to facilitate the comparison of the results of this study with previous research, the metrics at $4.8s$ will be the reference values for the tuning of the models.

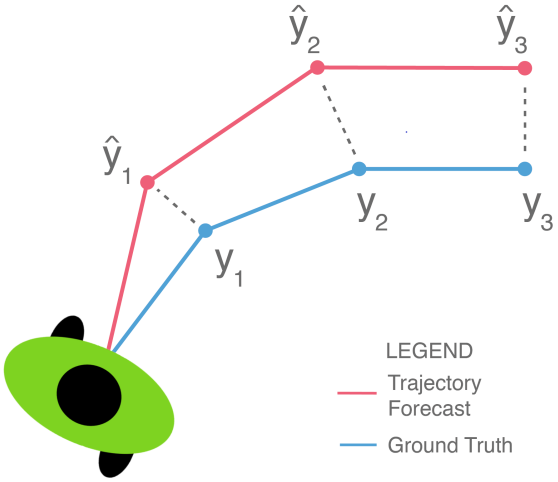

Figure 6.1: Visualization of ADE

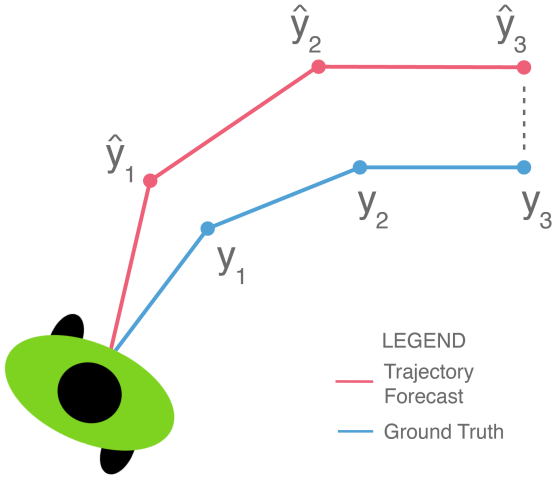

Figure 6.2: Visualization of FDE

6.3. Constant Velocity Model

In this section, the CV prediction model is presented. In the previous chapter, it is shown that many trajectories in Waymo have a considerable deviation from linear motion, but earlier research involving this model showed that it could compete even with advanced data-based models [44]. Therefore, an evaluation of a model based on it is relevant to this study. Moreover, it is one of the least computationally demanding algorithms that could be designed. For this reason, even low-power embedded systems would be capable of executing it in real-time. In its most basic form, the model would not even need an extended history, since it could use just one observation, and its design would be a

simple application of the kinematic laws. In fact, given the velocities at the last observed timestep: v_x and v_y, the predictions would be given by the following:

$$\begin{cases} \hat{x}_i = v_x \cdot t \\ \hat{y}_i = v_y \cdot t, \end{cases} \tag{6.3}$$

where t is the prediction horizon, and considering that at prediction time the pedestrian is located in the position $(0, 0)$.

However, since the data was sampled at a frequency of $10Hz$, and the accuracy metrics are computed on discrete values, it is more relevant to design a model that outputs discrete trajectories. These are given by the following:

$$\begin{cases} \hat{x}_i = v_x \cdot 0.1i \\ \hat{y}_i = v_y \cdot 0.1i, \end{cases} \tag{6.4}$$

where i is a given future step. In this case, the output rate was designed to be equal to the sampling rate. Therefore, each time step is equivalent to $0.1s$.

To mitigate the effect of noise on the velocity measurements on the performance of the CV predictions, a dependency on history is introduced. Averaging the velocities of the k timesteps before the prediction time is chosen as regularization parameter, as expressed in the following:

$$\begin{cases} \overline{v}_x = \frac{1}{k} \sum_{i=n-k}^{n} v_{x_i} \\ \overline{v}_y = \frac{1}{k} \sum_{i=n-k}^{n} v_{y_i}, \end{cases} \tag{6.5}$$

where $\overline{v}_x$ and $\overline{v}_y$ are the average velocities. Considering a $2.0s$ time window sampled at $10Hz$, the model could exploit the information from 1 up to 20 previous timesteps. In this case, k can be considered a hyperparameter of the model. As previously discussed, to select its best value, the FDE at $4.8s$ was monitored. Fig. 6.3 shows the ADE at $4.8s$ of this model in function of the length of the history. The optimal value is at $k = 7$.

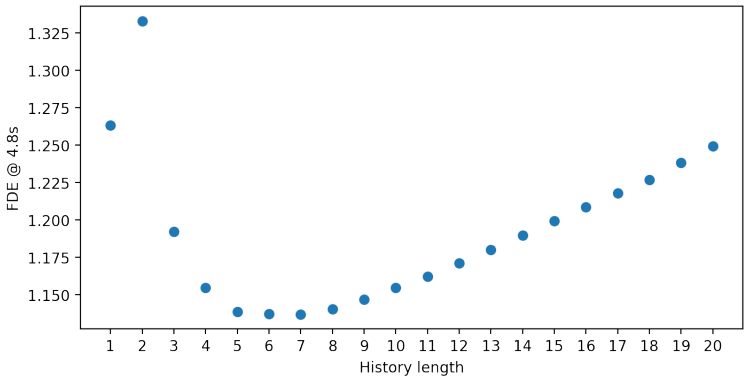

Figure 6.3: Hyperparameter tuning of the CV model

6.3.1. Performance evaluation

As Fig. 6.4 shows, the predictions produced by this model approximate pedestrian motion to a straight line, which could be precise if the person does not change their heading or speed during their future steps, but in case of any curved trajectory, this model is not suitable to predict them.

Considering the consistency of the predictions, Fig. 6.5 reports the mean, μ and the standard deviation, σ of the ADE and the FDE over the entire prediction horizon. As the prediction horizon increases, μ and σ of both metrics increase exponentially, indicating that as the prediction horizon increases, the positions become less dependent on the observation.

6.4. Neural Network

In this section, the pattern-based model selected for the analysis are presented. Given that most recent works have obtained the best results using a neural network with recurrent layers, multiple configurations of a this kind of neural networks were tested. The models were implemented using Keras [21] and Tensorflow 2 [3], and trained on a machine equipped with an AMD Ryzen 9 5950X CPU, 16GB of RAM, and an NVIDIA RTX A5000 GPU.

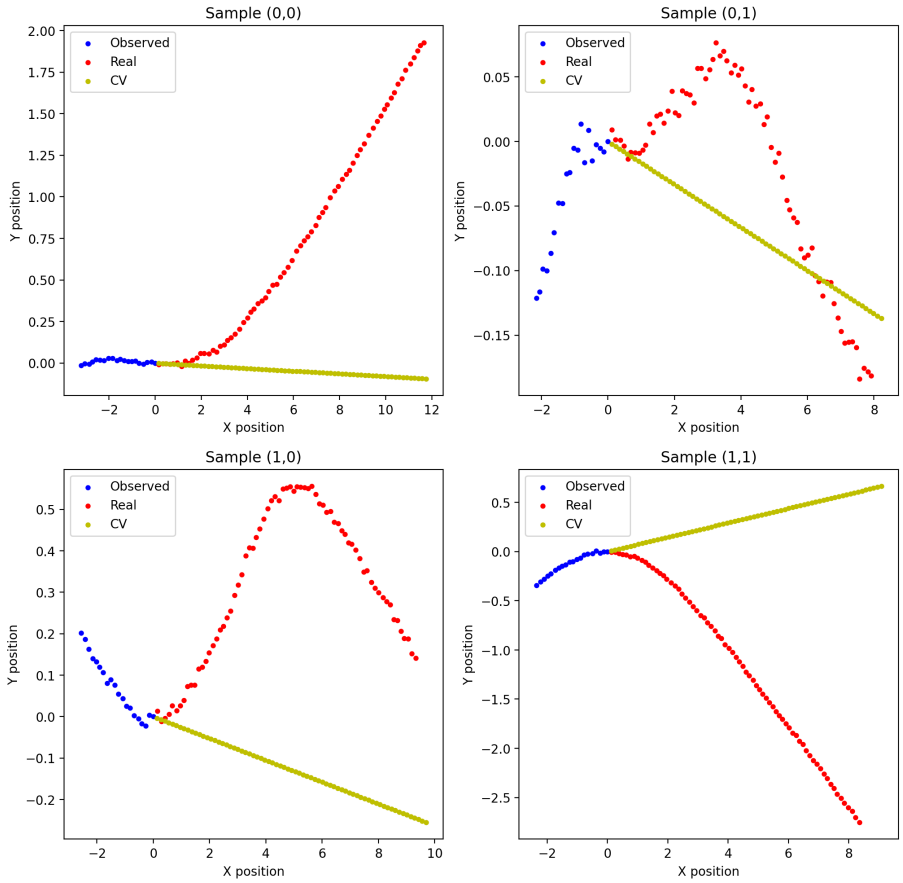

Figure 6.4: Predictions of the CV model

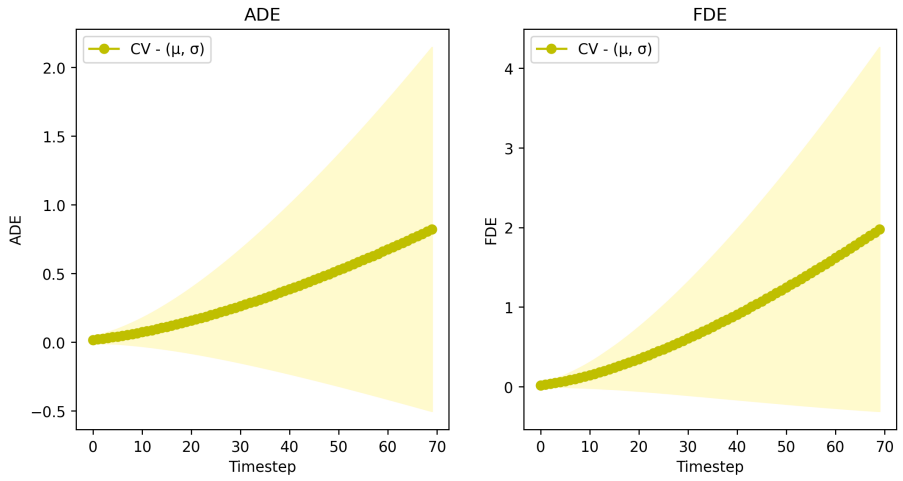

Figure 6.5: Performance of the CV model: mean and standard deviation of ADE and FDE at progressive timesteps

6.4.1. Configurations

Three architectures have been considered, with a progressive level of complexity: *i) Vanilla LSTM, ii) LSTM Encoder-Decoder, iii) Conv-LSTM*. The models were trained minimizing

the mean squared error, using a batch size of 128, a learning rate of 10^{-4}, and Early Stopping has been used to avoid overfitting. For each model, the tuning was performed by selecting the best combination of the number of LSTM layers and the number of neurons that provided the best FDE at 4.8s on the test data. The results of the tuning can be examined in Appendix B.

Vanilla LSTM

The first architecture is referred to as *Vanilla LSTM* as it is a simple configuration. Analyzing Fig. 6.7 from left to right, the Input Layer receives the observed trajectory and passes it to the LSTM. A Dropout layer is added between the LSTM and the Dense layer, to contrast overfitting. In the end, a Dense layer is added to obtain the final output. The final Dense layer has a number of neurons of 142 because the target is a vector of coordinates of size $[71, 2]$. The Reshape layer transforms the output of the Dense layer to the target shape.

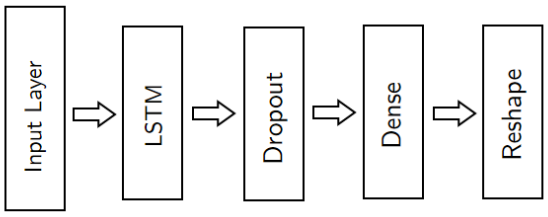

Figure 6.6: Architecture of Vanilla LSTM

LSTM Encoder-Decoder

This architecture, which is commonly used for sequence-to-sequence prediction, adds to the previous one an additional level of complexity. As depicted in Fig. 6.7, a first LSTM layer, the *Encoder*, elaborates an input and produces an encoded fixed-length representation of the trajectory, which is obtained using the RepeatVector layer. Next, an additional LSTM layer, the *Decoder*, further elaborates the encoded state to output a new sequence. The same combination of Dense and Reshape layers is used to produce a vector of the required shape.

Conv-LSTM

This model is a mixed recurrent and convolutional neural network. As the recurrent layers are useful for capturing the dependencies between the positions over time, the convolutional layers can reason over the shape of the trajectories, identifying spatial patterns.

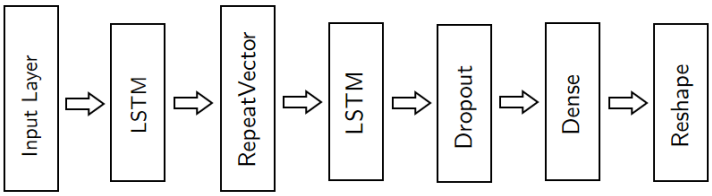

Figure 6.7: Architecture of LSTM-ED

	CV	Vanilla LSTM	LSTM E-D	Conv-LSTM
ADE @ 4.8s - (μ/σ) (m)	0.48/0.77	0.42/**0.63**	0.41/0.64	**0.40/0.63**
FDE @ 4.8s - (μ/σ) (m)	1.14/1.33	1.04/1.08	1.01/1.12	**1.00/1.09**

Table 6.1: Measurements of the mean and standard deviation of ADE and FDE at 4.8s in meters of the presented models

For these reasons, a model that uses both types of layers is relevant for this task. Fig. 6.8 represents the architecture of this model, where the Conv1D and MaxPooling1D operate on the sequences returned by the LSTM. Finally, a GlobalAveragePooling1D layer is used instead of a Flatten layer to reduce the number of weights in the Dense layer and therefore to reduce the chances of overfitting.

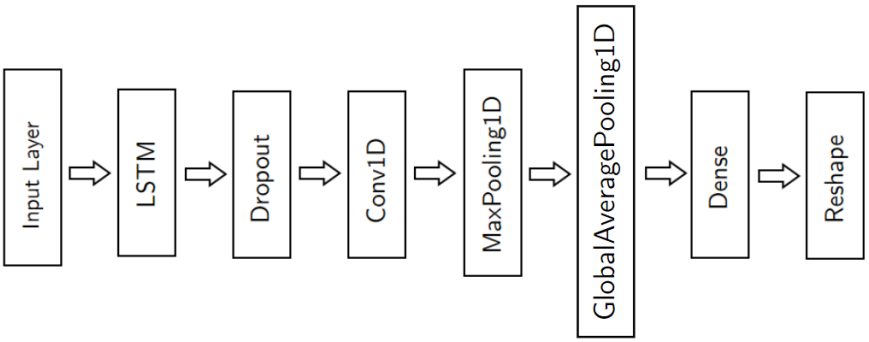

Figure 6.8: Architecture of Conv-LSTM

6.4.2. Model selection

After implementing, training, and selecting the optimal hyperparameters of each configuration, the best DB model was selected by comparing the metrics of each model, evaluated at 4.8s, summarized in Table 6.1. Overall, the Conv-LSTM had the best performance and was selected as the reference for DB models for the rest of this study.

6.4.3. Performance evaluation

As Fig. 6.9 shows, in contrast to the CV model, the Conv-LSTM is able to identify patterns in the observation, and in general, its predictions are more accurate, confirming what depicted in Table 6.1. It is worth noticing that this model is capable of recognizing a curved trajectory only if it starts during the observation window. For example, Sample (0,0) has a curve that starts after the observation ends and could not be predicted by the model.

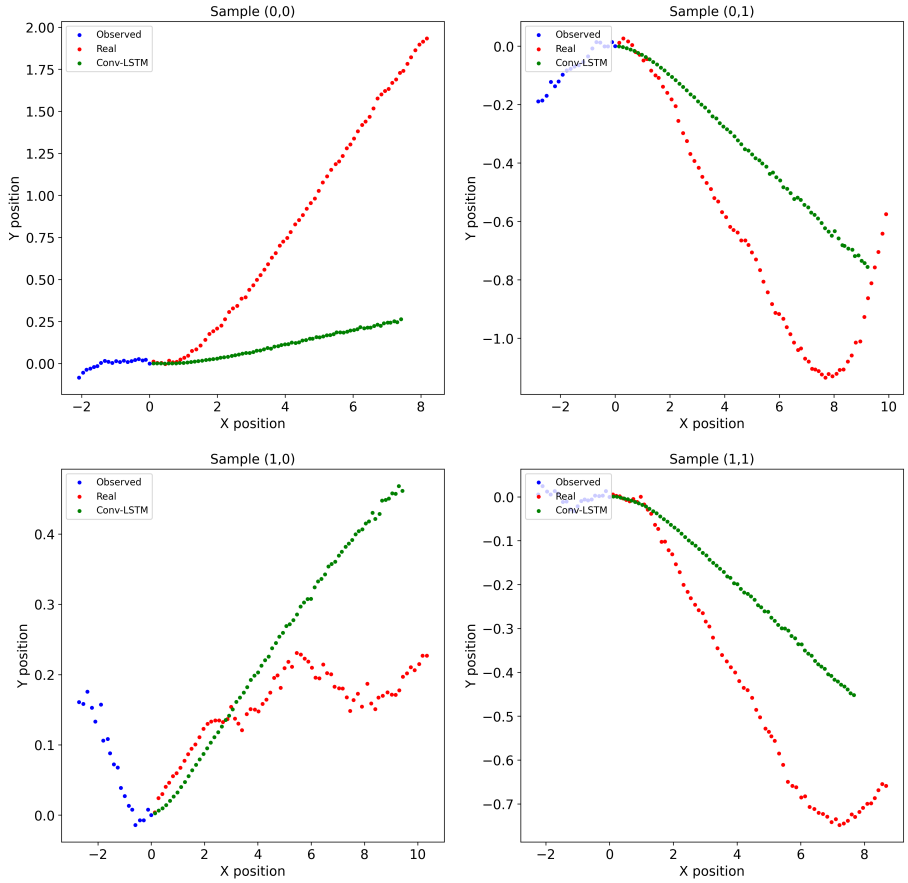

Figure 6.9: Predictions of Conv-LSTM model on clean data

Lastly, the ADE and FDE were computed on the outputs of the Conv-LSTM for the entire prediction horizon, as reported in Fig. 6.10. Also in this case, the mean and standard deviation of the metrics increase exponentially as the prediction horizon increases.

6.5. Conclusion

In this chapter, a selection of PB and DB pedestrian motion prediction models was designed, implemented and tested on clean trajectories extracted from the Waymo Open

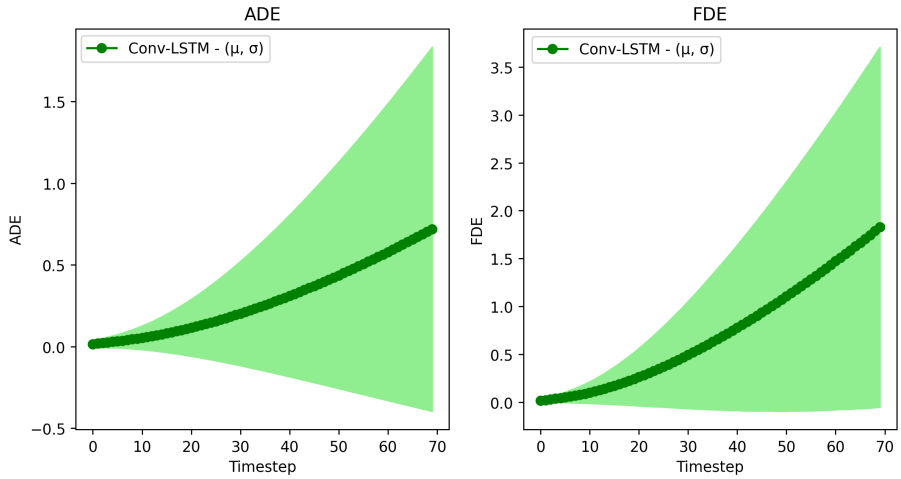

Figure 6.10: Performance of Conv-LSTM model on clean data

Motion Dataset. The CV model has the advantage to be lightweight, and since it doesn't need any type of pre-processing, it is very flexible. On the other hand, despite its higher complexity, the Conv-LSTM was the one able to predict trajectories with the best accuracy metrics. However, on longer prediction horizons, the quality of the predictions degrades consistently for both models. Under the conditions set for this experiment, a hybrid combining the two models analyzed is not necessary to improve the results.

7 | Prediction on realistic data

In the previous chapter, the performance of the two models was evaluated under ideal conditions, meaning that the pedestrian was correctly tracked for the entire observation time, in absence of obstructed views and sensor failures. As mentioned in Section 6.1, each position has an *observed* flag, that determines if the position is valid or not, and therefore, a full observation would look like the one in Fig. 7.1, where for each position in the observation, the flag is set to 1.

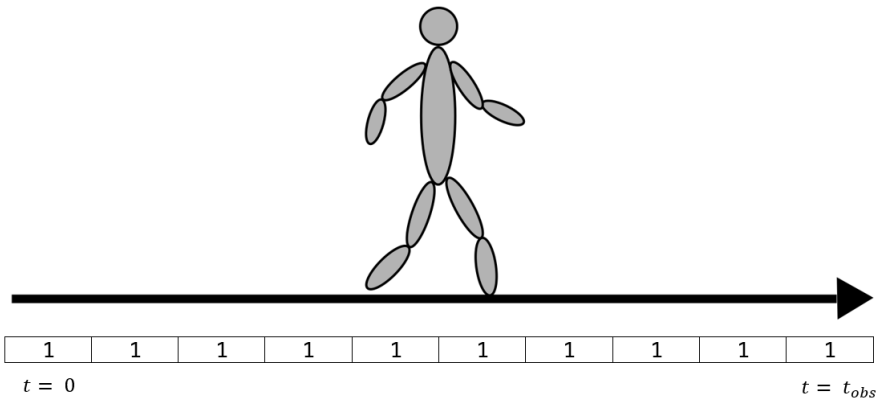

Figure 7.1: Prediction on clean data

However, on realistic data, this is hardly the case. Pedestrians may be hidden behind objects, which would make them invisible to the AV's field of view, and adverse meteorological conditions such as rain or direct sunlight can alter the perception of the sensors, which could fail to track a pedestrian. In these cases, the pedestrian is temporarily lost, and the *observed* flag is set to 0. Depending on the quantity and the position of the invalid positions in the observation, the quality of the predictions has a chance of degrading critically. In the case that the missing observations were not detected. A robust prediction model should be aware of these challenges and its design should properly address them in order to make accurate predictions, even in the most extreme cases, where only a few positions were recorded. In the following sections, a few types of typical prediction tasks in non-ideal conditions are presented, and a hybrid model is designed to deal with realistic

data.

7.1. Data alterations

In this section, three types of non-ideal observations are presented. These alterations are representative of cases that may happen when driving in real life.

Missing beginning

This alteration simulates the case of a pedestrian hidden behind an object and is only visible in the last part of the observation window, also known as late detection. An example including this alteration is the case of a pedestrian trying to cross while being hidden by a car parked on the side of the road. Fig.7.2 represents the altered observation, showing that the pedestrian is detected and tracked only when they are visible to the sensors.

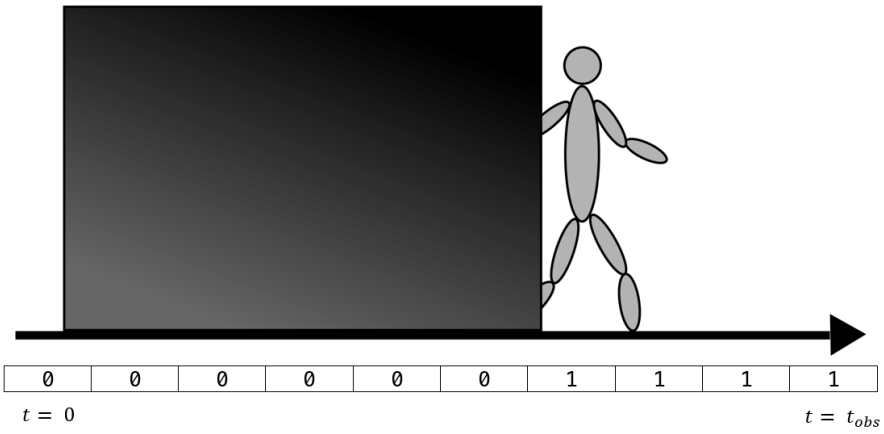

Figure 7.2: Missing beginning

Missing end

As pictured in Fig. 7.3, this alteration is analogous to the previous one. However, in this case the observation was interrupted at the end of its time window. This alteration may have different meanings. In the best case, it implies that a pedestrian exited the field of view of the vehicle and does not represent a threat to the AV anymore. In the worst case, it means that a pedestrian is hidden behind an object and could unexpectedly cross.

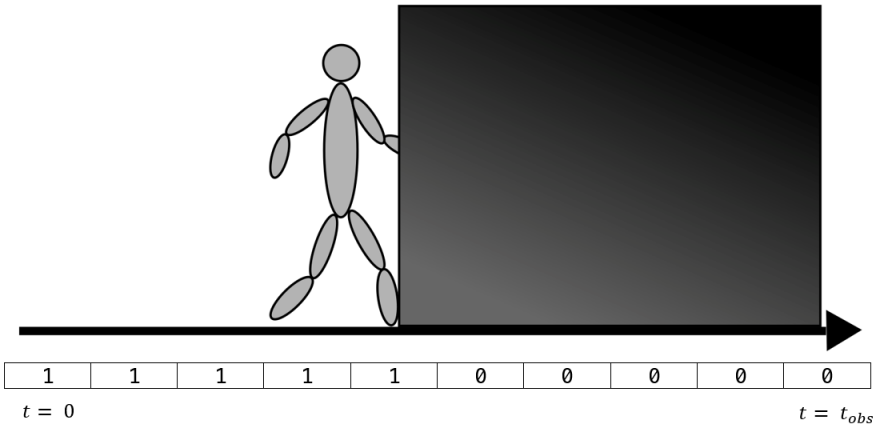

Figure 7.3: Missing end

Missing random

In this case, depicted in Fig.7.4, some data points were collected, while others were not. This alteration simulates unstable tracking due to any factor that may have influenced the perception of the sensors, or their failure.

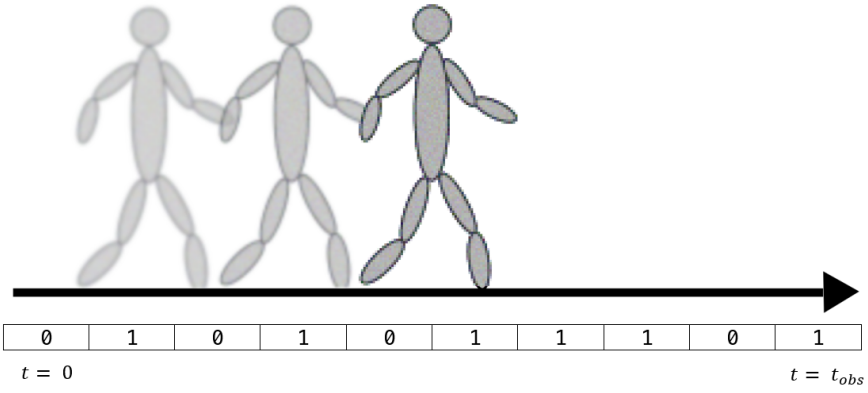

Figure 7.4: Missing random

7.1.1. Threat to the prediction models

After presenting the alterations, their threat to the prediction models introduced in the precious chapter is discussed. The first implication of using a different kind of data is that the models need to be re-tuned to reset their parameters. Next, assuming that the raw data is collected in real time by the AV's sensors and that it needs to be pre-processed before using it as input to the neural network, the *missing end* and *missing random* observations introduce a challenge in their handling. As seen in Section 6.1, the pre-processing needed for a trajectory to be normalized relies on the last observed position,

and if this is missing, the process would fail.

7.2. Hybrid model

In this section, possible solutions to overcome the problem described above are discussed. For long-term predictions, the Conv-LSTM model is better than CV in terms of the mean and standard deviation of the metrics chosen. To achieve this goal, two approaches are evaluated: *i) Shifting the observation* and *ii) Completing the observation.*

7.2.1. Shifting the observation

The most trivial option to have a valid position at the end of the observation would be to right-shift the positions by an offset equal to the number of missing positions at the end of the array, and fill the left part of the observation with invalid positions. This method would only require one simple operation and would be fast to compute. However, as it fixes one problem, it would introduce another, since the predictions would have to be shifted by an offset equal to the number of missing positions, and different output trajectories would be valid for a different number of future timesteps. This situation is not ideal because we would like to have a prediction of every agent for a set prediction horizon so that the motion planner in the following step of the pipeline can have a complete overview of the scene. This problem could be easily solved using CV to complete the missing positions in the prediction. However, if the trajectory is not straight, the completed positions would not represent the real behavior.

7.2.2. Completing the observation

Completion using Conv-LSTM

Considering the results of experiments done in the previous chapter, the Conv-LSTM model would be the choice that would grant the best accuracy. However, the benefits in accuracy would be mitigated by the need for a cumbersome procedure to handle the operation. In this case, the observation vector should be shifted using the process explained in 7.2.1, and the prediction should be clipped and added to the original observation, in the place of the invalid positions. Finally, the observation would be complete, and ready to be used as input to the final prediction model. The downside of this method is that it requires some heavy processing, by executing many array operations and transformations to different sets of coordinates, which will eventually increase the run-time for an operation that could be trivial if only a few final positions are missing.

Completion using CV

Another approach that can be considered is using CV to complete the missing observations. The greater flexibility of this model allows it to compute predictions without any kind of pre-processing and, as seen in the previous chapter, in the very short term its performance is comparable to the Conv-LSTM. For this reason, this model is the most suitable for the considered task. After completing the track using the CV model, it can be successfully pre-processed and the Conv-LSTM model can be used to predict further into the future, exploiting its better prediction power. An approach of this kind is *hybrid* because it combines models of different kinds, and has the potential to deliver good performance, exploiting the flexibility of a physics-based model and the prediction power of a data-based model. This model will be referenced as *Hybrid Conv-LSTM* (HC-LSTM).

7.3. Model re-tuning

As seen in the previous sections, altered data has many added challenges with respect to the clean data. Therefore, the models need to be re-tuned so that they can be used at their full potential.

7.3.1. Dataset with realistic trajectories

To simulate realistic, non-ideal trajectories, a new dataset was created. To resemble the alterations presented in 7.1, the clean trajectories composing the original dataset were artificially processed in order to contain a random number of invalid positions, in specific locations according to the definitions given. The final dataset was constructed to contain an equal number of trajectories for each of their kind: *i) Clean*, *ii) Missing beginning*, *iii) Missing end* and *iv) Missing random*. The tracks which did not have a final position were processed with an additional step to complete them using CV before the normalization, resembling the action performed during its real-time operation. The positions completed with this method were labeled with the validity flag equal to 2, to explicitly report that they were completed by the trajectory completion algorithm, and provide more information during the training phase of the neural network. Also in this case, the dataset was split into Train, Validation, and Test sets using respectively 70%, 20%, and 10% of the trajectories.

7.3.2. Constant Velocity Model

The version of the CV used in this section is the same one used in the previous chapter. Also in this case, the model was tuned by monitoring k, the size of the history used to compute the velocities.

In this case, Fig.7.5 shows that the optimal value of k is 6, one sample less than needed when predicting on clean data. This might be the case because the incomplete trajectories are completed using the proposed method. Therefore, the last data points already contain the average velocities.

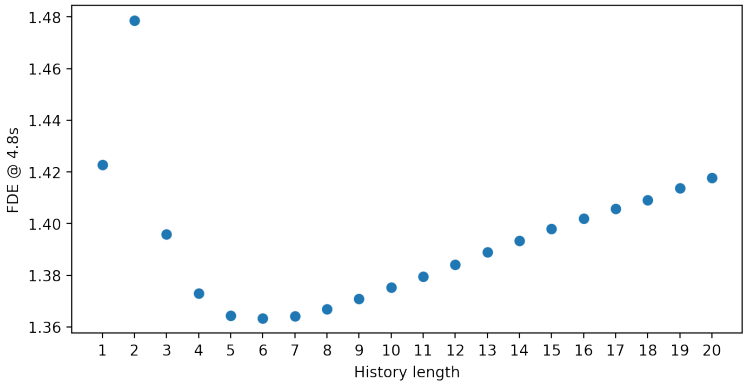

Figure 7.5: Re-tuning of CV model

Using the same approach used in the previous chapter, the mean and the standard deviation of the model were computed for the entire prediction horizon, as shown in Fig. 7.6. Also in this case the performance decreases exponentially as the number of future timesteps increases.

7.3.3. Hybrid Conv-LSTM

After training the Hybrid Conv-LSTM, the mean and standard deviation for the entire prediction horizon are reported in Fig. 7.7. Once again, the performance decreases exponentially as the number of future timesteps increases.

7.4. Performance analysis

After re-tuning the models, we can finally measure their performance and compare it to the case of prediction on clean trajectories. In Table 7.1 we can see the ADE and FDE computed for each model at a prediction horizon of 4.8s. As expected, predicting realistic trajectories harms the performance of the models. Comparing the performance

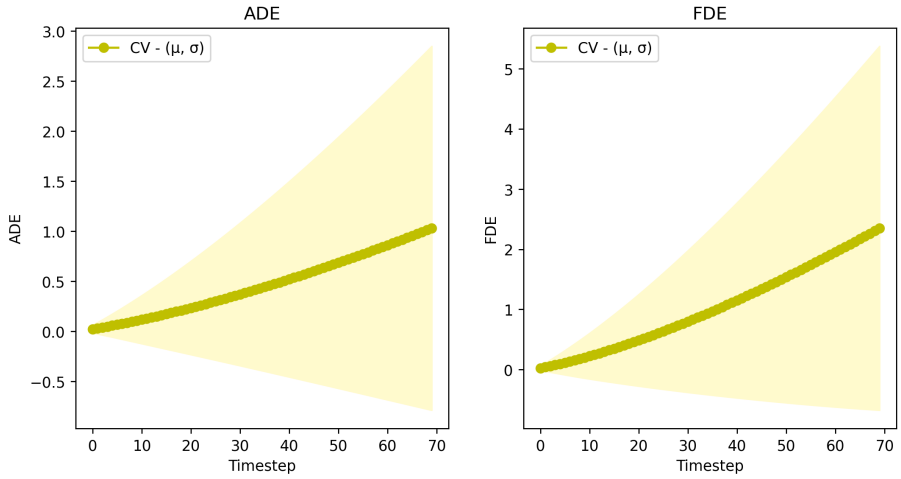

Figure 7.6: Performance of CV on alterations

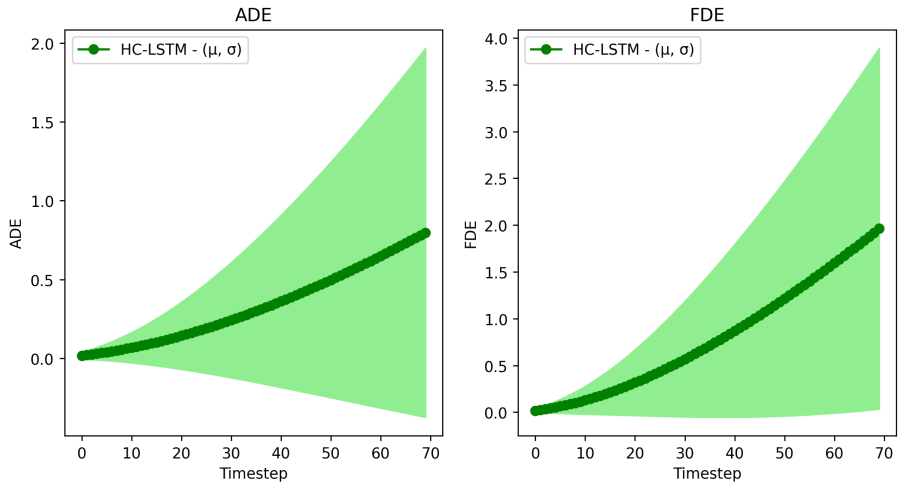

Figure 7.7: Performance of HC-LSTM on alterations

degradation from clean to realistic trajectories, for the CV the mean of the ADE and FDE at a prediction horizon of 4.8s degrade respectively by 25% and 19%, while for the HC-LSTM the same metrics degrade by 13% and 9%. This means that the HC-LSTM model is more robust to alterations in general.

7.4.1. Performance on alterations

To further assess the robustness of the models, an additional evaluation was performed, measuring the ADE and FDE at each timestep of the prediction, for trajectories with an increasing number of invalid positions. To create the datasets, the test dataset on clean data was processed using the same strategy described in 7.3.1. However, in this case, instead of creating a single test dataset containing trajectories with a random number

	CV - Clean	HC-LSTM - Clean	CV - Realistic	HC-LSTM - Realistic
ADE @ 4.8s - (μ/σ) (m)	0.48/0.77	0.40/0.63	0.60/1.13	0.45/0.68
FDE @ 4.8s - (μ/σ) (m)	1.14/1.33	1.00/1.09	1.36/1.88	1.09/1.16

Table 7.1: Comparison of the mean and standard deviation of ADE and FDE at $4.8s$ in meters between clean and altered data

of invalid positions, as done for the training of the models, 19 datasets were created for each alteration. Each of these datasets contained an additional invalid position, starting from 1 up to 19. This analysis shows which type of realistic prediction task is the most challenging and allows us to measure the performance drop from minimal alterations up to extreme cases when only a few positions were recorded.

Missing beginning

In this case, the performance of the models, depicted in 7.8 shows that the performance of the CV, as expected, starts its degradation when the observation contains less than 6 points, that is the history that the model considers. We can see a similar trend in the HC-LSTM, meaning that, when considering this kind of trajectories, a history longer than 0.5 seconds does not determine a considerable improvement of the prediction. Interestingly, the HC-LSTM is still capable of offering greater performance than the CV even when only the last data point was correctly recorded.

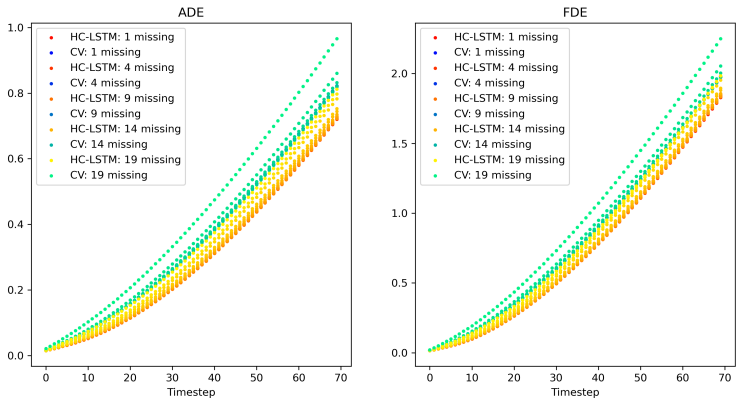

Figure 7.8: Performance on Missing beginning

Missing end

Analyzing the metrics in this case is important to understand the performance of the HC-LSTM for the purpose it was designed: prediction on data completed using the hybrid completion framework. As reported in Fig. 7.9, the HC-LSTM shows a considerable

improvement over the CV, showing that the hybrid model is able to exploit the synergy between the flexibility of the CV and the performance of the Conv-LSTM.

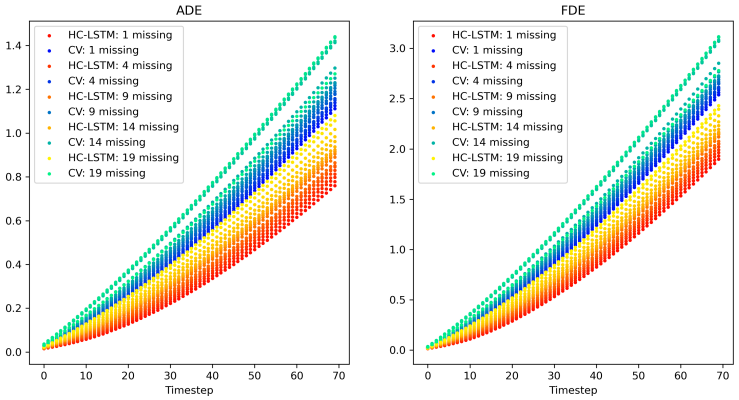

Figure 7.9: Performance on Missing end

Missing random

The last comparison was made between different cases that may happen in the *Missing random* alteration type. The most general comparison, reported in Fig 7.10, was made on sets composed of fully random positions. Due to the randomness, trajectories in these sets can be both complete and incomplete. Therefore, the incomplete ones were completed using the hybrid framework before their pre-processing. A similar trend can be noted in this section as well. A further comparison of other types of random trajectories is reported in Appendix C.

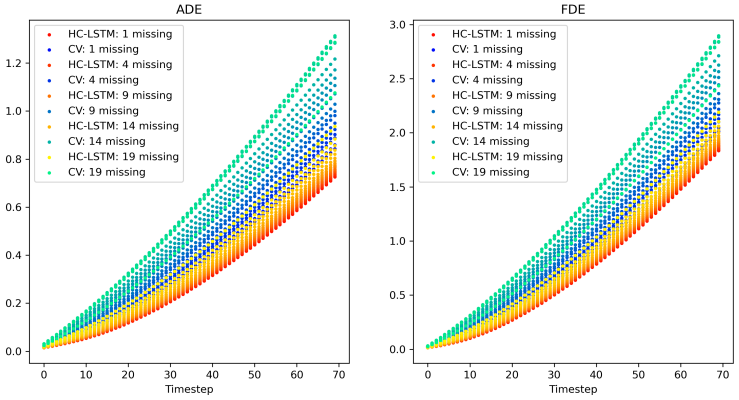

Figure 7.10: Performance on Missing random

7.5. Conclusion

In this section, the main challenges that AVs face during the prediction on realistic data were presented and discussed. The prediction using the Conv-LSTM is dependent on the last position in an observation. A hybrid model for pedestrian trajectory prediction was presented to handle this kind of incomplete observations. This model is suitable for the prediction on realistic trajectories by exploiting the flexibility of the constant velocity model and the accuracy of the Conv-LSTM. Its predictions were finally shown to be more robust to alterations with respect to the CV baseline, especially in the most extreme cases, when only a few positions were recorded.

8 | Prediction in real-time

After analyzing the challenges coming from prediction on realistic data and designing a hybrid model able to overcome them, in this chapter, an architecture for a real-time pedestrian motion prediction model is introduced. Finally, a section is dedicated to the implementation of this architecture on an AV, to demonstrate that an architecture of this kind is suitable for prediction in real-time.

8.1. Prediction framework

In the previous chapter, the realistic dataset composed of altered trajectories was tested on offline data. In this section, a framework to handle the alterations in real time is presented. This framework extends the pipeline presented in Section 2.1, enhancing the capabilities of the Scene Representation block by introducing a module to robustly predict non-ideal trajectories.

As represented in Fig. 8.1, a trajectory could be predicted using Model A: Hybrid Conv-LSTM, or Model B: CV. Depending on the computation capabilities of the vehicle, one model could be preferred over the other. In the case of CV prediction, the prediction is produced directly and no further action is required. On the other hand, the HC-LSTM needs an initial condition to decide if the trajectory needs to be completed using CV completion. If the last position in the observation was successfully recorded, then this is not necessary, and the trajectory is pre-processed directly. Otherwise, CV completion is needed as an intermediate step.

8.2. Experiment

This section reports the experience obtained during the experiment, where the prediction model was implemented on an AV. This experiment had the goal of demonstrating that the prediction model presented is suitable for the task of real-time prediction. The test vehicle used for the experience is a Volkswagen Jetta with an Axiomtek embedded PC.

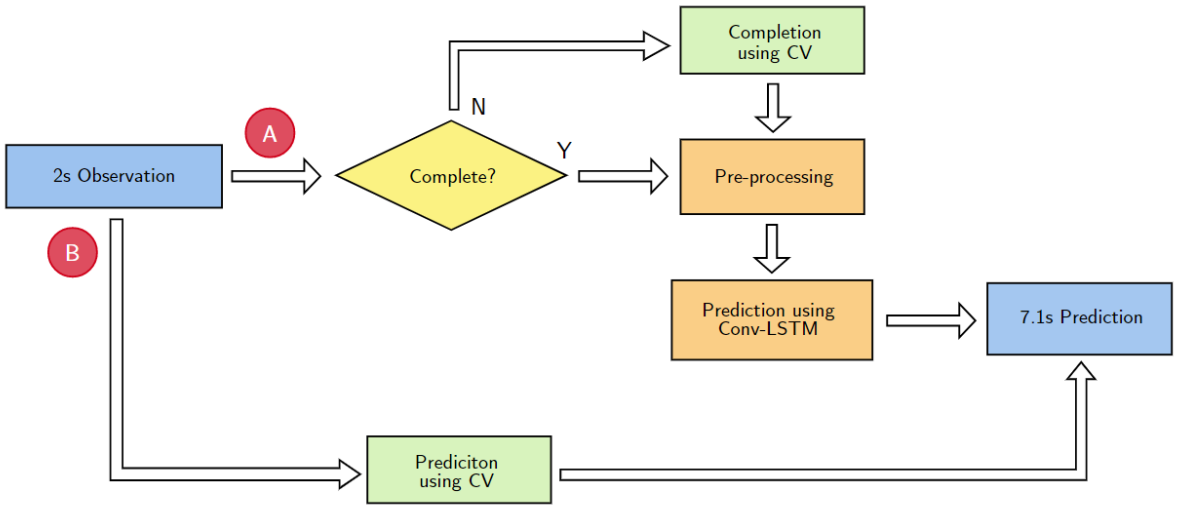

Figure 8.1: Real-time prediction framework

The sensor stack included a Mobileye 550 camera, a Continental radar, and embedded IMU and wheel encoders. The prediction model used ROS 1 as middleware. Fig. 8.2 depicts the real-time prediction model executing in real-time in the vehicle. On the left, the image from the camera is reported, showing that the pedestrian is correctly tracked and their future positions, produced by the hybrid model, are marked with red dots in the scene. On the right, the world representation of the vehicle is reported, showing its abstraction of the scene.

Figure 8.2: Predictions in real-time from the AV point of view.

8.3. Conclusion

In this chapter, a real-time trajectory prediction architecture was presented and tested on an AV. The experiment showed that the presented model is capable of being run by an AV in real time.

9 | Conclusions

This thesis aimed at designing a hybrid model for pedestrian motion prediction. This model improved pedestrian trajectory prediction under non-ideal conditions and is suitable for real-time applications in AVs. In this final chapter, the main contributions of this work are presented. Finally, recommendations for future work are given.

9.1. Main contributions

The first contribution of this research is a comparison of the trajectories found on a sample of AV motion datasets and pedestrian datasets. This analysis was performed using the OpenTraj framework, and highlights the differences between the two kinds of datasets, especially in terms of the complexity of the trajectories, given by greater deviation from linear motion and speed variations, and lower predictability. The higher complexity of the trajectories implies that the prediction task using this kind of data is more challenging and that AV motion datasets are more suitable than pedestrian datasets for motion prediction in urban scenarios. Further research in the area of pedestrian trajectory prediction for AVs should use these datasets because they allow for the development of models more suited for the pedestrian data an AV encounters.

The second contribution of this research is a benchmark of physics-based and pattern-based pedestrian trajectory prediction models. Under ideal conditions, when an agent is correctly observed for the entire duration of the observation window, all the examined neural networks outperformed the constant velocity baseline in short-term and long-term prediction. Among the neural networks studied, the Conv-LSTM had the best accuracy. At a prediction horizon of $4.8s$, the mean of the ADE and the FDE of the Conv-LSTM were respectively 20% and 14% lower than the ones of the CV.

Next, a hybrid model for the prediction of realistic trajectories is presented. Prediction on this type of trajectories is more difficult due to the limited number of valid positions they contain and the additional step required to pre-process them. However, a hybrid

model was designed to perform this task by exploiting the flexibility of the CV, to complete the missing observations, and the pattern recognition abilities of the Conv-LSTM, to make accurate predictions. The accuracy of predictions on realistic data was measured including the most severe cases, on highly degraded data. Under these conditions, the mean of the ADE and the FDE of the HC-LSTM at 4.8s were respectively 33% and 25% lower than the ones of the CV. Moreover, comparing the performance degradation from clean to realistic trajectories, for the CV the mean of the ADE and FDE at 4.8s degrade respectively by 25% and 19%, while for the HC-LSTM the same metrics degrade by 13% and 9%. This means that the HC-LSTM model is more accurate and more robust to alterations in general.

Finally, a novel real-time motion prediction architecture is proposed, to allow AVs to predict realistic trajectories using the proposed hybrid model. This framework was successfully integrated in the pipeline of a test vehicle and shown to be functional.

9.2. Future works

To further improve the accuracy and the reliability of the presented model, in this section recommendations for future works are given.

This study included a robustness analysis of data alterations typical of real-world driving scenarios. These alterations were selected considering the situations that may limit the perception of the AV's sensors, assuming they were working correctly, or at least that the positions collected were valid. Different types of alterations, that may happen in case of other hardware failures, were not investigated in this work. These cases, which may include errors in the calibration of the sensors, should be addressed by future research, to have a broader view of the limitations and the robustness of prediction models.

Another research direction would involve an extension of this model to include context awareness and intention prediction. Context-aware models are more complex because they can reason on the surrounding environment and the movement of other agents in the scene. On the other hand, analyzing the visual features of a target can reveal their intentions, and make future behaviors more or less probable. In this way, further improvement in prediction accuracy is expected.

Finally, future work should be directed toward the deployment of the model in a vehicle. This study included a demonstration of an integration of a hybrid pedestrian motion pre-

diction model in the pipeline of an AV, but several more aspects of its operation should be studied. In particular, it is needed to investigate the issues related to data retrieval, understand the limitations from a hardware perspective, and identify the challenges of prediction in real time. Once these aspects are verified, it will be possible to improve the performance of the model on real-time prediction.

Bibliography

[1] Honda launches world's first level 3 self-driving car - nikkei asia. https://asia.nikkei.com/Business/Automobiles/Honda-launches-world-s-first-level-3-self-driving-car.

[2] Road accident fatalities - statistics by type of vehicle - statistics explained. https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Road_accident_fatalities_-_statistics_by_type_of_vehicle.

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[4] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[5] J. Amirian, B. Zhang, F. V. Castro, J. J. Baldelomar, J.-B. Hayet, and J. Pettré. Opentraj: Assessing prediction complexity in human trajectories datasets. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[6] A. Antonucci, G. P. R. Papini, L. Palopoli, and D. Fontanelli. Generating reliable and efficient predictions of human motion: A promising encounter between physics and neural networks. *arXiv preprint arXiv:2006.08429*, 2020.

[7] F. Bartoli, G. Lisanti, L. Ballan, and A. Del Bimbo. Context-aware trajectory prediction. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1941–1946. IEEE, 2018.

[8] I. Batkovic, M. Zanon, N. Lubbe, and P. Falcone. A computationally efficient model

for pedestrian motion prediction. In *2018 European Control Conference (ECC)*, pages 374–379. IEEE, 2018.

[9] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. *CoRR*, abs/1903.11027, 2019. URL `http://arxiv.org/abs/1903.11027`.

[10] T. Chavdarova, P. Baqué, S. Bouquet, A. Maksai, C. Jose, T. Bagautdinov, L. Lettry, P. Fua, L. Van Gool, and F. Fleuret. Wildtrack: A multi-camera hd dataset for dense unscripted pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5030–5039, 2018.

[11] E. Commission. Road safety in the european union: Trends, statistics and main challenges. *Technical Report*, 2015.

[12] O.-R. A. D. Committee et al. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *SAE International, Standard J3016*, 2018.

[13] A. Elnagar. Prediction of moving objects in dynamic environments using kalman filters. In *Proceedings 2001 IEEE International Symposium on Computational Intelligence in Robotics and Automation (Cat. No. 01EX515)*, pages 414–419. IEEE, 2001.

[14] C. M. Farmer. Relationships of frontal offset crash test results to real-world driver fatality rates. *Traffic injury prevention*, 6(1):31–37, 2005.

[15] K. Fitzpatrick, M. A. Brewer, and S. Turner. Another look at pedestrian walking speed. *Transportation research record*, 1982(1):21–29, 2006.

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[17] H. Girase, H. Gang, S. Malla, J. Li, A. Kanehara, K. Mangalam, and C. Choi. Loki: Long term and key intentions for trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9803–9812, 2021.

[18] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

[20] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. *CoRR*, abs/1803.06184, 2018. URL `http://arxiv.org/abs/1803.06184`.

[21] keras team. Keras. `https://github.com/keras-team/keras`, 2022.

[22] B. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. Sallab, S. Yogamani, and P. Pérez. Deep reinforcement learning for autonomous driving: A survey. 02 2020.

[23] R. Korbmacher and A. Tordeux. Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches. *arXiv preprint arXiv:2111.06740*, 2021.

[24] P. Kothari, B. Sifringer, and A. Alahi. Interpretable social anchors for human trajectory forecasting in crowds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15556–15566, 2021.

[25] I. Kotseruba, A. Rasouli, and J. K. Tsotsos. Joint attention in autonomous driving (jaad). *arXiv preprint arXiv:1609.04741*, 2016.

[26] A. Lerner, Y. Chrysanthou, and D. Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007.

[27] Z. C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[28] A. K. Lund and S. A. Ferguson. Driver fatalities in 1985-1993 cars with airbags. *Journal of Trauma and Acute Care Surgery*, 38(4):469–475, 1995.

[29] S. Malla, B. Dariush, and C. Choi. Titan: Future forecast using action priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11186–11196, 2020.

[30] P. Mallozzi, P. Pelliccione, A. Knauss, C. Berger, and N. Mohammadiha. Autonomous vehicles: state of the art, future trends, and challenges. *Automotive Systems and Software Engineering*, pages 347–367, 2019.

[31] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund. Trajectory analysis and prediction for improved pedestrian safety: Integrated framework and evaluations. In *2015 IEEE intelligent vehicles symposium (IV)*, pages 330–335. IEEE, 2015.

[32] K. Myny, A. K. Tripathi, J.-L. van der Steen, and B. Cobb. Flexible thin-film nfc tags. *IEEE Communications Magazine*, 53(10):182–189, 2015.

[33] R. Okuda, Y. Kajiwara, and K. Terashima. A survey of technical trend of adas and autonomous driving. In *Technical Papers of 2014 International Symposium on VLSI Design, Automation and Test*, pages 1–4. IEEE, 2014.

[34] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE, 2009.

[35] M. Pfeiffer, U. Schwesinger, H. Sommer, E. Galceran, and R. Siegwart. Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2096–2101. IEEE, 2016.

[36] A. Poibrenski, M. Klusch, I. Vozniak, and C. Müller. M2p3: multimodal multi-pedestrian path prediction by self-driving cars with egocentric vision. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 190–197, 2020.

[37] A. Rasouli, I. Kotseruba, T. Kunic, and J. K. Tsotsos. Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6262–6271, 2019.

[38] A. Rasouli, M. Rohani, and J. Luo. Bifold and semantic reasoning for pedestrian behavior prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15600–15610, 2021.

[39] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pages 549–565. Springer, 2016.

[40] J. J. Rolison, S. Regev, S. Moutari, and A. Feeney. What are the factors that contribute to road accidents? an assessment of law enforcement views, ordinary drivers' opinions, and road accident records. *Accident Analysis & Prevention*, 115: 11–24, 2018.

[41] A. Rudenko, L. Palmieri, and K. O. Arras. Joint long-term prediction of human motion using a planning-based social force approach. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4571–4577. IEEE, 2018.

[42] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020.

[43] SAE-International. `https://www.sae.org/`.

[44] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll. What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters*, 5(2):1696–1703, 2020.

[45] A. Seyfried, O. Passon, B. Steffen, M. Boltes, T. Rupprecht, and W. Klingsch. New insights into pedestrian flow through bottlenecks. *Transportation Science*, 43(3): 395–406, 2009.

[46] M. Silvestri, M. Lombardi, and M. Milano. Injecting domain knowledge in neural networks: a controlled experiment on a constrained problem. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 266–282. Springer, 2021.

[47] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[48] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, D. Ramanan, P. Carr, and J. Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021.

[49] Y. Yao, E. Atkins, M. Johnson-Roberson, R. Vasudevan, and X. Du. Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *IEEE Robotics and Automation Letters*, 6(2):1463–1470, 2021.

[50] Y. Yao, E. Atkins, M. J. Roberson, R. Vasudevan, and X. Du. Coupling intent and action for pedestrian crossing behavior prediction. *arXiv preprint arXiv:2105.04133*, 2021.

A | Implementation of relevant functions

In this section, the implementation in Python of a few relevant functions is provided. Listing A.1 reports the implementation of the pre-processing functions, to normalize a trajectory with respect to the last observed position. Listing A.2 reports the implementation of the CV model, and Listing A.3 reports the implementation of the CV completion function.

```
import numpy as np

def center_origin(track, pred_time):
    # Normalize xy positions and headings
    # wrt origin (0,0) at timestep pred_time
    track[:, 3] = track[:, 3] - track[pred_time, 3]
    track[:, 4] = track[:, 4] - track[pred_time, 4]

    return track


def rotate(track, pred_time):
    theta = track[pred_time, 5]
    track[:, 5] = track[:, 5] - track[pred_time, 5]

    # Rotate clockwise (theta = heading at prediction time)
    rot_mat = np.matrix([[np.cos(theta), np.sin(theta)],
                         [-np.sin(theta), np.cos(theta)]])

    for i in range(track.shape[0]):
        # Multiply pos_x and pos_y for rot_max
        track[i, 3:5] = rot_mat @ track[i, 3:5]
```

```
        # Same for velocities
        track[i, 6:] = rot_mat @ track[i, 6:]

    return track
```

Listing A.1: Implementation of the translation and rotation pre-processing functions.

```
import numpy as np

def cv_predict(trajectories, history=7, horizon=91):
predictions = np.zeros((trajectories.shape[0],
                        horizon-trajectories.shape[1], 2))
vel = np.zeros((trajectories.shape[0], 2))
for i in range(trajectories.shape[0]):
    last_5_valid_steps = trajectories[i,
                    trajectories[i, :, 0] != 0][-history:]
    vel[i, 0] = np.average(last_5_valid_steps[:, 4])
    vel[i, 1] = np.average(last_5_valid_steps[:, 5])
    predictions[i, 0, 0] = 0.1 * vel[i, 0]
    predictions[i, 0, 1] = 0.1 * vel[i, 1]
    for j in range(1, horizon-trajectories.shape[1]):
        predictions[i, j, 0] = predictions[i, j - 1, 0]
                             + 0.1 * vel[i, 0]
        predictions[i, j, 1] = predictions[i, j - 1, 1]
                             + 0.1 * vel[i, 1]

return predictions
```

Listing A.2: Implementation of the constant velocity model.

```
import numpy as np

def complete(trajectory, pred_time):

supp = trajectory[:pred_time]
i = pred_time - 2

while i >= 0:
    if trajectory[i, 1] == 1:
```

```
            last_valid_index = i
            break
        i += -1

    last_5_valid_steps = supp[supp[:, 1] != 0][-5:]
    vel_x = np.average(last_5_valid_steps[:, 6])
    vel_y = np.average(last_5_valid_steps[:, 7])
    theta = np.average(last_5_valid_steps[:, 5])

    for j in range(i + 1, pred_time):

        trajectory[j, 1] = 2
        trajectory[j, 6] = vel_x
        trajectory[j, 7] = vel_y
        trajectory[j, 3] = trajectory[j - 1, 3]
                           + vel_x * 0.1
        trajectory[j, 4] = trajectory[j - 1, 4]
                           + vel_y * 0.1
        trajectory[j, 5] = theta

    return trajectory
```

Listing A.3: Implementation of the CV completion function.

B | Tuning of the Neural Networks

In this section, the results of the model selection of the neural networks is provided. For each figure, the FDE at $4.8s$ for each combination of layers and number of neurons is reported. Fig B.1, Fig. B.2 and Fig. B.3 respectively represent the training of the Vanilla LSTM, the LSTM-ED and the Conv-LSTM on clean data. Finally, Fig. B.4 shows the training of the HC-LSTM.

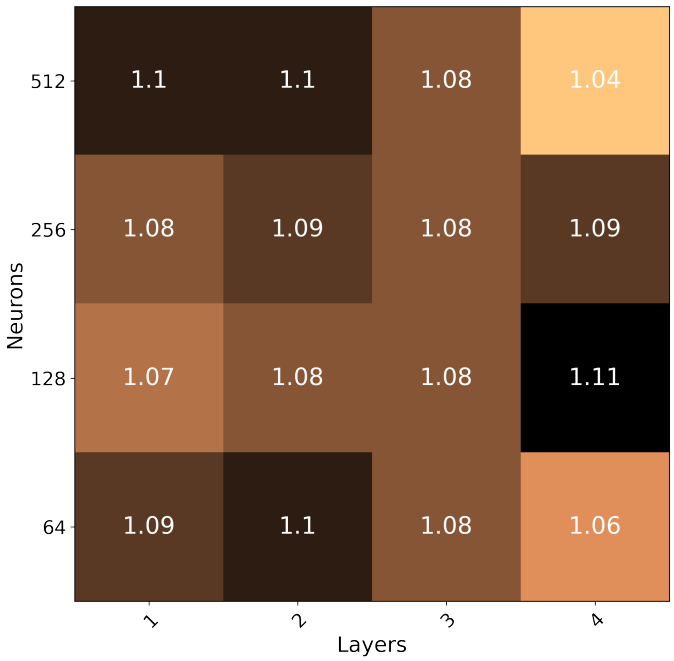

Figure B.1: Training of Vanilla LSTM model

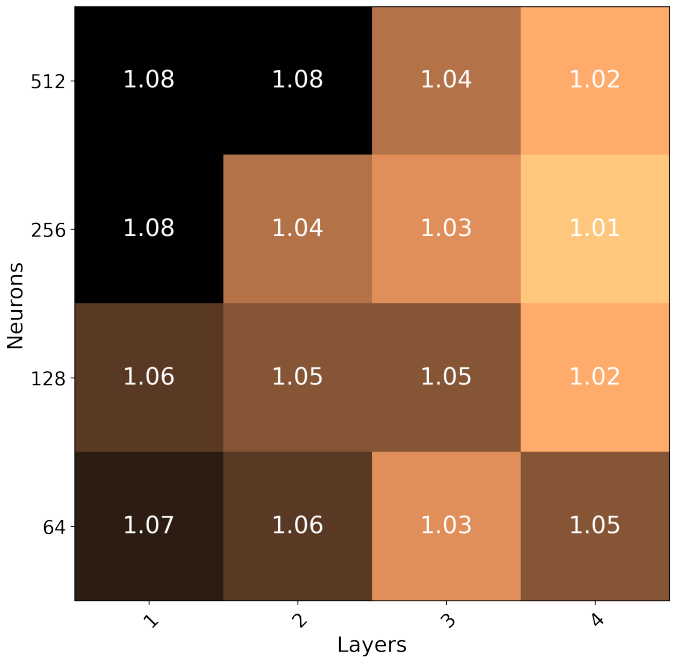

Figure B.2: Training of LSTM Encoder-Decoder model

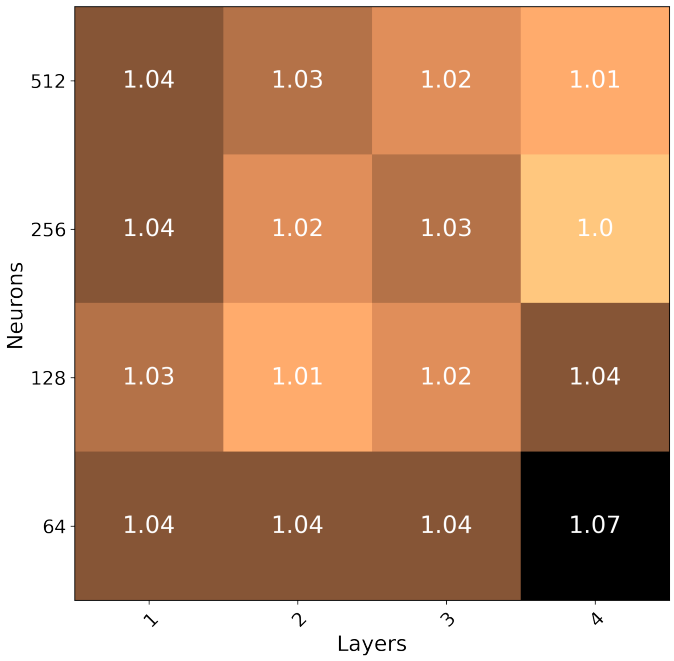

Figure B.3: Training of Conv-LSTM model

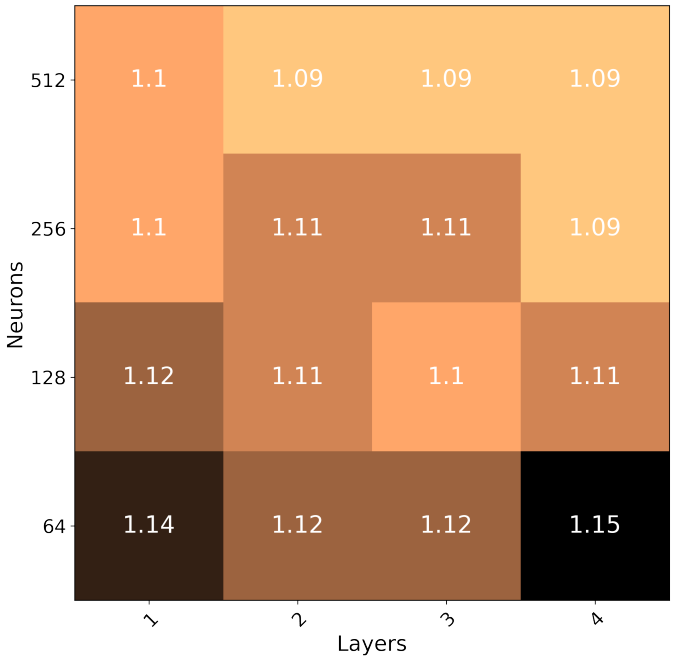

Figure B.4: Re-training of HC-LSTM model

C | Performance comparisons

In this section, the performance comparison of the CV and the HC-LSTM on a subclass of the alterations is reported. These alterations are sub-classes of the *missing random*, where for Fig. C.1, the last position was recorded and no CV completion was needed, and for Fig. C.2, each trajectory had a missing value on the last position.

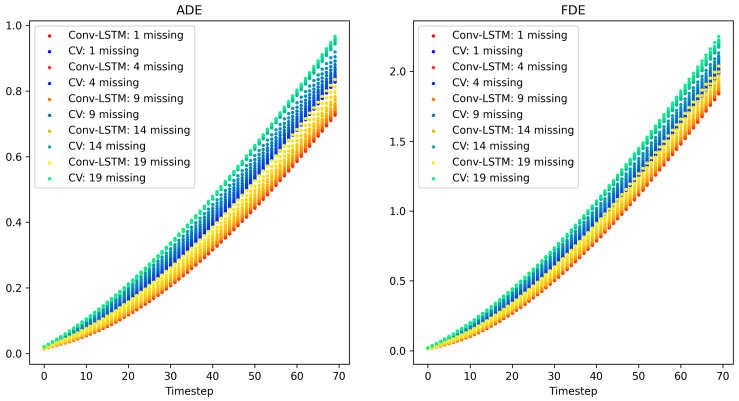

Figure C.1: Performance on Missing random last

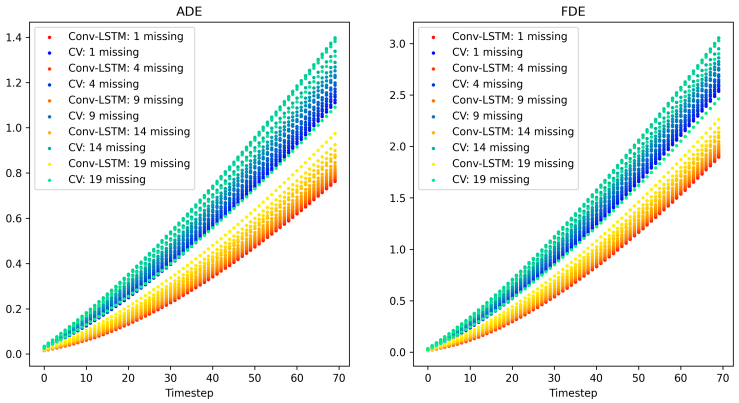

Figure C.2: Performance on Missing random full

List of Figures

List of Tables

Acknowledgements

This thesis marks the last chapter of my path as a student in the EIT Digital Master School, in the Data Science track. During the two years of my Master's, I had the pleasure to study at two prestigious universities in Italy and in the Netherlands, and to create bonds with interesting people from all over the world. I must say that this has not been easy, and especially the last months have been a real challenge to me. However, I am proud that I could always find the motivation to overcome difficulties and successfully finish my studies. Overall it has been an enriching experience and a chance to grow personally and culturally, and I look forward to the next steps in my career and life.

I would like to thank my supervisors from TU/e and TNO, who have guided me through these past months, providing their support, feedback, and ideas during our weekly meetings. I am grateful for the time you dedicated to me.

Next, I would like to thank my friends from EIT, who have made this past year in Eindhoven unforgettable. Thanks to you I have never felt alone during this time away from home.

Without the support of my family, all of this would have not been possible. Thank you for helping me find the motivation when I felt like I could not make it, and for always being there for me.

Last but not least, my biggest thanks go to my dad. Thanks to you I learned to live in the pursuit of continuous improvement, aiming for infinity. I hope you are proud of me.

Enrico Dalboni
Eindhoven, September 2022