User-centered development of a pedestrian assistance system using end-to-end learning

vorgelegt von
M.Sc.
Hasham Shahid Qureshi

von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
-Dr.-Ing.-

genehmigte Dissertation

Promotionsausschuss:
Vorsitzerin: Prof. Dr. Begüm Demir
Gutachter: Prof. Dr. Olaf Hellwich
Gutachter: Prof. Dr. Tobias Glasmachers
Gutachter: Prof. Dr. Felix Wilhelm Siebert
Tag der wissenschaftlichen Aussprache: 23. Feb 2023

Berlin 2023

Zusammenfassung

Das Ziel dieser Dissertation ist die Entwicklung eines Assistenzsystems, dass ältere Zufußgehende bei der Straßenquerung unterstützt. Dazu wurde ein Algorithmus ausgearbeitet, der den Bordstein aus Perspektive der älteren Zufußgehenden erkennt. Die Borsteinerkennung ist essenziell, um die Straße zu identifizieren und kann darüber hinaus zur Vermeidung anderer Hindernisse und sicherheitsrelevanter Objekte beitragen. Die Herausforderung dabei ist die geringe höhe der Bordsteine, deren unterschiedliche Struktur und andere im Weg befindliche Objekte. Zur Lösung dieser Probleme wurde mit einer Kamera und einem lichtbasierten Abstandssensor auf zwei Detektionswerkzeuge zurückgegriffen, die mit Hifle des End-to-End Learnings fusioniert wurden. Das Convolutional Network wurde entwickelt, um die von der Monokamera beim Filmen des Bordsteins und seiner Umgebung aufgenommenen Bilder zu verarbeiten. Das künstliche neuronale Netz wurde ausgewählt, um die Daten des Abstandssensors zu verarbeiten, die in Form von Arrays aus dessen 16 Kanälen vorliegen. Ein Prototyp wurde für die Datenerfassung und zu Testzwecken entwickelt. Dieser besteht aus einer Halterung, die an einem Rollator befestigt wurde und an der die beiden Sensoren angebracht wurden. Zu Trainingszwecken wurden die Daten beider Sensoren unter Berücksichtigung verschiedener Faktoren wie beispielsweise Wetter, Lichtverhältnissen und Annäherungswinkel an den Bordstein erhoben. Um den Algorithmus zu Trainieren wurde eine End-to-End Learning entwickelt, bei dem die gesamten Bilder bzw. Arrays und nicht etwa einzelne Pixel gelabelt wurden. Die Netzwerke des jeweiligen Sensors wurden trainiert und anschließend miteinander verkettet, um so das gesamte Netzwerk zu trainieren. Die experimentellen Ergebnisse attestieren dem verwendeten End-to-End Learning eine Genauigkeit von mehr als 99%. Beide Sensoren sind kostengünstig und können im Zusammenspiel den Bordstein aus Perspektive der älteren Zufußgehenden erkennen. Um die Nutzenden vor potenziellen Gefahren zu warnen, wurde eine vibrotaktile Schnittstelle entwickelt, die drahtlos mit dem Assistenzsystem und der Technik, die die Sensoren steuert, verbunden ist. Im Rahmen der vorliegenden Dissertation wurde ein Prototyp entwickelt, der potenzielle Gefahren für ältere Zufußgehende erkennt. Dazu wurde erfolgreich ein Alogrithmus mit End-to-End learning entwickelt, der zwei heterogenen Sensoren fusioniert und mit einem Schnittstellenmodul verknüpft, um die Nutzenden zu warnen.

Abstract

The goal of this dissertation is to develop an assistance system for supporting road crossing among older pedestrians. In order to accomplish this, a curb detection algorithm has been proposed from the pedestrians point of view. Curb detection plays a significant role in road detection and obstacles avoidance among other safety related variables. However, it also presents significant challenges such as the small size of the detection target as well as obstacles and different structures of the curb. To tackle these problems, two sensors were selected, a Camera and a distance sensor; a light based Radar. These were fused using the end-to-end learning approach. The convolutional neural network was designed to process the images acquired from the mono camera by filming the curb and its surroundings. The artificial neural network was selected to process the data of the distance sensor acquired in the form of arrays from the 16 channels of the sensor. A prototype was developed for data collection and testing purposes. It consists of a structure carrying both sensors mounted on a walker. The data from both sensors were collected with multiple factors taken into consideration, such as, weather, light conditions and, approaching angles. For the training of algorithms, an end-to-end learning approach was designed where the complete image or array was labeled as the hazardous scenario or not rather than labeling the individual pixels or features in the acquired data. The networks were trained and the features from the parallel networks were concatenated and given as the input to the fully connected layers to train the complete network. The experimental results show an accuracy of more than 99% with the designed end-to-end learning approach. Both sensors are relatively inexpensive and in fusion together are able to efficiently accomplish the task of detecting the curb stone from the pedestrian's point of view. To alarm the user of the potentially hazardous scenario, a vibro-tactile interface was developed which is connected wirelessly with the the machine that also control the sensors. In this dissertation, a prototype was developed to detect the potentially hazardous situations for the older pedestrian's while walking on the street. An algorithm with end-to-end learning approach to fuse two heterogeneous sensor was successfully designed and the alarms generated from the algorithm were transmitted to an interface module to alert the user.

For my family...

Acknowledgements

I would like to take this opportunity to express my heartfelt thanks to all those who have contributed in various ways to the realization of this work. First and foremost, I would like to express my deepest gratitude to Professor Dr. Hellwich, who as a supervisor and first examiner had supported me in all the phases with his knowledge and experience. With his professional competence and openness to new ideas, he provided me with a platform where it was possible for me to implement my own ideas. I am deeply grateful to Professor Dr. Glasmachers, who as a supervisor and second examiner, made a significant contribution to this work in terms of the ideas, technical support and continuous positive feedback. His insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. I would like to thanks the professors again that despite all the postponements and imponderables, they always showed unwavering support and belief in me throughout this dissertation. I would like to offer my special thanks to Dr. Felix Siebert for his insightful suggestions and comments, which helped me extensively to compile this work in the best way possible. My gratitude extends to the junior research group FANS for the funding opportunity to undertake my studies. I would also like to extend my sincere thanks to the group leader of FANS, Dr. Rebecca Wizcorek, for her treasured support which was really influential in shaping my experiment methods and critiquing my results.

Furthermore, my thanks go to the doctoral students of the junior research group FANS, who worked together with me for the success of the project and who are pursuing their doctoral projects in different disciplines. Especially Dr. Florian Breitinger, with whom it was an honor to work together in the same project. He had always inspired and motivated me with the thoughtful discussions and helped me with revealing new angles and perspectives on my research. His invaluable support helped me deal with the unavoidable intangibles of a doctorate. I would also like to thank Ms. Rima Akil, Mr. Youssef Bagueri and Mr. Benjamin Paulisch, who as technical staff, were an important part of FANS and always supported me competently with all the technical situations. I would also like to thank the student assistants who worked at FANS and who also supported me in my work.

Finally, I would like to thank my family and friends, who were always there for me before and during my doctorate and whose support all through my studies. I would like to thank my parents and my brothers Mr. Muhammad Ibtisam Qurashi and Mr.

Muhammad Waqas Shahid for their lifelong and unconditional love and support, which has made everything I have achieved so far possible. I would also like to thank them for helping me to develop an open, critical and reflective mind. I would also like to thanks my friends, for their inspiring strength and support, which made me realize the proportionality of my efforts and their contribution in different ways to the success of this work.

By far the greatest thanks, however, are due to my wife Aqsa Khan. From the countless discussions we had about the work, to the sacrifice of countless evenings which we could have spent together, her contribution to this work cannot be overestimated. I hope I can make it up for it one day. Without her tremendous understanding and encouragement in the past few years, it would have been impossible to complete it. Thank you for always being there for me!

Table of Contents

List of Figures

List of Tables

1

Introduction

To support the older pedestrians in the street environment and traffic situations, an assistance system was planned by the research group FANS (Fußgänger Assistenzsystem für ältere Nutzerinnen und Nutzer im Straßenverkehr - Pedestrian Assistance System for Older Road Users). The development and evaluation of the planned assistance system were participative and user-centered. The procedure is characterized by an iterative character in which analysis, development, and evaluation are alternated. Several qualitative and quantitative data surveys were conducted in the initial stage with experts and the target group, i.e., older pedestrians [1]. This dissertation focuses on the sensory recognition of the traffic and road environment. One of the common reasons for traffic crashes involving older pedestrians is insufficient attention to road traffic [2]. Therefore, this assistance system aims to draw their attention to the oncoming traffic to improve their traffic awareness. When pedestrian approaches a road, an assistance system can be used to increase the safety of the elderly in traffic. For example, by detecting when a pedestrian intends to cross a road and reminding him/her to check the road for traffic. An intelligent feedback system can help achieve this by detecting the pedestrian's path. This information or warning will be at a predefined distance from the road. Therefore, it is necessary to analyse the environment to predict when the user is approaching the road. It is essential to mention that using an assistance system is only one of the possible measures to improve the safety of older pedestrians.

One approach to achieve this is to detect the road itself. One of the problems that arise in detecting the road is that the road is not always in a clear line of sight. The parked cars, trees, and other passers-by are just a few examples of potential visibility obstacles. Consequently, the focus was on identifying the curb and its surroundings. It can be considered a relevant transition between the pavement and the road. Initially, we analysed the requirements to define the objective function of environmental identification. It was based on technical and user-oriented requirements.

Road environments, which include infrastructure, rules, and regulations, vary for each country. Even cities within the same country have different regulations on road design and standards. The developed assistance system is for the streets of Berlin, where the legal minimum width of the sidewalks is 2.5m [3], and the average speed of older people ranges from 80-130 steps per minute [4]. Taking these data into account, a detection window of $2m \pm 1$ was defined.

Within this distance from the curb, the system must detect the pavement and report it to the users to ensure an adequate time window for reaction (e.g., approach, stopping, visual hazard detection). This system also requires that the data processing should be computationally fast. These parameters were varied in systematic tests to define the limits of detectability and the functionality and restrictions of the system derived from these.

Since different sensors measure in different ways, it is important to ensure that their respective weaknesses compensate for each other, thereby increasing the reliability of the measurements.

Detecting the curb with only one sensor might not be a reliable approach. Therefore, we considered using Multi-Sensor Data Fusion (MSDF) for this project. We aimed to ensure reliable detection of the relevant road features using multiple sensors. The sensors selected for this project were cost-effective so that the final system becomes valuable and affordable. After the analysis, we used two sensors, a camera and LEDDAR.

Images had to be analysed accordingly to detect the curb and other road features with a camera. Different techniques of computer vision were compared for this. The basic concept is extracting the road features and then classifying them to identify the risks and accidents. Deep Learning is an ideal technique for this type of analysis. It is a method of extracting features from the dataset, such as images of the roads and their features. We considered the Convolutional Neural Network (CNN) technique to process the images. The end-to-end learning method with binary classification was selected. In end-to-end learning, CNN goes beyond feature extraction by processing the image. The goal is to train a CNN for the desired detection task. For this purpose, the dataset was collected from scratch, representing the problem as optimally as possible. This dataset must consider various conditions such as weather, traffic, obstacle, parked cars, pavement structure, pavement condition, and other related attributes. In addition to a broad new dataset of the curb and road from the pedestrian point of view, a novel algorithm has been presented that utilizes the CNN with end-to-end learning and can detect the curb efficiently.

The LEDDAR technology works similarly to a light-based radar. It requires or senses the time of flight. LEDDAR sensor generates 2D data, and the Artificial Neural Network (ANN) is explicitly trained with this data to detect the curb. The LEDDAR is used mainly to detect the difference in height between the curb and the road. The LEDDAR

sensor has a total of 16 channels. These channels transmit light pulses that hit the curb and road at a predefined angle.

For the fusion techniques, instead of using standard filtering techniques like the Kalman filter [5], a novel algorithm has been presented to fuse the two heterogeneous input data streams from the Camera and LEDDAR. In this way, the system learns and defines the rules and features. Human intervention is still required to classify the images based on the defined scenarios. However, due to end-to-end learning, the human did not specifically mark the location of the curb in the annotated data. In this way, there is less human intervention. Hence, there is a lesser chance of human error.

A prototype in the form of a walker was custom designed. The sensors and the computing devices were mounted on this walker. The reason behind using the walker lies in the need and requirements of the older pedestrians. Moreover, this prototype was also used for development and testing purposes in real-time scenarios.

This thesis is structured as follows:

Chapter 2 defines the theoretical framework of this research. This chapter also covers the related research that has been done previously in this field. The proposed methodology has also been defined in this chapter.

Chapter 3 lays the theoretical foundation of the concepts and algorithms that have been used throughout this research. It explains the theoretical knowledge of deep learning and computer vision, especially Artificial Neural Networks (ANNs), and Convolutional Neural Networks (CNNs).

Chapter 4 describes the creation of the datasets used to train the algorithm. The requirements analysis, collection, and preparation of the dataset have been discussed in detail in this chapter. This chapter also entails the prototype that was developed for the collection of the dataset and testing in the real-time scenario.

Chapter 5 shows how the curb can be detected using camera images. This chapter explains how CNNs can be trained using end-to-end learning. Moreover, it also covers the experimental setup used to weigh the theoretical idea presented in this thesis.

Chapter 6 covers the multi-sensor data fusion technique. It explains how the two sensors, the camera, and LEDDAR, can be efficiently fused using a deep learning algorithm with end-to-end learning.

Chapter 7 describes the field test organized to test the developed assistance system conducted with the target group. This chapter also includes the human machine interface developed to convey the information generated by the assistance system to the user.

Chapter 8 concludes this thesis and discusses the possible future applications and contributions.

2

Theoretical framework

This chapter discusses the mobility of older pedestrians and the need for an assistance system for older pedestrians. Afterwards, it summarises the previous research and literature review. In the end, the methodology to design the assistance system has been described.

2.1 Mobility of older pedestrians

The definition of older people links closely to the concept of old age. It is difficult to grasp the idea of old age as it lacks a universally based definition that is based on facts [6]. Colloquially, it is mostly meant as the calendrical or chronological age, which refers to the number of years of life spent [7]. It can be said that age is one of the dominant structural principles of modern societies [8]. In different areas of life, it determines a person's options for action, his or her social position, and the expectations placed on him or her [9]. Therefore, the retirement age of 65, which was set in this context and has been valid for a long time, generally marks the beginning of the phase of older age [7].

Older people significantly impact society in terms of resilience building, facilitating independence and healthy ageing. Many elderly people contribute to society by continuing work and providing family support and care to those in need. They also share their lifelong experiences with society. This impact is increasing with time. Several key factors are playing their part in this increasing impact. First, the older people will outnumber the younger people. The generation of the 1960s is soon reaching retirement age. Simultaneously, life expectancy has increased thanks to better living conditions and advances in medical sciences. Nowadays, it stands at 78 years for men and 83 years for women. Germany is one of the nations in the European Union where the most advancement has been seen in demographic change. More than a quarter of Germany's population is aged 60 or more. And this ratio is expected to increase to more than a

third by the end of 2050 [10]. This makes a great case for the elderly as they require special attention to traffic safety and the environment.

For people of all ages, outdoor mobility plays a crucial role in living an independent and safe life [11]. To live such a life, it is necessary to leave the home to perform daily tasks such as shopping, engaging in leisure activities, use of medical services[12]. The need to stay connected with community services and to keep social interactions alive is considered crucial to the well-being of all ages and is a key factor in successful ageing [13]. Therefore, mobility is an important aspect of the lives of older people. In the context of mobility, walking is the most common form of physical activity among older people [14], which also impacts positively on the health, cognition functions, and well-being of older people [15, 16].

Walking is the form of mobility which lasts longest in the life [17] and with the advancing age, more and more daily tasks are accomplished on feet. This is particularly true for cities, therefore, the importance of walking increases [18]. Moreover, mobility in addition to participation in society is the key factor that makes society functional and helps in preventing disability [19].

On one side, walking is crucial for older pedestrians, while, on the other hand, it also presents dangerous situations and exposes older pedestrians to the risks of crashes and falling. In Germany in 2013, older pedestrians were involved in 20% of road traffic crashes [20]. Taking the total number of crashes into account, they were not involved in the crashes more than other age groups. However, concerning the distance walked, older pedestrians are more prone to crashes as compared to younger pedestrians [2]. The statistics suggest that the recovery time required by the older person is quite high as compared to the time required by a young person after being involved in a traffic crash, moreover, the fatality rate of people aged 65 years or older because of a road related crash is four times as compared to the younger adults [20]. One of the main reasons for these crashes is that the older pedestrians lack in paying sufficient attention to the traffic, which reduces the performance in hazard detection [21, 22].

The statistics emphasize the importance to figure out and understand the underlying problems the older pedestrians face in traffic situations and developing suitable and applicable solutions which in turn can help to reduce older pedestrian crashes in traffic. Thus, to promote safety and help the older pedestrians, a research group FANS (Fußgänger Assistenzsystem für ältere Nutzerinnen und Nutzer im Straßenverkehr - Pedestrian Assistance System for Older Road Users) was founded. This research group aimed to investigate the underlying causes of these crashes and develop an assistance system that can help older pedestrians cross the roads [23]. Therefore, the work presented here was conducted within the scope of a funded research project that also involved other research areas such as Human factors, Psychology and Geography. This work followed a user-centered approach which involved the target group (older pedestrians) in all the stages of development, i.e., analysis, development, and evaluation.

This research mainly focuses only on the statistics and data that were available for the city of Berlin. It is Germany's largest city and has a modest percentage of around 19% of older people aged 65 or above [24]. Therefore, throughout the development and testing stages, only the data from Berlin has been considered and used.

2.1.1 Investigation of the underlying causes

At the start of this study, it was necessary to investigate the older pedestrians' behaviour at road crossings and to analyse the underlying problems. Several studies have suggested different behavioural deficits of older people in traffic, such as they leave smaller safety margins and they walk comparatively slower, they focus more on the walking path and less on the traffic situations, or they don't consider the speed but only the distance when accepting gaps in streets [25, 26, 27, 28]. The reasons behind these deficits have been identified as the decline of motoric and cognitive function in old age.

However, age-related deficits are not the only elemental causes behind older pedestrians' shortcomings in traffic scenarios. In addition to these, multitasking also has an effect in such situations [29]. While walking outside or crossing the street, pedestrians of every age do multitask by performing several different tasks at a time, e.g., navigating, looking for the pathway's apparent condition (e.g., bumps, etc.), and walking itself. During the road-crossing, the main task is to analyse the surroundings and perform hazard detection, however, the parallel activities which are performed while walking can distract the pedestrians from the main task.

Another important aspect that had to be analysed is the gait of the older pedestrians. It is to ensure that the older pedestrians should be alerted promptly to ensure that sufficient time is provided to the older pedestrians to take precautionary measures. Therefore, a detection window had to be defined that also considers the speed of the older pedestrians. This window ensures that the pedestrian has enough time to stop, quit all the parallel activities and focus completely on the traffic.

Considering all these scenarios, two main reasons were identified for older persons' unsafe behaviour in traffic. First, older people tend to scan the ground more often and carefully as compared to younger pedestrians [28]. This requires the visual attention of the pedestrians and this additional task, therefore, distributes the attention and reduces their performance in detecting hazardous situations in the traffic [29]. Second, while walking when older pedestrians approach the road, most people observe and watch out for cars. This task can also be catalogued in multitasking behaviour. Walking demands cognitive resources and therefore concurrently walking while visually observing the oncoming car can also reduce their performance in hazard detection [30].

Based on these findings and after the basic analysis of the official traffic crash statistics, the requirements of the assistance system were defined.

2.1.2 Technical and user-centered requirements for the environmental perception

This research aimed to develop an assistance system that can help older pedestrians in traffic scenarios. An alert can make them stop at the edge of the street. This warning also helps them to focus their complete attention on the traffic and to minimize road accidents.

Considering the street environment, there are a handful of features to detect, e.g., the cars, the curb, the obstacle (trees, poles, etc.), traffic signals, the road, etc. However, not all these features are available in every street scenario but the road and the curb. The problem with detecting the road is that it is rarely in the clear line of sight of the sensors. Parking cars, trees and other passers-by are just a few examples of potential visual obstacles.

On the other hand, every street in Berlin is equipped with a curb with the length and width officially defined in the laws of Berlin [3]. Therefore, it was decided to detect the curb by analysing the street surroundings. For this system to be effective, the system should perceive the environment reliably and should convey this information to the user at a distance that it would allow the user to take the precautionary measures in a timely manner.

For the system to be effective, it should be as accurate as possible in its detection in every possible situation. For example, it should perform reliably in various weather conditions such as snow, leaves, and wet conditions. It should also be able to handle the situation with different types of existing pavement structures in Berlin, hydrants, trees and poles, etc. Another important aspect to consider is the approaching angle to the curb. The pedestrians can approach the curb at any angle, therefore, the system should be able to handle the angle of approach of the pedestrians.

However, the core functionality of this assistance system can be summarized as:

- Detect the curbstone

  As the proposed assistance system is intended for Berlin now and considering the legal minimum width of the sidewalk in Berlin which is 2.5m [3], the average speed of older pedestrians ranges from 80 - 130 steps per minute [4], a detection window of $2m \pm 1$ was established. In this range, the system should be able to detect the curb and its surroundings reliably and accurately.

- Communicate with the user

  After the detection of the curb, this information must be conveyed to the user of the system as a feedback modality. It is done to make them alert while crossing the road. This information should be relayed promptly at a sufficient distance so that it provides them with enough time to react to this alert.

The system is aimed to detect the curb and convey this information to the user promptly. It prevents them from stepping onto the street without checking for the oncoming traffic. Therefore, there should not be an excessive number of *misses*. A miss would be considered if the system was not able to detect the curb while it was available in the frame. Another important point to consider in this context is that it should also not generate *false alarms* abundantly. A false alarm is when the system has detected the curb while there is none present in the frame. From previous research, it can be safely assumed that false alarm is as important as misses or more [31]. As with too many false warnings, the user loses their trust in the system which in turn makes them ignore the alarms [32]. It can make the user misses important events which could be accidental for them [33]. Moreover, the system should be robust to work reliably in different weather and light conditions. Most of the crashes occur in low visibility conditions such as in the evenings or on rainy days [22].

As the system was developed for older pedestrians so additional aspects had to be considered regarding the older people. First, it should be designed in a manner that older people should be able to trust and accept the system, because if the system lacks their trust, older people will not pay much attention to the oncoming warning and might ignore it.

The sensors needed to perceive the street environment should be economical. The reason behind that was, in Germany, the older people (aged 65+) are mostly pensioners, hence, the overall cost of the system should not be excessive so that they can afford this assistance system. Another thing to consider is that the whole system should be lightweight. As older people are sometimes fragile, it should not put too much weight on them.

2.2 Related work

Curb detection presents an important research challenge in the field of mobile robotics. It is particularly significant in the domain of Intelligent Transportation Systems (ITS) essentially in ADAS (Advanced Driver Assistance Systems). ADAS are electronic systems designed to assist the driver while driving. ADAS systems aim to increase driver and road safety. Curb detection contributes as a pivotal aspect of the ADAS research domain [34, 35]. Consequently, most of the research in this domain has been done in terms of road boundary detection and presents thorough research with sensing modalities and algorithms.

Curb detection presents a significant research value in the field of mobile robotics or autonomous service robots. Therefore, the influential contribution in recent research can be reviewed. Useful information can be extracted from it regarding the sensors and the methodologies that have been used to solve this problem thus far. The research in curb detection can be classified based on the variety of sensors used to accomplish this task.

2.2.0.1 Monocular camera

Methodologies using monocular cameras have been researched thoroughly to detect the curb because of the accessibility, the excessive amount of information, and in terms of cost and higher processing efficiency as compared with 3D sensors. These methods utilize the appearance information based on image processing techniques, for example, Prinet *et al.* [36] detect the curb using monocular images acquired by a fish-eye lens. They used the Support Vector Machines (SVM) classifier with the Histogram of Oriented Gradients (HOG) to extract the curb points and refine these using the Kalman filter. These techniques utilize monocular camera information. They rely heavily on the appearance information itself e.g., in this case, it can only be applied to clear and dominant curb structures. Image processing methods used for curb detection are vulnerable to the slightest changes in the intensity of images, for example, various illumination conditions, shadows, road marking, etc.

2.2.0.2 Stereo vision

Stereo vision can utilize 3D geometry information to detect the curb [37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48]. Stereo vision cameras can produce high-resolution appearance information which is not available in other 3D sensors like LiDAR or monocular cameras. However, the accuracy of this 3D information gained by the stereo matching is lower than that of LiDAR sensors. Nevertheless, these appearance and geometry feature available from a stereo sensor have been widely researched in curb detection methods. For example, in [48], a stereo vision camera with the Canny algorithm was used, it was based on the multi-frame persistence map to detect curbs. Cheng *et al.* [37] proposed a 16-dimensional descriptor to distinguish curbs from the road and obstacles from a flat area. They estimated based on disparity and v-disparity maps acquired from stereo matching. The SVM and Dijkstra Road model was used to distinguish the road and curb. Siegemund *et al.* [47] used Conditional Random Fields (CRM) to propagate the curb detection outcomes. Moreover, stereo vision-based methods also exploit geometrical phenomena such as height step [38], curvature [39, 40] and heights variation [41, 46, 49].

2.2.0.3 Mapping strategies

Mapping strategies are used in 3D computer graphics to represent a certain terrain using the elevation data. Therefore, to detect the curb stone, mapping strategies are also widely used for 3D sensors. In [41, 45], the Digital Elevation Map (DEM) is used to calculate the Height Gradient Feature for the curb feature extractor. Kellner *et al.* [38] constructed the road curb map to acquire the road curb features and updated it with the detected feature for curb candidates' extraction. In [44, 47], for curb detection, the column disparity match is generated. The mapping methods can benefit widely from the geometric information, but they require a high density of 3D point cloud

data. Moreover, they tend to benefit from the smaller cell sizes which are sometimes susceptible to accuracy loss and require higher computational efforts making it hard to use them in a real-time environment.

2.2.0.4 LiDAR

Light Detection And Ranging (LiDAR) sensors are used to determine the distance between objects by targeting the object with a Laser and by calculating the time the reflecting light took to return. LiDAR is also used to create the digital 3D representation of a surface, i.e., LiDAR can be used to acquire accurate 3D information from the environment [50, 51, 52, 53, 54, 55, 56, 57]. Zhang *et al.* [53] extracted the curb position based on its spatial position using the 3D LiDAR data for Unmanned Ground Vehicles (UGVs) and fitted them with a parabola model, a Kalman filter was used to predict and update the curb position in real-time. In [54], the scan line from Velodyne LiDAR was taken as a processing unit and feature points were extracted from these scan lines. These feature points were then selected as initial curb points using the distance criterion and Hough transform. It was then used as a seed point the for iterative Gaussian process to model the curb. The regression method Least Trimmed Square was used in [56] to model the road curb, especially the occluded scenes.

2.2.0.5 Mobile laser scanners

Mobile Laser Scanners (MLS) have gained popularity in acquiring high-density 3D point cloud data. MLS uses Global Navigation Satellite System (GNSS) and Inertial Measurement Unit (IMU) to provide georeferenced 3D point cloud data which has a higher density than LiDAR data. Xu *et al.* [57], proposed a method to extract the curb from a 3D mobile LiDAR point cloud by using an energy function to extract candidate points of the curb, a least cost path model was used to refine these candidates. In [58], the 3D point could data is processed with a trimming operation. Afterwards, the candidate curb points were extracted from the two maps delineating the height and density of 3D point could data.

2.2.0.6 Fusion of sensors

The researchers investigate the fusion of 3D sensors and cameras for curb detection. In [59], curbs were extracted using road surface curvature, for curvature estimation, dense 3D point cloud data was generated with the help of LiDAR and a stereo camera using the Iterative Closest Point (ICP) algorithm. Tan *et al.* [60] fused the sparse 3D LiDAR point cloud data and high-resolution camera images to recover depth images. Based on the depth images, normal direction within the image was estimated and based on the curbs' geometrical properties and curb point features fitting the patterns were detected. Afterwards, the Markov chain model is utilized to model the consistency of

curb features and the curb path is estimated using the dynamic programming algorithm. Finally, outliers were filtered by performing post-processing operations.

2.2.0.7 Line or surface fitting methods

The line or surface fitting approach has also been utilized regardless of the sensor choice to detect the curb, for example, vertical surface [44], spline [45] or straight line [46]. In [44], the curb area is considered as a vertical surface which divides the sidewalk and road, and both are fitted with a horizontal surface. In [45], DEM was used to extract the curb measurements and a spline model was used to fit the global set of curb measurements. In [46], weighted Hough Transform was used to estimate the curb lines.

2.2.0.8 Summary

The cameras or distance sensors have mainly been used for curb detection. Assignment methods, such as the Digital Elevation Model (DEM), were used either alone or in combination with edge detection techniques. The work based on DEMs is accurate, but its high computational power does not allow its use in real-time. A more promising approach, where this problem does not occur, is based on appearance information using image processing by using a Mono-camera. However, the potential weakness is the loss of accuracy that can be caused by changes in the intensity of the pixel values of the images. Further progress was made by combining different sensors, such as 3D distance sensors, which require an extensive setup and a high computation power and are expensive, with a mono camera. These approaches are quite accurate but cannot manage this accuracy in real time because of the requirement of high computational power [49].

Moreover, most of these techniques were not suitable for geometrically complex curbs. To tackle these challenges, it was decided to advance and exploit this approach, i.e., fuse the appearance information available from a Mono-camera with a cost-effective distance sensor, and does not require a bulky setup to implement. It must also have a lower computational power requirement.

2.3 Proposed Methodology

The research in ADAS targets curbs detection from the driver's perspective. The experimental setup assumes that the curb will appear only on the right or left side of the vehicle. However, the curb detection from the pedestrian's point of view has particular relevance, e.g., a pedestrian's view angle towards the curb can be entirely different compared to the vehicle's line of sight. The pedestrians walk on the pathway specially designed for them. This also raises the point that the curb can be approached from multiple viewpoints with multiple angles. Moreover, to name a few, curb detection provides the following challenges [37]:

- The height of the curb alters broadly in different scenarios, even in the same detection frame
- The position of the curb in the frame and angle of approach also determines whether the pedestrian intends to cross the street or not
- The curbs can have different sizes which in turn makes it difficult to extract information
- Occlusions also play an important role in correctly overlaying the curb
- The shadows impact significantly the visual information

2.3.1 Hardware

Most of the previous research has used a 3D sensor like LiDAR with the Mono-camera. LiDAR is not a viable option in the context of an assistance system. LiDAR requires a 360° field of view which is not available in our case as the assistance system is being governed by the user. LiDAR is also comparatively an expensive sensor; thus, it doesn't make the solution cost-effective. Cost-effectiveness is an important aspect to consider when the livelihood of older pedestrians is considered. Moreover, the computation power of the system also needs consideration. As mentioned before, most of these techniques have their applications in the automotive industry where the whole system can be placed in a prototype system, i.e., a car, which has enough space and computational power to use. In this research, the assistance system is carried and used by the older pedestrians, hence, we don't have the luxury to put too much strain on the weak bodies of older pedestrians.

Therefore, for the hardware, in addition to the monocular camera, we opted to use the LEDDAR sensor [61]. It is a propriety sensor from Leddartech and works based on the principles of light-based radar technology. It is small-sized, cost-effective and doesn't require a large amount of computational power. As this study involves the pedestrian moving toward the curb on a horizontal surface, therefore the LEDDAR was used in the 2D plane and the difference in height of the curb about the road was detected using the channels available from the LEDDAR. For the appearance information, a mono camera was used to capture the curb and its surroundings in the streets of Berlin. Considering the requirements of our assistance system, it was decided to fuse these two sensors, i.e., a camera and LEDDAR.

The multi-sensor data fusion aims to ensure reliable detection of the relevant feature by using both sensors. The concepts of redundancy and diversity are combined. Different sensors measure the same thing in different ways. This is to ensure that they compensate for each other's weaknesses and increase the reliability of the measurements. For example, if the camera is unable to detect the curb in the current frame because of the bad light, then the LEDDAR sensor can assist and generate the alarm. Similarly, if the curb is too low and LEDDAR is not able to detect the curb then the camera can help in generating

the alarm. Hence, the fusion of the sensors can help in achieving the goals of detecting the curb and its surroundings in the relevant scenarios efficiently.

2.3.2 Software

In the context of fusion techniques, more conventional techniques have been used in the past. Fusion of the sensors has relied heavily on filtering techniques like Kalman filters. Where the future states can be predicted using the past knowledge of states and the input data, however, Kalman filters are not efficient in fusing the sensors with two different input dimensionalities. Moreover, it relies heavily on human intervention in decision-making situations which makes it prone to human mistakes.

Therefore, deep learning methods opted for the fusion of appearance information from the monocular camera and the data available from the LEDDAR [62]. Deep learning is a subset of machine learning algorithms based on Neural Networks with representation learning. It uses multiple layers of neurons to extract higher-level features from the given raw input. This type of learning can be categorized into supervised, semi-supervised or unsupervised.

In recent times, a lot of Deep learning algorithms have been developed, hence, it offers various networks to choose from. This selection can vary based on the use case and its requirements.

In this research, Convolutional Neural Network (CNN) was chosen to train the network on images taken from a camera. CNN are the best available model to extract the underlying features of an image [5]. CNN belongs to Deep Learning, and it takes images as an input, allot learnable weights and biases to various objects or aspects in the image so that they can be distinguished and differentiated from one another. The pre-processing of the data required in a CNN is much lower when it is compared with other classification algorithms. In previous classification methods hand-crafted features have been used. While in CNN, with enough training, it can identify and learn these features/characteristics on its own.

For the LEDDAR sensor, as it generates the data in the form of a 2D array, Artificial Neural Networks (ANN) were chosen. ANNs are the type of algorithms that are modelled considering the functionality of the human brain. ANN can learn from the provided data and can provide output in terms of classifications or predictions.

For the training of the networks, it was decided to use deep learning methods with end-to-end learning. In end-to-end learning, networks learn the features and regulate the processing pipeline. Tobias Glasmachers shows how end-to-end learning is embedded in a deep learning context [63]:

"This elegant although straightforward and somewhat brute-force technique [E2E] has been popularized in the context of deep learning. It is a seemingly natural consequence of deep neural architectures blurring the classic boundaries between learning machine

and other processing components by casting a possibly complex processing pipeline into the coherent and flexible modeling language of neural networks."

End-to-end learning has been successfully applied in other areas of research such as Speech Recognition and Autonomous Driving [64, 65]. Moreover, in end-to-end learning, there is less human interference hence there is less probability of human error. The fusion of the sensors is also done using end-to-end learning. By doing this, the system has more freedom to define and learn the rules.

2.3.3 Interface modality

After detecting a hazardous situation, this alert must be conveyed to the older pedestrian. To find out the most suitable modality, a study was conducted by the Human Factors experts within the FANS group with older people to test the different modalities and their effects on the older people. From the study, it was deduced that, while the visual and auditory channels in road traffic are occupied by a variety of different sensory impressions, tactile feedback in the form of vibration is an alternative for drawing the attention of older pedestrians to road traffic. Therefore, it was decided to use the vibrotactile modality which would be mounted on the upper arms of the older pedestrians. An alarm system was developed in this regard with the help of embedded devices and vibration sensors.

Therefore, whenever a hazardous scenario is detected by the algorithm it would generate an alarm. Consequently, this information was wirelessly given to older pedestrians in the form of vibrations [66]. This was also found feasible and beneficial in terms of extensive testing of the assistance system in real-time scenarios, i.e., on the streets of Berlin.

3

Theoretical background

This chapter discusses the theoretical background of the methods which have been used in this thesis. The basic definitions and relevant theoretical details of machine learning, furthermore, deep learning and its impact on computer vision have been discussed.

3.1 Machine learning

Machine learning is a field of science that comprises of the study of statistical algorithms and models that a computational machine can use to perform a task without providing explicit instructions [67]. Usually, the programmer breaks down the big task into small tasks that a computer can perform in a simple and smooth manner. On the contrary, a platform based on machine learning techniques tries to learn from the sensory data, comprehending the problem on its own. It interprets this data in terms of machine perception, clustering raw input, labelling, etc. to perform the desired tasks.

In recent years, the field of machine learning has advanced immensely. There is a great pool of new and innovative algorithms available to choose from for a lot of applications. Here, the algorithms that have been used in this thesis are discussed briefly.

3.2 Artificial Neural Network

Artificial Neural Networks (ANNs) are one of the prominent tools used in machine learning nowadays. Neural networks are the assortment of algorithms that are designed to recognize the underlying patterns or to find knowledge and model from the data. For example, to recognize a car in an image, it is quite difficult for a human programmer to extract the features or patterns, which can include but are not limited to the shape, color and make of the car, manually and then program it to teach an algorithm to

predict the category car in the given images. Therefore, NNs are excellent in recognizing patterns or fetching information that is complex for a human programmer to derive and train the machine.

One of the simplest definitions of a neural network written by Robert Hecht-Nielsen can be read as:

"*...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs* [68]"

The processing element mentioned in this definition is an artificial neuron inspired by the human brain. One example of a neuron is the *perceptron* proposed by Warren McCulloch [69] and the first-ever perceptron model was presented by Frank Rosenblatt [70]. An artificial neuron receives input from the input sensors or other artificial neurons, defined by its weights (w_n), bias (b) and the activation or transfer function (φ). An activation function is typically represented by the sign φ, which determines the final output of the neuron. The function of a perceptron can be seen in equation 3.1.

$$y = \varphi\left(\sum_{n=1}^{N} w_n \cdot x_n + b\right) \tag{3.1}$$

Perceptron performs a linear combination of the input (x_n), the weights (w_n), and the bias (b) and then applies the activation function (φ) on the result to obtain the output (y). The most used activation function is a sigmoid function.

A neural network is a parallel network composed of layers of interconnected neurons with connections in three different types of layers. These are input layer, the output or visible layer and the hidden layer. Each layer processes the information provided by the previous layer and once the computation is done the result is then sent to the next layer.

Every ANN consists of one input layer and one output layer. The number of neurons in the input or output layer depends upon the number of input variables and the number of output variables respectively. The hidden layers lie between input and output layer . These are responsible for the processing of the data received from the previous layer (either input or previous hidden layer).

The configuration of connections among the neurons of an ANN makes it possible to classify the ANNs into two categories: if the connections are in the direction from input to output through a hidden layer, then the ANN is called a *feed-forward* network. On the other hand, if the neurons are connected to the neurons of the same layer or with the neurons from the previous layers then the ANN is called a *feedback* neural network. Feedback neural networks are considered more powerful because they are stateful. While feedforward networks are universal function approximators, recurrent networks can

implement every algorithm and that makes them much harder to train. Therefore, feed-forward neural networks are used extensively for practical applications. Feed-forward networks can be further classified into *fully connected* and *partially connected*. It depends upon whether all the neurons from the previous layers are connected to all the neurons of adjacent layers or if there are neurons missing the connection with neurons of the adjacent layers. Moreover, all the connections between the neurons are weighted, whether they are between the input layer and hidden layers or between the hidden layers and output layer.

The input to the Neural Network is the image pixels which are then flattened into a single vector. For example, a grayscale image with 28Ö28 dimensions if flattened results in a single vector of 784 pixels, and therefore, the input layer of this example has 784 neurons, which form the first layer of the Neural Network, as can be seen in Figure 3.1. Each neuron is fully connected to each neuron in the next layer [71].

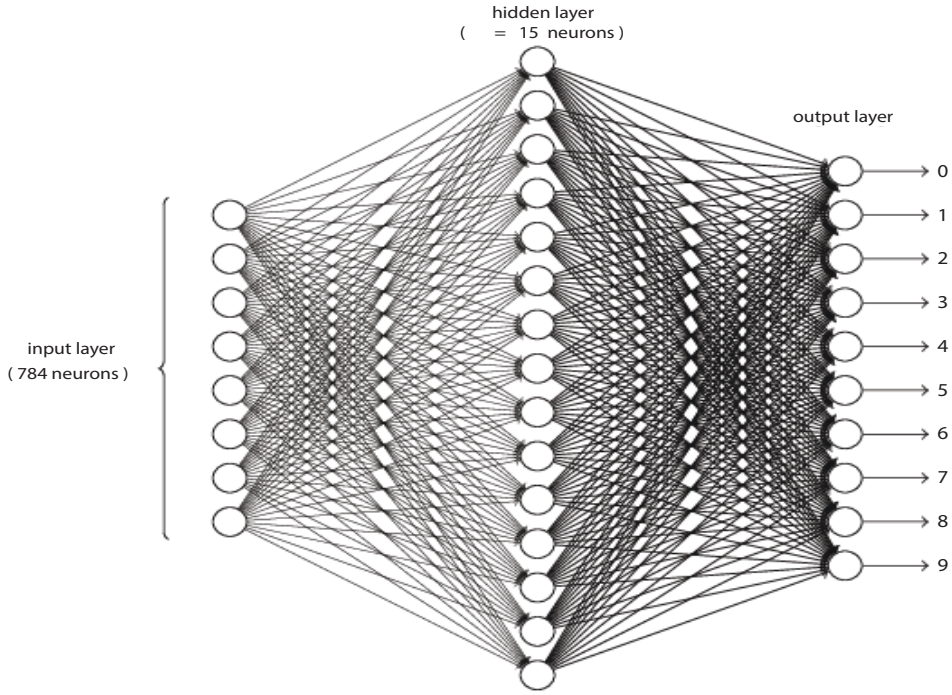


Figure 3.1: Neural Network architecture for an image of 28 x 28 with a single hidden layer [71]

After the designing of the network, a learning technique must be applied which is technically called training, to adjust the weights and biases of the network. The initial step is to assign the random values to the weights, then an iterative training algorithm is used to adjust the weights to converge them to optimal status. These training algorithms are usually based on gradient descent optimization techniques. Gradient descent is a first order iterative optimization technique usually used to find the local minima of a function.

The most used training algorithm, employed to train the feed-forward neural networks, is the *backpropogation* algorithm which was developed by Werbos [72] and Rumelhart [73]. This algorithm uses Lvenberg-Marquardt method, which is the combination of the gradient descent method and Gauss-Newton optimization [74, 75]. The performance of the algorithm can be evaluated from the quantity and quality of errors. To assign a cost to rectify these errors, a loss function is then used [76]. The objective is to reduce this loss in the training phase. The learning process in backpropagation is based on the minimization of the Mean Squared Error (MSE). It measures the difference between the output provided by the network $y^{'}$ and desired output y for the input x. This error is denoted in equation 3.2.

$$Error(x) = \frac{1}{2} \cdot \sum_{n=1}^{N} (y(x) - y^{'}(x))^2 \tag{3.2}$$

where the desired output y is given by:

$$y(x) = \begin{cases} 1 & \text{if } x \in \text{class c} \\ 0 & \text{otherwise} \end{cases}$$

Another important aspect in the ANN is, that due to some random conditions the adjustments of weights and biases can become a non-deterministic procedure regardless of the fact that the training algorithms can be deterministic. Therefore, the initialization of the weights and the selection of an activation function are important factors which have a significant effect on the output of a network.

3.2.1 Initialization of the weights

The weights initialization is a hyper-parameter which has a significant impact on the training to reach the local minima. There are several initialization methods available, such as including Zeros or Ones, RandomUniform, RandomNormal, Kaiming Initializer [77] and Xavier [78] etc. Out of these mentioned, the first four initializers can lead to a vanishing gradient when it is too small or exploding gradient when it is too large. The most popular approach used by the research community is the Xavier initializer. Its main objective is to keep the variance equal on all layers. It uses Gaussian distribution to initialize the weights which has zero mean and a variance of $1/N_{Avg}$, where N_{Avg} represents the average number of input neurons as Equations 3.3 and 3.4. $n_i + n_{i+1}$ refers the number of input and output network connections, respectively.

$$Var(W_i) = \frac{1}{N_{Avg}} \tag{3.3}$$

where

$$N_{Avg} = \frac{n_i + n_{i+1}}{2} \tag{3.4}$$

Kaiming *etal*. [77] profess that Xavier initialization uses a deep net with 30 hidden layers or more, whereby the activation function used in the hidden layers is ReLu which suffers from the vanishing and exploding gradient. To cater this problem, the authors propose the Kaiming initializer. The Kaiming initializer uses the standard normal distribution multiplied by $\frac{\sqrt{2}}{\sqrt{n}}$ and with the bias set to zero. To summarize, the weight initialization determines the initial point from which the backpropagation algorithm converges to the minimum. This concludes that the time consumes in the training depends considerably on the weight initialization method. Choosing the appropriate initialization methods to have an impact on the speed of the training process.

3.2.2 Activation functions

The activation function in a Neural Network is responsible to calculate the output 'weighted sum' of each neuron in each layer of the network. It is taken as a hyper-parameter that can be tuned to improve the modelling accuracy of the whole network. There are quite a few activation functions available, including Sigmoid, Softmax, Tanh, ReLU, and Leaky ReLU. A brief description of these is provided in this section. In Figures 3.2 and 3.3, the curve and scope of each function are shown.

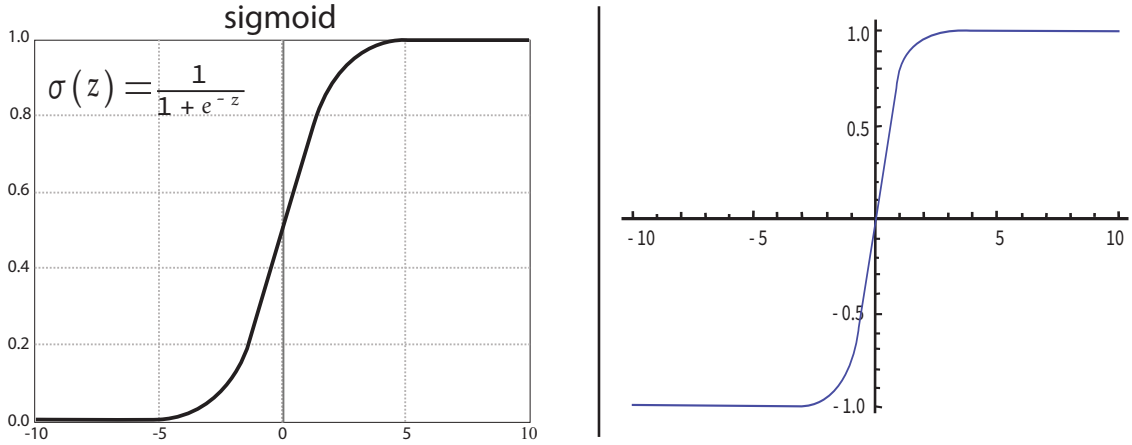
Sigmoid Function is a non-linear activation function whose input is a real value and output is between 0 and 1, as shown in Equation 3.5.

$$f(z) = \frac{1}{1 + e^{-z}} \tag{3.5}$$

The z in this equation is a linear function and can be seen in Figure 3.2a. The major drawback of the Sigmoid function is the gradient vanishing, and the struggle in loss optimization. This occurs as it is not zero centralized, and while updating the gradients, it extends in various directions which becomes a hindrance in finding optimal minima. However, it is still being used with small and medium-sized architectures to produce the desirable results [71].

Tanh Function takes a real value as an input and maps it between -1 and 1, Equation 3.6, and therefore, it converges the output to zero. Although the gradient of the Tanh function is stronger as compared to the Sigmoid function, the vanishing of the gradient still occurs in very deep networks [79]. The Tanh function can be seen in Figure 3.2b.

$$f(z) = \frac{(e^{z} - e^{-z})}{(e^{z} + e^{-z})} \tag{3.6}$$



(a) Sigmoid Function

(b) Tanh Function

Figure 3.2: The scope of activation function used in Neural Network [79]

Rectified Linear Unit (ReLU) gives the maximum of x and *zero* as the output. The x is the linear function and can be seen in Figure 3.3a. The ReLU function returns the identical x for all values which are greater than zero (positive values) and zero for all negative values, Equation 3.7 [80]. As ReLU is considered the fastest activation function, it is used in deep networks to minimize computation time and complexity. Therefore, these days most CNN networks apply ReLU in the hidden layers.

$$f(x) = max(x, 0) \tag{3.7}$$

Leaky ReLU is another form of ReLU function which allows a small negative value to be present which in turns then helps to reduce the number of vanished neurons in the backpropagation process, also termed as dying ReLU; see Equation 3.8 [77].

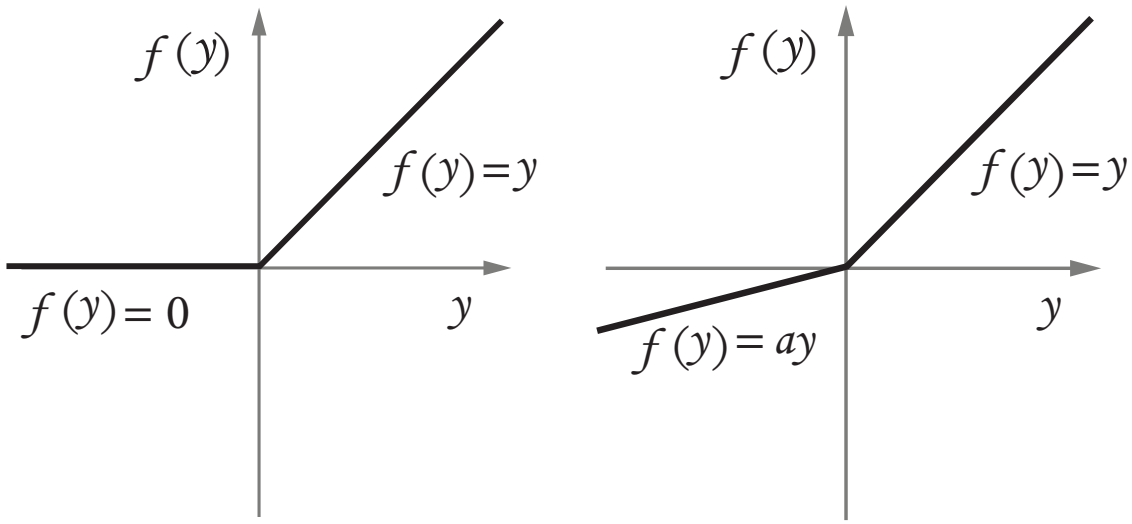
$$f(x) = \begin{cases} 0.01x & x > 0 \\ x & x \geq 0 \end{cases} \tag{3.8}$$

The x is the linear function as can be seen in Figure 3.3b.

SoftMax Function is usually applied in the last fully connected layer. It calculates the probability (between 0 and 1) of the input x among the defined classes. It as can be seen in Equation 3.9, where the sum of all values is 1.

$$\sigma(\overrightarrow{x})_i = \frac{e^{x_i}}{\sum_{j=0}^{n} e^{x_j}} \tag{3.9}$$

The x here refers to the elements of the input vector and n is the total number of classes. The term used on the bottom in Equation 3.9 is a normalization term which makes it certain that the output of the function sums to 1 [81]. For example, if the



(a) ReLU Function **(b)** Leaky ReLU Function

Figure 3.3: The scope of activation function used in Neural Network [79]

network is trained with three different classes, then the output of the last layer will be a vector and it will contain three values with a sum equals to 1.

To summarize, the activation function is used to activate the neurons in the network. The main difference is the scope of the function which then decides which activation function should be used in the network. Most modern networks use ReLU and Leaky ReLU in the hidden layers. And Softmax in the fully connected layer for the classification task. The effectiveness of an activation function can be determined by applying in the network. It has been tested as a hyper-parameter that researchers tune to find the best classification results.

3.3 Deep learning

Modern neural networks are often referred to as deep neural networks. Although multilayer networks have been around since the 1980s, few reasons restrict the efficient training in multi-layers neural networks [82].

One of the significant obstacles is the *curse of dimensionality* [82, p. 152], which refers to the situation that if the number of variables increases, the number of different configurations these variables can grow exponentially, which in turn results in forcing to increase the amount of training data so that the optimal accuracy can be achieved. Collecting the data for training in sufficient size is time-consuming and costly and, in some scenarios, could be futile.

However, the real-world data doesn't often have a structure. It is also not uniformly distributed where the useful information lies on a low-dimensional manifold. This manifold hypothesis assumes that most data configurations are invalid or rare [82, p. 159]. Thus, we can decrease the dimensionality by representing the data using the coordinates of manifold. The generalization can also be improved by assuming *local*

constance [82, p. 154]. It assumes that the activation function the network learns to approximate should not modify within a small specific region.

Another theoretical breakthrough was the replacing of the mean squared error function with cross entropy-based functions, i.e., sigmoidal activation functions were replaced with rectified linear units (ReLU) [82, p. 222].

All these features made the way for deep learning, where there is less need for hand-tuned features. They were used extensively in machine learning. For example, a classical pattern detection task includes a hand-tuned feature extraction phase before feeding them to the machine learning algorithm. On the other hand, in deep learning, the lower layers of the network are used to extract the basic features, which are then fed to the advanced layers for entire feature detection.

3.3.1 Computer vision

Computer vision deals with how machines can gain meaningful and high-level understanding from digital images or videos. Computer vision application includes visual detection, image classification, augmented reality, 3D scene reconstruction from 2D images, etc. [83].

Computer vision is improved tremendously because of the deep learning. Because of the availability of large amount of data, nowadays. Deep learning is considered a vital component of various computer vision algorithms [84], these algorithms combine image processing with deep learning. To gain the desired outcome, these algorithms can handle the vast amount of information available in the images, and critically for various application, are able to carry out the computation in real time.

3.3.2 Convolutional neural network

There are problems when solving computer vision problems using traditional machine learning techniques. For example, even a small-sized image holds a substantial amount of information. To envisage this, consider a monochrome image with a size of 620×480. This image contains 297,600 pixels. In a fully connected neural network, each neuron would require 297,600 weights if each pixel is connected to a neuron as an input separately. Similarly, a 1920×1080 full HD image would require 2,073,600 weights. If the images used are polychrome then the weights are also multiplied by the number of colour channels used, usually three for RGB. Thus, as the image size increases, the number of free parameters in the network becomes immensely large. This can result in *overfitting* and slow performance [76, p. 9].

Moreover, various pattern detection tasks require the algorithm to be translation invariant. In this way, the algorithm recognizes the pattern presented in different sections of the image irrespective of the position of the pattern in the image. It doesn't train the

separate neurons for the same pattern. A fully connected neural network is not able to handle the translation invariant task.

A special case of deep neural network, which can handle a large amount of information in the images, is a Convolutional Neural Network (CNN). The neocognitron, inspired from biology and developed by Fukushima [85] in 1988, presented a neural network model for translation invariant object detection. This method was combined with the learning algorithm i.e, backpropagation by Le Cun et al. [86] and its first application was the recognition of hand-written characters from MNIST-dataset.

3.3.2.1 Basic structure

The basic idea is inspired by a concept taken from biology called the receptive field [85]. The Receptive fields represent a feature of animal visual cortex [87] and act as detectors for certain types of stimulus e.g. edges.

These biological functions can be hypothesized for machines using the convolution operation [88]. In computer vision, convolution can be used to filter the images to generate the visible effects. These filters are also referred as *Kernels*. The discrete convolution operation between an image I and a kernel K is defined in equation 3.10:

$$H[x, y] = I[x, y] * K[x, y] = \sum_{n} \sum_{m} I[n, m]K[x - n, y - m]. \tag{3.10}$$

The kernel K and sub-image of I have the same dimensions, and the dot product of both produces the pixel value H at coordinates x, y. The size of the receptive field is determined by the size of the filter matrix. Aligning the kernel K with each sub-image of I generated the output pixel matrix H. This output matrix is also called a *feature map* [82, p. 328].

The convolutional kernels can be combined to make a convolutional layer. The values of the kernels are treated as parameters of the neurons and are trained. The multiplication operation of the regular neural network is replaced by the convolution operation. The height and width of the output of this operation depend upon the dimensions of the activation map and the depth depends upon the number of kernels.

As the same kernels are used to detect patterns from all parts of the image, it reduces the number of free parameters drastically compared to the fully connected layer [86]. The neurons of the convolutional layers are connected to local regions of the input and share the same parameters. This parameter sharing ensures the translation invariance in CNN. In this way, neurons share weights at multiple spatial locations and have zero weights outside the receptive field.

Ensuing convolutional layers form a convolutional neural network. These layers are often combined with other types of layers such as pooling layers (described below). Theoretically, the layers which are closer to the input learn to recognize low-level features and patterns such as corners, edges, etc. On the other hand, the layers deeper in the

model learn to incorporate these features to recognize high-order or more abstract features [85]. If we go into the details of CNNs, there are three features of CNNs which make them so effective, namely *"Local receptive field"*, *"Shared weights"* and *"Pooling"*.

3.3.2.2 Local receptive field

If we consider a fully connected network, as shown in Figure 3.1, and an input image of dimensions 28×28, that means our network will have 784 input neurons for a fully-connected network and these inputs are given as a vertical line of neurons. However, in a convolutional neural network, instead of a vertical line, 28×28 square of neurons are used as an input, whose values depict the intensities of the pixels. It can be seen in Figure 3.4

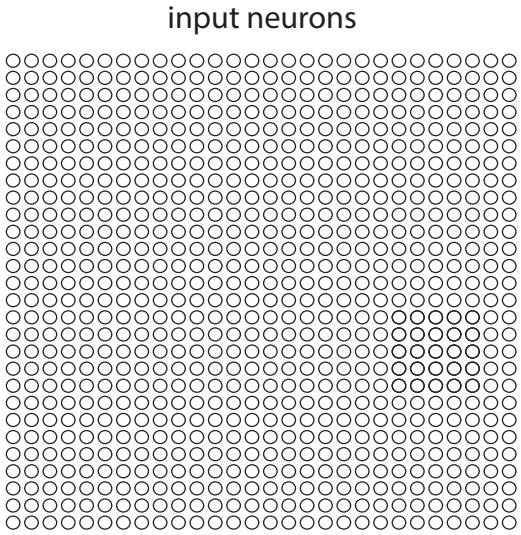


Figure 3.4: Input neurons for an image of 28×28 as a square [71]

The next step is to connect the input with the hidden layer' neurons. However, unlike fully connected network, where every input pixel is connected with every neuron in the hidden layer, in CNN, connections are made in localized and small regions of an input image. That means every neuron in the hidden layer will be connected to a small region or field in the input domain. For example for a 5×5 region, which then refers to the 25 input pixels and for a particular neuron in a hidden layer, the connection with the input pixel might look as shown in Figure 3.5.

This region is referred as the local receptive field. It is a small window where the connection between the hidden neuron and input pixel lies. A particular hidden neuron learns to analyze its receptive field. Therefore, each neuron in the hidden layer learns an overall weight and bias for its own local receptive field.

To cover the entire input image, this local receptive field is then slid. For every local receptive field occurring because of the sliding, there is a particular hidden neuron in the first hidden layer. The sliding concept can be seen in Figure 3.6

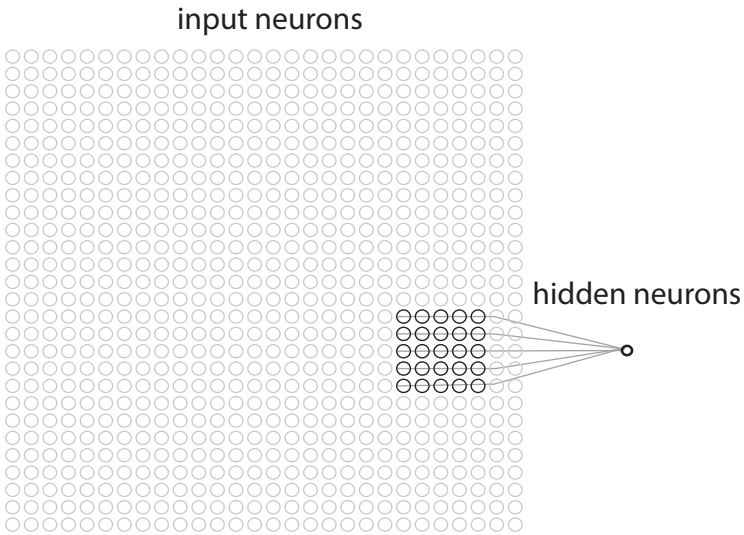


Figure 3.5: A hidden neuron connections with the region of 5×5 in the input domain [71]

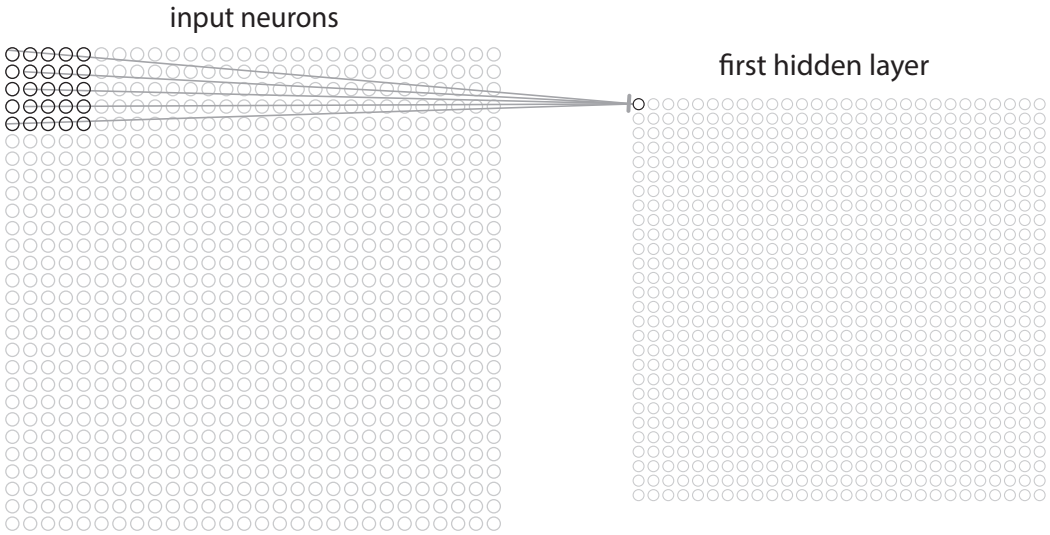


Figure 3.6: Sliding of the local receptive field in the input domain [71]

In Figure 3.6, the first neuron in the hidden layer is connected to the first local receptive field. Then to connect the second hidden neuron, the field is moved by one neuron (i.e. one pixel), as shown in 3.7

The entire input image is then covered using this strategy which in turn constructs the hidden layer. It is important to note that for an input image of 28×28 and local receptive field of 5×5, hidden layer will have the dimensions of 24×24. This is due to the reason because the local receptive field can be moved across 23 neurons (either across or down) before it collide with the bottom or right-hand side of the input image.

In Figures 3.6 and 3.7, the stride of one was shown, i.e., region or field moved by one pixel. However, the stride is a parameter that can be tuned and different lengths can be chosen depending upon the dataset and problems at hand. Stride controls the convolution output. It determines whether this output is generated for every pixel of the input image (stride 1) or for every nth pixel (stride n). Stride regulates how the

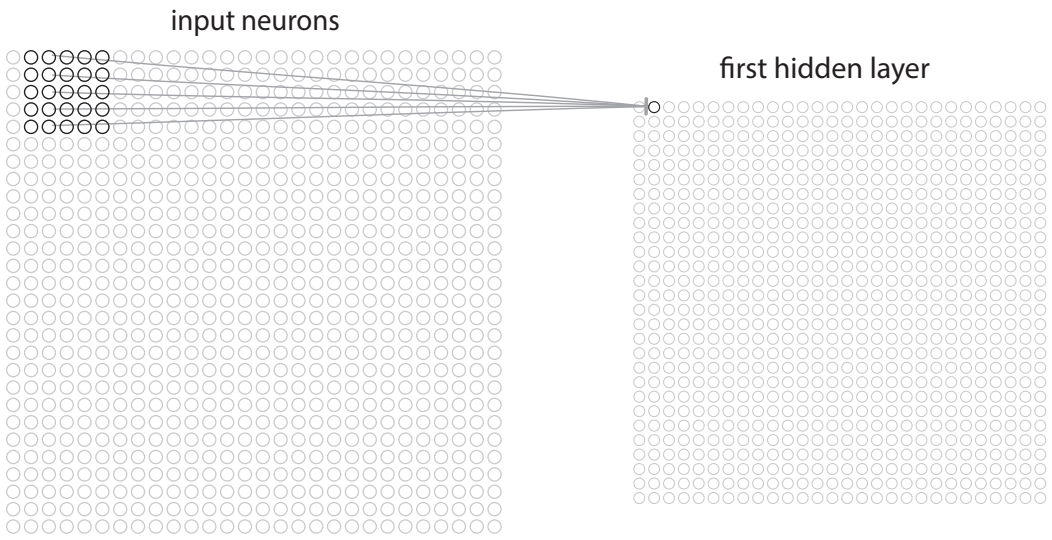


Figure 3.7: Sliding of the local receptive field in the input domain [71]

kernel will convolve around input data. This the output data can be downsampled, which results in the faster computation of the CNN.

3.3.2.3 Shared weights and biases

In the aforementioned example with an input image of 28×28 and local receptive field of 5×5 with a stride length of 1, each neuron in the hidden layer has 5×5 weights and a bias corresponding to its local receptive field. Another important feature of the CNNs is that the weights and bias are shared among every hidden neuron in a single hidden layer, i.e. same bias and weights are being trained for every neuron in the 24×24 hidden layer. The output of the j, kth hidden neuron can be represented as:

$$\sigma\left(\sum_{i=0}^{n}\sum_{i=0}^{n} w_{l,m} a_{j+l,k+m}\right) \tag{3.11}$$

Where, σ is the activation function which is being used to trigger the neuron, b is the bias which is being shared, $w_{l,m}$ is the array of shared weights, in this example this array would be of size 5×5 and $a_{x,y}$ is the activation of input at position x, y.

This shows that the entire first hidden layer will detect the same feature. Here the feature can be defined as a pattern, for example, an edge in an image or a curve, or the geometry of the detection, which is detected by the hidden neuron that activates the neuron. The complete first hidden layer will detect the same feature in the entire image but in different locations in the image. Suppose a hidden neuron learned the weights and bias to detect a horizontal edge in a particular receptive field in an image. Now this feature detector can be used anywhere in the image to find the horizontal edge in the given image. In the abstract terms, CNNs can handle the translation invariance in an image, for example, if the position of the pedestrian is moved in the image, but the CNN can still detect it as a pedestrian.

Because of this single feature detection, sometime the mapping from input layer to the hidden layer is also called a *feature map*. The weights and bias which define the feature map is called shared weights and shared bias. These shared weighs and shared bias are often termed as a *"Kernel"* or a *"filter"*.

A single feature detector or a feature map can detect a single localized feature in an image. To detect a complete object in an image, usually more than one feature maps is required. Thus, the combination of several feature maps builds one convolutional layer. This can be illustrated in Figure 3.8.

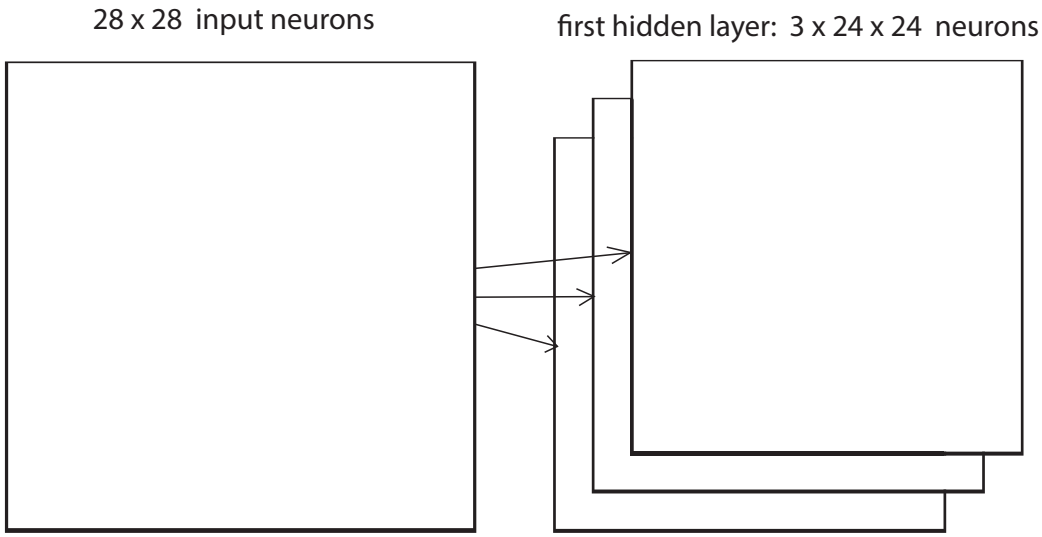


Figure 3.8: Sliding of the local receptive field in the input domain [71]

In Figure 3.8, 3 feature maps are shown. Considering the example, each feature map is characterized by a set of 5×5 shared weights, and a single shared bias. This results in detecting 3 different types of features by the network, where each feature is detectable throughout the image. However, as mentioned, in practice CNNs use many more feature maps to detect the object in an image.

Another advantage of using the shared weights and biases is that it reduced the number of trainable parameters in a CNN network. In our example, for a 5×5 local receptive field the number of weights needed will be $5 \times 5 = 25$ and a shared bias. Hence, each feature map comprises of 26 trainable parameters. If 30 feature maps are needed, then it will make $30 \times 26 = 780$ parameters and it will define a convolutional layer. However, in comparison with a fully connected layer, the input images are presented as a vertical layer of neurons which will be $28 \times 28 = 784$, and if a hidden layer has 30 neurons then the total trainable parameters would be $784 \times 30 \times 30(biases) = 23,550$. Therefore, a convolutional layer would require 40 times fewer parameters than the fully connected layer. This results in faster training of the network and helps in building a deep network of multiple convolutional layers.

3.3.2.4 Activation function

In CNN, it is typical to add non-linearity to the convolution layer using the non-linear activation function. They are described as a separate layer coming after each convolutional layer. An effective way to create a non-linear network is to use Rectified Linear Units (ReLU). A rectified linear function uses a ramp function to generate the output:

$$f(x) = max(x, 0) \tag{3.12}$$

This function is easily computable and differentiable for back propagation. In practice, ReLU has replaced sigmoidal functions, which offer smooth derivatives but suffers from slower computations and gradient saturation problem.

For multi-class classification tasks, the *Softmax* function is used:

$$\sigma(\overrightarrow{x})_i = \frac{e^{x_i}}{\sum_{j=0}^{n} e^{x_j}} \tag{3.13}$$

The Softmax function computes a vector of i arbitrarily large values and generates the vector of i values ranging from 0...1 and sums up to 1. These output values can also be utilized as class probabilities. A more detailed analysis of different activation functions can be found in Section 3.2.2.

3.3.2.5 Pooling

Convolutional layers in a CNN apply kernels to the input images to create the feature maps that detect the input features. These feature maps are sensitive to the location of features in the input, whereas, at the deep end of the network, these maps are used to recognize the multiple high-level patterns rather than the exact spatial location of the features [82].

A common approach to solve this problem is *downsampling* where a lower resolution of the images are used which still hold the information about the large or important structural elements without the fine details that may not hold much value for the underlying task. In other words, it simplifies the information in the output from the convolutional layer. Moreover, it also helps to keep the computational time manageable in a CNN. The downsampling can be achieved in two ways i.e. by introducing a *pooling layer* or *stride* (as mentioned in the previous section).

In detail, the pooling layer is added after a convolutional layer, specifically when an activation function (e.g., ReLU) has been applied to the feature maps. It efficiently reduces the size of activation maps. A CNN model with a convolutional and pooling layer may look as follow:

1. Input image

2. Convolutional layer
3. Activation function
4. Pooling layer

The pooling layer operates on each activation map to create a new pooled feature map of the same number. These pooled feature maps are the summarized version of the patterns detected in an input image. Pooling also results in making the network more translation invariant by causing the detectors to be less accurate. However, adding a pooling layer can also result in neglecting the spatial relationship information between sub-parts of the patterns. The two commonly used functions for the pooling operations are:

- **Maximum pooling** or max pooling outputs the maximum value for each patch of the activation map
- **Average pooling** calculates the average value for each patch of the activation map

An example can be seen in Figure 3.9, where the max pooling with the input region of 2×2 was used to output the maximum activation.

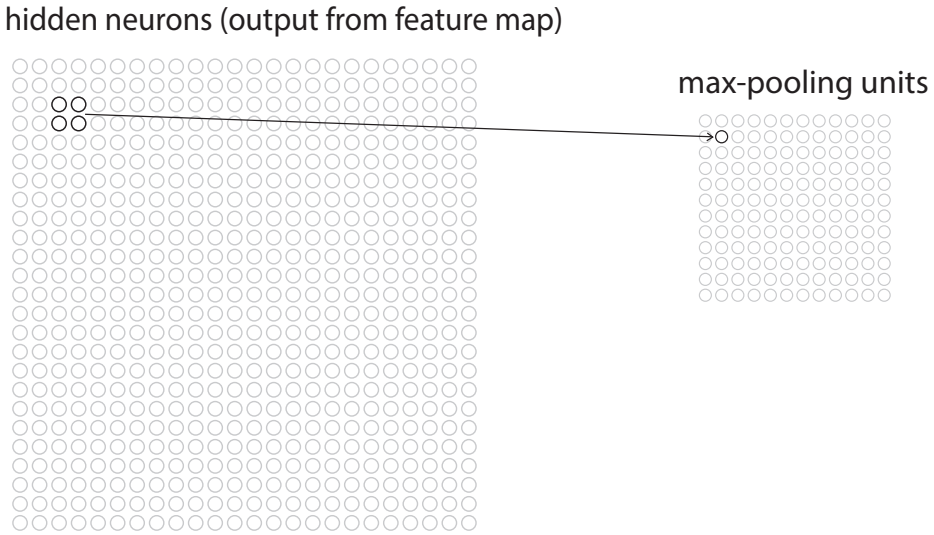


Figure 3.9: Max pooling applied on a single hidden layer [71]

In the example we have used so far, we received the hidden layer with 24×24 neurons, after the pooling layer with the 2×2, the output will be 12×12 neurons. As mentioned, more feature maps are needed to detect a particular object in an image. Therefore, multiple feature maps can form a single convolutional layer. Pooling is applied to each feature map in the convolutional layer. An example can be seen in Figure 3.9.

3.3.2.6 Fully connected and output layers

The final layer typically used for a CNN is fully connected layer. This is also called the "decision" or "classification" layer. In the final layer, the feature map matrix is flattened

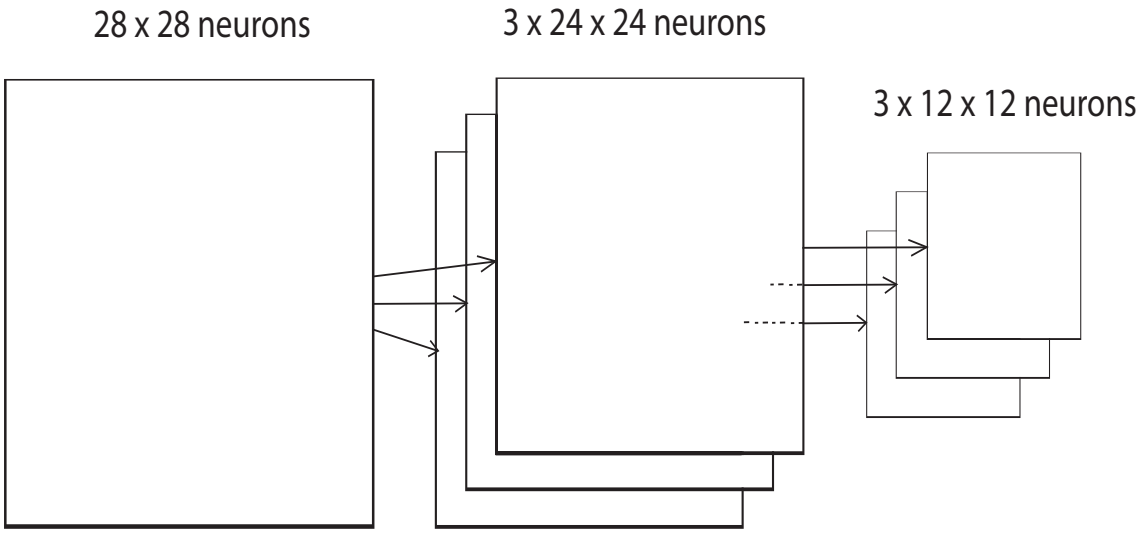


Figure 3.10: Max pooling applied on the complete convolutional layer [71]

into a vector and fed to a fully connected layer like a neural network. By the end of this, the neural network generates its output.

The error in the prediction is calculated in terms of the cost function, commonly referred to as the loss function in CNN, usually cross-entropy is used to achieve this. This loss function helps in determining the accuracy of the network. This information is then used to optimize the network to increase its effectiveness.

In the example we used with 28×28 neurons, the final architecture can be seen in Figure 3.11. Each neuron in the final layer is connected to each neuron from the pooling layer. The number of neurons in the final fully-connected layers refers to the classes defined in the training stages.

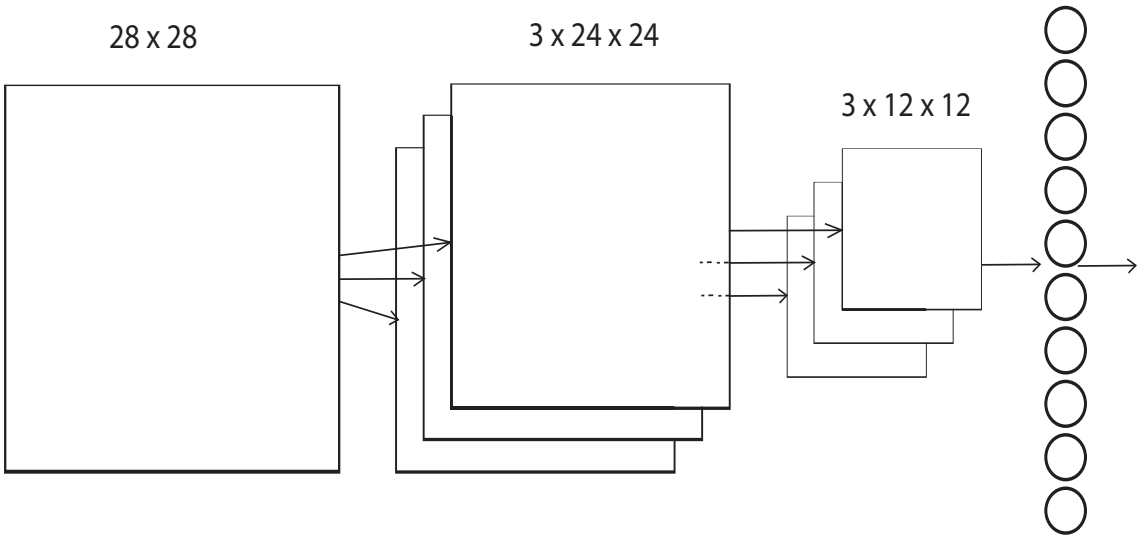


Figure 3.11: The CNN architecture [71]

3.3.3 Important terminologies

In this section, the important terminologies that are commonly used in neural networks and convolutional neural networks are defined.

3.3.3.1 Supervised and Unsupervised learning

Machine learning algorithms can be divided into two main categories; supervised and unsupervised [76, p. 3]. In supervised learning, data with their respective labels are fed to the model for training. Once the model is trained, another dataset similar to training data whose labels are kept hidden or unknown can be further fed to the model for the prediction. On the other hand, in unsupervised learning, prior labels are either inaccessible or accessible but they don't hold much importance for the application. Unsupervised learning, thus, dwells on studying how a model can infer functions to define hidden patterns or structures from unlabeled data. Semi-supervised learning is another category of machine learning. It is gaining its root where the aim is that the algorithm tries to exploit the use of comparatively small-sized data and rather large-sized unlabeled data. In this thesis, supervised learning is used, where the labels were given to the network while training and the network were asked to classify the output based on pre-given labels.

Another notable learning technique is Reinforcement learning. It is based on rewarding required behaviours and/or oppressing undesirable behaviours. Overall, in reinforcement learning an agent can perceive and interpret its environment, take actions, and learn through trial and error. Reinforcement learning is based on scalar feed which can be termed as weaker than providing labels, however, it depends upon the application.

3.3.3.2 Parameters and Hyperparameters

The parameters that are tuned or changed before starting the training of the model are referred to as hyper-parameters. These parameters include the Kernel size, the choice of activation function, the number of iterations to train the model and the learning rate of the model mostly termed as α. Tuning the hyper-parameters can have a significant impact on the outcome of the model.

The parameters of the model refer to the Weights (W) and Bias (b) of the model. These are learned by the model through the back-propagation algorithm. Therefore, it is important to identify and discriminate between the parameter and hyper-parameters.

3.3.3.3 Overfitting and Underfitting

Overfitting refers to a statistical model that captures the underlying patterns and noise in the training to the extent that it has memorized the training examples. But it is not

able to generalize the new data. Overfitting occurs if the model shows low bias but high variance.

Underfitting usually happens when the model has not learned enough from the dataset. And it is unable to define the correct relationship between the input and output which in result generate a high error and results in unreliable predictions. Underfitting occurs when the model shows high bias.

3.3.3.4 Epochs and Iterations

Epoch is the term which is used when the network has seen the complete data available in a training dataset, in other words when all the data has been passed through the network. Whereas iteration is used when the network sees one batch of the training data. If all the data has been passed in one batch, then iteration and epoch will be the same. Another example would be, if a training dataset consisting of 960 samples and a batch size of 32 has been used, then while training 30 iterations will make 1 epoch. The Number of epochs is a hyper-parameter which can be tuned. However, if the value is too large or too small, the network can tend to overfit or underfit and if the number of epochs is too small then the network needs more iterations to see all the data and ultimately converge.

3.3.3.5 Training, Validation and Test dataset

There are three different types of datasets that are being used by the research community. When a large dataset is available, it is then divided into three categories, *training, validation, and test* datasets. Training dataset refers to the data that is used to train the model. The validation dataset is used to check the performance of the trained algorithm. The test dataset is used to evaluate the performance of the final model after the training and fine-tuning of the hyper-parameters. This is done to see if the model is biased or not towards the unseen data.

4

Datasets

In this chapter, the dataset, to detect the curb from both sensors, i.e., Camera and LEDDAR, has been discussed in detail. The necessary steps, e.g., the essential analysis of the streets/pavements structure of Berlin, the statistics involving the older pedestrians, the prototype developed to collect the data, the software used to refine the dataset, etc. have been discussed in detail.

Dataset collection is one of the most pivotal tasks in Deep learning. Any supervised deep learning task requires good, structured and labelled data. This data could contain the meaning of images, videos, emails, driving patterns, phrases and so on. This dataset is then fed to a learning model for the desired prediction task. Unfortunately, despite our world being flooded by data, quintillion bytes a day, a huge amount of it is not structured or labelled. It means that for most supervised learning techniques, it is somewhat unusable.

The primary goal of our research is the detection of the curb and its surroundings accurately. It shows the transition between the sidewalk and the road. To train the network and detect the curb, appropriate data is required. This data must be representative of the task at hand, that is, in this case, the representations of the road junction that the older pedestrians would use. Since there was no existing dataset available, a new dataset had to be collected and annotated from scratch.

For the detection of the curb, two sensors were used: A camera and LEDDAR as discussed in Chapter 2, section 2.3. Both the sensors fulfil our requirements. The **camera** used in this research was chosen based on its weight and price. Moreover, it also should be computationally inexpensive. Hence, the camera selected was the USB Webcam HD C270 from Logitech which has a focal length of $4.0mm$ with an optical resolution of 1280×960 and a maximum frame rate of 30fps @ 640×480 with the $60°$ field of view [89].

Similarly, the **LEDDAR** which was used in this research is LEDDAR M16, it is an economical distance sensor which doesn't weigh too much and like a Camera, it is also computationally inexpensive. The LEDDAR used in this research is a 16-segment solid-state LiDAR sensor module. The LEDDAR M16 sensor module uses 16 independent detection channels to deliver continuous and precise detection combined with exceptional lateral discrimination. It has a detection range of 146m and a data acquisition rate of up to 100 Hz [61].

4.1 Pre-analysis as a basis for the data collection

As this prototype was developed specifically for Berlin, before we could start collecting the dataset for training, it was necessary to analyse the different sidewalks existing on the streets of Berlin. This analysis was imperative in concluding what the data should be comprised of concerning the existing pavement structure and its frequencies. The algorithm should be robust against the different weather conditions, illumination effects etc. Therefore, the dataset should also take into account important aspects like weather conditions, crash statistics of Berlin etc.

4.1.1 Analysis of Berlin sidewalks

To gain a better understanding of the different structures and types of sidewalks in Berlin, the sidewalks in Berlin were thoroughly analysed. According to the Department of Urban Development in Berlin, the minimum legal width of the sidewalks existing in Berlin is $2.5m$ [3]. Officially, the typical existing sidewalk structure is divided into three main zones and are called "upper stripe, central walkway and lower stripe". If we look at the sidewalk standing on the street, the first zone after the curbstone is the "lower stripe" and after it comes to the "central walkway". Once the central walkway ends, the upper stripe begins. The upper and lower stripes are primarily made of concrete slabs and cobblestone. The central walkway is mostly made of artificial stone or granite slabs [3]. For this analysis, the sidewalk was divided into three parts, where zone 1 represents the transition between curb and road or the lower stripe. Zone 2 represents the central walkway and zone 3 represents the upper stripe. An example of a sidewalk in Berlin with zone divisions can be seen in Figure 4.1.

Another important structural aspect to consider is the height of the curb stone. The height of the curb can vary depending on the location of the curb. If it is situated on the pedestrian signal, the height sinks significantly, to keep it convenient for pedestrians. Therefore, we can call it an official crossing. On the other hand, the height of the curb at the pedestrian crossings, where there is no pedestrian signal present, is elevated and therefore termed an unofficial crossing. The height of the curb at the unofficial crossings in Berlin remains constant with little or no variations. Therefore, the official

and unofficial crossing also plays an important role in the analysis. In official crossings, there is a traffic signal present to facilitate the pedestrian, hence, the main concern is the unofficial crossing, where there is no traffic signal present to help the pedestrians, and this is where the most accidents occur as older pedestrians sometimes are not able to see the oncoming traffic.

Figure 4.1: Subdivision of the sidewalk in three different zones (taken from [62])

After a basic structure analysis, it was necessary to find out the different types of curbs. Berlin is divided into 12 districts. To figure out the existing types of sidewalks, each of these 12 districts were analysed. For this purpose, a total of approx. 1000 random points were selected from all over Berlin by keeping the share of each district homogeneous. These data points were categorized using ArcGIS; ArcGIS is the Geographic Information System Software, which is used for creating maps and analysing geographic data. A rough sketch of the points distribution can be seen in Figure 4.2.

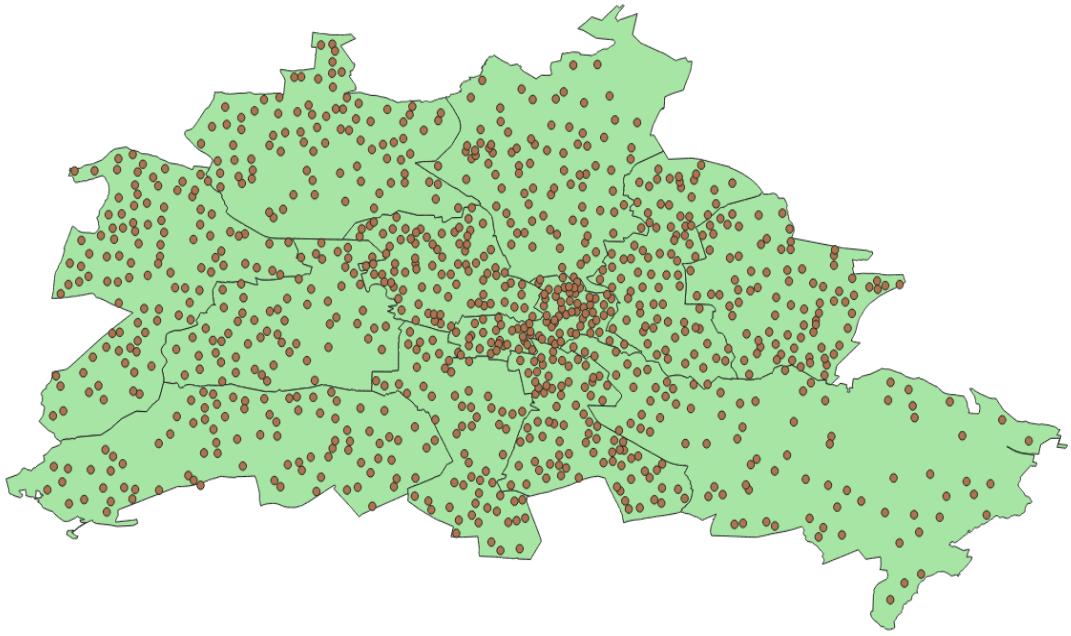


Figure 4.2: Map of Berlin depicting 1000 randomly selected points

To visualize these data points Google Earth Pro was used. Google Earth Pro is a program that is used to render the 3D presentation of the Earth based on satellite imagery. Each data point was conceived in Google Earth Pro to find out the existing sidewalk structure. An example of a data point being visualized in the Google Earth Pro can be seen in Figure 4.3. A few points where there was no street view available were discarded from the analysis.

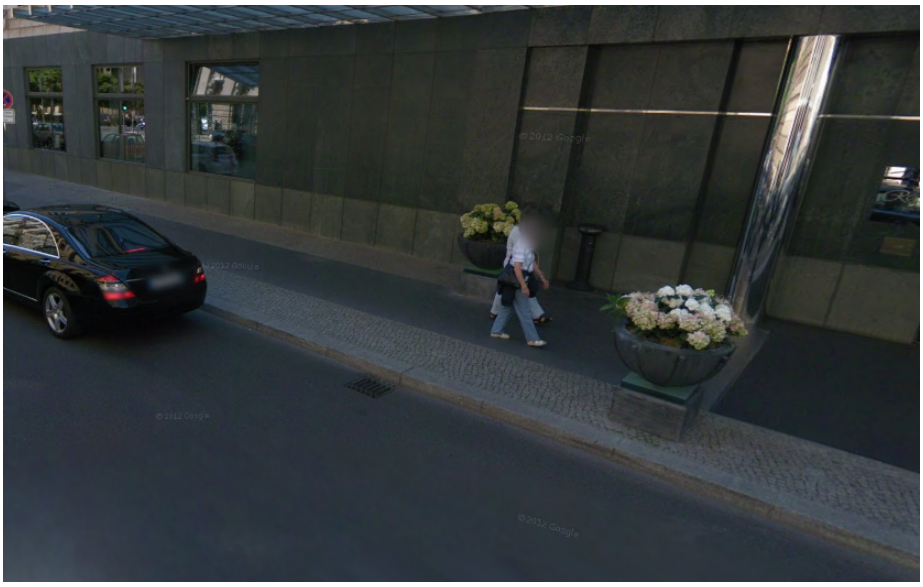


Figure 4.3: A street view of Charlotten street in Berlin taken from Google Earth Pro

While analysing each data point, multiple things were inspected:

- **Sidewalk structure:** The type of sidewalk and number of existing zones
- **Curbstone:** Characteristics of curbstone were analysed, like type and shape
- **Road structure:** The material of the road was also analysed to determine whether it is Asphalt, stones or dirt
- **Height of the curb:** The height of the curb was also analysed at the official and unofficial crossings

Our focus lies on determining the types of structure existing in predefined zones, and a conclusion was drawn from this analysis. The results show that there are eight different types of sidewalks generally exist in Berlin, namely:

1. Cobblestones
2. Concrete stones
3. Grass shoulder
4. Concrete pavement
5. Asphalt pavement
6. Natural stone

7. Large stone
8. Concrete slabs

From the analysis, it was also concluded that the most commonly occurring pavements on the sidewalks in Berlin are cobblestone, concrete stone, and grass shoulders. An example can be seen in Figure 4.4

(a) Cobblestone **(b)** Concrete stone **(c)** Grass shoulder

Figure 4.4: Examples of the three most occurring walkway surfaces in Berlin

The statistics exhibiting the most common sidewalk structures for the three zones are shown in Figure 4.5.

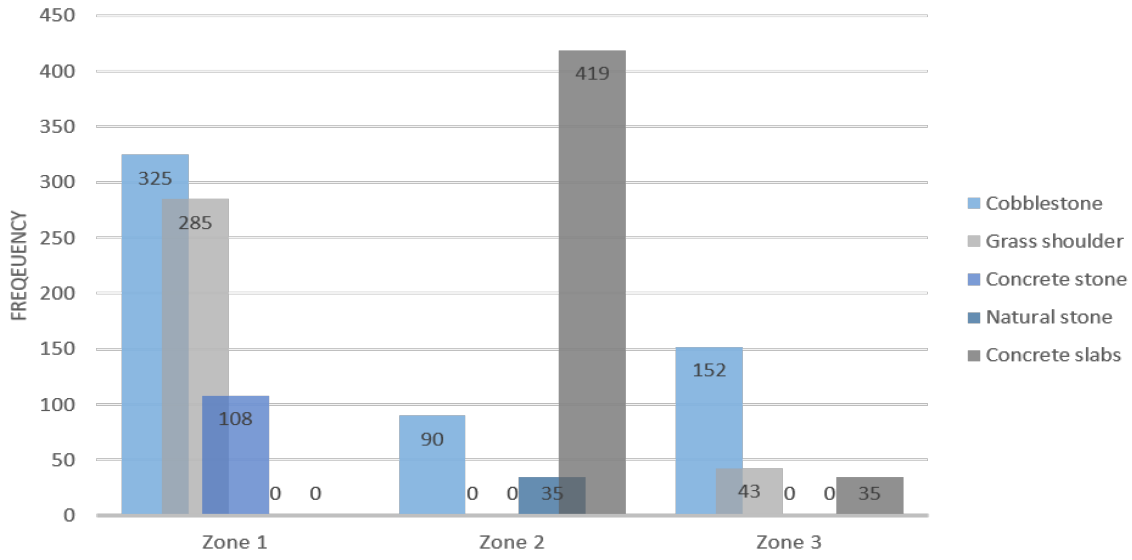


Figure 4.5: Frequencies of different pavement surfaces in the three zones

The focus for the dataset characterizing a sidewalk lies in Zone 1, which is closest to the curb and is captured by the camera accordingly. The most used material is cobblestone (Figure 4.4a). Zone 2, on the other hand, has relevance, especially for pictures that don't show the road. This information is captured by the camera when the users walk along the street with no intentions to cross the street and thus move parallel to the curb. In Zone 2, the most occurring material is concrete stones (Figure 4.4b). Although the statistics were also created for Zone 3, however, the occurrence of

Zone 3 in many walkway structures is relatively less. Also, it doesn't matter much for the training session because the distance of the road from Zone 3 is more than 2 meters, hence, exceeds our basic requirement. This is the reason that Zone 3 was not considered for the creation of the dataset.

From each district analysis, Charlottenburg and Spandau districts were chosen for the data collection because they have the highest ratios of cobblestone as pavement structures i.e. 10.28% and 8.06% respectively. Moreover, these districts also provide a good combination of pavement structures regarding their availability and their combinations with other pavement structures.

4.1.2 Creation of data matrix

To make the system more robust, the detection algorithm must also work efficiently in diverse weather and light conditions. The majority of the accidents occur in lower visibility conditions i.e., in the evening or on rainy days [22]. Therefore, the system should also consider the time of the day. Moreover, it must be robust against obstacles such as parked cars which can provide hindrance and bring a new aspect while designing the detection algorithm. Similarly, leaves on the ground can hide the curbstone partially or completely and make it difficult for the system to perform efficiently. Another important point to mention is that pedestrians do not necessarily cross the street at an angle of 90° to the road, hence different possible angles of crossing the street should be considered while crafting a design for the assistance system. To systematize this data collection, a matrix was created, which contains all these aspects and considers the conclusions drawn from the Berlin sidewalk analysis. This matrix also considers the number of crashes in each district, weather statistics e.g., the number of sunny/rainy days in a year etc. Percentages were assigned according to relevance based on the frequency of roadside accidents involving older pedestrians.

The overall percentage (100%) of the dataset was divided into three main categories, 1) Morning - 30%, 2) Noon - 30%, and 3) Evening - 40%. Afterwards, these three categories were then further divided into two categories High visibility (HV) and Low Visibility (LV) because visibility plays an important role in traffic crashes and in training an algorithm based on camera images. Another important aspect considering the training of algorithms can be the state of pavement which means either it is wet or dry because the intensity of pixel values changes comprehensively, therefore after the visibility category, a new setting was introduced comprising of the wet and dry floor. And then these two categories (dry and wet floor) were divided into four sections which are 1) leaves, 2) parked cars, 3) obstacles and 4) free images. Free images represent a category where a clear pavement structure can be visualized e.g., without parked cars, obstacles, leaves, etc. Obstacles represent the main holes, poles, trees, bicycles on the

sidewalk, people, etc. An overview of the categorical division of the dataset can be seen in Matrix 4.1.

Morning	30%						
HV	10%			**LV**	20%		
Dry Floor	7%			Dry Floor	12%		
Leaves	P. Cars	Obs.	Free Img	Leaves	P. Cars	Obs.	Free Img
2%	2%	1%	2%	2%	4%	2%	4%
Wet Floor	3%			Wet Floor	8%		
Leaves	P. Cars	Obs.	Free Img	Leaves	P. Cars	Obs.	Free Img
0.5%	1%	0.5%	1%	1%	3%	1%	3%
Noon	30%						
HV	20%			**LV**	10%		
Dry Floor	15%			Dry Floor	7%		
Leaves	P. Cars	Obs.	Free Img	Leaves	P. Cars	Obs.	Free img
3%	5%	3%	5%	2%	1%	2%	2%
Wet Floor	5%			Wet Floor	3%		
Leaves	P. Cars	Obs.	Free Img	Leaves	P. Cars	Obs.	Free img
0.5%	2%	0.5%	2%	0.5%	1%	0.5%	1%
Evening	40%						
HV	20%			**LV**	20%		
Dry Floor	10%			Dry Floor	10%		
Leaves	P. Cars	Obs.	Free img	Leaves	P. cars	Obs.	Free Img
2%	3%	2%	3%	2%	3%	2%	3%
Wet Floor	10%			Wet Floor	10%		
Leaves	P. Cars	Obs.	Free img	Leaves	P. Cars	Obs.	Free Img
2%	3%	2%	3%	2%	3%	2%	3%

Table 4.1: The matrix showcasing the categorical division of the dataset

The abbreviations used in the Matrix are:

- HV - High Visibility
- LV - Low Visibility
- P. Cars - Parked Cars
- Obs. - Obstacles
- Free img - Free images

4.2 Prototype

Considering the requirements of the system and after the detailed analysis of Berlin sidewalks, a prototype was developed to collect the datasets for the camera and LEDDAR. While developing this prototype it was also considered that this will also be used to test the system in a real-time environment with the target group i.e., older pedestrians. Hence, a walker was used to mount all the sensors and devices so that it doesn't put too much strain on the older pedestrians.

The prototype consisted of the following items:

- Walker (Invacare Banjo P452E/3)
- USB Webcam HD C270 from Logitech
- Leddar M16
- Notebook (Lenovo Y720-15IKB)
- Arduino Uno WiFi

Figure 4.6: Prototype with the customized foundation and sensors

Figure 4.6 shows the walker which was used for data collection and later for testing purposes. A customised structure was made for the walker to mount the sensors safely and fixedly. A synthetic housing was developed for the camera using a 3D printer. The notebook is used to connect the camera and the LEDDAR using a USB interface. This prototype served to record the datasets, from camera and LEDDAR, for the experiments in the vicinity of the curb and its surrounding in different conditions of light, weather, and obstacle.

4.3 Camera dataset

The camera was used to film the curb and was mounted in a way on the walker that it is looking at the surface with an angle to fulfil the requirement of detecting the curb within the distance of $2m \pm 1$. The video was sampled at ten frames per second (FPS) as the higher sampling rate led to redundancy and thus did not provide additional beneficial information. The images were collected following the data matrix 4.1, created solely for data collection.

To collect the data, the multiple streets in chosen districts of Berlin were filmed in various scenarios mentioned in Table 4.1, and obstacles such as pedestrians, parked cars, bicycles, trees, poles, sewage holes, fire hydrants, etc. were filmed. The video were filmed at different times of the year and in every season to capture the essence of each season and to incorporate all the scenarios. This video was then used to extract the images which were then used to train the network. Initially, for the detection, around 150,000 images were extracted from the films made of curb stone and the obstacles.

The main goal of our research is to detect the scenarios where the user i.e., the older pedestrians, faces potentially hazardous situations. Therefore, rather than detecting each scenario or obstacle as a separate class, it was decided to opt for the binary classification task. The two classes represent the scenarios: a) where the users should be given an alarm considering the hazardousness of the situation or, b) where the users should not be given an alarm when the surroundings don't represent a situation where the older pedestrians can be hurt potentially. The classes were initially named positive and negative classes. Hence, the dataset was collected and divided based on these two classes categorised as positive labelled images and negative labelled images. In terms of the data collection, it is important to mention that different recording sessions were conducted to create the dataset for each class and then the images were extracted respectively.

4.3.1 Positive labelled images

Positive class depicts the scenarios when the pedestrian is going to cross the street either it could be an official or unofficial crossing. It is important to remember that most crashes happen at unofficial crossings. Moreover, different approaching angles were used to cross the street.

The data collection was initialized by collecting the images from the category of free images. Free images are the images in which the curbstone can be seen clearly without the cars, obstacles and leaves present. A few examples can be seen in Figure 4.7.

In Figure 4.7, different images in a collage are shown showcasing the curbstone in different environmental conditions. While collecting the images the visibility conditions, type of pavement, angle of approach, the height of the curb, dryness/wetness of the pavement, etc. were taken into account.

Figure 4.7: A few examples from the category "free images"

After the collection of the free images, the date was expanded by taking the images from the leaves category, which can be seen in Figure 4.8. All the scenarios were exploited again while collecting the images from this category. These images had to be categorized because they hold importance when the pavement is covered in leaves, especially in the season of Autumn when everything is relatively camouflaged with leaves.

Figure 4.8: A few examples from the category "leaves"

Afterwards, the images were collected with the obstacles in the frames. Here, all the obstacles lying in the pedestrian's way were filmed irrespective of their location, i.e., closer to the curb or not, because the obstacle provides a hindrance in any case. It can also create problems for older pedestrians even if they are walking alongside the curb and have no intentions to cross the street. An obstacle could be a pole, sewage holes, trees, bicycles, etc. A few examples can be seen in Figure 4.9.

Figure 4.9: A few examples from the category "obstacles"

Another important aspect that was covered while collecting the dataset was parked cars. It represents the importance in the case when the pedestrian is going to use the unofficial crossings where the cars are parked. In Berlin, cars can be parked alongside the curb stone. Hence, it was vital to consider this information in the dataset. Different angles of the cars were considered while collecting such images. A few examples can be seen in Figure 4.10. Moreover, the idea was that system should also be able to detect the cars if they are parked partially on the curb and partially on the street either in horizontally or vertically.

Figure 4.10: A few examples from the category "cars"

Another important factor which impacts hugely on the training of the network to detect the curb is the condition of the pavement, whether it's dry or wet. Because, in some cases, e.g., rain, the surface of the pavements changes the colour significantly and

have a significant impact on the pixel values in the images which in turn hold importance in the training of the network. To elaborate on this, a few images have been shown in Figure 4.11 which show the difference in the appearance of different surfaces.

Figure 4.11: A few examples from the category "wet floor"

Till now, all the categories in the positive labelled class have been shown. These collages also demonstrate various visibility conditions. These categories were important in terms of training a network to detect the curb stone at a predefined distance in various scenarios.

4.3.2 Negative labelled images

The negative class represents the case when the pedestrian has no intention to cross the street. In this case, the pedestrian is walking on the pavement or can also walk alongside the curb if the pavement is too small.

Hence, for the negative class images, scenarios were considered where the pedestrian is walking parallel to the curb and isn't going to cross the street. In addition, while collecting these images following aspects were also taken into account: the visibility conditions, surface condition (in terms of pavement structure, dry/wetness of the surface, etc.), leaves, obstacle, shadows, etc. Some of the samples are shown in Figure 4.12.

4.4 LEDDAR dataset

The LEDDAR has been mounted on a walker so that the channels are facing the curb stone vertically with an approximate angle of 45° in the predefined distance of $2m \pm 1$.

Figure 4.12: A few examples from the negative labelled class

4.4.1 Schematics

The schematics can be seen in Figure 4.13.

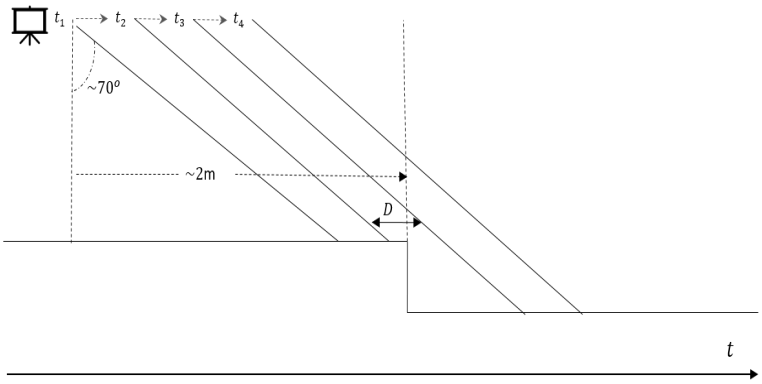


Figure 4.13: Schematic diagram of the LEDDAR sensor

In Figure 4.13, "t" represents the channels of the LEDDAR in different time intervals. When the walker is moved towards the curb stone and approaches the predefined window, the LEDDAR detects the difference in height between the curb and the road. In consequence, it indicates a change in value, as the lateral distance between the LEDDAR receiver and the deflecting surface increases. This means that, for one channel e.g., channel number 16, at t = 0, the deflecting surface is the pavement and, at t = 1, the deflecting surface is the road. Therefore, a profile can be made where this channel indicates this height difference. This provides an array of data in the shape $(16, 1)$. Similarly, a profile can be made for each channel. The software Lab Streaming Layer (LSL) was used to collect the data with the relative timestamps. LSL makes it

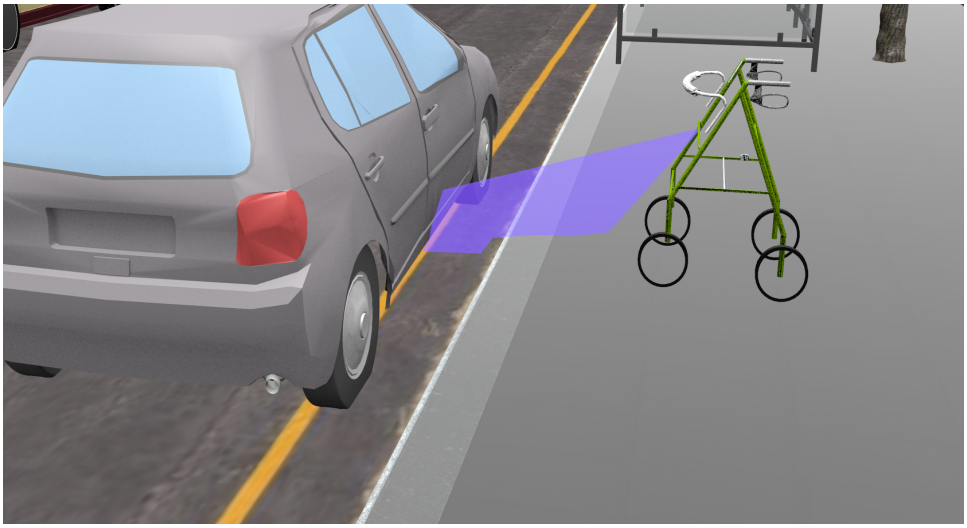


Figure 4.14: A 3D model of the LEDDAR sensor to detect the curbstone

possible to provide different data streams with timestamps and refers to the same clock (of the notebook). Recordings with LSL are implemented in the form of a publisher receiver design pattern. On a notebook, a receiver, the so-called lab recorder, record data streams and saves them in a file, including metadata and timestamps.

Similar to the images' dataset, two classes were defined: positive and negative. A 3D model of the LEDDAR can be seen in Figure 4.14, where it can be seen how the LEDDAR looks at the curb.

4.4.2 Visualization and collection of LEDDAR data

To visualize the arrays from each channel of LEDDAR, LEDDAR Software Development Kit (SDK) was used. It can be seen in Figure 4.15.

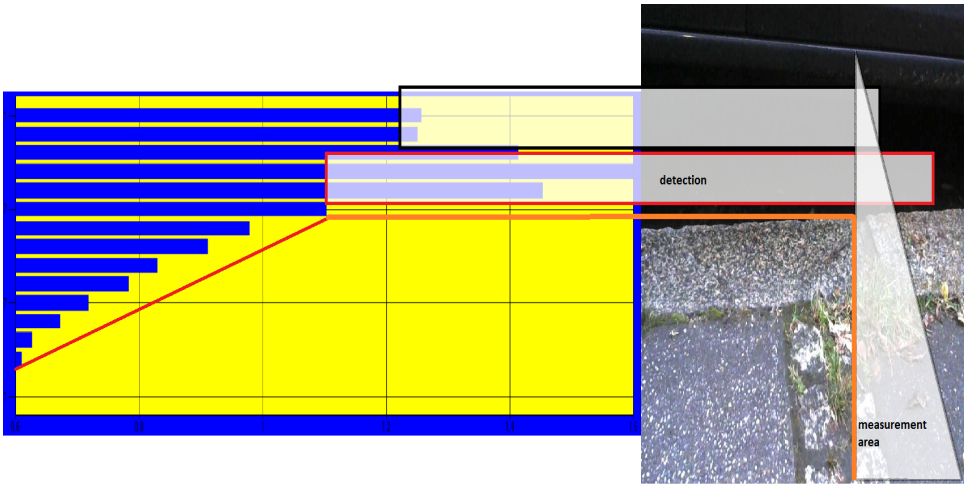


Figure 4.15: Real-time depiction of the LEDDAR's 16 channels

The left half of the figure shows the 16 channels of the LEDDAR. The y-axis represents the number of channels, and the x-axis represents the lateral distance between the sensor and the rebounding surface. The right-hand side of the figure depicts the

scene of a curb stone with the parked car at the predefined distance of $2m \pm 1$ from the pedestrian. Both

images in the figure are related in a way that both sensors (Camera and LEDDAR) are capturing the same information. It can be seen in the figure that channel number 16, 15 and 14 are rebounding from the surface of the parked car. Whereas, channel number 13 and 12 are rebounding from the surface of the road. It is also noticeable as the difference in distance values between channel number 13 and 12 is comparatively large. If the LEDDAR is moved towards the curb stone, the lower channel will come into action and it will detect the lateral difference between the curb and the street. Therefore, the information can be extracted from the channel values to find out which sensor is detecting the curb. Similarly, it can also calculate the distance between pedestrians and curb stones. The dataset for training was collected and the categories from data matrix 4.1 were considered. Moreover, the data was also collected for the negative labelled class.

5

Detection with the Camera

This chapter presents a novel approach for detecting the curb stone and its surroundings from a pedestrian's point of view using a mono camera.

To recognize the curb and its surroundings with a camera, the images must be analysed accordingly. Due to this, different computer vision techniques were compared. The basic idea was to extract the features from the images and classify them to identify the potentially hazardous situations for older pedestrians in traffic scenarios. Research and experiments were done on the more traditional image processing techniques like edge detection, therefore, different filters e.g., Canny filter [90], Sobel filter [91], and Prewitt filter [92]. These were tested to extract the features of the curb in real-time scenarios. These techniques are computationally fast but are not robust enough to tackle the detection task in different scenarios. For example, when different structures of the curb and pavement, weather, obstacles, etc. are considered. Therefore, it was decided to use a convolutional neural network (CNN) using end-to-end learning. It provides more flexibility in terms of feature learning.

CNNs are the best choice so far for extracting valuable features from image datasets. CNNs have reformed pattern recognition techniques. Before the extensive acceptance of CNNs, most of the pattern recognition tasks were achieved using hand-crafted features followed by a classifier. The leap of the CNNs is that using the CNN features are extracted from the images automatically using training examples. The CNN techniques are specifically influential towards image recognition tasks where multiple datasets have been leveraged using CNN (e.g., MNIST [93], CIFAR-10 [94], ImageNet [95], etc.)

The approach used in this research was inspired by the work of Bojarski *et. al.* [96]. They successfully implemented an end-to-end learning algorithm using a CNN in the context of autonomous driving, where they used a 9-layer network including normalization and three fully connected layers to develop an end-to-end control system

for the vehicle's steering. This work was an extension of the work of Muller et al. [97], who pitched the idea of end-to-end learning.

End-to-end learning is, if a network is being used in the context of end-to-end learning, it learns the whole processing pipeline without the need to label explicit parts of the data. For example, in the case of the image dataset, it is sufficient to label the whole image rather than label the individual pixels in the image, which saves a considerable amount of time and effort regarding the annotation of the data.

5.1 Training of algorithm

The major components required to train the CNN network are the data and the appropriate network architecture. Once the camera dataset was built (see Chapter 4.3)), the next step was to select the data that represented all the scenarios in the traffic environment. This also include training the network on the selected dataset to detect the curb efficiently.

5.1.1 Selection of the data

One of the biggest problems the scientific community faces while training a deep neural network is the collection and selection of appropriate data in a usable format. This data should be able to correlate with the problem at hand i.e., the data should portray the problem. It must contain all the possible scenarios that are part of the defined problem. The data selection stage is important. The outcomes of a trained network must equate with the expected predictions, and the efforts to build a trained network must return to the data collection and selection stage.

Deep learning algorithms generally need a good amount of training data to work adequately. This data is called a training set which acts as a criterion against which an algorithm is trained. Selecting the required training dataset requires time and expertise. It defines that the most important thing for the network is to extract the features.

Deep learning usually works with three main types of datasets: training, validation, and testing. Feeding the training set to the network results in the training of the model. Using the training dataset, a network learns to weigh different features. It adjusts the parameters according to their likelihood to minimize errors. The parameters or coefficients are accommodated in the forms of tensors and collectively they are called the model. Because they encapsulate a model of the training set.

The validation dataset is used to validate the accuracy of the trained algorithm. Once the network has been trained and optimized, validation data is fed to the network. It is to validate whether it can recognize the images correctly. If the accuracy is less than the desired percentage then, the training parameters need rechecking. Another

dataset used in this research is the test dataset. This dataset is used to have an unbiased estimate of the final model whence it is passed through the model for model evaluation.

The data used for the training, validation and testing purposes should be able to comprehend the task at hand effectively i.e., detection of the curb. In this research, the whole dataset was divided into two classes: positive and negative with a ratio of 50 : 50. Which means half of the dataset is comprised of the images that contained information about the presence of the curb in the data and if the pedestrian is going to cross the street. While the other half of the dataset contained images which depicted the scenarios when there is no curb present in the frame or when a pedestrian has no intention to cross the street and is walking alongside the curb.

5.1.2 Data augmentation

Deep learning generally uses a tensor or a multi-dimensional array as a format. Therefore, the data pipelines used in deep learning usually convert all data, which could be image, video, sound, voice, text, or time series, into vectors and tensors on which linear algebra techniques can easily be implemented. However, this data generally needs to be standardized, normalized or cleaned for effectiveness.

In other words, deep learning algorithms are statistical algorithms, hence, there is a possibility that the algorithm doesn't classify the images with curb in them correctly. And declares those falsely as the negative labelled scenario which could potentially raise a hazardous situation for the user. There could be many possible reasons for this, the most common could be iterated as the ineffectiveness of the data. It was mentioned in the previous section that the dataset is as important as the network itself.

In the case of curb detection, the reasons could be coined as insufficient illumination, motion blur, or real-time scenarios which could not be captured while collecting data but are plausible to be seen by the model. For example, different angles of approach, different heights, width, and colours of the curb, etc. among many others. Therefore, to tackle such problems, data augmentation techniques were used. These techniques are also being widely used to artificially increase the amount of training data. Domain-specific techniques can be applied to generate a transformed version of the images from the existing training data sample that belong to the same class as the original sample.

Deep learning models like CNNs can extract features from the images, invariant to their location in the image. Nevertheless, augmentation techniques can further help to transform invariant courses into learning. It can also help the model to extract features that are invariant to transform i.e., illumination level in images, ordering of the images, etc.

Data augmentation can further aid the model to counter the overfitting problems (see section 3.3.2) by making the dataset large or diverse. Therefore, after the selection of the final frames the data was augmented using different techniques.

The techniques that were used frequently in the training of our model include artificial rotation, shear effects, artificial shifts, and horizontal flipping.

Artificial rotation is a widely used augmentation technique and it helps the model to become immune to the orientation of an object in the image. The image can be rotated between 0° to 360°. Once the image has been rotated, some pixels move out of the image, and it leaves behind an empty area. There are different techniques available to fill this empty area. The system was designed in a way that the nearest pixel values are used to fill this empty area. Moreover, it was designed in a way that it rotates the images to 40°. An example can be seen in Figure 5.1. The left-hand side of the image shows the original images and the right hand side shows the augmented images.

Figure 5.1: Images showing the effects of artificial rotation augmentation

Shear effects are used to shear angle in counter clockwise direction in degrees. An example can be seen in Figure 5.2.

The artificial Shift range controls the amount of horizontal and vertical shift, i.e., it moves all the pixels in a predefined direction, either horizontally or vertically. Keeping the dimensions of the image intact. Figure 5.3 shows the vertical shift augmentation in an image.

Figure 5.4 shows the horizontal shift augmentation in an image.

An **image flipping** refers to reversing the rows or columns. In horizontal flips, the columns are reversed to alter the image appearance. This can be seen in Figure 5.5.

Using the data augmentation techniques, the model was trained on the various superficial conditions that can differ significantly in appearance in real time. It is important to mention that the data augmentation techniques are usually applied only to the training datasets. It is not applied to the test or validation dataset, in this research, these techniques have only been applied to the training datasets. Moreover, the data augmentation techniques are different from data preparation techniques such as pixel scaling or image resizing.

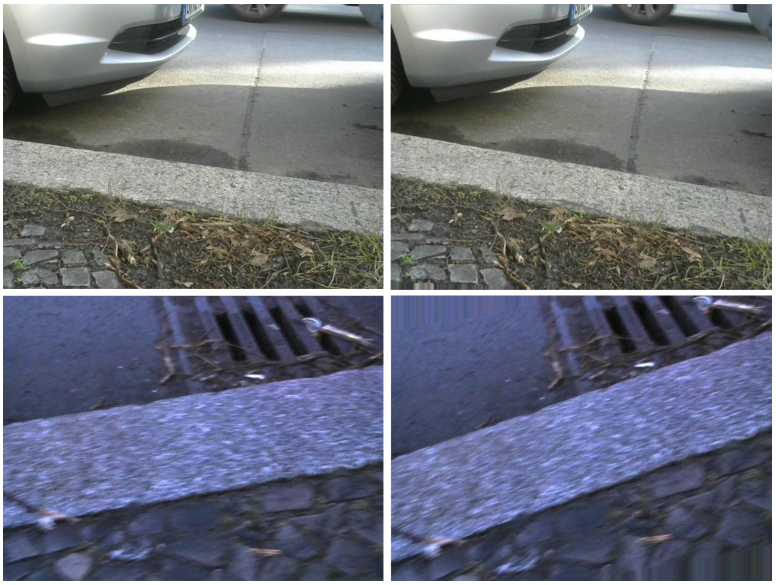


Figure 5.2: Images showing the effects of shear augmentation

Figure 5.3: Images showing the effects of vertical shift augmentation

5.1.3 Network architecture

Another important stage which has a huge impact on the outcome of a CNN is the selection of the architecture of CNN. A reliable architecture can be a key factor in determining the performance and efficiency of the network.

The key elements that are used in the architecture (e.g., convolutional layers, pooling layers, filters, etc.) are relatively straightforward to comprehend. There are almost limitless ways to design a deep net to solve a computer vision problem [98]. The challenging part of using a CNN is to design a model architecture which best exploits these simple layers. Because how the layers are assembled, or the elements are used often impacts the accuracy and speed of the network for a prediction problem. In designing an architecture, the most important steps are:

Figure 5.4: Images showing the effects of horizontal shift augmentation

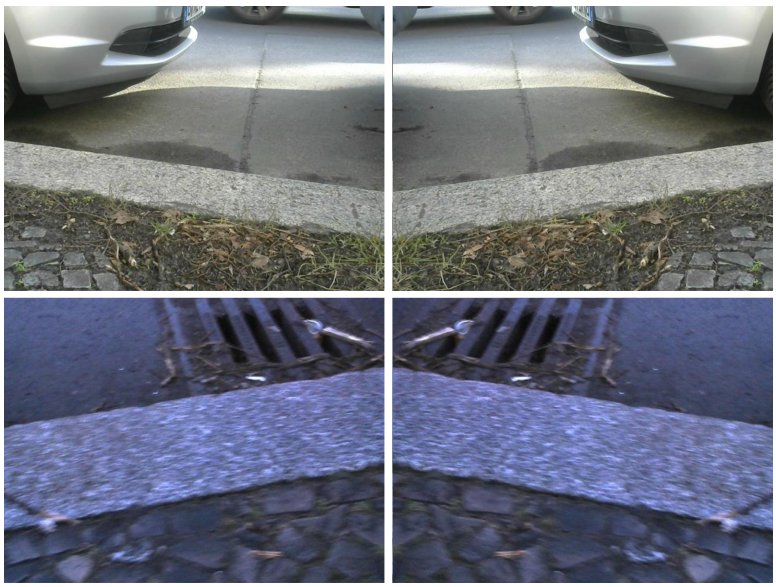


Figure 5.5: Images showing the effects of horizontal flip augmentation

1. To arrange convolutional and pooling layers in such a way that the model is well suited to perform the task at hand efficiently
2. To design the number of filters and filter sizes in the CNN
3. To tune the rest of the hyper-parameters of the network optimally

It is also important to mention here the hyper-parameters related to the training of a CNN will be discussed in this chapter. The hyper-parameters related to the structure of a CNN have already been discussed in detail in Chapter 3 and section 3.3.2.

- **Learning rate** represents how fast a network revises its parameter. A high learning rate speeds up the learning process but might not be able to converge, conversely, slow learning slows down learning, but the convergence can be smooth.

- **Number of epochs** is a hyper-parameter which is defined before training a model. When the entire dataset has been passed through the CNN both forward and backward once then it refers to one epoch.

- **Batch size** is the number of samples or sub-samples given to the CNN after which a parameter update occurs.

Fortunately, in the world of computer vision, the appropriate and adequate information regarding the above-mentioned points can be found in the previous research in terms of architectural innovations and common patterns to develop an efficient network [95, 99, 100, 101, 102, 103]. After studying the state-of-the-art research architectural designs, intuition and a rationale can be developed on how to design a network efficiently to perform the prediction task.

Subsequently, it was decided to design an innovative end-to-end deep net that can handle the prediction task of curb detection precisely instead of using the already existing architecture. To assess the network efficiency and capability for this task, the training was done in several stages. As already mentioned, the task was approached as a binary classification task where the classes were defined as, positive and negative classes. The positive class represents the scenarios when the curb is present in the frame and the curb lies at an angle in the frame which shows the pedestrians' intent to cross the street. The negative class depicts the cases where there is no curb present in the frame. The negative class also depicts where the pedestrian is walking alongside the curb but has no intention to cross the street.

To find out the best CNN architecture, the correct type and the amount of data to predict the curb with the camera, the training of the CNN was done empirically in several stages. Here, only two cases have been mentioned: the first, where we started with the camera detection and the final case when the maximum accuracy was achieved.

5.1.3.1 First simulation

Initially, to ascertain the system's competence, relatively simple images of the curb as the positive class were used, these images were termed "free images". Free images are images without obstacles, leaves, parked cars, etc., as shown in Figure 5.6. Contrarily, the negative class, images were used where no pavements or curbs were visible in the image. The images used are made available online by the visual geometry group from the University of Oxford [104] and contain images of 11 different landmarks of Oxford. A few sample images can be seen in Figure 5.7. The idea behind using the completely different datasets as positive and negative classes was to ascertain the competence of the CNN towards the data and to monitor the performance of the CNN. In this way, CNN will be able to extract the features from both classes. The extracted features are completely different hence it is easier for the CNN to classify these images.

Figure 5.6: A few examples from the category "free images"

For the training of the CNN in this case, 7000 images were selected as a training dataset with each class comprising 3500 images. As a validation dataset, 3100 images were selected and 1550 images belonged to each class. The ratio between the training and validation dataset was approximately $2:1$.

The data augmentation techniques were implemented during the training of the network to generate batches of tensor image data. The data was looped over in batches for training purposes. The data augmentation was used only to process images during the training to expand the scenarios domain while training, not beforehand to expand the size of the dataset. For the image processing during training, few methods or arguments were used, e.g., shear effect, shift effects, zooming and horizontal flips.

Our focus was to find and develop the best architecture. Different combinations of layers were analysed and tested with different dropout parameters and filter sizes to find out the best architecture. After thorough experimentation, an architecture was chosen which was able to work best with the available data as shown in Figure 5.8. This architecture had in total $2,797,665$ trainable parameters.

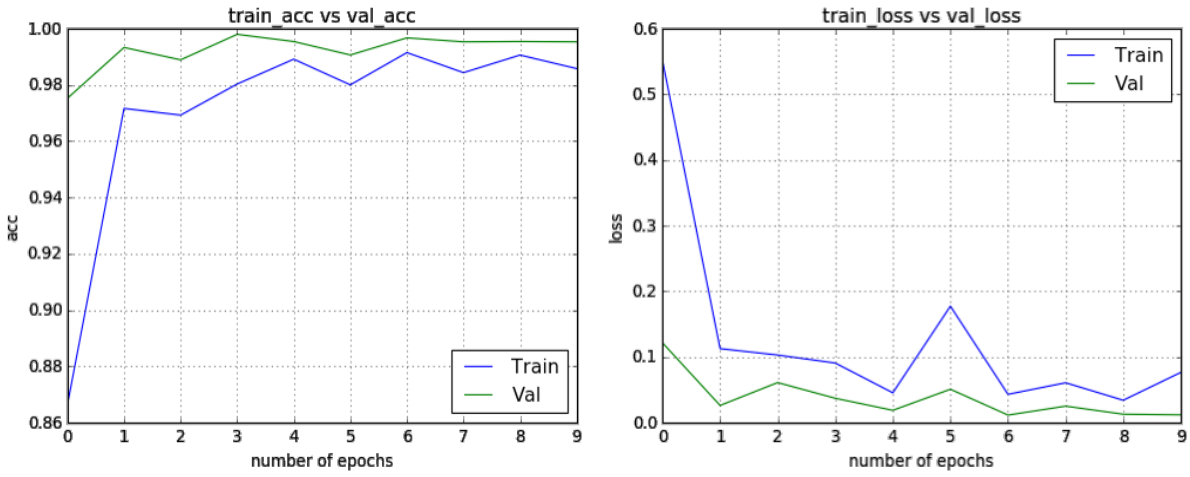
The features from the CNN network also helped in determining the type and amount of data it needed to work efficiently. The data was adjusted accordingly, e.g., to deal with overfitting, one way to remove overfitting from the network is to feed more and more diverse data to the network.

For the final architecture, image size $(227, 227, 3)$ was used. It means the input plane consisted of RGB images. A different number of training epochs were used to find out the best accuracy matrix where the system doesn't underfit or overfit. Finally, it was found out that after training the model for 10 epochs, the model remains stable, i.e., no overfitting or underfitting occurs. We used the batch size of 64.

Once the parameters have been set and learned after training, the accuracy of the model can be determined using the validation accuracy and validation loss. It tells how

Figure 5.7: A few sample images taken from the dataset provided by the University of Oxford [104]

well the model works after each iteration. In our case, we observed a validation accuracy of 0.995 and a validation loss of 0.0133. This indicated that the model was able to train well on the given data. The training and validation results in the simulation can be seen in Figure 5.9. It can also be seen from the graphs that there is no underfitting or overfitting in the model.



(a) Accuracy graph **(b)** Loss graph

Figure 5.9: Accuracy graphs of the first simulation for every epoch

5.1.3.2 Final simulation

After the successful training of the model with the basic image category from the street environment i.e., free images, further scenarios were introduced step-by-step that contained more features to capture from the training's point of view. For example, in the positive class, categories like leaves, obstacles, illumination effects, multiple angles,

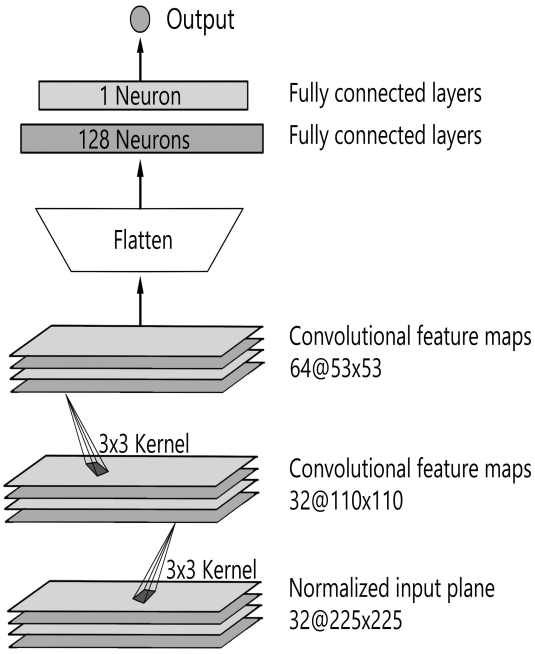


Figure 5.8: Network architecture of the first simulation

parked cars, etc. were added. Similarly, the negative class was also supplemented with different types of pavements and their combinations, illumination effects, leaves, wet pavement etc. With the induction of each new category, the model was trained and then the optimization of the hyper-parameters was done after observing the model's behaviour towards the new features. With the successful integration of each new category, the dataset was also expanded and as a result, the computation time of the algorithm also increased.

For the curb detection using the camera, approximately $150,000$ images from all the categories, following the category matrix mentioned in Chapter 4, Matrix 4.1, were collected. These images were extracted from the filmed curb and pavements and represented the task at hand, i.e. helping the older pedestrians in the road crossing tasks. Out of this sum, $108,000$ images were chosen to train the model. The data compromised of two classes as defined; positive and negative. Each class had a 50% share of the dataset i.e., $54,000$ images. This data was then divided into the training and validation datasets with a ratio of $60 : 40$. For the training dataset, $64,800$ images and for the validation dataset $43,200$ images were used. This dataset contained all the possible scenarios described in the requirement analysis.

In the end, this data was fed to the model to get trained. The image size of $(227, 227, 3)$ was used comprising of RGB channels. The data was augmented in real-time during the training as was done in the first simulation. That means, the batches of images were processed and given to the model.

We treated this problem as a binary classification, hence the best available loss function *binary cross-entropy* [105] was used. It can be represented as:

$$L_p(q) = -\frac{1}{M}\sum_{i=1}^{M} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i)) \tag{5.1}$$

Here, y is the label (1 in the case of the positive class and 0 in the case of the negative class) and $p(y)$ is the predicted probability of the image being positive for all the N images. For each image from positive class ($y = 1$), it adds $log(p(y))$ to the loss function, i.e., the log probability, predicting the road crossing. For every negative class image ($y = 0$), it adds $log(1 - p(y))$, i.e., the log probability of it predicting the no crossing scenario.

For the optimizer, we used RMSprop [106]. It is a gradient-based optimization technique which normalizes the gradient by utilizing a moving average of squared gradients. It results in balancing the step size. It decreases the step size for the large gradients to prevent exploding and avoids vanishing by increasing the step size for the small gradients.

Another method that was used in the training of the CNN was early stopping. One of the major challenges in training a CNN is the choice of the number of epochs. If a model is less trained, there is a chance that the model will underfit and too many epochs can result in an overfitted model. Early stopping is a method which allows the user to assign many arbitrary training epochs. When the training starts and the model's performance stop improving on the validation dataset, then the early stopping aborts the training, no matter how many epochs the model has used for training.

Once the prerequisites and conditions were met, the training was started. The architecture of CNN was chosen empirically. The number of layers and their configuration, kernel size, stride, and batch sizes were optimized through a series of experiments. It was to find out the best possible model and architecture that can yield the highest prediction accuracy. After each training, the system was tested thoroughly in the simulation (using the test dataset, a dataset that the model has never seen before). The real-time environment on the street of Berlin was also tested to find out where the system doesn't predict efficiently. The testing in real-time is important because there could have been quite a few cases in which the model has never learned before. Even a small change in the illumination could have a significant impact on the training. Therefore, it was necessary to test the model in real-time scenarios thoroughly.

The final CNN model, which generated the best results, consisted of five CNN layers. It had a total trainable parameters of $2,656,461$. The stride of 2×2 and kernel size of 3×3 was used throughout the model. Two fully connected layers with varied neuron sizes were added at the end [62]. The complete network architecture can be seen in Figure 5.10.

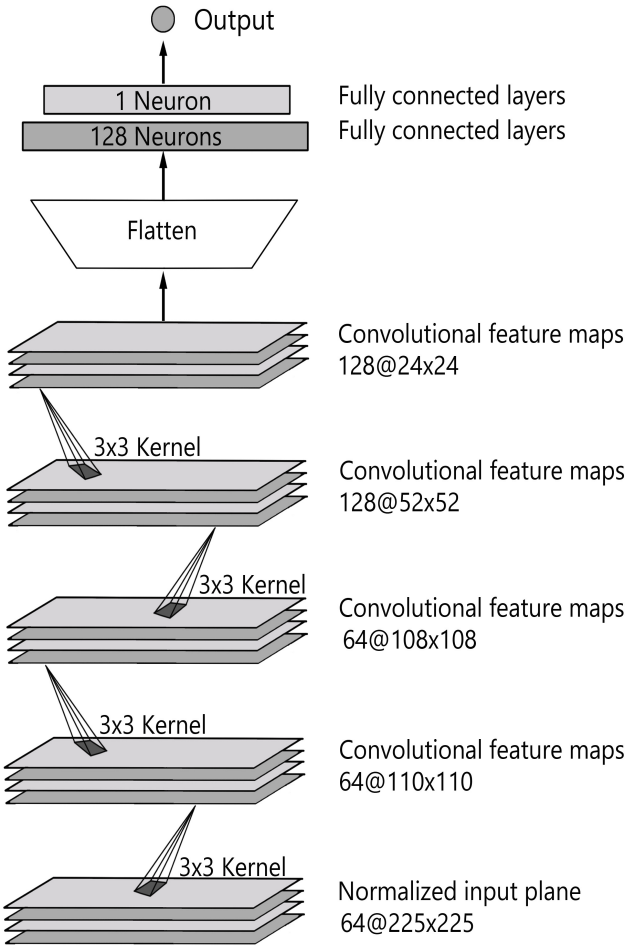


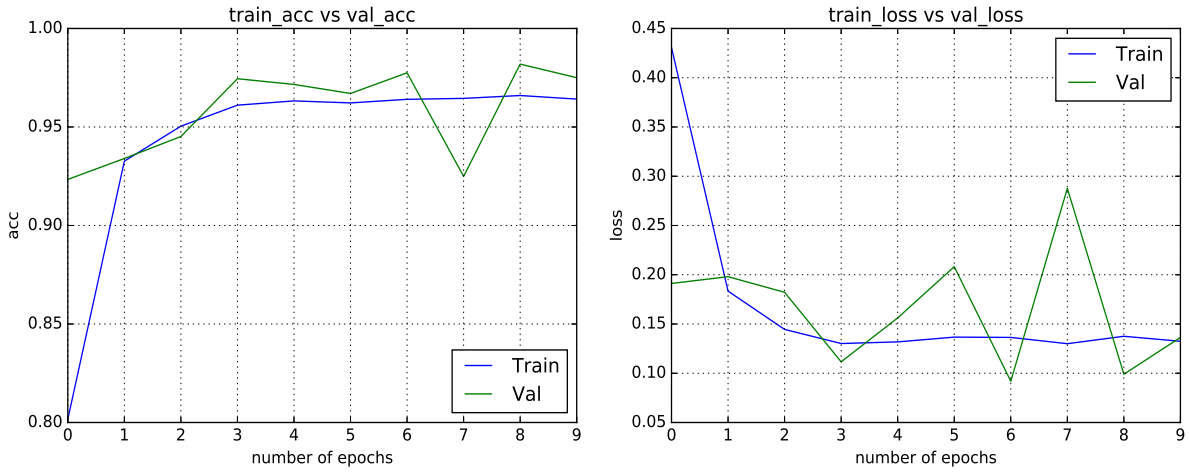
Figure 5.10: Network architecture of the final simulation (taken from [62])

The CNN layers are commonly used to extract features while the fully connected layers on top of the CNN layers are usually considered as the classifiers performing classification tasks. However, we used the end-to-end learning methodology, hence it is hard to differentiate which module acts as the feature extractor and which serves the role of a classifier.

5.2 Evaluation of the model

The model was trained for 10 epochs after that we achieved a validation accuracy of 97.49% and a validation loss of 0.136. The accuracy of more than 97% shows that the system has been able to learn the underlying features of the network efficiently. The training and validation losses of the model for every epoch are shown in Figure 5.11. The graphs also shows that there is no overfitting or underfitting in the model.

In addition to the usual accuracy and loss evaluation, we tested the model with techniques available within the deep learning community. One such way to determine the efficacy of the model is the *confusion matrix*. The confusion matrix, also known as the error matrix, is used to visualize the performance of an algorithm. The accuracy



(a) Accuracy graph **(b)** Loss graph

Figure 5.11: Accuracy graphs of the final simulation for every epoch

is visualized by detecting whether the model can assign the labels to the unseen data correctly or not. Hence, 2400 test samples (1200 from each class) which haven't been used in either training or validation were fed to the model to draw the confusion matrix. The confusion matrix can be seen in Figure 5.12. From the confusion matrix for the positive class, the model was able to predict 1144 times correctly out of 1200 and for the negative class, it was able to predict 1179 correctly out of 1200.

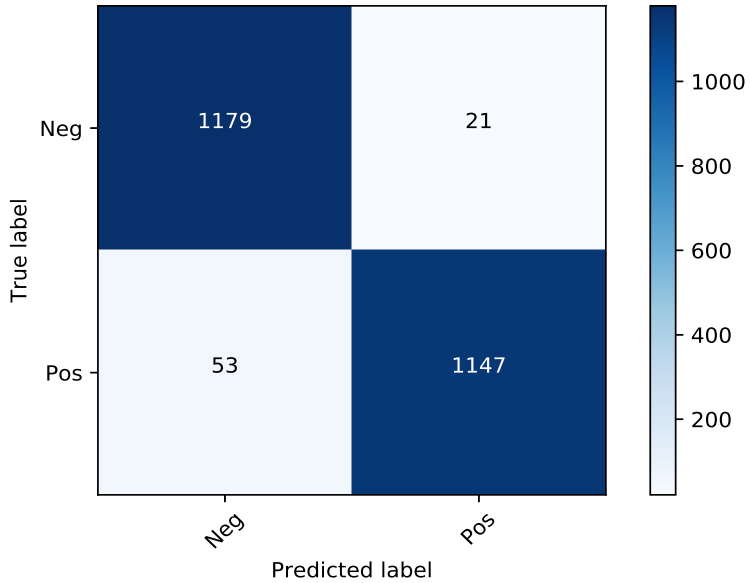


Figure 5.12: Confusion matrix for 2400 test images

On the same test samples used for Confusion matrix, we also evaluated our model for *Precision, recall and F1 score.*

Precision is the ratio of correctly predicted positive samples to the total predicted positive samples and is represented as:

$$Precision = \frac{True\ Positives}{True\ Postives + False\ Positives}$$

Recall, also known as sensitivity, is the ratio of correctly predicted positive samples to all samples in a class

$$Recall = \frac{True\ Positives}{True\ Postives + False\ Negatives}$$

F1 Score is the weighted average of Precision and Recall.

$$F1\ Score = 2 \times \frac{Recall \times Precision}{Recall + Precision}$$

The Precision, Recall and F1 score can be seen in Table 5.1.

	Precision	Recall	F1 score
Positive	0.98	0.96	0.97
Negative	0.96	0.98	0.97
Avg/total	0.97	0.97	0.97

Table 5.1: Precision, Recall and F1 score of the system

This description of the data also makes it possible to analyse the model's performance using Signal Detection Theory (SDT) [107] as it is widely used in the field of Psychology. Using SDT we can distinguish between the detection ability (sensitivity d') and the response bias (criterion c). Sensitivity d' represents the accuracy after taking into consideration the various types of errors and various types of correct decisions. Criterion c states that whether the system is biased towards positives or negatives. SDT parameters can be calculated as follows:

$$d' = z[p(Hit)] - z[p(FA)] \tag{5.2}$$

$$c = -0.5(z[p(Hit)] + z[p(FA)]) \tag{5.3}$$

$$p(Hit) = Hits/(Hits + Misses) \tag{5.4}$$

$$p(FA) = FAs/(FAs + CRs) \tag{5.5}$$

SDT tells us in our case that the model has a very high sensitivity of $d' = 3.81$ and a conservative (high) criterion $c = 0.2$, which means that model is rather susceptible to misses. As described in the requirements section (chapter 2 and section 2.1.2), we favour misses over false alarms. However, a neutral unbiased criterion for the final system is

our aim. Hence, the response bias can be adjusted either by making the improvements in algorithm or hardware, if possible, or by using additional information e.g. by inducing more data.

From the simulation results and the real-time testing of the system, we came to find out that the novel algorithm developed for the camera using end-to-end learning with CNN for curb detection performs efficiently. The benefit of end-to-end learning is that the model provided the data without the hand-crafted rules and it was able to derive useful information from the data itself. Another benefit that was observed that the network was trained with a relatively lesser amount of data and yet it proved to be a reliable solution. The results showed that the system works efficiently with different types of curbs, pavement structure, condition of pavements etc. and is robust against obstacles and leaves.

The system with only the camera works reliably with an accuracy of more than 97%. However, there are a few scenarios where the system is prone to misses and a few cases where the system detects the crossing when there is none in the camera frame, meaning a false alarm. It was also noticed that the system is more prone to false alarms in case of the sunny condition, especially when there are shadows in the frame. The reason can be the change in slight illumination in the camera frame immensely changes the pixel values of the images which in result creates problems for the algorithm while classification resulting in a false alarm. Hence, for the development of an assistance system, to achieve the unbiased criterion, the aim was to reduce all the false alarms as well as all the misses which means to improve the overall accuracy of the model. To achieve this criterion, a distance sensor, LEDDAR, was fused with the camera.

6

Sensor fusion using end-to-end learning

Using only the camera images, trained with the CNN using end-to-end learning algorithm, the system was able to achieve the accuracy of more than 97%. However, the results from the SDT theory and other evaluation techniques used in simulation, it was observed that there were some misses and false alarms generated by the system also in some dangerous situations for older pedestrians. To tackle this problem and to further improve the accuracy of the system, a distance sensor, LEDDAR, was integrated with the camera. This chapter discuss the details about the fusion technique that has been used to integrate both sensors and provides the analysis and the results.

6.1 Methodology

The CNN was initially trained for the camera using end-to-end learning with the RGB images. The image dimensions of $(227, 227, 3)$ were used to train the model. With these dimensions, model was able to extract the useful features and was able to identify the road crossing in an optimal manner.

LEDDAR generates the data in the 2D format with 16 points per frame. As explained in Chapter 4, in LEDDAR section (section 4.4), LEDDAR is being used with the 16 channels, where each channel generates a value in a particular time frame making it an array of $(16, 1)$ for one single time frame. Therefore, because of the heterogeneous input planes comprising images and arrays, the use of the same CNN which was trained for the camera images, is not a viable option. Also, training of a new CNN for the LEDDAR data is not feasible either, because the CNN tends to reduce the dimensions of the input data when the input data propagates through its layers which in turn may results in losing part of the relevant information. This information loss can thus be critical in performing the classification task. Thus, we decided to employ two different training algorithms to process the input data from both sensors.

For the camera images, CNN was chosen again to detect the curb as it gives the best opportunity to explore, extract and learn the underlying features in the images. To cater for the illumination effects (as discussed in the last chapter) in the RGB images, this time it was decided to use the grey-scale values of the pixel. This way the effects of light are comparatively diminished (e.g. shadows, brightness levels, etc.), especially in sunny conditions and this resulted in a better prediction model. The image's input size was also slightly reduced to cater for the longer training time of the CNN. The new input size had a resolution of $(225, 225)$.

Multiple algorithms can be considered as a feasible option to train the LEDDAR data to classify the road crossing scenarios. For example, Random Forests [108], ANNs, etc. We decided to use ANN for the LEDDAR data as ANNs do not tend to lose the dimensionality of the data. Moreover, ANNs are easier to fuse with the CNN and they are also computationally fast. The reason for them being faster in our scenario could also be that the dimensions of the LEDDAR data are comparatively too small.

For the fusion task, both networks were trained in parallel. Once the networks were trained, we extracted the features from each network and concatenated them. These concatenated features were given as input to a fully connected network. Afterwards, the complete network was trained to optimize the hyper-parameters of the network which helped in establishing conformity between the two networks. These fully connected layers also perform the classification and generated output prediction. However, as mentioned in the Camera detection part, in end-to-end learning it is hard to distinguish which part work works as a feature extractor and which part performs the classification [109].

6.2 Training of the algorithm

Once the methodology was defined, the next step was to train the algorithm to find out whether the proposed model can perform the desired task accurately.

6.2.1 Data selection

Before we could start training the data, it was important to select and prepare the data. For the CNN, images were chosen from all the categories mentioned in the category matrix 4.1. When only the CNN was used to perform the classification task with the camera images, the data augmentation was done in real time during the training of the algorithm. This time the images were augmented beforehand. These augmented images were given to the CNN network for training. The data was augmented using artificial shifts, rotations, sheer and zooming effects, etc. The purpose was to present the network with the various conditions that may present themselves in real-life situations e.g. motion blurring, insufficient illumination effects, etc. Moreover, it also helps in

preventing the overfitting of the network. For the training of the network, 40,000 images were chosen. It is a binary classification, so the image dataset was divided into two classes with a 50 : 50 ratio. A few samples from the positive and negative labelled classes can be seen in Figure 6.1 and Figure 6.2 respectively. The dataset was further divided with a ratio of 70 : 30 into training and validation datasets.

Figure 6.1: A few samples after augmentation from positive labelled class

Figure 6.2: A few samples after augmentation from negative labelled class

For the LEDDAR, 16 channels of LEDDAR were used to generate the 2D data, where each channel produce a data point and hence an array of $(16, 1)$ was obtained. LEDDAR data was collected keeping in mind the two classes which we also used for the Camera dataset: *positive* and *negative*. As the array obtained from the LEDDAR channel consisted of 16 values, data augmentation techniques were not used because adding the artificial noise to a single value could have changed the total outlook of

the data. For the training, 40,000 data points from LEDDAR were used. Similarly, the LEDDAR dataset was also divided into classes with a ratio of 50 : 50 and for the training and validation dataset, the dataset was divided with a ratio of 70 : 30.

6.2.2 Network architecture

After the selection and preparation of the data, various experiments were performed to find the suitable architecture that can perform the desired task accurately. The training was started with the simplest of cases of the images and LEDDAR dataset, e.g., free images and free walkway categories without any obstacles, cars, leaves, etc. The complex scenarios were given to the network to monitor the performance and to see how the network adapts to the new data. The complexity in the network was increased by adding multiple scenarios, such as illumination effects, motion blur effects, leaves, obstacles, various weather conditions, angle of approach, type of pavements, etc.

After thorough experimentation, to achieve maximum accuracy, the final network architecture consists of the following attributes:

The final CNN architecture, for the image's dataset, consisted of 5 layers with a stride of size 2×2. The kernel size of 3×3 was used in all the layers. The ANN architecture, for the LEDDAR data, is comprised of 4 layers. Afterwards, the features were extracted and fed to the three fully connected layers. The whole network was then trained on the combined features from CNN and ANN. The complete network architecture can be seen in Figure 6.3.

6.3 Evaluation of the system

After the training, the efficacy of the network was accessed through the overall accuracy and loss values. A validation accuracy of 99.04% and a validation loss of 0.043 was observed. The analysis for each epoch is shown in Figure 6.4.

Similarly, as it was done for the detection with the camera, different evaluation methods were also used to evaluate the performance of the algorithm in simulation. The confusion matrix was drawn for 10,000 samples to find out how the model behaves towards the unseen data samples. The confusion matrix and can be seen in Figure 6.5.

The Precision, Recall, and F1 scores were also evaluated for the fusion algorithm and can be seen in Table 6.1.

	Precision	Recall	F1 score
Positive	1.00	0.99	0.99
Negative	0.99	1.00	0.99
Avg/total	0.99	0.99	0.99

Table 6.1: Precision, Recall and F1 score of the system

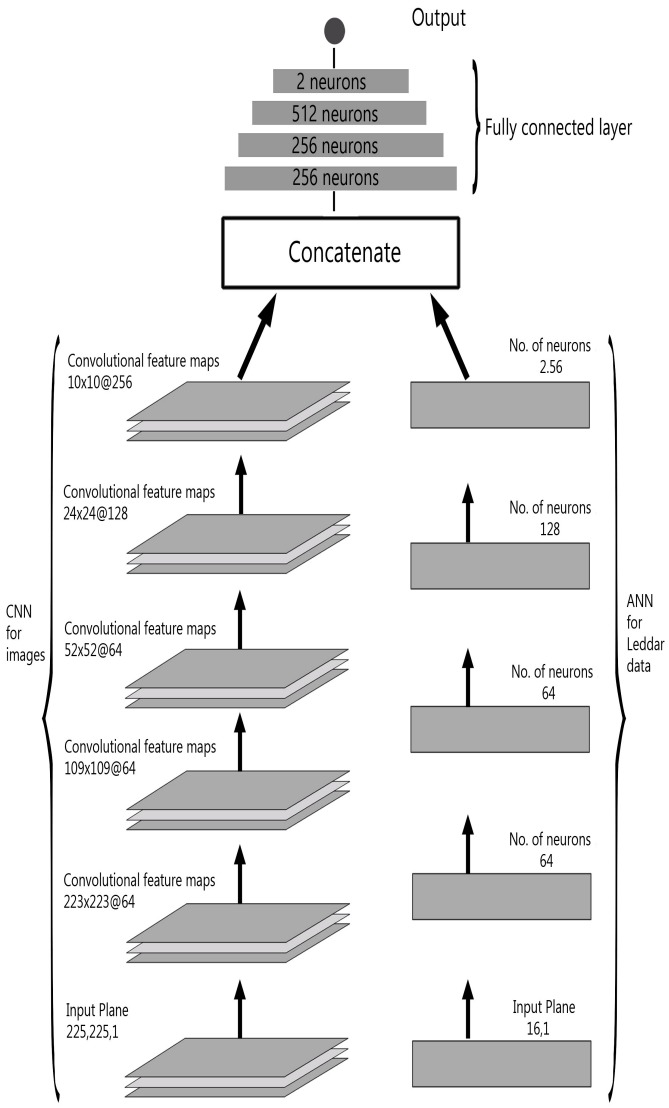
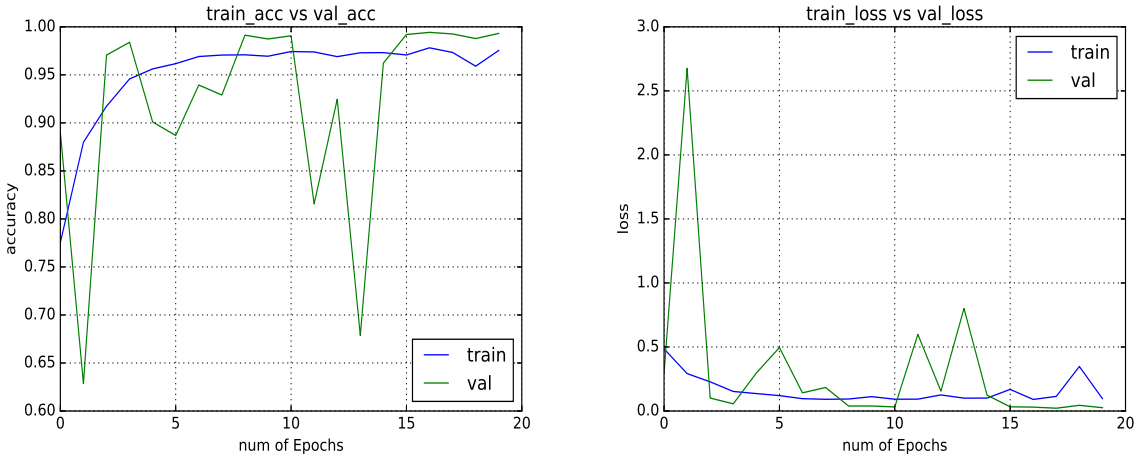


Figure 6.3: Network architecture for the sensor fusion

For real-time testing, we tested the algorithm on the streets of Berlin. For this purpose, the prototype, mentioned in Chapter 4.2, was used. The algorithm was running on the Lenovo Y720-15IK notebook, which has NVIDIA GTX 1050 GPU. The aim was to test and evaluate the system on the streets of Berlin in real-time. The system was tested in various conditions. The real-time results show that after the fusion of both sensors the accuracy has been improved. The system was able to detect the curb in various scenarios. However, as it is a probabilistic system, there will always be a probability for the system to miss some events or generate a false alarm.

From the real-time testing, we also found out that the detection speed is impressive. The detection rate is 22 Frames Per Second (FPS). This means that the algorithm can process and perform the prediction 22 times per second.



(a) Accuracy graph **(b)** Loss graph

Figure 6.4: Accuracy and loss graphs of the final simulation

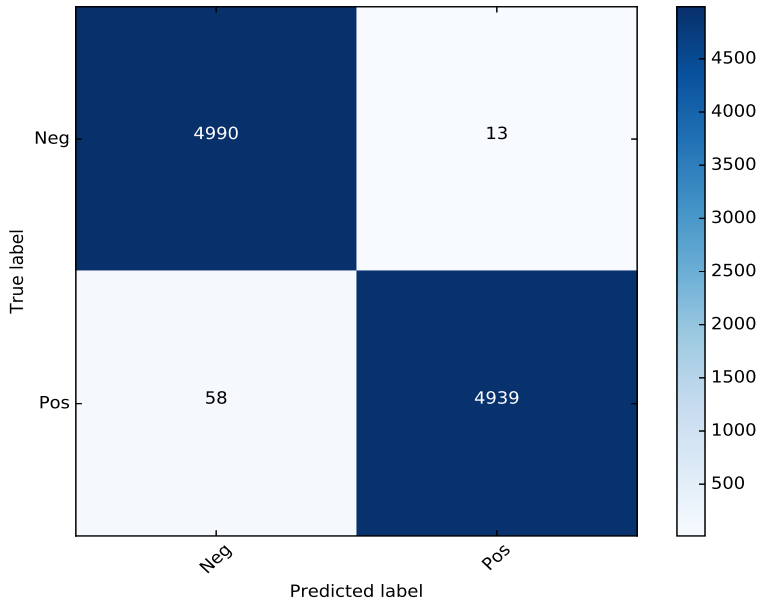


Figure 6.5: Confusion matrix for the test images (taken from [109])

7

Actuators and field test

To test the system in a real-time environment with the relevant audience, i.e., older pedestrians, a field test was performed. However, before we could evaluate the system in real-time with the elderly, an interface had to be developed between the system and the user. For this purpose, an alarm system had to be designed for the cases when the system detects the curb. For example, generating a warning to the user, prompting them to stop and focus on the oncoming traffic. This chapter focuses on the field test and the user interface was developed to carry out the field test.

7.1 Selection and evaluation of actuators

To find the best medium to communicate with the older pedestrians, the state of the art methods and interfaces had to be researched. Moreover, it was also important to research that how perception works in older people and what kind of interface modalities are available and applicable in terms of older people.

In humans, the sensory perceptions deteriorate with the age. It also depends strongly on individual differences and various lifestyles. Furthermore, not all sensory perceptions show an equally strong decline. While visual and auditory perceptions, for example, decline sharply, however it is not the case with the sense of smell which declines at later stages of life. However, when it comes to tactile perception, there have been far fewer findings so far, since it often plays a rather subordinate role in our everyday life. However, recent studies suggest that this little-used modality may offer advantages when used, especially for older people [110]. Moreover, in contrast to visual and acoustic modalities, it represents an immediate form of communication that users can neither miss nor overhear and that runs a little risk of being overlaid by other environmental information of the same type. The intuitive response to tactile stimulation is a turning of attention. If it is possible to use this attention to unsafe environmental conditions,

an assistance system with a tactile interface represents reliable support with a simple form of communication at the same time.

To ensure a broad gain of knowledge and a high degree of innovation in the project, two different tactile interfaces i.e., vibration and temperature stimuli, were empirically evaluated against each other and an auditory interface by the Human Factor experts in the group [111]. One interface uses vibration as a communication signal. This is perceived via vibro-receptors located on the skin. The other interface uses temperature stimuli, which are perceived by human thermoreceptors. This modality has not yet been used in practical applications, as there has been quite limited research on it. However, initial studies demonstrate the potential of this type of information transfer [112].

A few factors had to be taken into account to evaluate the interface modality for the pedestrian assistance. To evaluate the interface modality for the pedestrian assistance system regarding the performance in hazard detection, the trust and acceptance of the users as well as their attention distribution when using the assistance system, a laboratory study was carried out with the target audience in the research group by the Human factors experts [23, 29, 30, 66, 113]. For this purpose, 27 elderly test persons, aged 65+ were examined. In the course of the four-hour survey, they were asked to complete a hazard detection task, each accompanied by a visual or a cognitive secondary task. The participants used three assistance systems one after the other in a different order, each with a different interface modality (vibrotactile, thermo-tactile, acoustic). In a baseline measurement, they completed the tasks without an assistance system for comparison. The response times and errors as well as the acceptance of the system by the users and the proposed workload were collected. Various types of errors in the assistance system were simulated to investigate misconduct and trustworthiness. After the evaluation, it was decided to use the vibrotactile sensors as they proved to be the best medium to convey the information generated by the system to the participants.

7.2 Development of the system and user interface

After the selection of the interface modality, the next step was to develop the modality and link it with the prediction algorithm. For this purpose, multiple components were used. To generate the vibrations, flat button vibration motors with a diameter of $12mm$ were used. It is an Eccentric Rotating Mass (ERM) motor. Evaluated with 3 VDC, it offers a solution with low energy consumption and low noise level. The vibration motors are implemented in two cuffs which were designed for this specific purpose. When approaching the road, they generated vibrotactile signals that are intended to make the users stop and focus their attention on the road traffic.

The vibration motor was enclosed in a band made of elastic cloth to avoid direct contact with the user's body. The band containing the vibration module was provided to the user to wear on their biceps. The biceps were chosen, because when the alarm is

generated and the user feels the vibration on their biceps, they will not focus towards the source of vibration as the lateral distance between source and eyes are relatively small, hence it will not create a hindrance in focusing their attention towards the traffic.On the contrary, if we had asked the user to wear the device on the arms, e.g., wrists, in such case when they receive the alarm, they tend to look at the wrist rather than the street traffic as it has some lateral distance from their eyes. Therefore, we chose to put the vibration motors on the upper arms of the users.

These motors were controlled using the Arduino Uno WiFi. The intensity and duration of vibration were set beforehand so that it doesn't irritate the user and they feel comfortable with the alarm. The Arduino was connected using a wireless interface with the Notebook and it was controlled within the algorithm. The algorithm detects the curb stone and generates the alarm signal. The alarm signal is then relayed to the Arduino which turns the vibration motor on to alarm the users.

Another important aspect that had to be analysed is if the user stops after receiving the alarm and rotates their head to analyse the oncoming traffic. For this purpose, an extra sensor Inertial Measurement Unit (IMU) was chosen to record the head movement, velocity, acceleration, etc. The IMU used for this purpose was from the company "Adafruit" and the model was "BNO055". The advantage of this model is that it contains a high-speed ARM Cortex-M0-based processor which takes the values from the accelerometer, gyroscope and magnetometer and does the sensor fusion to provide accurate absolute orientation and acceleration among others.

One IMU was installed on the walker itself facing forward. The accelerometer with the 3-axis was used to determine the direction of the IMU. This IMU, mounted on the walker, was used as a reference point, as it is always facing forward. Another IMU was mounted on a helmet which the participants were asked to wear. Hence, if the person rotated his head, the difference in both IMU orientations will detect the movement of the head. These IMUs were also controlled by the same Arduino which controlled the vibration motors. In this way, all the sensors were synchronized with the same time stamp. Therefore, it can be then determined when the person received the alarm, then either he or she stopped and moved his head to analyze the oncoming traffic.

Another camera was installed on the helmet to track the motion of the head and to record the surroundings from the user's point of view in the form of a video. The camera used for this purpose was a GoPro 7 black edition. A GoPro is a durable and portable camera that is used to record the surroundings in the form of a video. It can also be used to take pictures. GoPro can also record the location data (i.e., GPS data). In this work, GoPro was only used to record the video and the location of the participant. Therefore, if a person moves his head the camera will also be moved with the head rotation, hence it can be analysed if the person stopped at a crossing and analysed the road traffic. The video recording from GoPro was also used to analyse the users' frequency and duration of the standing still. As GoPro is a standalone camera with an

internal clock, therefore the time cannot be synchronized with the time of an Arduino. A complete setup of the system with the user can be seen in Figure. 7.1.

Figure 7.1: An elderly person with the walker

Multiple simultaneous predictions were used to generate the alarm to increase the accuracy and certainty of the prediction. 15 frames were used in total to generate the alarm, i.e., if the system detected the curb in 15 consecutive frames, the alarm will be generated. As the system's processing speed lies around 18 fps that means about 1 sec of predictions were used to generate the alarm. It was also helpful in reducing the number of alarms generated. Furthermore, to reduce the number of received alarms, an alarm was generated after every five seconds. It is done because if the user has stopped and waiting on the road to get cleared, the user must not receive too many alarms while just standing and waiting at the road crossing. It is done to prevent the user from getting annoyed by too many alarms. The drawback was that it could be possible that the system generated a false alarm and soon after there is a crossing in the frame and it detected this crossing, however it didn't generate the alarm as 5 seconds waiting window hasn't passed. Also, the other way around, after the detection of the curb, it detected an object which was in the detection window and is relevant for our use case like a car or a person in front however it didn't generate the alarm because the system was in the 5 seconds waiting for the window but in this field study our focus was only to detect the curb and to generate alarm and to observe how older pedestrian reacts to the alarms hence it doesn't have too many repercussions.

7.3 Field test with the target group

After the development of the test setup to test the functionality of the system in real-time, a field test was organized. The main research questions considering the detection algorithm were:

- Does the system reliably detect the road crossing?
- Does the system accurately generate the alarms when it detects a crossing?

A few more perspectives were also analysed in the field test to answer the further questions posed in this study. For example, the attention of the user towards the road traffic with the assistance system, the data for this was collected using IMUs by calculating the movement of the head for gaze frequency and its duration. Another important perspective analysed was to see if the system helps in reducing the multi-tasking effect of the user so that they can focus on the oncoming traffic. This was detected using a GoPro camera by measuring the frequency and duration of standing still. Subjective data such as experience and acceptance of the assistance system were assessed via the questionnaires specially designed for this study.

This study was approved by the "Ethik-Kommission des Instituts für Psychologie und Arbeitswissenschaft (IPA) der TU Berlin" under the name "Studie zur Nutzung eines Fußgängerassistenzsystems im 310 Straßenverkehr" (serial numbers BRE_02_201808803). The field test was conducted with 26 older people between the age of 65 and 85. The average age was 73.15 and the standard deviation of the age was 5.38. The participants were chosen with a gender distribution of 13 males and 12 females and one preferred to stay anonymous.

The route was chosen in such a way that the systems' accuracy can be tested variously. This route is situated in Berlin and consists of multiple streets. It consists of multiple road crossings. For the field test, 13 road crossings were selected which included both official and unofficial road crossings. There were 7 official crossings and 6 unofficial crossings. The complete route with the marked road crossings can be seen in Figures 7.2 and 7.3:

Figures 7.2 and 7.3 also show some of the crossings as sub-images. These images show that different types of crossings were considered to test the system's accuracy. This route contained crossings with sunken curb, which is also considered official crossing, and elevated curb, which is supposed to be unofficial crossing. The route also contained different monuments, such as trees, poles (as can be seen in Figure 7.2), parked cars, main holes, etc.

The field test consisted of one complete round trip of the route. One half of the trip (either moving towards or coming back) was approximately 500m long. The complete round trip was approximately 1 km long. Each half of the trip consisted of 13 road

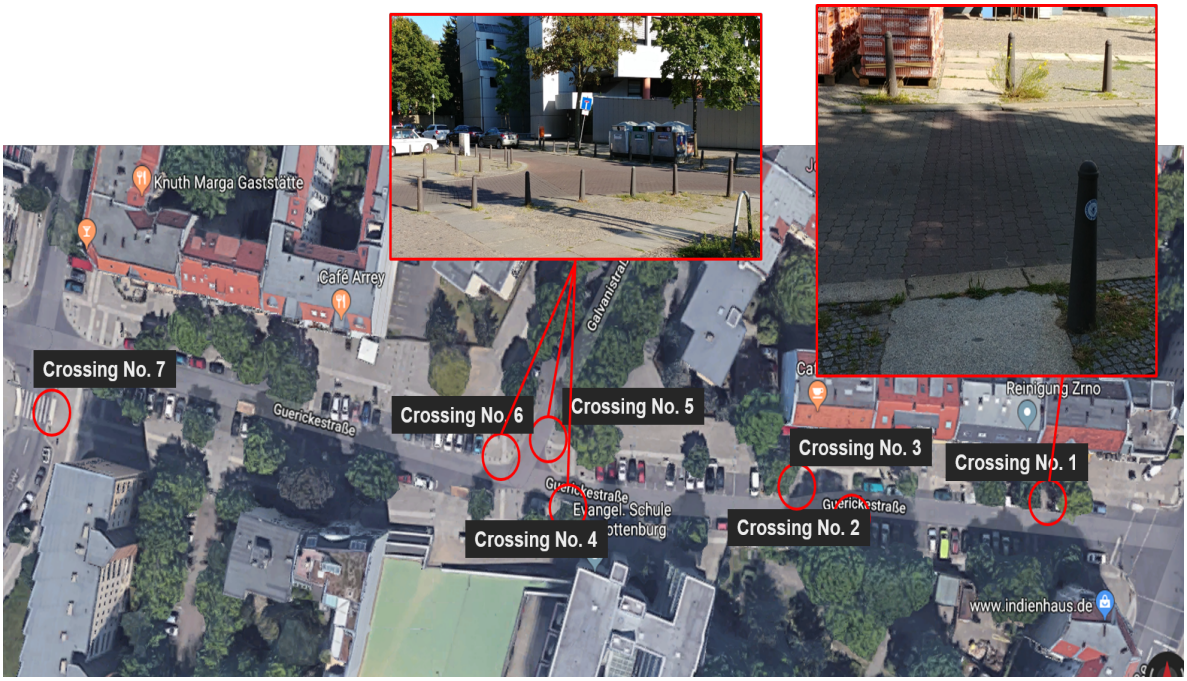


Figure 7.2: Figure depicting the first half of the route with the official crossings

crossings. On one half of the round trip, the system with the alarm modalities was turned on i.e., whenever the system detected a curb in the frame, it generated the alarm. However, on the second leg of the trip, the alarm modality was switched off i.e., the system didn't generate the alarm whenever it detected the curb. It was done to observe how the participants behave in the road crossing scenarios with or without the alarm.

For the data analysation, there are a few important factors are important to discuss. In the analysis, only the data has been presented which shows that if the system can detect road crossings or not in real-life situations. The system was able to detect the objects such as cars, pedestrians, poles, trees, etc. but they are not the scope of this study. It is because there is no ground-truth data available with which it can be compared how many of such objects were available to detect and how many of these available objects the system detected. For example, it can be analysed how many road crossings the system was able to detect out of the pre-defined crossings, but the rest of the elements either pedestrians or cars are the variable factors so there is no ground truth value available, hence the results are not presented here.

Another important aspect is that the detection itself was impacted by a lot of factors, e.g., weather conditions, and the walking speed of the older people (sometimes they walked fast, and the system detected the street but couldn't gather enough positive samples to generate the alarm). Sometimes at the crossings when there were other people, the system detected them as an obstacle and generated the alarm. It also then detected the curb but didn't generate an alarm as it was designed in a way that it shouldn't generate alarms one after another too soon. The detection rate of overall samples was also different in almost every case, it could be because of the hardware used in the field study. The detection rate also depends upon the laptop's condition

Figure 7.3: Figure depicting the second half of the route with the unofficial crossings

(e.g., hardware, heating, etc.) other than the programming itself. Moreover, as it was a test of the real system, the algorithm was tweaked during the field test to tackle the problems which could have only been seen in real-life scenarios but can't be observed in the simulations. For example, the amount of the positive number of detections to generate the alarm, the selection of the waiting window etc., factors also have an impact on the overall detection.

The field test was conducted in the months of November and December of 2018 and took almost 35 days to complete. The data was then analysed for the assistance system. The preliminary results concerning the detection algorithm can be seen in Table 7.1 and Table 7.2.

The first column in each table represents the Participants. The thirty participants included the trial participants' results have also been listed here. The second column represents the route and it also contains two elements which represent each half of the trip. One is going towards the endpoint with the assistance system turned on which means with the support of the assistance system and another one represents the second half of the trip i.e., coming back to the starting point with the assistance system turned off. The third column shows the number of total samples the algorithm processed in each half of the trip. This number also depends upon the speed of the user, if the person walks slow or takes small steps then this number would be higher and vice versa. The fourth column shows the number of positives i.e., the number of times the system detected the curb and generated the alarm. The fifth column shows the False positives, i.e., the number of times the system detected the curb, but it was not present in the camera frame. The sixth column depicts the False Negative, i.e., the system was not able to detect the curb in the images.

The preliminary evaluation shows that, in most cases, the system was able to generate the alarm whenever it detected the curb stone but there were a few misses in some cases.

Participant No.	Route	Total Detections	Positives	False Positives	False Negatives
1	A	18080	10	10	3
	B	15286	9	8	4
2	A	18925	12	9	1
	B	17328	10	10	3
3	A	17690	12	10	1
	B	14879	10	11	3
4	A	21242	13	11	0
	B	17606	11	10	2
5	A	19430	11	8	2
	B	14891	12	5	1
6	A	16626	10	5	3
	B	15371	12	6	1
7	A	15990	12	7	1
	B	13139	11	8	2
8	A	18247	9	4	4
	B	16649	9	3	4
9	A	17794	12	8	1
	B	18693	12	9	1
10	A	17741	13	10	0
	B	18150	13	11	0
11	A	20367	13	9	0
	B	19409	12	8	1
12	A	16368	13	10	0
	B	14823	13	11	0
13	A	22606	13	11	0
	B	22594	12	12	1
14	A	22929	13	12	0
	B	24715	11	8	2
15	A	19703	12	8	1
	B	20068	12	9	1

Table 7.1: First part of the table shows the preliminary results of the field test

Participant No.	Route	Total Detections	Positives	False Positives	False Negatives
16	A	18157	12	10	1
	B	14243	11	7	2
17	A	18701	13	13	0
	B	17264	13	13	0
18	A	23377	13	12	0
	B	21002	13	13	0
19	A	15301	12	11	1
	B	13975	13	9	0
20	A	19518	11	8	2
	B	18042	12	8	1
21	A	15892	11	8	2
	B	15764	10	9	3
22	A	18839	13	7	0
	B	19003	11	7	2
23	A	14749	10	6	3
	B	14487	11	5	2
24	A	24501	12	8	1
	B	24796	12	10	1
25	A	21929	13	11	0
	B	20375	12	10	1
26	A	14948	11	5	2
	B	13097	12	6	1
27	A	16518	12	7	1
	B	18482	13	8	0
28	A	18937	11	6	2
	B	17102	12	4	1
29	A	28828	12	8	1
	B	13406	13	12	0
30	A	18585	12	6	1
	B	16308	12	5	1

Table 7.2: Second part of the table shows the preliminary results of the field test

The results can be evaluated using the Signal Detection Theory [107] which was one of the criteria that was used to evaluate the results in simulation. SDT tells us the hit rate of the system which describes how good the system is, in detecting the targets. The hit rate of the field test ranges from 0.769 to 1 with M = 0.922 and SD = 0.068. It is comparatively lower than the hit rate achieved in the simulation i.e. 0.988 [109]. There could be a few reasons for this. The fourth column in the tables represents the value where the system generated the alarm, however from the evaluation of the data collected from the field it was observed that the system was able to detect every curb but didn't generate the alarm each time. For example, the system was designed in such a way that it should generate an alarm after considering multiple positive detections, 15 to be precise, to increase the accuracy. That means the system considered the 15 positive sequences to generate the alarm. It is possible that not every time system was able to detect the curb continuously for 15 times, hence it didn't generate the alarm. One of the other reasons could be that system was designed in such a way that it should not generate the alarm again within 5 seconds once it has generated the alarm to avoid the redundancy of the alarms which could prove to be a nuisance for a user and then could lose interest in the warnings generated from the system. Therefore, for example, near the road crossing, the system detected a standing person and generated an alarm. But soon after it detected the crossing it didn't generate the alarm as it had to wait 5 seconds. The algorithm was trained using end-to-end learning therefore it only detected the hazardous situation. Hence it doesn't provide us with a way to turn off the detection of the objects which were not relevant for this study, e.g., persons, trees, cars etc. However, it can be observed that the parameters like the waiting window or the number of predictions to generate the alarm can be tuned in such a way that system reacts better to the situation. Moreover, this study spanned over 35 days, it also included various weather conditions which also implied different lighting conditions which has a direct impact on the detection. The fifth column shows the number of False Positives which are also high in some cases. The same reasoning, discussed above, can also be applied to the number of False Positives detected from the system. In addition to that, it is quite possible that in the real-time environment, quite a few objects were not part of the training algorithm. They have prompted the system to generate an alarm which leads to more false positive alarms. Since it was not possible to calculate the false alarm rate as the underlying number of events is not known, true negatives and false positives could not be thoroughly analysed.

The psychological factors were also analysed by the Human factor specialist of the FANS group. One of the aims of this field test was to check if the prototype helped older pedestrians to increase safety in their behaviour. The system was supposed to increase the head rotation to look for the oncoming traffic and to increase the stopping frequency to minimize multitasking. In addition to this workload and acceptance of the system were also analysed. Regarding head rotation, it was observed that the users

significantly rotated their heads to observe the environment when the system was turned on which is a major success. Considering the workload, it was encouraging to observe that the participants did not feel an extra workload with the assistance system which could be a problem if it imposed an additional workload which hinders the acceptance of the system. A more detailed analysis regarding the psychological aspects of the field test can be found elsewhere [114].

8

Summary

This chapter concludes with the assistance system that has been presented in this thesis to support older pedestrians in Berlin. Moreover, it also discusses some future work and possible applications of the system in the various application fields.

8.1 Conclusion

Outdoor mobility plays an important role in human lives. It also represents a way to interact with society. With the passing age, mobility as a form of walking is a preferred way for older people to do their usual daily tasks. The statistics suggest that older pedestrians are involved in 20% of road crashes. And because of their frail physique, they suffer the most fatalities. The reason is that they don't pay the required attention to the traffic. To help older pedestrians crossing the road, in this thesis, a prototype has been proposed that can warn pedestrians before they cross the road. The system aimed to generate an alarm when the user is in the vicinity of the street so that the user can focus his attention on the traffic. This assistance system fulfils the needs and requirements of older pedestrians such as low weight, cost-friendly, easily integrable, etc.

Based on the requirement analysis, the sensors used to develop this system should be inexpensive and lightweight. Accordingly, a novel approach has been introduced to develop the assistance system. For this purpose, two sensors, LEDDAR and a camera, in combination with Multi-Sensor Data Fusion (MSDF) techniques are used to detect the curb in the road environment. This cheaper and more efficient technology is, therefore, better suited for the target group of older pedestrians as it fulfils the requirements.

This work started with the detection of the curb using the camera only. In the beginning, the traditional route uses image processing techniques and afterwards, the more advanced techniques like object recognition were researched. Finally, an advanced approach of deep learning with end-to-end learning was chosen for its suitability to the

task at hand. A Convolutional Neural Network was designed empirically to detect the curb using the camera data.

To train the network on the camera images, a dataset had to be built from the scratch. For this dataset, an analysis of the streets of Berlin was made to find out the physical features of the curb and its relevant pavement. Moreover, this dataset covers the entirety of the street environment and considers the various scenarios such as pavement structure, types of curbs, weather, obstacles, different pedestrian approaching angles, etc. This dataset now contains 130,000 images. The dataset itself represents a great value in the computer vision community and can be used in many other applications, including autonomous vehicles, pedestrian robots, and the walker industry.

In addition, to the creation of a new dataset, a novel algorithm was designed using CNN with end-to-end learning to detect the curb in the image frame. CNN was designed and trained empirically on a binary dataset. CNN trained using end-to-end learning requires less effort and thus has less human interference. It allows modelling the freedom to select and choose the features itself. This turned out to be a better solution as the model required a comparatively less amount of data. The results show that the trained CNN is very well suited for the task at hand. It was able to detect the curb with an accuracy of more than 97%. It was shown that the system could process and categorize new input data very well.

To improve the accuracy and to take into account the few errors of the camera such as the illumination effects, etc., the LEDDAR sensor was integrated into the camera. Both sensors, camera and LEDDAR, generate the data in different formats. Therefore, one of the biggest hurdles was to find an algorithm that processes both data streams efficiently. Instead of conventional techniques, like Kalman filters, a new approach was implemented, where end-to-end learning is used with CNN for the camera data and ANN for the LEDDAR data. With this innovative approach, the system finds and learns the underlying data representation and fuses the sensors to detect the curb in an efficient way with minimal user intervention, making the system invulnerable to human design flaws. Sensor fusion using this method increased the accuracy of curb detection to 99%. Both sensors were able to support each other. The model was able to detect the curb robustly and efficiently. This capability was derived from the dataset and algorithms themselves without the need for specific hand-crafted rules.

From this thesis, it can be concluded that the new approach described, the fusion of Convolutional Neural Networks and Artificial Neural Networks with end-to-end learning can be used to merge the two sensors with heterogeneous data streams. Both the created dataset and the developed algorithm represent novelties in the community. This is not least due to the use of the end-to-end learning method, which is still at the beginning of its promising career in environment detection.

8.2 Future work and application

Future work can focus on expanding the current dataset, mainly the Camera dataset. Moreover, it includes optimizing the system performance and computational requirements.

The camera dataset currently consists of 130,000 images and covers a wide range of scenarios. This dataset represents a considerable amount of value and will be made available to the computer vision community. This dataset can be expanded to cover more scenarios, e.g., the shadows created by the sun or the shelters, etc.

The next step includes the enhancement of the proposed CNN architecture by designing it deeper to achieve even higher accuracy for curb detection in multiple scenarios. More classes can be added as per requirements to detect the specific object in the images e.g., other pedestrians, bicycles, trees, cars, etc. to name a few. The algorithm's computational performance can be improved using the cascades of GPUs instead of a notebook with a GPU. This will provide further system resources allowing for a deeper design and implementation of complex models. This can also benefit in training higher resolution images which can help to increase the accuracy of the overall system. Moreover, the proposed algorithm can be compared using the already existing state-of-the-art algorithms using the same image dataset. A comparison analysis can be made to compare the efficiency and reliability of the system. The famous available ConvNets are ZF Net [100], VGG Net [101], GoogLeNet [102], Microsoft ResNets [103] and so on.

Moreover, object detection techniques can be used to classify the objects in the images. As such the images need to be labelled to specify the objects or features in the images and will be outside the domain of end-to-end learning. However, it will be interesting to compare the results of end-to-end learning with the object detection techniques. The famous object detection algorithms are Faster R-CNN [115], You Only Look Once (YOLO) [116], and Single Shot multibox Detector (SSD) [117].

In the context of the application, this assistance holds substantial importance. A workshop was carried out at the end of this project with the potential market partners to find out the possible application domain. A huge enthusiasm was shown by the participants in this workshop. The demonstrations were also given to the companies from various sectors such as walker companies, automotive companies, software companies and the health sector. Great interest was shown from the possible industry partners. The walker companies, already working to support the older pedestrians, are willing to integrate the system with a few modifications keeping in mind the walker specifications. The automotive industry showed great interest in the curb detection and fusion algorithm. The software companies were interested in developing it as a mobile application with some more features such as heart rate monitor or blood pressure monitor etc. The health

sector showed interest in the system as it helps their customer, which also includes older pedestrians, in everyday scenarios.

References

[1] Florian Breitinger and Rebecca Wiczorek. “Außerhäusliche Mobilität älterer Menschen als Voraussetzung für ein selbstbestimmtes Leben: ein technisches Assistenzsystem zur Unterstützung der Verkehrssicherheit”. In: *IT für soziale Inklusion*. Ed. by Aljoscha Burchardt and Hans Uszkoreit. De Gruyter Oldenbourg, 2018, pp. 121–140. DOI: 10.1515/9783110561371-012.

[2] Michael Rytz. “Senioren und Verkehrssicherheit”. In: *VCS Verkehrs-Club der Schweiz, Bern* (2006).

[3] Senatsverwaltung für Stadtentwicklung. *Ausführungsvorschriften zu § 7 des Berliner Straßengesetzes über Geh- und Radwege (AV Geh- und Radwege)*. 2010. URL: https://www.berlin.de/senuvk/service/gesetzestexte/de/download/bautechnik/AV_Geh-und_Radwege.pdf (visited on).

[4] Michael Whittle. *Gait analysis: An introduction*. 4th ed. Edinburgh and New York: Butterworth-Heinemann, 2007. ISBN: 9780750688833.

[5] Rudolph Emil Kalman et al. “A new approach to linear filtering and prediction problems”. In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45.

[6] Stefan Pohlmann, Christian Leopold, and Paula Heinecker. “Richtungsentscheidungen für Jung und Alt”. In: *Altern mit Zukunft*. Ed. by Stefan Pohlmann. Wiesbaden: VS Verlag für Sozialwissenschaften, 2012, pp. 19–40. ISBN: 978-3-531-19418-9. DOI: 10.1007/978-3-531-19418-9_1.

[7] W. D. Oswald. “Gerontopsychologie - Gegenstand, Perspektiven und Probleme”. In: *Gerontopsychologie*. Ed. by Wolf D. Oswald, Gerald Gatterer, and Ulrich M. Fleischmann. Wien: Springer, 2008, pp. 1–12. ISBN: 978-3-211-75685-0. DOI: 10.1007/978-3-211-78390-0_1.

[8] Heinz-Hermann Krüger and Winfried Marotzki, eds. *Handbuch erziehungswissenschaftliche Biographieforschung*. 2., überarbeitete und aktualisierte Auflage. EBL-Schweitzer. Wiesbaden: VS Verlag für Sozialwissenschaften, 2006. ISBN: 9783531900100. URL: http://swb.eblib.com/patron/FullRecord.aspx?p=749897.

[9] Kirsten Aner, ed. *Handbuch Soziale Arbeit und Alter*. 1. Aufl. Wiesbaden: VS Verlag für Sozialwissenschaften, 2010. ISBN: 9783531155609.

[10] Thomas Haustein et al. *Older People in Germany and the EU*. Wiesbaden, Germany, 2016. URL: https://www.bmfsfj.de/blob/113952/83dbe067b083c7e8475309a88da89721/aeltere-menschen-in-deutschland-und-in-der-eu-englisch-data.pdf (visited on).

[11] Maria Limbourg and Stefan Matern. *Erleben, Verhalten und Sicherheit älterer Menschen im Strassenverkehr: Eine qualitative und quantitative Untersuchung (MOBIAL)*. Vol. 04. Mobilität und Alter. Köln: TÜV Media, 2009. ISBN: 3824912619.

[12] Rebekka Oostendorp. "Aktiv im Alter in der Stadt". In: *Standort* 34.2 (2010), pp. 62–67. ISSN: 1432-220X. DOI: 10.1007/s00548-010-0137-x.

[13] Irene H. Yen and Lynda A. Anderson. "Built Environment and Mobility of Older Adults: Important Policy and Practice Efforts". In: *Journal of the American Geriatrics Society* 60.5 (2012), pp. 951–956. ISSN: 1532-5415. DOI: 10.1111/j.1532-5415.2012.03949.x. URL: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1532-5415.2012.03949.x.

[14] Janice B. McPhillips et al. "Exercise Patterns in a Population of Older Adults". In: *American Journal of Preventive Medicine* 5.2 (1989), pp. 65–72. ISSN: 0749-3797. DOI: 10.1016/S0749-3797(18)31107-3. URL: http://www.sciencedirect.com/science/article/pii/S0749379718311073.

[15] Fox, Kenneth R. and Stathi, Afroditi and McKenna, Jim and Davis, Mark G. "Physical activity and mental well-being in older people participating in the Better Ageing Project". In: *European Journal of Applied Physiology* 100.5 (2007), pp. 591–602. ISSN: 1439-6327. DOI: 10.1007/s00421-007-0392-0.

[16] Arthur F. Kramer and Kirk I. Erickson. "Effects of physical activity on cognition, well-being, and brain: Human interventions". In: *Alzheimer's & Dementia* 3.2, Supplement (2007), S45–S51. ISSN: 1552-5260. DOI: 10.1016/j.jalz.2007.01.008. URL: http://www.sciencedirect.com/science/article/pii/S1552526007000040.

[17] Barbara Lenz et al. *Mobilität in Deutschland 2008*. Ed. by Bundesministerium für Verkehr, Bau und Stadtentwicklung. 2010. URL: https://elib.dlr.de/68010/.

[18] Florian Breitinger et al. "Design of a template for a scenario-based evaluation as part of the user-centered development of an assistance system for older pedestrians". In: *Tagungsband der 11. Berliner Werkstatt Mensch-Maschine-Systeme*. Berlin, 2015, pp. 314–317.

[19] Heidrun Mollenkopf et al. "Social and behavioural science perspectives on out-of-home mobility in later life: Findings from the European project MOBILATE". In: *European Journal of Ageing* 1.1 (2004), pp. 45–53.

[20] Statistisches Bundesamt. *Unfallentwicklung auf Deutschen Strassen 2012: Begleitmaterial zur Pressekonferenz am 10. Juli 2012 in Berlin*. 2013. URL: https://www.destatis.de/DE/PresseService/Presse/Pressekonferenzen/2013/Unfallentwicklung_2012/begleitheft_Unfallenwicklung_2012.html (visited on).

[21] Statistisches Amt Berlin Brandenburg. *Verkehrsstatistiken*. 2015. URL: https://www.statistik-berlin-brandenburg.de/publikationen/stat_berichte/2014/SB_H01-02-00_2013j01_BB.pdf (visited on).

[22] Stab Des Polizeipräsidenten. *Sonderutnersuchung Fußgängerverkehrsunfälle in Berlin 2013*. 2013. URL: https://www.berlin.de/polizei/_assets/aufgaben/anlagen-verkehrssicherheit/fussgaenger2013.pdf (visited on).

[23] Hasham Shahid Qureshi, Florian Breitinger, and Rebecca Wiczorek. "Entwicklung und Evaluation eines Fußgänger- Assistenzsystems für ältere Nutzerinnen und Nutzer". In: *IT für soziale Inklusion*. Ed. by Aljoscha Burchardt and Hans Uszkoreit. Berlin and Boston: De Gruyter Oldenbourg, 2018, pp. 141–150. ISBN: 9783110561371. DOI: 10.1515/9783110561371-013.

[24] Amt für Statistik Berlin-Brandenburg. *Statistischer Bericht: Einwohnerinnen und Einwohner im Land Berlin*. 2020. URL: https://www.statistik-berlin-brandenburg.de/publikationen/stat_berichte/2021/SB_A01-05-00_2020h02_BE.pdf.

[25] Aurélie Dommes et al. "Crossing a two-way street: comparison of young and old pedestrians". In: *Journal of Safety Research* 50 (2014), pp. 27–34. ISSN: 0022-4375. DOI: 10.1016/j.jsr.2014.03.008. URL: http://www.sciencedirect.com/science/article/pii/S0022437514000395.

[26] Jennie Oxley et al. "Differences in traffic judgements between young and old adult pedestrians". In: *Accident Analysis & Prevention* 29.6 (1997), pp. 839–847. ISSN: 0001-4575. DOI: 10.1016/S0001-4575(97)00053-5. URL: http://www.sciencedirect.com/science/article/pii/S0001457597000535.

[27] Erel Avineri, David Shinar, and Yusak O. Susilo. "Pedestrians' behaviour in cross walks: The effects of fear of falling and age". In: *Accident Analysis & Prevention* 44.1 (2012), pp. 30–34. ISSN: 0001-4575. DOI: 10.1016/j.aap.2010.11.028. URL: http://www.sciencedirect.com/science/article/pii/S0001457510003726.

[28] Rebecca Wiczorek, Jan Siegmann, and Florian Breitinger. "Investigating the impact of attentional declines on road-crossing strategies of older pedestrians". In: *Proceedings of the Human Factors and Ergonomics Society Europe*. 2016, pp. 155–169.

[29] Jan Siegmann, Janna Protzak, and Rebecca Wiczorek. “Modality effects of secondary tasks on hazard detection performance of younger and older pedestrians in a simulated road crossing task”. In: *Proceedings of the Human Factors and Ergonomics Society Europe*. 2017, pp. 21–34.

[30] Janna Protzak and Rebecca Wiczorek. “On the Influence of Walking on Hazard Detection for Prospective User-Centered Design of an Assistance System for Older Pedestrians”. In: *i-com* 16.2 (2017), pp. 87–98.

[31] Stephen R. Dixon, Christopher D. Wickens, and Jason S. McCarley. “On the independence of compliance and reliance: Are automation false alarms worse than misses?” In: *Human Factors* 49.4 (2007), pp. 564–572.

[32] Poornima Madhavan, Douglas A. Wiegmann, and Frank C. Lacson. “Automation failures on tasks easily performed by operators undermine trust in automated aids”. In: *Human Factors* 48.2 (2006), pp. 241–256.

[33] J. P. Bliss, R. D. Gilson, and J. E. Deaton. “Human probability matching behaviour in response to alarms of varying reliability”. In: *Ergonomics* 38.11 (1995), pp. 2300–2312.

[34] Massimo Bertozzi, Alberto Broggi, and Alessandra Fascioli. “Vision-based intelligent vehicles: State of the art and perspectives”. In: *Robotics and Autonomous Systems* 32.1 (2000), pp. 1–16. ISSN: 0921-8890. DOI: 10.1016/S0921-8890(99)00125-6.

[35] J. C. McCall and M. M. Trivedi. “Video-Based Lane Estimation and Tracking for Driver Assistance: Survey, System, and Evaluation”. In: *IEEE Transactions on Intelligent Transportation Systems* 7.1 (2006), pp. 20–37. ISSN: 1524-9050. DOI: 10.1109/TITS.2006.869595.

[36] Véronique Prinet et al. “3D road curb extraction from image sequence for automobile parking assist system”. In: *2016 IEEE International Conference on Image Processing (ICIP)*. 2016, pp. 3847–3851.

[37] Mingmei Cheng et al. “Curb Detection for Road and Sidewalk Detection”. In: *IEEE Transactions on Vehicular Technology* 67.11 (2018), pp. 10330–10342. DOI: 10.1109/TVT.2018.2865836.

[38] Martin Kellner et al. “Multi-cue, model-based detection and mapping of road curb features using stereo vision”. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. 2015, pp. 1221–1228.

[39] Danish Sodhi et al. “Crf based method for curb detection using semantic cues and stereo depth”. In: *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*. 2016, p. 41.

[40] C. Fernández et al. “Road curb and lanes detection for autonomous driving on urban scenarios”. In: *IEEE 17th International Conference on Intelligent Transportation Systems*. 2014, pp. 1964–1969.

[41] Martin Kellner, Mohamed Essayed Bouzouraa, and Ulrich Hofmann. “Road curb detection based on different elevation mapping techniques”. In: *IEEE Intelligent Vehicles Symposium*. 2014, pp. 1217–1224.

[42] Markus Enzweiler et al. “Towards multi-cue urban curb recognition”. In: *IEEE Intelligent Vehicles Symposium*. 2013, pp. 902–907.

[43] Alexander Seibert et al. “Camera based detection and classification of soft shoulders, curbs and guardrails”. In: *IEEE Intelligent Vehicles Symposium*. 2013, pp. 853–858.

[44] J. Siegemund, U. Franke, and W. Förstner. “A temporal filter approach for detection and reconstruction of curbs and road surfaces based on Conditional Random Fields”. In: *IEEE Intelligent Vehicles Symposium*. 2011, pp. 637–642.

[45] F. Oniga and S. Nedevschi. “Curb detection for driving assistance systems: A cubic spline-based approach”. In: *IEEE Intelligent Vehicles Symposium*. 2011, pp. 945–950.

[46] Tingbo Hu and Tao Wu. “Roadside curb detection based on fusing stereo vision and mono vision”. In: *Fourth International Conference on Machine Vision (ICMV 2011): Computer Vision and Image Analysis; Pattern Recognition and Basic Technologies*. Vol. 8350. 2012, 83501H.

[47] Jan Siegemund et al. “Curb reconstruction using conditional random fields”. In: *2010 IEEE Intelligent Vehicles Symposium*. 2010, pp. 203–210.

[48] Florin Oniga, Sergiu Nedevschi, and Marc Michael Meinecke. “Curb detection based on a multi-frame persistence map for urban driving scenarios”. In: *2008 11th International IEEE Conference on Intelligent Transportation Systems*. 2008, pp. 67–72.

[49] Markus Enzweiler et al. “Towards multi-cue urban curb recognition”. In: *IEEE Intelligent Vehicles Symposium*. 2013, pp. 902–907.

[50] Gangqiang Zhao and Junsong Yuan. “Curb detection and tracking using 3D-LIDAR scanner”. In: *2012 19th IEEE International Conference on Image Processing (ICIP 2012)*, pp. 437–440. DOI: 10.1109/ICIP.2012.6466890.

[51] Sebastian Thrun et al. “Stanley: The robot that won the DARPA Grand Challenge”. In: *Journal of Field Robotics* 23.9 (2006), pp. 661–692. ISSN: 15564959. DOI: 10.1002/rob.20147.

[52] Jonathan Bohren et al. “Little Ben: The Ben Franklin Racing Team’s entry in the 2007 DARPA Urban Challenge”. In: *Journal of Field Robotics* 25.9 (2008), pp. 598–614. ISSN: 15564959. DOI: 10.1002/rob.20260.

[53] Yihuan Zhang et al. “A real-time curb detection and tracking method for UGVs by using a 3D-LIDAR sensor”. In: *2015 IEEE Conference on Control Applications (CCA)*. Piscataway, NJ: IEEE, 2015, pp. 1020–1025. ISBN: 978-1-4799-7787-1. DOI: 10.1109/CCA.2015.7320746.

[54] Tongtong Chen et al. “Velodyne-based curb detection up to 50 meters away”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. 2015, pp. 241–248.

[55] Martin Kellner et al. “Laserscanner based road curb feature detection and efficient mapping using local curb descriptions”. In: *IEEE 17th International Conference on Intelligent Transportation Systems*. 2014, pp. 2602–2609.

[56] A. Y. Hata, F. S. Osorio, and D. F. Wolf. “Robust curb detection and vehicle localization in urban environments”. In: *IEEE Intelligent Vehicles Symposium Proceedings*. 2014, pp. 1257–1262.

[57] Sheng Xu, Ruisheng Wang, and Han Zheng. “Road Curb Extraction From Mobile LiDAR Point Clouds”. In: *IEEE Transactions on Geoscience and Remote Sensing* 55.2 (2017), pp. 996–1009. ISSN: 0196-2892. DOI: 10.1109/TGRS.2016.2617819.

[58] Borja Rodríguez-Cuenca et al. “An approach to detect and delineate street curbs from MLS 3D point cloud data”. In: *Automation in Construction* 51 (2015), pp. 103–112. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2014.12.009. URL: http://www.sciencedirect.com/science/article/pii/S0926580514002532.

[59] Carlos Fernández et al. “Curvature-based curb detection method in urban environments using stereo and laser”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. 2015, pp. 579–584.

[60] Jun Tan et al. “Robust curb detection with fusion of 3D-Lidar and camera data”. In: *Sensors (Basel, Switzerland)* 14.5 (2014), pp. 9046–9073. DOI: 10.3390/s140509046.

[61] Pierre Olivier. *Leddar optical time-of-flight sensing technology: A new approach to detection and ranging: A new approach to detection and ranging*. 2015. URL: https://d1wx5us9wukuh0.cloudfront.net/app/uploads/dlm_uploads/2016/02/Leddar-Optical-Time-of-Flight-Sensing-Technology-1.pdf (visited on).

[62] Hasham Shahid Qureshi, Tobias Glasmachers, and Rebecca Wiczorek. “User-Centered Development of a Pedestrian Assistance System Using End-to-End Learning”. In: *17th IEEE International Conference on Machine Learning and Applications*. Ed. by M. A. Wani. Los Alamitos, CA: IEEE Computer Society,

Conference Publishing Services, 2018, pp. 808–813. ISBN: 978-1-5386-6805-4. DOI: 10.1109/ICMLA.2018.00129.

[63] Tobias Glasmachers. *Limits of End-to-End Learning*. URL: http://arxiv.org/pdf/1704.08305v1.

[64] Mike Lewis et al. *Deal or No Deal? End-to-End Learning for Negotiation Dialogues*. URL: http://arxiv.org/pdf/1706.05125v1.

[65] A. C. Serban, E. Poll, and J. Visser. "A Standard Driven Software Architecture for Fully Autonomous Vehicles". In: *2018 IEEE International Conference on Software Architecture Companion (ICSA-C)*. 2018, pp. 120–127. DOI: 10.1109/ICSA-C.2018.00040.

[66] Hasham Shahid Qureshi et al. "Umwelt-System-Mensch-Interaktion: Auswahl und Evaluation von Sensorik und Aktuator eines Fußgänger-Assistenzsystems für ältere Menschen im Straßenverkehr". In: *INFORMATIK 2017*. Ed. by Maximilian Eibl and Martin Gaedke. Gesellschaft für Informatik, Bonn, 2017, pp. 713–716. DOI: 10.18420/in2017_68.

[67] Tom M. Mitchell. *Machine Learning*. International ed., [Reprint.] McGraw-Hill series in computer science. New York and London: McGraw-Hill, 1997. ISBN: 0070428077.

[68] Maureen Caudill. "Neural Networks Primer, Part I". In: *AI Expert* 2.12 (1987), pp. 46–52. ISSN: 0888-3785.

[69] Warren S. McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The Bulletin of Mathematical Biophysics* 5.4 (1943), pp. 115–133. ISSN: 0007-4985. DOI: 10.1007/BF02478259.

[70] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain". In: *Psychological Review* 65.6 (1958), pp. 386–408. ISSN: 0033-295X. DOI: 10.1037/h0042519.

[71] Michael Nielson. *Neural Networks and Deep Learning*. 2019. URL: http://neuralnetworksanddeeplearning.com/index.html.

[72] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Harvard University, 1975. URL: https://books.google.de/books?id=z81XmgEACAAJ.

[73] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing – Explorations in the Microstructure of Cognition*. MIT Press, 1986, pp. 318–362.

[74] Kenneth Levenberg. "A method for the solution of certain non-linear problems in least squares". In: *Quarterly of Applied Mathematics* 2.2 (1944), pp. 164–168. ISSN: 0033-569X. DOI: 10.1090/qam/10666.

[75] Donald W. Marquardt. “An Algorithm for Least-Squares Estimation of Nonlinear Parameters”. In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441. ISSN: 0368-4245. DOI: 10.1137/0111030.

[76] Christopher M. Bishop. *Pattern recognition and machine learning (Information science and statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 9780387310732.

[77] He Kaiming et al. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1026–1034. DOI: 10.1109/ICCV.2015.123.

[78] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *AISTATS*. 2010.

[79] Sagar Sharma. *Activation Functions in Neural Networks*. URL: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[80] Abien Fred Agarap. “Deep Learning using Rectified Linear Units (ReLU)”. In: *CoRR* abs/1803.08375 (2018).

[81] Thomas Wood. *Softmax Function*. URL: https://deepai.org/machine-learning-glossary-and-terms/softmax-layer.

[82] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[83] Richard Szeliski. *Computer vision: Algorithms and applications*. Texts in computer science. London: Springer, 2011. ISBN: 9781848829343. DOI: 10.1007/978-1-84882-935-0.

[84] Nicu Sebe, Ira Cohen, and Ashutosh Garg. *Machine Learning in Computer Vision*. 1. Aufl. Vol. v. 29. Computational imaging and vision. s.l.: Springer-Verlag, 2005. ISBN: 1402032749.

[85] Kunihiko Fukushima. “Neocognitron: A hierarchical neural network capable of visual pattern recognition”. In: *Neural Networks* 1 (1988), pp. 119–130.

[86] Y. LeCun et al. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Computation* 1.4 (1989), pp. 541–551.

[87] D. H. Hubel and T. N. Wiesel. “Receptive fields and functional architecture of monkey striate cortex”. In: *The Journal of Physiology* 195.1 (1968), pp. 215–243. ISSN: 0022-3751.

[88] D. Marr and E. Hildreth. “Theory of edge detection”. In: *Proceedings of the Royal Society of London. Series B, Biological sciences* 207.1167 (1980), pp. 187–217. ISSN: 0950-1193. DOI: 10.1098/rspb.1980.0020.

[89] Logitech. *Logitech HD Webcam C270 Technical Specifications*. 2012. URL: https://support.logi.com/hc/en-us/articles/360023462093-Logitech-HD-Webcam-C270-Technical-Specifications (visited on).

[90] John Canny. "A computational approach to edge detection". In: *Readings in computer vision*. Elsevier, 1987, pp. 184–203.

[91] Irvin Sobel. "Neighborhood coding of binary images for fast contour following and general binary array processing". In: *Computer graphics and image processing* 8.1 (1978), pp. 127–135.

[92] Judith M. S. Prewitt. "Object enhancement and extraction". In: *Picture processing and Psychopictorics* 10.1 (1970), pp. 15–19.

[93] Yann LeCun, Corinna Cortes, and C. J. Burges. "MNIST handwritten digit database". In: *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[94] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. "CIFAR-10 (Canadian Institute for Advanced Research)". In: (). URL: http://www.cs.toronto.edu/%C2%A0kriz/cifar.html.

[95] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc, 2012, pp. 1097–1105.

[96] Mariusz Bojarski et al. *End to End Learning for Self-Driving Cars*. URL: http://arxiv.org/pdf/1604.07316v1 (visited on).

[97] Urs Muller et al. "Off-Road Obstacle Avoidance through End-to-End Learning". In: *Advances in Neural Information Processing Systems 18*. Ed. by Y. Weiss, B. Schölkopf, and J. C. Platt. MIT Press, 2006, pp. 739–746. URL: http://papers.nips.cc/paper/2847-off-road-obstacle-avoidance-through-end-to-end-learning.pdf.

[98] J. Brownlee. *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery, 2019. URL: https://books.google.de/books?id=DOamDwAAQBAJ.

[99] Y. LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. ISSN: 00189219. DOI: 10.1109/5.726791.

[100] Matthew D. Zeiler and Rob Fergus. "Visualizing and Understanding Convolutional Networks". In: *CoRR* abs/1311.2901 (2013).

[101] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (9/4/2014). URL: http://arxiv.org/pdf/1409.1556v6 (visited on).

[102] Christian Szegedy et al. "Going Deeper With Convolutions". In: 2015, pp. 1–9. URL: http://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf.

[103] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: 2016, pp. 770–778. URL: http://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf.

[104] J. Philbin et al. "Object retrieval with large vocabularies and fast spatial matching". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2007, pp. 1–8.

[105] Kevin P. Murphy. *Machine learning: A probabilistic perspective / Kevin P. Murphy*. Adaptive computation and machine learning series. Cambridge, Mass. and London: MIT Press, 2012. ISBN: 0262018020.

[106] T. Tieleman and G. Hinton. *Lecture 6a overview of mini–batch gradient descent: Neural Networks for Machine Learning*. 2012. URL: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf (visited on).

[107] David M. Green and John A. Swets. *Signal detection theory and psychophysics*. Oxford, England: John Wiley, 1966.

[108] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[109] Hasham Shahid Qureshi and Rebecca Wizcorek. "Curb Detection for a Pedestrian Assistance System using End-to-End Learning". In: *Journal of WSCG* 27.1 (2019). DOI: 10.24132/JWSCG.2019.27.1.9.

[110] Brandon Pitts and Nadine Sarter. "Two is company, three is a crowd: Age-related differences in processing concurrent visual, auditory, and tactile cues". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 60. 2016, p. 1544.

[111] Rebecca Wiczorek. "Evaluation of Thermotactile and Vibrotactile Cues to Improve Hazard Perception of Older Pedestrians [Under Review]". In: ().

[112] Frank Bolton, Shahram Jalaliniya, and Thomas Pederson. "A Wrist-Worn Thermohaptic Device for Graceful Interruption". In: *IxD&A* 26 (2015), pp. 39–54. URL: http://dblp.uni-trier.de/db/journals/ixda/ixda26.html#BoltonJP15.

[113] Rebecca Wiczorek and Janna Protzak. “The Impact of Visual and Cognitive Dual-Task Demands on Traffic Perception During Road Crossing of Older and Younger Pedestrians”. In: *Frontiers in psychology* 13 (2022), p. 775165. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2022.775165.

[114] Rebecca Wiczorek and Janna Protzak. “Evaluation of an assistance system supporting older pedestrians’ road crossing in virtual reality and in a real-world field test”. In: *Frontiers in psychology* 13 (2022), p. 966096. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2022.966096.

[115] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2017), pp. 1137–1149. DOI: 10.1109/TPAMI.2016.2577031.

[116] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 6/27/2016 - 6/30/2016, pp. 779–788. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.91.

[117] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 21–37. ISBN: 978-3-319-46448-0.