November 26, 2018

MASTER THESIS

DEVICE-FREE SENSING AND DEEP LEARNING: ANALYSING HUMAN BEHAVIOUR THROUGH CSI USING CONVOLUTIONAL NETWORKS

Jeroen Klein Brinke

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Pervasive Systems

Exam committee:
dr. N. Meratnia
prof.dr. P.J.M. Havinga
ir. E. Molenkamp

UNIVERSITY OF TWENTE.

Abstract

Monitoring human beings is becoming a more pressing matter in this society. It is common these days for people to wear wearable sensors in the form of smartwatches or activity trackers in order to sustain a healthy lifestyle, increase activity or improve sport performances. The desire to monitor ourselves becomes apparent by how fast different technologies are being adapted into monitoring systems.

Device-free sensing is one of these new monitoring systems. This is a method that allows the monitoring and localization of humans through measuring the characteristics of these wireless signals. A method to do so is by looking at the way these signals propagate through the channel state information (CSI). Deep learning is an upcoming field within machine learning used to classify human activities. It attempts to replicates the way humans perceive information. Of all the techniques within deep learning, convolutional networks have proven to be capable of dealing with signal processing, yet it has not been widely adapted in human activity recognition through CSI.

This research presented in this thesis looks into how convolutional networks perform against current state-of-the-art systems through analyzing both static and dynamic activities. The experiments performed in this research were conducted with multiple participants over three days to investigate about the scalability. The findings indicate that for dynamic activities, convolutional networks achieve higher accuracies than static postures (98% and 60%). Scalability between participants and days falls short, but findings imply that this is due to a lack of data.

Keywords

device-free sensing; channel state information; csi; deep learning; convolutional networks; cnn; human activity recognition

Preface and acknowledgements

This research is part of the 'Create Health' project at the Pervasive Systems group, located at the University of Twente, Enschede, and connects to research done by Prachi Bagave for her Master thesis *Unobtrusive sensing using Wi-Fi signals*. Introductory work started in January 2018 in the form of Research Topics (a 10 EC literature research course before the graduation project), after I was contacted by Nirvana Meratnia in December 2017. I was done with Research Topics at the end of April 2018, but I did not actually started to work on the project until June 2018, as I was awaiting a grade for another course.

This project has been an interesting learning experience for me and has taught me many things. At times, this project was terrorized by delays: at first it was waiting on hardware for two months, after which it became waiting for my computer and laptop to be done rendering images and training neural networks. I have learned a great deal about planning and organizing a large scale research and I have learned a lot about analyzing and gathering data. I will never forget to collect ground truth data in future experiments and I will not sit idle while waiting for results when I can do other things. However, the most important thing this project taught me is that I enjoy research and it was a great boost in for my PhD.

I would like to offer my deepest gratitude to dr. Nirvana Meratnia for offering me a position in this project and by supervising me throughout the entirety of Research Topics and this final graduation project. She has offered me lots of helpful advice and always managed to motivate me to take the extra step. When I was stuck on the project, she would offer me insight and other opportunities to continue the project. It was thanks to her motivation I managed to finish my thesis in time.

For the hardware aspects of this project, I want to express my gratitude towards dr. ir. D.V. Le Viet Duc and ir. D.T.N. Nguyen for supplying me with the

appropriate hardware (the router and other attritbutes), as well as helping to combine all the hardware in a small node. Without their help, I would not have been able to create the nodes currently in use.

I would like to deeply thank Prachi Bagave for supplying me with her data and helping me develop a better understanding of the Linux CSI Tool and CSI in general. Without her data and help, I would not have been able to get as far with my research as I did.

I would also like to thank the rest of the Pervasive Systems group for the conversations and insights I gathered during the time of my graduation project and I hope to continue these conversations during my PhD at the Pervasive Systems group.

I also want to thank prof. dr. P.J.M. Havinga and ir. E. Molenkamp for freeing up the time to read my thesis and being part of my final presentation.

I also want to thank my parents, little sister, girlfriend and friends in general for supporting me throughout the graduation project and letting me let off steam regarding my weaker moments. I especially want to thank one of my friends for helping me develop the software for gathering data and being the first participant and therefore helping me fine-tune the experiments. I also want to thank my other friends for participating in my experiments as I know the experiments were physically quite demanding. I also want to thank my little sister and my girlfriend for taking part in the experiments. Lastly, want to thank my girlfriend again for helping me collect data over multiple days, design the images and figures for my thesis, and being a constant support over the past few weeks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In our current society, the desire to monitor the world around is increasing. As technologies are advancing, it becomes easier to monitor and make life easier and more secure. Techniques are being used to make homes and cities smarter, prevent illegal poaching and thus help animal species threatened with extinction, structural health monitoring to aid in preventive maintenance, and monitor humans in their behaviour and health. Especially within monitoring humans, there are a lot of different fields. These include, but are not limited to, the monitoring of physical conditions to improve sport performance or to aid in recovery, monitor activities to aid in a smart home situation, or monitor medical symptoms to prevent and predict how diseases are progressing to call experts in case of an emergency. Currently, the most accessible and defined tool to monitor the aforementioned situations are wireless sensor networks.

Wireless sensor networks (WSN) have proven to be an important tool in continously monitoring situations or detecting events and are often considered to be unobtrusive[1][2][3]. However, wireless sensor networks are not truly unobtrusive: they often require physical sensors worn on the structures or body parts. Wireless sensor networks consist of a network of small, potentially different sensors, measuring the same or different aspects of the environment, but combining the information and data to come to conclusions. This means that situations are often measured directly: sensors can be put directly on the subject (building, human, animal) in question and thus the actions and events affect the sensor directly.

Truly unobtrusive sensing can only happen when the events are measured in-

directly. Therefore, when it comes to truly unobtrusive sensing, the focus has more and more shifted to device-free sensing [4][5][6][7][8][9][10][11][12]. Device-free sensing uses the "traffic" in the transmission environment generated by the actions or events caused by subjects and are thus often measuring the effects and impact of said actions and events, thus measuring the actions and events themselves indirectly. While device-free sensing still uses nodes (laptops or routers) to measure the signals, they are often not part of the action (unlike wireless sensor networks) and are therefore considered to be truly unobtrusive.

In telecommunication, device-free sensing is often done through measuring the wireless medium, thus the radio waves that are traveling through the air. There are several techniques to do so, such as measuring the packet receiving rate (PRR) [5], received signal strength indicator (RSSI) [8] or the channel state information (CSI). Lately, it is believed that CSI is the best characteristic to look into, as it captures the entire propagation from transmitter to receiver [6][13][14][15][10][11][16][17][18][19]. None of these solutions measure the wireless medium directly, that is, listening to the frequencies and notify which frequencies are used. Instead of measuring the wireless medium directly, these solutions require one to send traffic through the medium and measure how this traffic is affected by the impact of the actions one wants to measure.

To understand the previous better, one can imagine two people separated by an ocean: they cannot directly see each other, making the only way to communicate with one another to create waves that eventually reach the other person. Even when sending the same message, the receiver will a difference in the waves each time: they could be taller, they could come in at a different angle, or the delay could be slightly larger. This is all caused by impacts on the water, which could be environmental (e.g. the wind), or actions by people (e.g. ships on the water, people on the shore fishing). Each of these activities has a specific signature on the water, and thus by analysing the waves received (by sending a known message), one could classify these activities. The same technique can be used for radio waves, albeit different characteristics due to the frequency spectrum being slightly more complex.

This technique of device-free sensing can be used in human activity recognition: detecting, classifying and monitoring humans has proven to be a demanding research field when it comes to device-free sensing [8][6]. Research currently focusses

on indoor localization through CSI analysis [14][20] and human activity itself [8][6]. An especially interesting field in this is the support for elderly or handicapped: creating such systems allow automatic fall or aid detection, while increasing privacy, as current systems require either wearable sensors [21][22][23][24][25][26] or video cameras [27][28][29][30]. When it comes to health care (whether it is for the elderly, handicapped or the average person), an important incentive to invest in device-free sensing is the ease of use: the wireless signals are already there, so one only requires a node (or several nodes) to measure the disturbances and warn in case of an emergency. This is much more convenient than wearable sensors, as one cannot expect the elderly or handicapped to always wear body sensors, nor can they be expected to always carry their phones or smartwatches. A device-free system also offers much more privacy than video cameras, in the sense that when people feel watched, they act differently and feel more uncomfortable (Hawthorne effect).

The most famous way to recognize human activity is machine learning. Machine learning is the field in artificial intelligence that uses mathematical techniques (from structured data) to give computers the power to learn from data, without humans interfering with such a system. While such systems have proven to be accurate in human activity recognition (both for wearable sensor systems [31][32][23][24][26] as for device-free solutions [33][8][5][16][17]), these systems often need to be trained by having structured data and by having the features picked by humans.

Deep learning is a field within machine learning. It is an old concept, but only in the last few years has technology advanced to the point where it can actually work with these deep, neural networks. Deep learning tries to replicate the way humans perceive and process, by creating neurons and activating these. Deep learning requires no structured data and no human intervention: it will pick its own features and learn from these automatically. They are also capable of adapting to changes that occur over time (by learning new features, or adapting existing features). Another important difference between deep learning and machine learning has to do with transparency: in machine learning, it is possible to analyse how a decision was reached, whereas in deep learning it is not. Even though it is a new field when it comes to research, within both wearable sensors and device-free sensing it is a field that is extensively used [3][34][20][2][35][13][18].

In deep learning, a distinction can be made into two main fields: convolutional networks (CNN) [3][34][20][35][19] and recurrent neural networks (RNN) [2][13][11]. The former focusses on recognizing and classifying of objects in images and the latter for recognizing and classifying recurring events (such as texts). Convolutional neural networks have been proven to work fine with CSI, as CSI can be classified as unstructured data and by analysing the graphs it will determine its own features.

1.1 Problem statement

Even though a lot of research goes into the analysis of CSI through convolutional networks, there are a few shortcomings. The first of which is the size of the nodes: these are usually full-sized computers or laptops, which are not at all convenient to place around rooms in inobtrusive manner. If the technology is to be used in practice, for example in health care to support the elderly, the nodes must be smaller in order to be more easily placed around a room.

Secondly, while plenty of research goes into human activity recognition and detection, research falls short when it comes to looking at the scalability of their solutions: it is often only one or two subjects for which the data is collected. If a system is to work properly in a real-life setting, the system must work for humans with a lot of different characteristics: height, weight, gender and style.

Lastly, different research often uses the same characteristic of CSI: the amplitude and phase. It is possible that other characteristics of the CSI provide more information. One of these characteristics is the slope analysis of the CSI data: slopes give a lot of information regarding how fast a change occurred, rather than how impactful the change was (amplitude).

1.1.1 Research aim and questions

The main aim of this research is to give insight in how convolutional networks can be used to classify human behaviour through measuring disturbances in the channel state information. More specifically, the focus will be on giving give insight in whether or not it is feasible to use convolutional network to classify behaviour and activities between different people.

Furthermore, this research aims to create CSI collecting nodes smaller than the current state of the art, in order to make the nodes more mobile and therefore more functional. Smaller nodes are easier to put in rooms and to leave unattended, rather than having entire laptops or computer collecting the CSI.

The main research question for this research is: *What is the influence of multiple people and days on CSI when classifying human activity through deep learning.* This research emphasizes convolutional network, as this is a promising deep learning technique and this is thus compared to the current state-of-the-art. In order to answer this question, multiple research questions will be answered:

RQ1 What is the correlation between feature extraction and the accuracy of CNNs when classifying human activities using a single node?

RQ2 What is the correlation between combining data of participants for training/classification and the accuracy of the system?

RQ3 What is the correlation between combining data of participants for training/classification and the accuracy of the system?

It should be noted that part of this research is dedicated to creating nodes smaller than the current state-of-the-art. This has to do with the current nodes not being small enough to unobtrusively place nodes around a room. As future research involves multiple nodes in a single room, it is important to create a smaller node.

1.2 Research overview

The research started by laying out the current state-of-the-art methodology and hardware through an extensive literature research. The start of the literature research was made in another paper, done for the course Research Topics.

The first part of the research consisted of creating the hardware. By reading through the literature, several requirements were found when it came to the hardware. These requirements included the maximum size of a node in order to be smaller than the state-of-the-art, as well as software restrictions. With these requirements in mind, several hardware solutions were attempted: two single-board computers (Raspberry Pi 3 and Hummingboard Edge i2eX). However, none of

these worked properly: the Raspberry Pi 3 failed because of hardware limitations, whereas the Hummingboard Edge i2eX failed due to a combination of hardware and software limitations. In the end, a small, barebone computer (Gigabyte Brix IoT) was modified in order to fit the components and function properly. To finish of the first part of the research, software was written that allowed i) adaptable CSI collection, ii) synchronizing the data to a central server, and iii) collect device status data.

Secondly, existing data was used to familiarize with the CSI data, as well as verify the functionalities of convolutional networks in these situations. The data used consisted of multiple activities and locations: i) sit and stand, ii) basic shapes (five basic hand movements), iii) static postures (five postures for which the CSI was measured), iv) multiple day data and v) standard activities (doing nothing, clapping, walking, jogging and jumping). First, it will be explained what data and how it was fed into the convolutional neural network. To finish the second part of the research, the network is trained for all of the aforementioned activities and the performance is compared to the results of the research from which the existing data originates. For static postures, the current implementation of the neural network scores lower than the previous research. However, for the dynamic activities, the convolutional network outperformed previous research in multiple cases.

Lastly, the experiments for this research were conducted. The activities were performed in an actual living room environment over the course of three days. Participants were encouraged to perform activities differently, in order to view how this would impact accuracies both per day and over different days. Another group of participants were asked to perform activities consistently over multiple days. Also, interference was added by always being engaged in conversation with the participants, as well as performing small tasks next to them. Accuracies were high for individual participants and even considerably high between participants on a given day. Accuracies dropped when classifying for a group of participants and trying to classify the others. Accuracies also dropped for classification for a single participant over multiple days, but not as low as for different participants. However, it seems likely that more data would greatly increase the accuracy of the system.

1.3 Thesis overview

After the introduction, background information will be given on both wireless signals (section 2.1) and deep learning (section 2.2). In general, it is recommended to read through these chapters as they can be a refresher to knowledge and furthermore, to lead immediately into the knowledge required to understand this research. Having a lack of knowledge in either of these fields, it is strongly recommended to read these chapters. In wireless signals, the basics regarding frequencies, modulation and influences will be discussed. This is necessary to understand the analysis of networks through RSSI, SNR and CSI, all of which are discussed in the backgrond afterwards. Finally, a quick look into CSI analysis through `MATLAB` is given.

For deep learning, the section first explains the basics of a neural network: the softmax function, cross entropy and (stochastic) gradient descent, as well as the different layers, propagations and regularizations. After these basic concepts, the two main neural networks are explained: convolutional and recurrent neural networks, respectively. Also, not just the basic principles of these neural nets are explained, also the main differences and their strengths are explained.

After the background, the state-of-the-art is listed and explained in section 3.1. A distinction is made between two main subjects: unobtrusive human activity classification through device-free sensing and analysing networks and their disturbances, respectively. After listing and explaining the state-of-the-art, the state-of-the-art is discussed and the main strengths and shortcomings are pointed out.

Next, the hardware creation is discussed. First, the requirements for the nodes are listed in section 4.1. These requirements are split in the functional and non-functional requirements. After the requirements are presented, a look is taken into possible solutions and ideas to create such a wireless node and the shortcomings of the attempted solutions. The chapter is concluded by given the specifications to the final hardware, as well as giving the written software to create an understanding on how the node (and a network of these nodes) work.

The first use of the convolutional networks is used in chapter 4, when existing data (provided by previous research) is analysed using alternative methods. First, the activities are explained: sitting, standing, standing in a fixed position and hand movements, as well as some unused data. The methodology and the shortcomings

of the mentioned methodology are mentioned next, together with the alternative approach provided by this research and the results of the alternative approach.

In chapter 5, the experiments that belong to this research are conducted and explained. First, the new list of activities is listed: walking, running, jumping, falling, clapping and sitting. It is then explained how this data is gathered and analysed in the methodology, after which the experimental setup is explained and finally the results of the conducted research.

The final two chapters are the conclusion and future work, respectively.

Chapter 2

Background

2.1 Wireless signals

Wireless signals are considered to be radio signals carried through the air.

2.1.1 Frequencies and channels

First, to understand the definition of a channel, one must understand frequencies. The technical definition of a frequency is the number of recurring events per second; thus, formally described a frequency f in Hz (Hertz) can be described as

$$f = \frac{1}{T}$$

where T is the period of a single event in these repetitions in seconds ([36], p. 29). Frequencies, combined with amplitude and their period, can be represented as a radio wave in a graph, as shown in figure 2.1. Combining this with a phase, it can be expressed in the following formula (known as a sinusoid) ([36], p. 29):

$$s(t) = A\sin(2\pi ft + \phi)$$

To give an example within the field of *wireless communications*, suppose a low frequency signal A and high frequency B, of which the period is 4 times smaller. The difference in their period can be seen in figure 2.2.

There is a reason specifically 2.4 GHz and 5 GHz were chosen in the previous

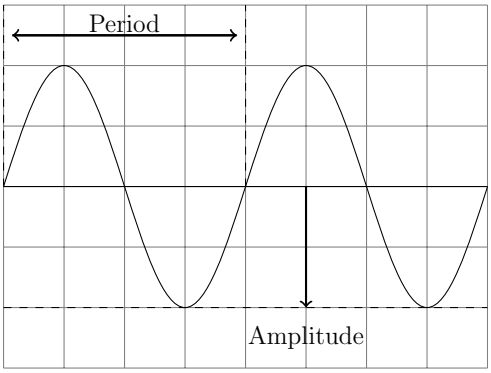


Figure 2.1: Example showing a radio wave in wireless communications

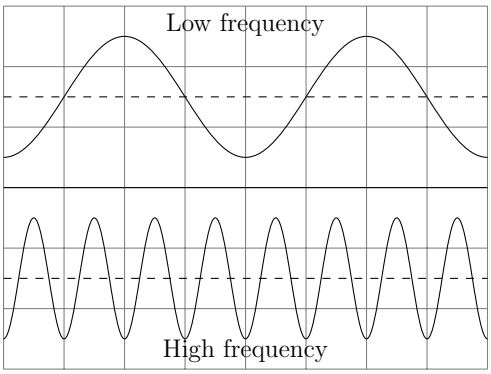


Figure 2.2: Difference between low and high frequencies

example: these are two famous WLAN channels specified by IEEE 802.11[1] ([36], p. 359). Now, with the basic understanding of frequencies, one can move on to *channels*. Every frequency range (such as 2.4GHz or 5GHz) is once again divided into different channels of slices of this frequency ([36], p. 50). Each country is allowed to apply their own rules and regulations on how many users and which power levels are allowed in these channels.

Frequencies can still affect the neighboring frequencies ([36], p. 225) and cause interferences in these. This is known as *overlapping* channels. As one might suspect, there is also such as a thing as *non-overlapping* channels: this is usually done by leaving several frequencies of space between each channel.

Another aspect worth mentioning between lower and high frequencies, is that

[1]IEEE 802.11 is set of MAC (Medium Access Control) and physical layer specifications for designing hardware

lower frequencies can travel further and through certain obstacles, whereas higher frequencies tend to lose their strength faster and can thus travel less far. However, due to higher frequencies having more "events per second", they can transmit data faster (that is, more data per second) and are more sensible to noise ([36], p. 46).

2.1.2 Modulation

In wireless signals, modulation is changing some aspect of the default waveform (as described in **Frequencies and channels**), now referred to as the *carrier signal*, using the signal that needs to be transmitted, hereafter mentioned as the *modulating signal*, into a combined signal that is transmitted (the *modulated signal*). In digital modulation, three key aspects of the signal can be changed: amplitude, frequency and phase, which are respectively called amplitude shift keying (ASK), frequency shift keying (FSK) and phase shift keying (PSK) ([36], p. 133-136).

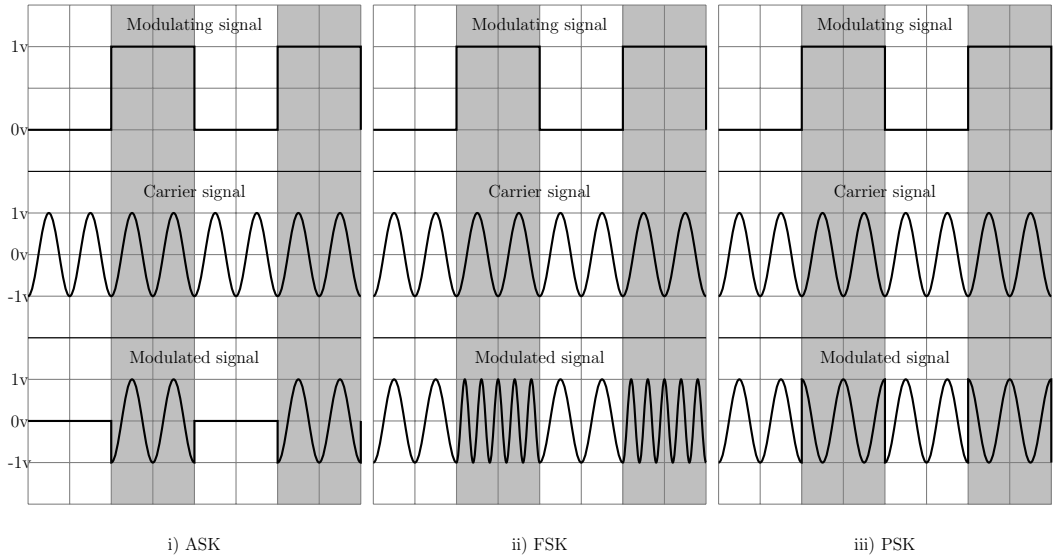


Figure 2.3: Figure showing difference in i) amplitude shift keying (ASK), ii) frequency shift keying (FSK), and iii) phase shift keying (PSK)

In amplitude shift keying, the amplitude is changed depending on the data that is being sent. In the example found in figure 2.3.i, an example of binary amplitude shift keying can be found. The reason this is considered binary, has to do with the fact that there are two levels: either there is a signal (maximum amplitude)

or no signal (no amplitude). You could also have multiple levels of amplitude, for example at 0V, 0.3V, 0.6V, and 1V. Using this, you could encode multiple bits at once, e.g. 00, 01, 10 and 11, respectively. This way, the signal can carry more information. However, due to noise, amplitude shift keying is very vulnerable: noise usually affects the strength and thus it is quite hard to differentiate between bits.

In frequency shift keying, the frequency of a signal is modified to transmit data. For example (figure 2.3.ii), if one wants a transmit a 0 then a low frequency is used; when sending a 1, a high frequency is used. However, the main issue with frequency shift keying is that it uses different frequencies: which is exactly what is a limited resource in wireless communication.

In phase shift keying, the phase of the signal is changed depending on the modulating signal (the data). The example found in figure 2.3.iii shows a binary PSK (BPSK - or 2-PSK): the signal is changed 180 degrees depending on a one or a zero. However, higher orders of PSK are also possible. In order to transmit an n number of symbols at the same time, one needs 2^n points on the phase circle; which means that there is a change of $\frac{360}{2^n}$ in phase between the points. This can be shown by the following formulas for 2-PSK (1 symbol), 4-PSK (2 symbols) and 8-PSK (3 symbols):

$$PSK_2 = 2^1 = 2 \Rightarrow \frac{360}{2} = 180$$

$$PSK_4 = 2^2 = 4 \Rightarrow \frac{360}{4} = 90$$

$$PSK_8 = 2^3 = 8 \Rightarrow \frac{360}{8} = 45$$

The last noteworthy modulation technique is a combination between ASK and PSK: amplitude phase shift keying (APSK). This is essentially combining the two: there is a phase shift; but during each phase shift there can also be a change in amplitude - thus heavily increasing the number of bits that can be sent without using excessive energy usage. For example, techniques exist to support up to 256-APSK, which is transmitting 7 symbols at a time.

2.1.3 Line-of-sight and influences

In order to talk about the influences that impact wireless transmissions, one must first discuss the concept of *(non-)line-of-sight* ([36], p. 115). In a perfect scenario,

there are no obstacles between the transmitting node and a receiving node and no obstacles that can reflect the signal. In this case, the transmission (radio waves) from the transmitting node can go directly to the receiving node with enough strength. This criterion alone is enough to define line-of-sight: if a direct line can be drawn between two nodes, they are in line-of-sight of one another.

Furthermore, as there is no way the signals are in any way changed in direction, the receiving node will receive the signal once and only once; not multiple times in different strength over time due to signals being changed in direction. The concept of receiving the same signal multiple times with a fluctuating strength over time is known as *shadowing*.

Now, to discuss the influences that can impact wireless transmissions, the following five are defined: *absorption*, *diffraction*, *reflection*, *refraction*, and *scattering* ([36], p. 115). *Absorption* is when a signal comes in contact with a material, which then converts the energy of the signal into heat - thus greatly reducing the signal strength (often completely consuming all energy).

Diffraction happens when a wireless signal finds a (big) obstacle and needs to travel around it: the strength and direction of the signal both change, often introducing the aforementioned *shadowing*.

Reflection is when a signal hits a (solid) material, such as a metal, and changes direction (and sometimes strength, depending on the reflectiveness of the material). Like *diffraction*, this can also result in *shadowing*.

Refraction is when a signal enters a different medium (for example, from air to water), which results in a bending of the wave (thus a change in direction).

Scattering is the most unpredictable of one, yet happens quite frequently: when the waveform hits a small object (such as dust), then the signal "scatters". This means that the signal is split into different directions, greatly decreasing the integrity and signal strength.
Chapre *et al.* [37] showed that human presence does affect signal strength due to it consisting of water for more than 50%, thus essentially absorbing the signal. Furthermore, people wear clothes and jewelry that often affects the signal as well.

2.1.4 Strength (RSSI)

This signal strength is an important metric in the connectivity and performance of networks. Strengths are expressed in dBm, as this is much more readable than the alternatives (for example, -60 dBm is 0.000000001 W, or $1 * 10^{-9}$ W). Within WLAN (IEEE 802.11 variants), there are different values for the minimum and maximum strength, depending on the variant [38]. In general, the minimum received signal power is around -100 dBm (0.1 pW, $1 * 10^{-13}$W). The maximum received signal power is described within these regulations around -10 dBm (100 µW, $1 * 10^{-4}$ W) [38]. Due to the low amount of power of the signals, all of the dBm values are negative; therefore, it is important to note that signals are stronger when their strength is closer to 0.

An important indicator to the signal strength on the receiving side is the calculated RSSI (Received Signal Strength Indicator). However, the values are arbitrary, that is, every wireless networking card can use a different measurement scheme. For example, some RSSI may be represented in a value between 0 and 100, some from -100 to 0 and others from 0 to 127 (or even 128) [38].

2.1.5 Signal-to-noise (SNR)

The quality of a signal is usually described in terms of noise and interference: it is how much noise and interference there is between the transmitting node and the receiving node. Therefore, the signal strength is also used to determine the quality of a signal. However, the received signal strength can also be used (together with measured noise) to determine the signal-to-noise ratio (SNR) ([36], p. 41,123):

$$SNR = \frac{P_{signal}}{P_{noise}} \Rightarrow SNR_{db} = 10log(\frac{P_{signal}}{P_{noise}})$$

2.1.6 Channel state information (CSI)

In current wireless communication, the transmitted data is often multiplexed. This allows for higher data rates, as multiple data streams (subcarriers) are available over the same frequency band at the same time. This means overall capacity of the link. The channel state information (CSI) is essentially a transmission matrix between receiving and transmitting antennas. The elements in the matrix

contain complex numbers describing the propagation from antenna N to antenna M, and these complex numbers describe the amplitude and phase variation. By capturing subsequent packets and logging the CSI, the propagation contains the link information between all antennas and thus essentially contain the combined effect of the described influences on the wireless signal over the subcarriers (absorptions, diffraction, reflection, refraction and scattering). Therefore, the CSI thus contains more information than the RSSI: RSSI merely contains the cumulative signal strength, whereas the CSI combines all the information of the path between two antennas.

CSI is essential in multiple-input and multiple-output (MIMO) systems, which are system that contain multiple transmission and multiple receiver antennas. The CSI is used to dynamically adapt either antenna to ensure reliable communication, especially in high data rate systems.

As mentioned before, between multiple receiving and transmitting antennas, multiple data streams are multiplexed. This data is then stored in a transmission matrix, which is an TxR matrix, where T and R are the number of transmitting and receiving antennas, respectively. This creates the following transmission matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1(T-1)} & h_{1T} \\ h_{21} & h_{22} & \dots & h_{2(T-1)} & h_{2T} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{(R-1)1} & h_{(R-1)2} & \dots & h_{(R-1)(T-1)} & h_{(R-1)T} \\ h_{R1} & h_{R2} & \dots & h_{R(T-1)} & h_{RT} \end{bmatrix}$$

where h_{rt} is the link between receiving antenna $r \in R$ and transmitting antenna $t \in T$. Thus, as an example, h_{21} is the channel information between transmitting antenna 1 and receiving antenna 2. The reason the receiving antenna is listed first has to do with the fact that it is the receiving antenna that estimates the channel link information. However, not only the CSI is measured; the noise is also measured and represented by noise vector $\mathbf{n}$. So, if one also considers the input and output vectors, respectively $\mathbf{x}$ and $\mathbf{y}$, one gets the following combination of vectors (with $\mathbf{H}$ shortened):

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{(R-1)} \\ y_R \end{bmatrix} \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_{(T-1)} \\ x_T \end{bmatrix} \mathbf{H} = \begin{bmatrix} h_{11} & \dots & h_{1T} \\ \vdots & \ddots & \vdots \\ h_{R1} & \dots & h_{RT} \end{bmatrix} \mathbf{n} = \begin{bmatrix} n_1 \\ n_2 \\ \dots \\ n_{(T-1)} \\ n_T \end{bmatrix}$$

which can be combined into the expression:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$$

The onboard chip calculates these linear expressions. However, it is nice to understand how these equations are solved. Therefore, as an example, for a $2x2$ MIMO setup this becomes:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}$$

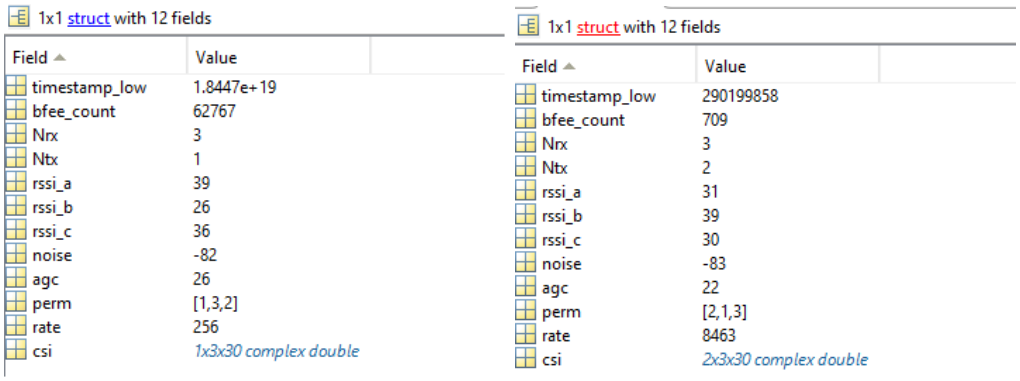
$$y_1 = h_{11}x_1 + h_{12}x_2 + n_1$$

$$y_2 = h_{21}x_2 + h_{22}x_2 + n_2$$

2.1.7 Exploring CSI frames and graphs

In this section, a quick look will be taken at the CSI frames and traces collected by the Linux CSI Tool [39], and more information can be found on their FAQ. Most information here is based on the given information there. All the data collected in a CSI packet can be seen in Figure 2.4. This figure is also a reference point for the explanation in the remainder of this section.

Starting with the easier components of the collected CSI traces: `Nrx` and `Rtx` are the number of receiving antennas and transmitting antennas, respectively. Connected to these two is the `perm`, which stands for permutation. This shows which signal was sent to which chain in order to process the measurements, thus the permutation of these antennas. In the example of Figure 2.4a, the permutation states $[1, 3, 2]$, implying that the first antenna was sent to the first chain, the second one on to the third chain and the third one was fed into the second chain. However, in Figure 2.4b, the permutation is $[2, 1, 3]$, thus having the first antenna going into



(a) One transmitter (b) Two transmitters

Figure 2.4: Examples of CSI packets collected during experiments

the second chain, the second into the first and the third going into the third chain. This is picked by the NIC itself, and is used in order to reduce RSSI. Speaking of RSSI, in the data three RSSI values can be found, namely `rssi_a`, `rssi_b`, `rssi_c`. These are the received RSSI values received at the three antennas, A, B, and C.

Moving a little away from the physical antennas, but still related to the RSSI, one can find the `noise` and `agc`. While noise is quite obvious, as it is the actual noise measured on the signal in dB, AGC is a bit harder to explain. AGC is a feedback circuit of (a chain of) amplifiers which tries to maintain a suitable and stable signal between the sender and the receiver. In order to get the actual dBm (as the values shown in the data are relative for the Intel chip), one most combine the RSSI values, noise and AGC. Luckily, the Linux CSI Tool provides supplemantary files, one of which is a MATLAB function called `get_scaled_csi(csi_entry)` which does precisely that.

With the signal information out of the way, there are only two actual values left (excluding the CSI matrix): `timestamp_low` and `bfee_count`. However, either of these are not used in this research. The first of these, `timestamp_low`, is the lower 32-bit count of the internal 1 MHz clock of the NIC, which wraps about every 4300 seconds. For this research, it would have been more useful if it was a timestamp of the moment it was captured, synchronized with the real time. The other one, `bfee_count`, is a count of the total number of beamforming measurements and also not used in this research.

The final piece of data is the actual CSI, which is aptly stored in `csi`. This is

```
val(:,:,1) =

   3.0000 +11.0000i  -3.0000 -36.0000i  -8.0000 +13.0000i
   3.0000 +14.0000i   5.0000 + 5.0000i  -6.0000 + 2.0000i


val(:,:,2) =

  11.0000 + 8.0000i -29.0000 -29.0000i   2.0000 +17.0000i
  13.0000 +10.0000i   8.0000 + 0.0000i  -4.0000 + 6.0000i


val(:,:,3) =

  15.0000 + 0.0000i -45.0000 - 5.0000i  13.0000 +14.0000i
  17.0000 - 1.0000i   6.0000 - 7.0000i   0.0000 + 8.0000i
```

Figure 2.5: H matrix in `MATLAB`

essentially the H matrix as described in the matrix (section 2.1.6). In its purest form, it shows the complex formula to describe the channel state information between the antenna pairs across 30 subcarriers, as can be seen in Figure 2.5. These values can be transformed into workable values using the `abs()` function in `MATLAB`, as it returns the complex magnitude (or modulus) of the value and by using the `get_scaled_csi()` function described previously. An example of the SNR can be found in Figure 2.6. As can be seen, for each receiving antenna two lines are plotted: this can be explained as the data from which this was generated is a $2x3x30$ CSI matrix, thus meaning there are two transmitting antennas (as in Figure 2.4b). This means that every antenna receives two signals.

However, all CSI traces can be analysed together per subcarrier. This gives an overview of how the CSI changes over the collected CSI frames and this gives an insight in how activities can affect the CSI data per subcarrier. An example of this can be found in Figure 2.7. Furthermore, in Appendix A one can find an example for all 30 subcarriers at once.

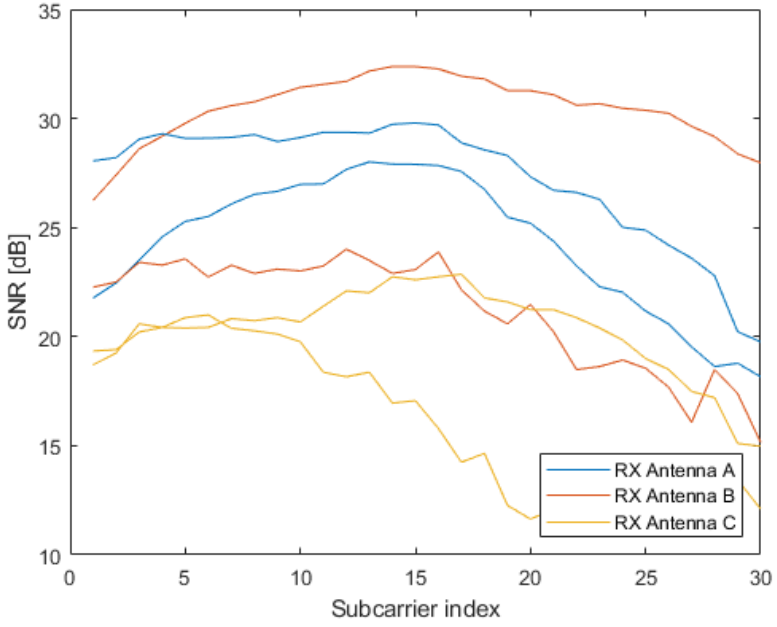


Figure 2.6: SNR plotted across the subcarriers. Two sets are plotted as there are two transmitting antennas

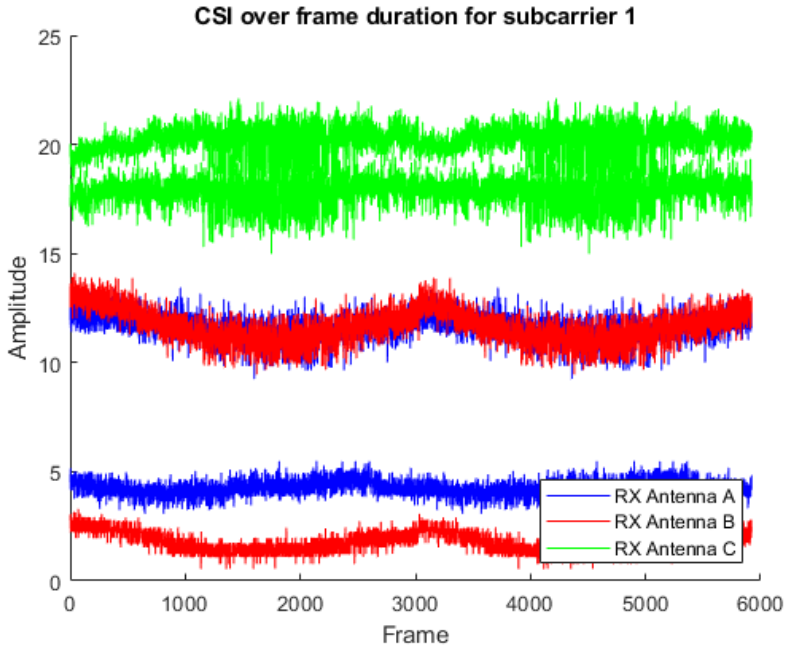


Figure 2.7: Amplitude of the signal plotted over the duration of the frames for subcarrier 1

2.2 Deep learning

Within the field of Machine Learning, there is a subcategory of algorithms called deep learning. Whereas machine learning is usually expert-tuned and needs to be trained using structured data, deep learning can survive on only tuning hyperparameters and unstructured data. Deep learning has become a more widely used technique recently, even though the concept originates from the 1980s: neural networks. A neural network (or deep network) is a network in which several layers of nodes that are connected that, together with weights and activation functions, dampen or strengthen certain outputs, in which ultimately a classification is reached in the end. Unlike classical machine learning approaches, in deep learning one cannot backtrack which choices the deep learning network made.

Within the deep learning field, two big modifications of a regular neural network exist: convolutional neural networks (CNN) and recurrent neural networks (RNN). These two networks are both neural networks, but have different characteristics: CNN share parameters over different classifications, thus making them great for recognizing patterns within different sets of data (making it great for image and audio recognition, as well as signal processing and detecting wireless interferences), whereas RNN share parameters over time, thus making them great for seeing patterns that repeat (making it in turn great for speech and text recognition).

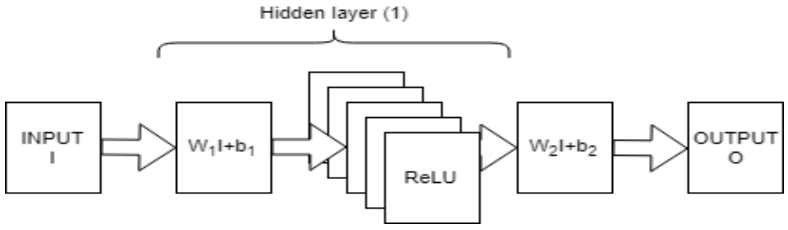


Figure 2.8: Showing a wider model, thus with more ReLU at one place

Neural networks are built from linear operations, which have multiple advantages. They are very efficient and hardware makes it possible to execute these faster and more efficient every day. Linear operations are also quite stable, in a way that a small change in the input of a function cannot result in a huge change in the output (as is the case with exponential functions, for example). Even their derivative is very stable: it is a constant. Therefore, deep learning models are still

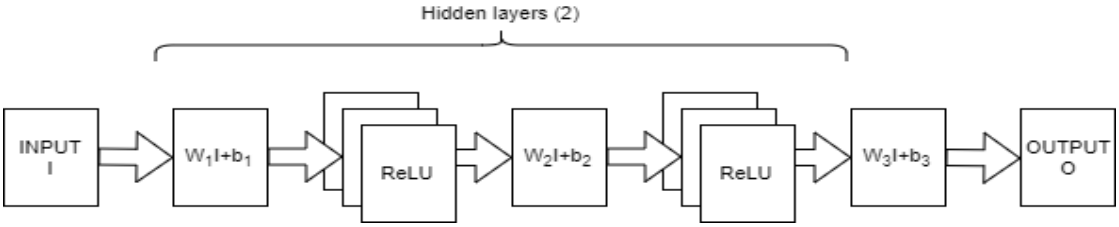


Figure 2.9: Showing a deeper model, thus with more (hidden) layers

based on these linear models and functions, but have an H amount of rectified linear units (ReLU) between them and the linear operations (which are at least 2: *2-Layer Neural Network*. However, increasing H (widening) does not result in a big difference when it comes to classification: it is usually better to apply more linear operations with rectified linear functions between them in a model (deepening). To make the difference clearer, two pictures were constructed to show the differences: Figure 2.8 (wider) and Figure 2.9 (deeper). In these pictures, also the hidden layers[2] are shown.

For this research, several terminologies need to be put on the table regarding deep learning; these will be discussed below.

2.2.1 Softmax

Deep learning and neural networks do not classify objects with a 100% certainty. Rather, they give an estimation how likely it is that an object belongs in a certain class. However, the neural network does not guarantee that the values are understandable, whereas it would be more preferable when all the probabilities sum to 1. Therefore, every neural net ends with a `softmax` layer: a function that takes an input vector $\mathbf{X}$, which is created by the neural net, and transforms it to a probability vector $\mathbf{P}$, which is a vector of which the elements sum to 1. This is done by solving the following equation, the softmax equation:

$$\mathbf{P} = \frac{e^{\mathbf{X}}}{\sum e^{\mathbf{X}}}$$

[2]A hidden layer is a combination of a linear operation (thus the weights and biases) and a ReLU, but only when the output is hidden to the outside of the network

where e^X is the exponential value of the elements in X, $\sum e^{\mathbf{X}}$ the sum of all these exponential values and $\mathbf{P}$ the prediction (likelihood of which value it is) of $\mathbf{X}$ (which now assures $\sum \mathbf{P} = 1$).

2.2.2 Cross entropy and (stochastic) gradient descent

Cross entropy is the distance between to vectors, and can be used to learn more about the weights and bias, that is, one can evaluate the cross entropy over different weights and biases and the one with the smallest loss (which is the average cross entropy over different vectors) are probably good weights and biases. The formula for solving the cross entropy is:

$$D(\mathbf{P}, \mathbf{L}) = -\sum_i \log(S_i)$$

where $\mathbf{P}$ is the probability vector and $\mathbf{L}$ the vector containing the labels and thus the truth.

However, instead of doing this randomly (with trial-and-error), one can apply gradient descent to find the bottom of these losses as a function over several parameters (for example, the weight and the bias).

There is also such a thing as stochastic gradient descent; the difference between stochastic gradient descent and the regular one, is that it uses a small, random portion of the data (between one and a thousand data points) to calculate the gradient descent. As this is a poor estimation of the actual gradient descent, a lot more steps are required to do this. However, each step costs a lot less than with actual gradient descent, so overall this process is a lot faster and it scales a lot better than the original method.

2.2.3 ReLU

A rectified linear unit, also known as an activation function, is a function that makes sure that values below 0 become 0 and values above 0 stay the same.

2.2.4 Forward- and Back-propagation

To use SGD in deep learning models, one should require a way to automatically update all parameters in the network. Note that only the linear operations (neu-

rons) have parameters (the weights), the ReLU (activation functions) do not. This can be done through forward- and back-propagation.

Forward-propagation is running the network (for example, with training data) as it is meant to run, from inputting data to the prediction of the label, with the initial weights w_0^0, w_1^0, ..., w_{n-1}^0, w_n^0. This is likely to give a bad estimation, as it is the first run and the initial weights are picked using a Gaussian distribution. This is where back-prop(agation) becomes handy.

Back-propagation is essentially running the network backwards and by differentiating the network using the chain rule [40]. The goal for this is to calculate a gradient (δw_0^0, δw_1^0, ..., δw_{n-1}^0, δw_n^0) for the (initial) weights. Then, using SGD, one can try to find a minimum depending on this gradient, as the new weights are calculated as $w_0^1 = w_0^0 - \delta w_0^0$, $w_1^1 = w_1^0 - \delta w_1^0$, ..., $w_{n-1}^1 = w_{n-1}^0 - \delta w_{n-1}^0$, $w_n^1 = w_n^0 - \delta w_n^0$. Doing it in this way means the entire network is updated at once and thus, the entire network is optimized.

To continue the example to make it clearer, after the first back-propagation is completed, the network is run again (as forward-propagation) with the new weights w_0^1, w_1^1, ..., w_{n-1}^1, w_n^1. Then the new gradients are calculated during the following back-prop again and the new weights are once again calculated as $w_0^2 = w_0^1 - \delta w_0^1$, $w_1^2 = w_1^1 - \delta w_1^1$, ..., $w_{n-1}^2 = w_{n-1}^1 - \delta w_{n-1}^1$, $w_n^2 = w_n^1 - \delta w_n^1$. Then the process is repeated once more.

2.2.5 Regularization and dropout

As deep networks can grow quite big, it becomes harder and harder to optimize them as the number of parameters to tune grow and thus overfitting becomes a threat. A way to prevent this is by applying regularization: one restricts the number of parameters. This is usually done in deep learning by punishing large weights, unless they offer a real contribution to the network. This is done by adding the following formula to the loss (during SGD):

$$\frac{\lambda}{2n} \sum_w w^2$$

which essentially sums the squares of all weights (w) in the model and then multiplies this by λ divided by double the data size (n). This shows that the network will edge towards picking the smaller weights. λ can essentially hold two different

kind of values: a small value (close to keeping the original weights) and a large value (preferring smaller weights).

Another regularization technique is dropout [41], which is different from L2 regularization in that it is not related to the weights, but related to modifying the network itself. Suppose a neuron fires activates four different neurons in the next layer. What dropout does, is randomly resetting half of those signals (disabling half the neurons in the next layer): thus, instead of learning a single network, one learns several (more robust) smaller networks. It should be noted that due to half of the nodes being activated, in the full network, the found weights and biases should be halved as well.

2.2.6 Convolutional Neural Networks (CNN)

Now, to delve deeper into neural networks, one can learn different types of neural networks that are more powerful for certain situations. An example of this is the Convolutional Neural Network [42]. These are especially useful in image recognition, in that they learn to find objects in an image, independent of the location of this object in the image.

Statistical invariance

To understand the reasoning to use convolutional neural networks, one can take a step back: how does a regular neural network (DNN) learn about different images? Imagine one has two images, both with an object y, but with different positions (p_1 and p_2). If one wants to teach this regular DNN to recognize y in both these pictures - it must feed the network a lot of similar pictures where y is around the position of p_1 and p_2 and therefore creating different weights for each input. However, what one would like to do is to share weights for object y and recognize this object y independent of a position. This is also known as statistical invariance: some things do not change over time (object y will remain object y, independent of its location).

Feature maps and layers

The idea behind a convolution neural network, is that a big picture with $depth = 1$ (thus a single picture) is reduced to several smaller pictures ($depth = K$) that when

put on top of each other, still paint the same image but smaller and with more layers. The depth of the picture is also known as the amount of features a network will learn, and a single one of these features is called a feature map. Thus, in the previous example, a 1-feature map is mapped onto a K feature map. The goal of these convolutions (thus turning an image into a stack of images), is that it does not matter where an object is on the picture: eventually (once made small enough), the features of this object will still shine through.

In convolution neural networks, there are four different aspects, which are all quite important:

Convolution

A convolution is mapping an N feature map onto a K feature map (where $K < N$). This is essentially done by scanning over the picture using a patch[3] of a fixed size (dependent on how big the images are) and with known weights (weights implying features such as a diagonal or straight line). This is done for every possible match. Putting the scores from the different feature maps on top of each other will usually paint something like the original picture.

Pooling

Pooling is done to shrink an actual feature map: that is, a set of x by x pixels are squeezed into a single pixel. This is usually done through `max pooling` (taking the maximum value of these pixels and noting that) or `average pooling` (taking the average value of these pixels and noting that). This creates a smaller, vaguer image of the original one before pooling. This makes it so that pictures can be slightly translated or rotated, and the pooling will still pick it up.

Normalization

This is essentially eliminating useless values so that the mathematics become easier: change all negative values to 0.

[3] Also known as a kernel

Fully-connected layers

Fully-connected layers come at the end of the convolutional neural network: they change the convoluted pixels (that no longer actually represent the original image, but the features of this image) to a list of votes. In training, these votes determine the weights which vote is important to predict which label; in validation/testing/deploying, these weights in turn predict the label by averaging the score and the highest score is the likely label.

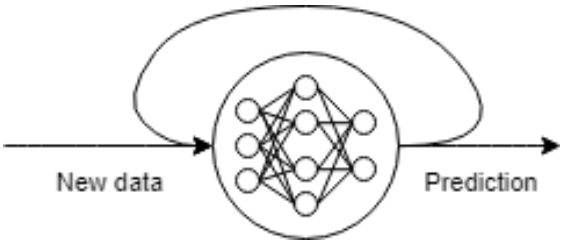


Figure 2.10: Showing the principle of a neural network

2.2.7 Recurrent Neural Networks (RNN)

Whereas Convolutional Neural Networks share parameters over space, for memory one would like to adapt this ability over time. Luckily, such a network exists: the recurrent neural networks (RNNs).

RNNs have two inputs: the past (or some information about the past) and new data. Combining these, they learn to make choices keeping the past in mind. However, a default RNN has several problems, as will be discussed in the next section about vanishing and exploding gradients.

Overall, an RNN is just a feedback loop that feeds into itself again after it makes a decision. This is shown in Figure 2.10.

2.2.8 Vanishing / Exploding Gradients

When it comes to computing the weights through SGD, this becomes troublesome for recurrent neural networks: as the derivatives need to be taken back in time (to the beginning or at least some point in the past), which results in correlated

updates for the same weight. This is troublesome for SGD, as this makes the gradient either go to infinity (exploding) or go to zero (vanishing).

Fixing these is quite easy for the exploding gradients is quite easy, as one can simply apply `gradient clipping`: given a maximum value max, if $x < max$ then x keeps its value, but if $x > max$ then $x = max$. This prevents gradient from exploding.

Fixing the vanishing gradient is harder. First of all, vanishing gradients equal memory loss: if a gradient becomes 0, then it has no information to remember and it will not learn anything. This is fixed by using an LSTM unit, as explained in the next section.

2.2.9 Long Short-Term Memory (LSTM)

A Long Short-Term Memory [43] (LSTM) unit is a building block for recurrent neural networks and essentially adds a memory block to the network. One can view memory as a block with three gates: read, write and erase. The values of these gates can be either 0 (closed) or 1 (open). Depending on the gate, memory can be written, read and erased.

The main difference between regular memory and an LSTM is that rather than discrete values (0 or 1), the gates are controlled by a continuous function with values between 0 and 1. The most important part of this is that these functions are differentiable and thus back-propagation throughout the network is possible. Thus, even though a memory cell can only remember a small piece of recent information (short-term memory), through back-propagation more can be learnt about the past - thus creating "long" short-term memory.

The gating values for these three gates are controlled by their own tiny neural networks and are learnt through training the network.

Chapter 3

Related works

3.1 State-of-the-art

3.1.1 Unobtrusive human activity classification through device-free sensing

The focus of research in wireless networks has shifted to device-free sensing: detecting wireless disturbances in an environment to classify and recognize human activities. This method is known as "device-free sensing". The first examples from this come from the early 2010s, like research conducted by Fang *et al.* [33] and Gu *et al.* [8].

Fang *et al.* used the data available from a smart home (such as temperature, motion and doors opening/closing) to classify what a person was doing (bed, toilet, breakfast, computer, dinner, laundry, outdoor, lunch, taking medicine) by using the available values. These values were fed to an RBM to classify, and later compared to more classical machine learning approaches like HMM and Naive Bayes. On the other hand, Gu *et al.* used an access point (transmitter) and laptop (receiver) to analyse the received signal strength (RSS) of the WiFi (2.4 GHz, IEEE 802.11b) signal and other ambient WiFi signals. The focus was on the basic activities, such as standing, sitting and walking. The algorithm used to classify the data was a k-NN algorithm.

Then, in 2016, Li *et al.* [6] looked to modify an access point to use the channel state information (CSI) of the wireless local area networks (WLANs) to classify

the activities walking, sitting, lying, standing, squatting, falling and crawling in order to support the elderly. It also used the 2.4 GHz frequency, but instead used a Random Forest classifier. The access point was modified in such a way that it could gather the CSI from nodes (other devices) in the room and based on the changing state, the features would change. Thanks to this, the system was robust in both line-of-sight as well as non-line-of-sight situations.

Later, in 2017, Huang *et al.* [5] used four corner nodes and an access point to track humans in a room based on their height: the access points would transmit data to the access point over different frequencies (2.41 to 2.49 GHz) and the packet receive rates (PRR) was the feature extracted to label the data on and an SMO and k-NN were trained and compared. Also in 2017, Wang *et al.* [20] used eight nodes and a laptop to localize people, as well as classifying four activities and four gestures. The system would place eight nodes around a room and connect these nodes with one-another, then the LOS properties, as well the RSSI was used to train a deep learning model and classify the activities. Another interesting development in 2017, was the use of a radio waves as a radar, proposed by Haider *et al.* [4]. A transceiver was placed in a room and would transmit an electromagnetic wave in the 3.3 to 10.3 GHz range. The signal would interact with the environment (and thus be reflected or broken) and the transceiver would pick up the signal again and based on this learn. However, as this was a proposed system, no actual learning techniques were discussed. Lastly in 2017, Murad *et al.* [2] deployed a recurrent neural network (RNN) with LSTM and tested it on benchmarked data (much like Zheng *et al.*). The evaluated activities included opening doors, opening fridge, opening dishwasher, opening drawers, cleaning table and toggling a switch.

However, the current state of the art shows how much research is going into the field of device-free sensing when looking at what has been concluded in the first few months of 2018 [44][45][14][15].

Booranawong *et al.* [44] proposed an algorithm to track and detect human movements. This proposed idea would focus on walking and moving in general and required a single base node, one receiving and three transmitting nodes that would use the 2.4 GHz frequency. All nodes were placed on the same altitude and the base station would look at the RSSI of each transmitting node. However, there was no machine learning involved, as the proposed algorithm used a threshold to detect movement.

Guo *et al.* [45] developed a hybrid system based on the 2.4 GHz (IEEE 802.11n) frequency to analyse the RSSI and CSI (and compare the two), as well as adding skeleton data by using a Kinect (image analysis). A total of sixteen activities were classified using k-NN, random forest and decision tree techniques. This showed that combining both methods greatly improved the accuracy of the system.

Han *et al.* [14] looked at a passive way to detect humans: CSI fingerprints in a room. This method used a receiving node (laptop, as the receiver) and looked at the CSI and RSSI, and was made robust in a NLOS scenario by using a sub-carrier matrix (of 30 subcarriers). The use of multiple antennas was also tested and increased the performance. The algorithm used was a self-developed voting algorithm to vote on which fingerprint it was.

Lastly, Shah *et al.* [15] looked at the medical world and focused on narcoplectic patients, detecting sleep attacks and sleepiness. The proposed method consisted of a receiver and transmitter that used the 2.4 GHz channel state information (CSI) to calculate the phase and amplitude, as well as using the S-Band Sensing technique.

With deep learning, Liu *et al.* [46] proposed a deep believe learning (DBL) based algorithm to successfully identify critical and weak links, which could in turn result in an optimization of networks. This was done by evaluating the link states and learning from this.

Earlier this year as well, Dang *et al.* [10] proposed a Kalman filtered CSI and PCA (principal component analysis) solution in LOS, NLOS and wall environment experiments and are compared. They used the classical machine learning technique SVM and the gathered data is matched against the data in the fingerprint database and they show an accuracy of 95%, while claiming to have the best algorithm in terms of average error and indoor activity recognition accuracy.

[11] *et al.* proposed an activity recognition algorithm that can be ran on commodity Wi-Fi enabled IoT devices, thus enables the cost for dedicated devices. Their designed a novel OpenWrt-based IoT platform to collect their CSI and used an RNN to recognize and classify the data. Their results show a 97.6% accuracy for 10 volunteers. The devices used are two routers, which are still quite large compared to the potential devices. Also, a convolutional network can potentially achieve higher accuracy.

Furthermore, [16] *et al.* also devised a system based on the CSI, using the

Linux CSI tool, and first use feature extraction and PCA to clean the data. They used the k=NN classifier to classify the activities in a LOS environment. There were three volunteers to gather and test data and to test against. While they have shown that the Linux CSI tool can correctly be used to gather the data, their system still needs manual feature extraction and only has three volunteers.

[**?**] *et al.* proposed a new system based on HMM. Their research focused on how to estimate the signal propagation in an adaptive complex environment. They proposed a new ambience identification method, called Ambience Sensor (Asor), in order to improve the performance of the applications on top. Furthermore, they integrated Asor into a localization method called Aloc) and claim their method is superior when it comes to the median detection rate of propagation ambience.

[18] *et al.* developed a device-free people counting system, based on CSI. The CSI makes sure the system is a non-image based counting system, thus meaning that no actual pictures of humans are being used, but rather the CSI data is analysed through a DNN. Their testbed showed an accuracy of 88% on average when it came to estimating a crowd size up to nine people.

Lastly, [19] Wang *et al.* proposed a system based on 5 GHz CSI data to create an indoor localization system. The images used were that of the angle of arrival (AOA), which were gathered in the online phase. During the offline phase, a DCNN was trained to localize the humans in two representative environments. The devices used are an access point and a laptop, both equipped with the NIC 5300, thus implying the Linux CSI Tool was used.

3.1.2 Analysing networks and their disturbances

To start with analysing networks using deep learning, the importance to this should be clear: as an attempt is made to classify human activity through measuring these networks, it is important to know if deep learning can be used for this. Luckily, Kulin *et al.* [35] has proven that deep learning (and specifically CNNs) can be used to detect interferences in wireless networks and can be improved using these.

Secondly, Wang *et al.* [47] (2018) has shown that more information can be gathered from signals (and increasing the accuracy of NLOS systems) by looking at the spatial information of the environment. This was done by providing structure blocks and a coherent histogram.

Also, Avci *et al.* [48] has shown that a CNN can be used to analyse structural health monitoring (SHM), and especially in the damage detection (SDD).

Choi *et al.* [13] proposed an RNN with LSTM block to detect NLOS and LOS based on the channel state information (CSI) of links, as well as the received signal strength indication (RSSI) in a cross-layer manner to accurately figure out whether a wireless device was in direct line-of-sight or not.

3.2 Discussion

First of all, device-free solutions seem to be quite useful in the tracking of certain (heavily-moving) activities and can even be used to track and identify multiple people in a single room. This is likely due to signals being redirected differently depending on the differences in physical appearances and the number of individuals within a room. As device-free solutions often monitor changes in areas, rather than monitoring changes on a physical level, these things can be identified by device-free solutions. However, device-free solutions seem to lack in the medical field. This is as they cannot (yet) be used to analyse things such as temperature, heart rate and oxygen levels of the individual beings.

The activities recognized in device-free solutions are the activities that heavily influence wireless signals: walking, running, falling, sitting. This is likely due to these signals heavily influences the environment and causes the wireless signals to refract, redirect and scatter in different ways, as the body changes the entire environment. The main focus here is found to be on walking (and accompanying basic movements, as standing and sitting), as these require moving around a room and thus consistently changing the aspects of the transmitted signals. There seems to be a minor focus of things such as walking stairs, falling/crawling and chores. This can be explained by two things: certain activities can only happen in certain areas and some activities do not do enough to change the environment.

The former are, for example, more isolated locations, in which activities occur but cannot be detected by device-free solutions due to the wireless signal not being redirected back to the receiver again. This could explain the lack of "stair walking" when it comes to device-free solutions: a stair is usually located in an isolated space (a hallway) where signals may not be redirected back to the receiver again. This does not mean that no wireless connection is available anymore, it merely implies

a signal transmitted from another room cannot bounce back to the original room anymore (either due to losing its strength or "getting stuck"). This problem is closely related to (non-)line-of-sight, but not directly. Research has shown that (non-)line-of-sight solution do exist, one must identify that "line-of-sight" in these occasions mean the line-of-sight between nodes, and not the line-of-sight between node and the occurring activity.

The latter of the issues related to a lack of interest in certain activities, can be contributed to the fact that certain activities do not change the environment enough. Going back to examples mentioned in the wearable sensor solution, typing on a computer, switching TV channels or reading a book do not impact the environment as much, in the sense that these chores could be potentially be identified by device-free solutions. This is quite possibly dependent on the location of the listening node: if it close to the activity, it may pick up more disturbances than when it is further away from the activity.

When comparing the tools used to measure the wireless environment, there seems to be a focus on using access points, laptops/PCs and node-based systems. Access points are often modified to measure these disturbances as they are already located and operation, and can contact any node (laptop, smartphone, etc) and get the information of the signal strengths and channel state from these nodes. They are therefore an easy solution to detecting the disturbances. However, modifying the access points is not always desired or even allowed, as these are usually already distributed among homes by manufacturers. As mentioned before, laptops/PCs are used to analyse signals as well, this is likely due laptop/PCs being quite powerful and personal, in the sense that researchers can change their own configuration settings and have enough computing power to analyse these disturbances. Lastly, node-based systems allow researchers to explicitly program what is being transmitted and analysed and to which nodes, resulting in a strong control over the environment and making it possible to collect data and information potentially unobtainable for access points or laptops/PCs.

Now, a single base station and transceiver options seem to be less popular when looking at the comparison table. A single base station likely has the downside of being located in a specific spot and not being able to access any of the mobile nodes (smartphones, laptops, etc) and can thus essentially only measure the immediate wireless environment, potentially losing out on information. A single base station

could also, potentially, lack the computing power to run and calculate neural networks. The other lacking solution, transceivers, are likely due to them requiring them to transmit their own signal (resulting in extra power consumption) and then basing their observations on the returned signal strengths and directions, essentially creating a radar system. There is a massive downside to this, as will be discussed in the next section, which is that it is a LOS-based solution.

Moving onto the gathered data and information, a look will first be taken at LOS/NLOS. It seems that both NLOS and LOS systems are being equally researched, with only a few hybrid (N)LOS systems. The aforementioned radar system, is such a LOS system: it requires line-of-sight with an activity in order to detect this activity. This is a characteristic of radar systems in general. However, in the more classical sense of LOS, it requires one node to see another node. This can be seen in the research as well, as the access points require a NLOS with their nodes (they communicate regardless of NLOS), whereas node-based solution require LOS (that is, they require to view each other). Using LOS systems has several upsides: the interlink connection between the nodes can be analysed and used to classify behaviour, whereas NLOS solutions can only base their information on the signal connection with the mobile nodes. While this may sound the same, it is not: fixed node positions that require LOS can gather more information about the position and movement of a person in order to classify, whereas NLOS solutions with mobile nodes do not have the information based on position or direction. They essentially lose out on information, which is also implied by looking at the table: LOS (or hybrid systems) are usually capable of classifying different and more activities. However, the interest in NLOS can be explained when looking at the table as well: NLOS systems are usually the systems that are easier to deploy (as they require a single laptop or access point), rather than deploying a network of nodes across a room.

When looking at the data and information used, the first thing that becomes apparent is that the current solutions do not keep multiple people in a room in mind. That is, no research has gone into the classification of activities of multiple humans in a room. Rather, all research conducted assumes there is only one person in a room. Research done by Han *et al.* [14] focused on identifying personnel based on CSI fingerprints, but not on the actual activities of these people. It is likely that a certain activity has a certain fingerprint (due to the high accuracy

rate of previous research), but it is also likely that multiple people performing the same or different activities will have different activity "fingerprints" and that these fingerprints could thus affect each other, potentially decreasing the accuracy of a system. It is therefore easier to propose solutions that are based on single humans, set in a fixed environment.

It is worth mentioned that, while left out of the comparison table, none of the reviewed work considered a dynamic environment. As the received signal at a node is the sum of the different signals caused by diffraction, reflection, refraction and scattering of the originally transmitted signal, it is likely that the signal is stable in a static environment. That means, as long as the environment does not change, only human interference will cause disturbances in the signals. However, it is likely that a person will interact with the environment and cause a permanent change in the environment: if this happens a lot, one could consider the environment to be "dynamic". Examples of dynamic environments would be warehouses or stores (where products are being moved frequently) or restaurants (where chairs and tables are being moved). While the environment of an elderly home is likely to be more static, changes still occur here: moving tables or chairs, new plants, etc.

The frequencies most used in analysing wireless environments, are the WLAN frequencies, and specifically the 2.4 GHz, rather than the 5 GHz (not specified in the comparison table due to its size). The difference in use of 2.4 GHz and 5 GHz signals is likely that 2.4 GHz signals are still more dominantly used and thus contain more information (essentially, each connection is a source of information). Furthermore, 2.4 GHz are a larger waveform than 5 GHz signals, and can thus travel further distances and are less vulnerable to noise and obstacles. On the other hand, other frequencies (in the range of 3 and 10 GHz) were proposed, but no developed. This was the radar-based system, and for this purpose this range could be used: it would not interfere with the WLAN frequency of 2.4 GHz and as it would just be to measure the rebounded signals, these higher frequencies would have been beneficial.

Looking at the actual performance measurements on which the classification is based, the most dominantly used is the RSSI, followed by the CSI. This is as RSSI is the most accessible performance metric to be gathered as it is contained in almost all packets (as defined in the IEEE 802.11 standard), but is not stable, in the sense

that it can fluctuate even when nothing is happening, due to it being implemented and defined per manufacturer. Therefore, it can at times be disadvantageous to use this metric. The interesting alternative is the CSI, as it contains the information on how a signal propagated (scattering, fading, decay) from the transmitter to the receiver. However, the CSI is a more challenging approach, as it requires the analysis of such CSI packets. This difference can be seen in the comparison table, as it seems the CSI is being used less frequently than the RSSI alternative.

There are also two less popular evaluation techniques, the PRR (packet receive rate) and using the phase/amplitude and direction of a signal. The PRR identifies how often packets are received, rather than the actual signal strength (how fast they are received). As this is not defined in a standard, this can only be used in node-based solutions, in which it is predefined on how often each node sends a packet. This is confirmed by looking at the comparison table, as the only solution that used the PRR is a system based on nodes. The latter of the two required a specific transmitter and receiver, in order to analyse the phase and amplitude of the signal.

Looking at the machine learning techniques, there seems to be a focus on using more classical approaches (k-NN, SVM, Random Forest) than there is a focus on more modern techniques, such as neural networks. This can be explained by it being a newer field and that the more established (and less complex) approaches are being analysed and discovered first. Another reason for the lack of deep learning in this field can be explained by the computational complexity and (lack of) computational power available.

Chapter 4

Hardware

A big part of this research was to design the hardware solution used in this research project. Currently, the state of the art uses laptops, computers or modified routers to collect CSI. In most of these cases, the smaller components are able to send the packets needed to measure the CSI on the main computer, but no small, CSI collecting nodes exist. Therefore, hardware was an important part of the research, as an attempt was made to create the smallest state-of-the-art CSI collecting solutions in order to make it possible to create small, CSI collecting nodes. Several requirements were written down, based on literature and research demand and several solutions were attempted, which will all be discussed in this chapter.

4.1 Requirements

The functional and non-functional requirements are listed below a single table (Table 4.1), starting with the functional and ending with the non-functional requirements, respectively. In these tables, their number (their identifier), their description and their origin can be found. Do note that the requirements were sorted using the MoSCoW method, but they are not listed as such below. Rather, they are written down in the "must, should, could and won't" principle. The origin is where the idea came from: research demand means that it is part of the research or for future research, literature means that the idea originated from the literature and requires knowledge from the literature and restriction is a limitation caused by a specific other requirement. The reasoning behind each requirement can be

Functional requirements		
Number	**Requirement**	**Origin**
FR1	A node must be able to collect CSI data	Research demand
FR2	A node must be able to store data locally for at least a day	Research demand
FR3	The system should be able to connect multiple nodes	Research demand
FR4	A node should function in a headless manner	Research demand
FR5	A node should be able to log its own device status	Research demand
FR6	A central server should be able to collect all data	Research demand
FR7	A node won't do onboard calculations towards deep learning	Research demand
Non-functional requirements		
Number	**Requirement**	**Origin**
NR1	The nodes must be smaller than the current state-of-the-art	Practicality
NR2	The system must use the Linux CSI Tool	Practicality
NR3	The system must run on Linux kernel between 3.2 and 4.2	Restriction (NR2)
NR4	The system must use an Intel Ultimate N Wi-Fi link 5300 NIC	Restriction (NR2)

Table 4.1: Short overview of the (non-)functional requirements

found in Appendix .

4.2 Attempted solutions

First of all, due to the requirements, the possible solutions were very limited, as any kernel version between 3.2 and 4.2 is considered outdated (at the moment of writing, the current Linux kernel version is 4.19). As Ubuntu is the most user-friendly Linux solution, Ubuntu was the targeted operating system. However, given the kernel version, only Ubuntu versions between 12.04 and 14.04.4 could be used. To put things in perspective: the way Ubuntu is named is that it first

denotes the year and then the month of release. Therefore, 14.04 means that it was released back in April 2014. The latest version as of writing is 18.10 (Cosmic Cuttlefish) and was released in October 2018.

It should further be noted that as the project moved forward and development started, it came to light that the regular Ubuntu versions would not work on the first two solutions: the single-board computers (the Raspberry Pi and the Hummingboard), as these use the ARM architecture (ARMv8 and ARMv7) and thus require their own firmware and operating system, usually provided and developed by either the manufacturer or the community (open source). This complicated the development, as will be discussed in the upcoming sections.

Going back to the requirements and looking at the fifth one, this is also a very limiting factor: the Intel Ultimate N Wi-Fi Link 5300 is outdated hardware, first released in 2008 and thus not widely available anymore. In fact, only stores like Amazon and eBay still provided these variants.

In Table 4.2, an overview of the hardware and their specifications can be found.

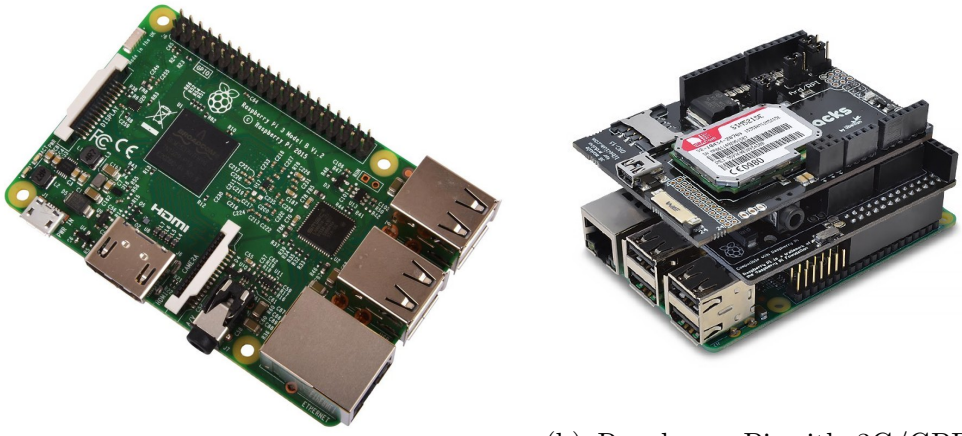
	Arch.	**CPU**	**RAM**	**Size (mm)**	**Output**
Raspberry Pi 3	ARMv8	Quad Core Cortex A53	1 GB	85x56x15	GPIO
Hummingboard i2eX	ARMv7	Quad 1GHz i.MX6	2 GB	102x69x20	PCIe
Gigabyte Brix IoT	Intel	Intel N3450	Own choice	165x105x27	PCIe

Table 4.2: Overview of the hardware and their specifications

4.2.1 Raspberry Pi (Micro PC)

The Raspberry Pi (Figure 4.1) is perhaps the most famous and utilized single-board computer on the market. It offers a lot of functionalities due to the general-purpose input/output (GPIO) pins, peripheral support and decent hardware for a low price. Besides all the out of the box niceties, there are a lot of shields (boards you can put on top of the Pi, Figure 4.1b) that extends its functionalities and allow the developer to easily add more features.

Therefore, the Raspberry Pi offered a lot of niceties: features, well-documented operating system and lively community. However, using the Raspberry Pi is not a



(a) A Raspberry Pi 3 Model B

(b) Raspberry Pi with 3G/GPRS shield

Figure 4.1: Examples of Raspberry Pi

strong PC: with a 1.2 GHz processor and 1 GB of memory it would not be able to do any heavy computations (which are required for neural networks). Thus, it was decided that there would be no real-time system that would automatically detect any events.

PCIe over USB (or reverse bridge)

It was known beforehand that the Pi did not have a PCIe x4 slot and that there were no PCIe shields available. However, there are solutions that can transport PCIe signals over USB to the Raspberry Pi, which is often called "PCIe over USB", or a "reverse bridge". This would allow a PCIe board to be connected to the Raspberry Pi, but the signals sent between the board and the Pi would not be PCIe x4 signals (at least 1 GB/s), but a USB connection (640MB/s), and thus much slower than the PCIe signal. This is a problem for wireless network interfaces (NIC), as these require high data rates in order to function.

PCIe over GPIO

An attempt was made to simulate PCIe over GPIO by connecting the PCIe to the pins and writing a custom driver that could capture and send the PCIe signal. However, after trying for a considerable time, several problems arose here:

1. The GPIO pins could not switch nor distinct between low and high voltages fast enough

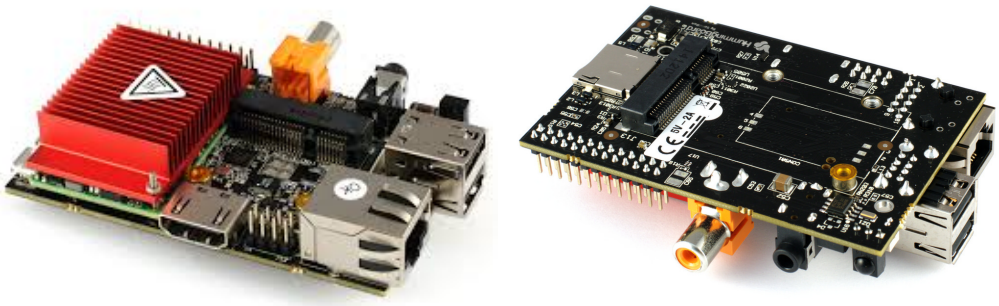
2. More than often, the voltages provided by the PCIe bridge were not high enough to capture

3. Boosting and improving the signal through hardware added a lot of components, increasing the size

4. Programming the actual driver for communication between both devices was more complicated than expected

As the design circuitry required to use the GPIO pins for PCIe communication would be highly complex, take a long time to build and be extremely expensive (which is why there are no PCIe shields for the Raspberry Pi), the choice was made to move on to another single board computer with a PCIe x4 slot.

4.2.2 Hummingboard Edge i2eX (Micro PC)



(a) Top view (b) Bottom view

Figure 4.2: Hummingboard i2eX Pro

After more research into single board computers, the Hummingboard Edge i2eX (Figure 4.2) was a good candidate, which had to do with the fact that it came with better specifications than the Raspberry Pi 3 (potentially allowing more on-board deep learning computation) and offered better passive cooling (as can be seen in Figure 4.2a). The most important aspect of the Hummingboard is that offers a

PCIe x4 slot, which is visible in Figure 4.2b. After installing the default operating system of the Hummingboard (a modified version of Solid Run's Debian Jessie for i.MX6 architecture) on an SD card and booting the device with the Intel Ultimate N Wi-Fi Link 5300, the device did indeed recognize the NIC when using `lspci`, a Linux command that shows all devices connected through PCIe.

Linux CSI Tool installation errors

Problems arose when trying to install the Linux CSI Tool software, as this requires a "plain" kernel version: which the modified Debian version was not. So, even though the kernel version had the correct version (kernel version 3.16, which is between 3.2 and 4.2), it could not install the Linux CSI Tool as the root-file system was still much too different from the default Debian version. It did have the correct modules for the Intel card (`iwlwifi`), as the N 5300 could be used for wireless transmissions after disabling the onboard WiFi card.

The other solutions provided by Solid Run themselves are Android and Yocto, where the file system of Android is too inherently different to work as well and the solution to their Yocto version did not work as their `git` was broken at the time.

There are a lot of community supported IMX6 distributions, including, but not limited to, Ubuntu, Fedora and ArchLinux, but all of these had the wrong kernel version. However, while some of these solutions had the right kernel and modules, they still were not "plain" kernel versions and the root system was still too inherently different to allow installation of the Linux CSI Tool. Even worse, some versions that correspond to a specific kernel on Windows did not do so on the ARM architecture. For example, Ubuntu 14.04.4 has the correct kernel on Windows, but on the ARM processor it had the wrong kernel version and could thus not be used to install the CSI Tool.

This meant that a custom kernel and modules needed to be compiled in order to get the right kernel version working on the board, while still maintaining the modules for the PCIe and the wireless drivers for the board. However, while there were plenty of solutions for the Intel architecture, the possibilities for the ARM architecture were very limited. In fact, they were so limited that most of these combinations were already used by the community supported distributions. The process at which the kernels and firmware were tried was mere trial and error:

compiling a kernel and rootfs (which could take quite some time to compile) on the SD card, put the SD card into the Hummingboard, boot and check the PCIe was recognized. More than often it was, and thus the next step was to try and install the Linux CSI Tool, which always failed.

In the end, no suitable and working combination of kernel and firmware could be found to work on the Hummingboard. This is likely due to the fact that no "plain" kernels exist for the ARM architecture and most ARM/i.MX6 kernels are modified in order to work properly with the development boards. Looking at community posts, one post from 2017 was able to install the Linux CSI Tool, but was still not able to properly log the CSI data. So, due to a lack of knowledge and documentation, any attempts to make the Linux CSI Tool work on an ARM processor were cancelled and the development was moved to an Intel architecture.

4.2.3 Gigabyte IoT Brix (Mini PC)

(a) Top view (b) Side views

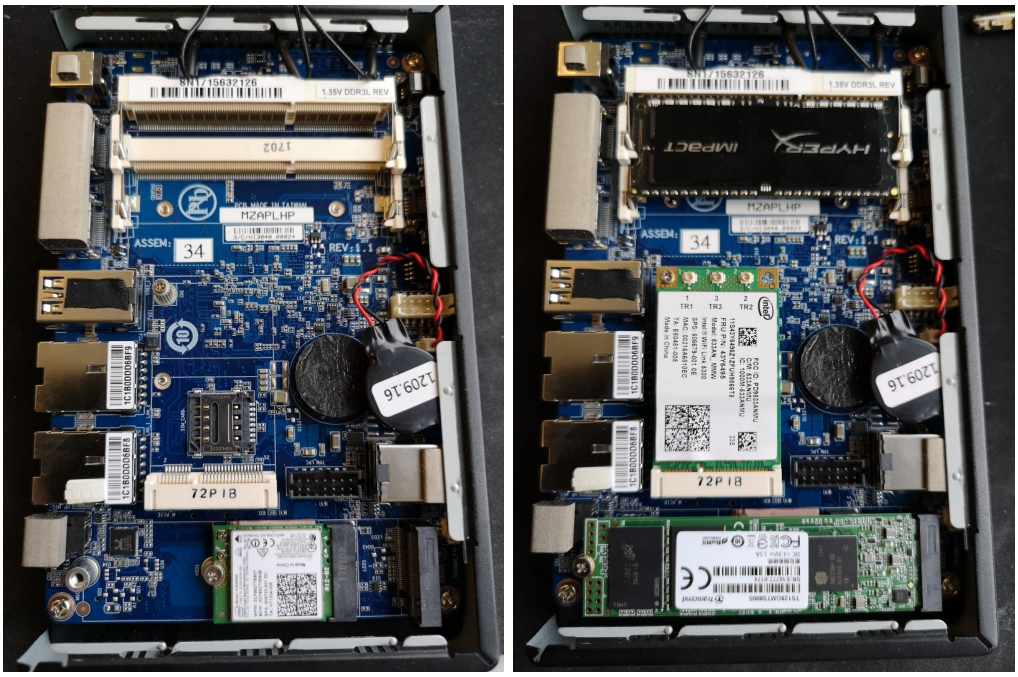
Figure 4.3: Gigabyte Brix IoT

The Gigabyte Brix IoT is a Mini PC (Figure 4.3), which means that, unlike the Raspberry Pi and Hummingboard, is an actual computer, rather than a single board device. This means that the Gigabyte is larger in size, but that it is more customizable then the single board computers. For example, the Brix IoT allows

for customizable memory and storage options. While the single board computers do allow for customizable storage options (through the SD card), using M.2 memory is a lot faster than SD cards.

mini-PCIe and M.2 slots

While the Gigabyte Brix IoT comes with its own wireless card (the Intel 3165NGW), this one needed to be replaced with the Intel Ultimate N Wi-Fi Link 5300. Unfortunately, the 3165NGW is an M.2 card, whereas the N5300 is a mini-PCIe card, meaning it could not fit in the same slot.



(a) With the 3165NGW (default) (b) With the Intel N 5300

Figure 4.4: Inside of the Gigabyte Brix IoT. *Note: while the antennas are disconnected, this was merely to make the picture clearer: during all experiments, all antennas were connected.*

However, the Brix IoT comes with an PCIe slot inside (as can be seen in Figure 4.4), which Gigabyte claims is for a 3G module. After connecting the N 5300 (Figure 4.4b), nothing happened. The NIC remained cold, which is strange for a wireless card, as it should heat up drastically, especially when no antennas

are connected, due to all the signals and vibrations on the card. Going through the boot logs, the card in the PCIe slot was not recognized at all. While it was not attempted to put another card in the PCIe slot, different N 5300 cards were tried among five Gigabyte IoT and none of these worked.

It is unclear as to why the NIC does not get powered by the board, but the working theory is that 3G modules need less voltage (1.5V), whereas a NIC needs the full 3.3V to be powered. The assumption is thus that the PCIe onboard delivers 1.5V and not the 3.3V, therefore not at all powering the card. This explains why the card remains cold and why the PCIe does not detect the card.

Fitting in the hardware

As the PCIe did not recognize the N 5300, another solution had to be found. Luckily, adapters exist that can connect PCIe to M.2 (Figure 4.5). This allowed the N 5300 to be detected and function with the Brix IoT.

It was another challenge to fit in all the hardware, as the adapter and full-sized N 5300 greatly increased the size of components. Furthermore, the M.2 slot is located beneath the hard disk (M.2), as can be seen when comparing Figure 4.4a and 4.4b). While the connection of the adapter is a flexible ribbon cable, it cannot be completely folded. By carefully placing the flexible cable underneath the hard disk, it can safely be moved to the center of the Brix IoT, where the adapter itself could be placed. By using only one memory slot and the half-sized N 5300 (HMW version), the adapter could be slipped underneath the memory stick and sit firmly in the Brix IoT. The final result can be found in Figure 4.6.

4.3 Final solution

The final solution is smaller than the current state-of-the-art, can easily be placed in rooms (even on the walls) and it fulfills all requirements. In total, five Gigabyte Brix IoTs were created using this method, but the main node (`minidfsn0`) was used to collect the data for the experiments conducted for this research (Chapter 6).

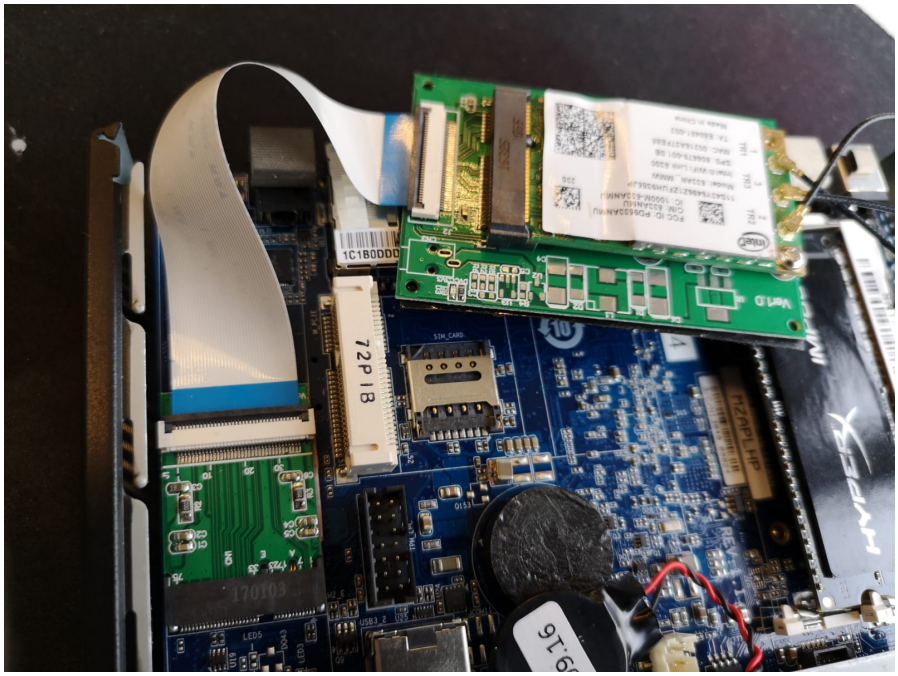


Figure 4.5: PCIe to M.2 adapter

Figure 4.6: Final version of the modified Gigabyte Brix IoT

4.3.1 Hardware specifications

The specifications of the final solution can be found in Table 4.3.

Component	Specifications
Processor	Intel Apollo Lake N34500
RAM	1x HyperX 8GB DRR3L-SO DIMM 1866 MHz
Hard drive	Transcend MTS800 SSD 128 GB (M.2 2280)
Graphics card	None
Wireless adapter	Intel N Ultimate Wi-Fi Link 5300
Size	165x105x27mm
Operating System	Ubuntu 14.04.4

Table 4.3: Hardware specifications Gigabyte Brix IoT

4.3.2 Available software

For this project, and with the intention in mind that multiple nodes were to be connected, three main software classes were written. All of the code below was written in both Python and Shell.

4.3.3 Collecting CSI

The most important requirement of this system is the collection of CSI data. As mentioned before, it was decided ahead of time to use the Linux CSI Tool, as the literature showed that it is a well-functioning software with extensive documentation. However, the tool is not completely headless, in the sense that it does capture CSI over a period of time until interrupted, but it does not allow for automatic consecutive measurements of fixed time periods. Therefore, scripts called `collectcsi.py / collectcsi.sh` were created that wrap the Linux CSI Tool around controllable software of which the features will be discussed down below. The pseudocode for `collectcsi` can be found below in Algorithm 1.

Setting duration of CSI frames

The wrapper allows to set a duration (and thus length) of the CSI frames. This length is passed as an argument to the function when called, and is in the format it, where i is an integer and t is a time unit (ms for milisecond, s for second, m for minute and h for hour). After the duration is over, the wrapper automatically saves the file with the appropriate filename and checks when to perform the next measurement. This allows for equally sized frames (assuming the packet receive rate does not change) and makes it easier to synchronize the CSI data more frequently for a more robust system.

Dynamic filenames

The wrapper automatically generates correct filenames, making it easy to understand when and where the data was captured and for how long. This is convenient for the analysis of the data, especially in `MATLAB`, as the neural network tools can use the folder and filenames for labeling. For the user, it is much more convenient to know when each frame was captured.

The layout of each filename is the following:

`$file_directory/$date_time+$duration_$node_name.mat`

where `$file_directory` is the directory where the node stores the data locally, `$date_time` the date and time at the start of recording (in the format DayMonthYear_HoursMinutesSeconds), `$duration` the duration of the CSI frame and `$nodename` the name of the node, currently set manually (the first node created is called `dfsnode0`).

Automated measurements

As previously mentioned, the wrapper allows for automated measurements. However, while there is a setting that just allows it to continually capture CSI frames, it also allows two other options: i) it allows for a break period (defined in the same format as the frame duration, thus it) and ii) it allows to only record CSI frames within certain hours of the day. This was originally meant so that CSI would only

be collected when it was necessary, and not just all the time, therefore saving space and energy.

Algorithm 1 Collecting CSI

```
1: date_time ← Current date and time
2: Create filename
3: while duration do
4:     function CSITOOL_COLLECTCSI(filename)
5: Save filename
6: if break is set then
7:     Wait until break is over
8: else if time period is set then
9:     Wait until time period is entered
10: else
11:     Go back to 1
```

4.3.4 Synchronizing CSI

When it comes to synchronizing the nodes, all of them should store their data on a central server and thus synchronize with it. This is done through a script that continously runs in the background of each node. This script periodically (a variable called `SYNC_TIME`) checks a few things, namely i) the folder size of both the new and archived data and ii) the dates on the files in both folders. This is done so that the sizes of both folders do not grow too big (variable `NEWF_MAX_SIZE` and `ARCHF_MAX_SIZE` for the new and archived folder, respectively). Furthermore, the date check on all files is so that even when the maximum size is not reached, files are still synchronized periodically (`NEWF_MAX_TIME` and `ARCHF_MAX_TIME` for new files and archived files, respectively).

synchronization from the node

synchronization happens from the nodes, that is, nodes contact the server that they have data to send every `NEWF_MAX_TIME` or `ARCHF_MAX_TIME`. After they have contacted the server, the server will notify a node when it is allowed to send the

files. While all nodes could perhaps simultaneously send their data to the server, it is cleaner for each node to send their data on confirmation of the server. Such a system also allows for expansion in the future: more functionally (such as status updates) can more easily be added in a system where nodes wait for their turn. When a node gets a confirmation that the files are received correctly, it moves the files from the new data folder into the archived folder.

synchronization to the server

On the server side, the script waits until it gets a request from a node. If the server is currently able to handle the request it sends a response to receive the files. The files are then stored in a temporary folder and once all the files are received and a confirmation has been sent to the node, the files are moved to their permanent folder. These folders are named after the nodes (thus extracted from the filename). Inside each node folder, the date itself is extracted and created as a subfolder. In the end, the file that is stored only contains the timestamp and duration. A s an example, we take the following file recorded at a node:

`24072018_114908+20s_minidfsn0.dat`

which corresponds to a file created on 24th of July 2018 at 11:49:08 and had a duration of 20 seconds. Once the server has received and processed the file, it ends up on the server as:

`minidfsn0/24072018/114908+20s.dat`

4.3.5 Device status

While the nodes do currently not synchronize their status with the server, it may be required in the future that these nodes do so. This has to do with the fact that all nodes should function headless, and if they synchronize their status with a server, it still becomes clear which nodes (mal)function when it comes to collecting CSI or their physical conditions and thus increasing the reliability and user-friendliness of the overall system.

The data that is currently gathered periodically (once every hour) by each node is the following and is stored in the following manner in a simple `.txt` file called `DEV_STATUS`:

```
DATE_TIME | CPU_TEMP | CPU_USAGE | MEM_USAGE |
CSI_COLLECTED | LAST_CSI | LAST_SYNC
```

where `DATE_TIME` is the current date and time, `CPU_TEMP` is the temperature of the CPU (the only temperature currently measurable), `CPU_USAGE` the CPU usage (in percentages from 0 to 100), `MEM_USAGE` the memory usage (in percentages from 0 to 100), `CSI_COLLECTED` the collected CSI between the last two device status synchronizations, `LAST_CSI` the time and date of the last completed CSI frame and `LAST_SYNC` the last synchronization when it comes to CSI data.

Chapter 5

Analyzing existing data

Previous research on this subject was performed by P. Bagave in the Master thesis Unobtrusive sensing using Wi-Fi signals [49]. In order to get familiar with the data and discover the potential use of convolutional networks, the data set created during this research will be used during this chapter. Furthermore, an attempt will be made to increase the accuracy of the proposed solutions by looking into a different feature (other than variance and amplitude): slope analysis. The findings from this research are compared with the results from previous research and conclusions are drawn regarding convolutional networks.

5.1 Activities

For multiple sets of different activities, the CSI was measured and stored for the research conducted by P. Bagave. This data was made available for this research, and a summary of the components of this data set can be found in table 5.1.

5.1.1 Sit and stand

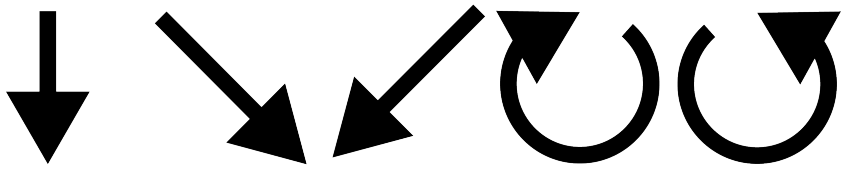
This was the introductory data to see if a system could detect sitting and standing. In the experiments, a single person was asked to stand for a few minutes and was later asked to sit for a few minutes, all of which was performed between the access point and the CSI collecting node. Furthermore, each of these activities had a repetition of 100 trials, thus leading to a total dataset of 200 data entries.

Activity	Summary	Data set	
Sit and stand	Sitting and standing	200	
Basic shapes	Five hand movements	500 (495 used)	
Static postures	Standing in fixed positions	4250	
unused data	Walking, jumping, clapping	178	

Table 5.1: Overview of the activities, their summary, size of their data set and amount of CSI frames collected

This is an easy exercise for any machine learning solution, as it offers a simple two class problem. The reason two class problems are easier to tackle is because it is a matter of "either ... or ...", which means a system can just say it is either A or $not A$ and that is much easier for a computer.

5.1.2 Basic shapes



(a) Down (b) Diagonal right to left (c) Diagonal left to right (d) Down (e) Counter-clockwise

Figure 5.1: Basic shapes drawn by hand

This experiment consisted of drawing multiple gestures with the fingertips. These gestures can be found in Figure 5.1. The idea behind classifying these gestures is an activity recognition system that detects and classifies what one is writing: each letter in the alphabet is a mere combination of straight lines and curves, thus detecting these straight lines and curves is fundamental to such a future system.

The basic shapes consist of a line straight down (Figure 5.1a, diagonal lines from top left to bottom right and top right to bottom left (5.1c and 5.1b, respectively), and clockwise and counterclockwise circles (5.1d and 5.1e, respectively). For each of these shapes, 100 trials were performed and thus a total dataset of 500 data

entries was available. However, for 5 entries only one transmitter was recorded, and these 5 entries were removed from the dataset, leaving only 495 entries. While in the thesis all the data is used for a single experiment, the data was actually gathered over two days, 50 trials for each day.

5.1.3 Static postures

(a) Both arms down (b) One arm forward (c) Both arms wide (d) Both arms forward (e) Bent through the knees

Figure 5.2: Static postures performed during the research

Like the basic shapes, there were 5 static postures (as shown in Figure 5.2) for which the CSI was measured over a period of time. As noted, these postures were all static: the person was standing with a fixed posture for the entire duration and only moved when the CSI was not collected. The idea was that a comparison could be made in how well CSI performs between dynamic (the fingertip moving for the basic shapes) and static (posture detection) situations. Furthermore, in order to see the influence of the environment on the CSI data, data was gathered in two situations for five subsequent days: in an office space and in an anechoic chamber. The office space resembles a real-life scenario, in which everything affects the CSI, whereas anechoic chambers eliminate all the environmental effects on the wireless signal.

The five static postures consist of standing straight (Figure 5.2a), having one arm out and one in front (Figure 5.2b), having both arms out to the side (Figure 5.2c), having both arms in front (Figure 5.2d), and finally crouching (Figure 5.2e). As described, for each activity two data sets were gathered: one in the office space and one in the anechoic chamber. For the office space, each data set consisted of a total of 500 to 600 for all days, which thus sums to around 3000 data entries. For the anechoic chamber this was 250 data sets per activity for all days combined

(thus 1250 data entries). Both of these data sets sum to a total of 4250 for both scenarios.

5.1.4 Unused data

The data set also contains data not presented in the thesis [49]. This includes multiple trials for some standard activities: talking, clapping, jogging, and jumping. Maybe most importantly, there is alto "empty" data, which is data that is just recorded without any activities happening. This data can be used in combination with the other data sets to compare activities against doing nothing.

It should be noted that there are no specifics given for this data, that is, no distance between the access point and the receiver are given. So, the assumption is that the data is recorded in the same manner as in the other office experiments, as discussed in the next section Used methodology (section 5.2). The data gathered is different for each activity and more information can be found in table 5.1, but on average there were 30 to 35 trials per activity.

5.2 Used methodology

5.2.1 Detecting a moving person

In order to detect a moving person, an experimental setup was fit into an office room. Within this room, several tables, cupboards and chairs were located. The experiments used a transmitting router and receiving PC equipped with the N 5300, with two and three antennas respectively. There was approximately 4 meters between the two devices and both were located at 0.75 meters high.

The variance over all channels was used as the most important feature to train an SVM classifier. The accuracy was between 96-99%. After this verification, a real-time system was developed using the SVM classifier and the variance. This system achieved an 86.26% accuracy, given a 91.24% certainty, and the accuracy dropped to 68.59% given a certainty of 70%.

5.2.2 Basic shapes recognition

For the recognition of basic shapes, a similar setup was used as before. However, subjects were asked to draw the shape facing the receiver and by trying to make the shape 60 centimeters long. A buffer of stationary CSI was recorded before and after the event itself.

After collecting the data, the data was modified by using noise reduction (moving average window), a low pass filter (cut-off frequency 25Hz, 1000 order FIR filter) and the signals were cropped to only contain the activity itself. After the data was modified and cleaned, the cropped signals were interpolated to all be of the same length. Finally, 90% of the data was used to train SVM, decision tree and linear discriminant machine learning models, and the remainder 10% was used to test the data. Also, a non-cropped version of the signal was used in an RNN network, with two LSTM layers containing 150 and 80 hidden layers, respectively.

Looking at the results, the highest accuracy measured is around 70% for a single antenna pair and a decision tree. The average for the first antenna pair was 50% on average when taking all the other classifiers in mind, with the linear discriminant without filter being the lowest at 20%. Taking all the other antenna pairs in mind, the accuracy was to 30% to 40% on average, with antenna pair 6 seemingly being the worst.

5.2.3 Static postures

This was a single person experiment, but the experiment was conducted over multiple days and in two different environments. The reasoning for having it in two different locations is explained in section 5.1.3, and the reasoning for collecting the same data over multiple days has to do with verifying the stability of the CSI in this situation over multiple days.

In order to create a system with verified features, the features are measured through data stability, significance of the features and system reliability. The first, data stability, means that all the trials are compared and classified to see the accuracy there; a high accuracy implies a correlation between the CSI logs within the same activity. Significance is proven by comparing by making a distinction in the data and classify the data through several different features (amplitude, phase, AGC and zero mean). Lastly, system reliability is proven by classifying the data

in relation to dates on which it was gathered.

Data for both the office room and anechoic chamber were stable, having a 99% classification rate for a given trial. However, when cross validating over the five different trials, the accuracy dropped to 15%, which could be explained by the use of AGC.

The significance of features and reliability will be discussed for both the office room and anechoic chamber.

Office room

For the office room, the same setup is used as in the previous experiments, which is a practical environment with some form of unpredictability and dynamic events. Data was gathered for about five seconds and trained and tested with a 90:10 ratio with non-overlapping trials.

CSI amplitude, phase, AGC and zero mean data were all considered as potential features by classifying them through an SVM. Phase seemed to be a worthless feature, only achieving an accuracy of 20%, while the other features achieved at least 80%.

When cross validating the remaining features over the five days, it became clear that the data from one day could not be used for another day (with accuracies not even reaching 45%). This could be caused by the office setting changing over the days (chairs moved, tables moved, etc).

Next, more data was collected over another five random days. When training with 5 days and testing the data on another 5 days, resulting in an accuracy of 27.61%. When using 9 days for training and only one day for testing, the accuracy dropped even more: 16.02%. Therefore, it was concluded no reliable system could be created for multiple days with the 10 days of data recorded.

Anechoic chamber

The anechoic chamber is a 3x3 meters room, causing the perfect environment, as there is no interference. For this experiment, the computer was placed outside of the room, but the receiver and router were placed inside the room. Therefore, the CSI was only affected by the setup and the subject in the room.

The features used in the anechoic chamber were amplitude, AGC, zero mean

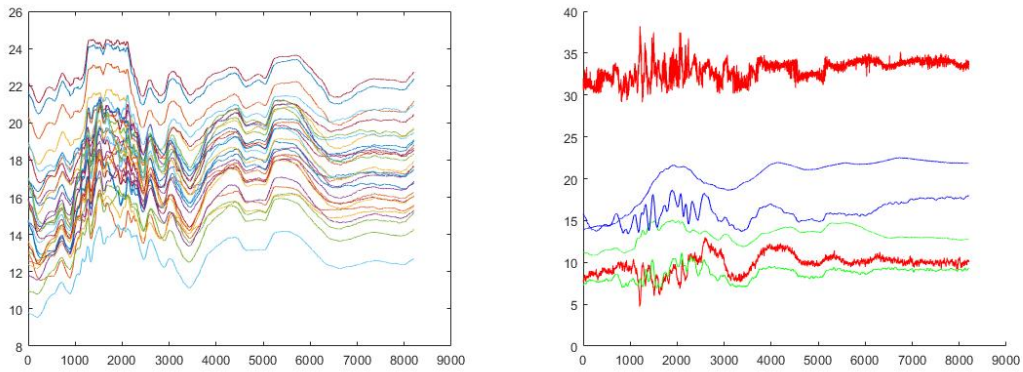
and regularly profiled data. All of these features seemed to do well, achieving at least 70% accuracy. Most importantly, zero mean and normal profiled data do not seem to hold much information in a controlled environment.

Like the office room experiments, data was also instable over given days: accuracies did once again not get above 42%.

5.3 Alternative approaches

5.3.1 Viewing data differently

The data was considered by looking each antenna pair individually and then considering all the subcarriers for each antenna pair. Then, for the basic shape recognition, classification was done for each antenna pair and then compared to see how accurate each antenna pair was. This did not feel quite as intuitive, as for each antenna pair, the subcarriers are mostly the same: differences come to light when one views each antenna pair per subcarrier. It is likely that by viewing the antenna pairs per subcarriers, more information can be extracted, as viewing data per antenna pair has a lot of overlap.



(a) Viewing all subcarriers per antenna pair (b) Viewing all antenna pairs per subcarrier

Figure 5.3: Different interpretations of CSI signals for the same activity and trial

Furthermore, while the data could potentially be reduced by only analysing one antenna pair and therefore it would make more sense to analyse individual antenna pairs, this may not be the best aspect to cut down on the data size. It

is likely that activities affect the antenna pairs differently per subcarrier and that this information combined thus provides a better signature of an activity.

5.3.2 Slope analysis

While multiple features were extracted and compared against each other, one important feature was left out: the slope of the data. Whereas amplitude shows the impact of the activity on the CSI, slopes show the speed at which an activity affects the signals. This is a big difference: the amplitude could be the same for multiple activities, but still have different slopes. On the other hand, activities with the same slopes but different amplitudes are still recognizable: with a different amplitude, the duration of a slope is still different.

The argument can be made that when one looks at the length of the slope, one is essentially looking at the amplitude of the signal, as one can calculate the amplitude of the signal in the following way:

$$A_{old} + s_c t_s = A_{new}$$

where A_{old} is the current amplitude, s_c the current slope, t_s the duration of the slope and A_{new} the new amplitude. However, looking at the slope still provides more information: it provides the amplitude and the time at which the amplitude is reached, thus it provides both impact and speed.

The claim made earlier that different amplitudes can still have different slopes can also be proven by the aforementioned formula. Assume $A_{old} = 0$ and $A_{new} = 10$, one gets the following:

$$0 + s_c t_s = 10 = s_c t_s = 10$$

Solving this for $s_c, t_s \in \mathbb{N}$ (t_s is always positive, as time cannot be negative, and thus c_s must be positive), one gets for s_c and t_s the following solutions: $(1, 2, 5, 10)$ and $(10, 5, 2, 1)$, respectively. When one multiplies the elements from the same index, these all provide the amplitude of 10.

While this is easy to understand, the slope and time are not bound to the natural numbers. Instead, it becomes slightly more complex, as the slope and time are all positive rational numbers, $\mathbb{R}_{>0}$. Now, when one considers the previous

formula, in order to solve it for $s_c, t_s \in \mathbb{R}_{>0}$, two formulas can be solved:

$$\frac{10}{s_c} = t_s, \frac{10}{t_s} = s_c$$

However, when one considers the range $1-10$ in $\mathbb{R}_{>0}$, in theory there are an infinite amount of rational numbers within this range, and thus an unlimited amount of solutions for either formula. In practice there is a finite number of solutions, as the resolution (the number of digits leading the number) is limited by the hardware. However, due to these reasons, the slope potentially contains more data.

5.3.3 Cropping and thresholding

While it is understandable data is cropped in order to reduce the data size, cropping often comes with challenges and loss of data. In this research, cropping the data was done based on the variance of the signal: no variance meant that there was no activity, whereas an increase in variance meant there was an activity. While this is a reasonable assumption, there are a few problems with this assumption.

First of all, an issue occurs when determining a threshold in general. The problem with thresholding is that it is a fixed number which determines whether or not an activity is happening. Setting a wrong threshold can either remove useful data or keep useless data. Either can impact the classification accuracy of machine learning techniques.

In this research, it is said that no variance means that no activity is happening and that anything above 0 indicates that an activity is happening. This could explain the cropping issues described in the paper, as environmental events also cause the signal to change and thus cause the variance to increase and cause the signals to contain a lot of useless data, which in turn drop the accuracy of the machine learning techniques. It is likely that the threshold should be higher in order to detect the activities more accurately, but what this threshold should be is unknown at this point. It is even likely that no fixed threshold can be set, as this is dependent on the environment, distance to the router and other external influences. This would mean that an adaptable threshold should be implemented. An adaptable threshold can be implemented using a moving window, that would automatically adapt the threshold to any permanent changes.

Variance is an expression of how far a random variable the mean of the data set. While an increase in variance does imply there is something affecting the signal, it does not necessarily mean that it is the activity affecting the signal. Now, this can be said for everything: a change in amplitude does not mean the activity caused, a change in slope does not mean the activity caused it, a change in anything does not mean the activity caused it. However, it could be that another feature of the signals is a better threshold to detect the activities. This research focuses on slope analysis (section 5.5), and therefore it will be attempted to create a better activity detection system based on slopes, as well as attempted to improve the current detection system based on variance.

Lastly, it should be noted that real-time systems are likely to suffer immensely from a fixed threshold, as real-time systems are in environments that are not controlled and thus likely to change. Therefore, it is important to invest time in inventing an adaptable detection system.

5.3.4 Interpolation of the data

Another point of critique is on the interpolation of the data. By cropping the data to only contain the activity and not the buffer zone, pieces of information are removed. Understandably, this results in pieces of CSI data that all have different lengths, as not even performing the same activity over and over again takes the same time. It is also understandable why cropping could be necessary, as it contains lots of information where nothing happens due to the buffer data. However, the problem lies by interpolation: interpolation is an educated guess on how data would continue to transform over time. CSI data is not stable and fluctuates over time due to environmental changes, so interpolating the data based on a single trial leads to uncertainty.

Furthermore, it seems contradictory to first remove actual data and then to interpolate the signal. The actual data could be used to make frames of the same size. The theoretical explanation will be given below, the implementation can be found in section 5.4.

First, all the trials are known as T, and a specific trial i as T_i. Now, T_i contains all the frames for the specific trial i, which will be called f_{total}. Within each trial, the following functions are assumed:

- `event` returns the frames containing the event: $f_{event} = event(T_i)$
- `left` returns the frames on the left of the event: $f_{left} = left(T_i)$
- `right` returns the frames on the right of the event: $f_{right} = right(T_i)$

This means that one knows that $f_{total} = f_{left} + f_{event} + f_{right}$. Next, one needs to identify the trial with the longest event (in frames): $f_{event_{max}}$. This is the maximum number of frames needed to fit any event from all other trials. The trial containing the longest event trial T_m. Thus, it is implied that all other events are shorter than $f_{event_{max}}$:

$$\forall i \in T, i \neq m, f_{event} = event(T_i) : f_{event} \leq f_{event_{max}}$$

To make it easier, $f_{event_{max}}$ can be denoted as f_{max} and is thus the maximum number of frames allowed. Note that for a trial $T_i, i \neq m$ it is still possible for $f_{total} > f_{max}$ for different trials. However, because the other events are shorter (or the same size), it can be concluded that there are either frames left, or no frames left when they share the same size. This means that $f_{left} + f_{right} \geq 0$ and the next question becomes how many frames are preferred on the left and right side of f_{event} or, in other words, how many frames one wants prior to the event (f_{prior}) and how many frames subsequent to event (f_{sub}). This means that i) $f_{max} = f_{prior} + f_{event} + f_{sub}$, ii) $f_{prior} \leq f_{left}$, iii) $f_{sub} \leq f_{right}$ and iv) $f_{max} \leq f_{total}$.

Now, as it is unknown whether data before or after the frame contains more valuable data, the assumption is made that they contain equal (useless) data. Therefore, the preference is that there is an equal amount of data prior and subsequent to the event, thus $f_{prior} = f_{sub}$.

However, this is only possible for the case in which $f_{prior} \leq f_{left}$ and $f_{sub} \leq f_{right}$, thus in which there is at least enough data on each side to evenly put the activity in there. When this is not the case either $f_{prior} > f_{left}$ or $f_{sub} > f_{right}$, as it would have fit otherwise. For example, if $f_{prior} > f_{left}$, then there is not enough data on the left to equally divide it. Therefore, the final frame count for $f_{sub} = f_{sub} + f_{prior} - f_{left}$ and for $f_{prior} = f_{left}$. The same can be said for the case with $f_{sub} > f_{right}$, by switching f_{sub} and f_{prior} around and replacing f_{left} for f_{right}.

Note that this does not work for frames of which $f_{total} < f_{max}$. In that case, some interpolation is needed. However, in this case, one could also consider to not

use the data (if there is enough alternative data left), or use data synthesis (which is based on all trials and not just interpolating the single trial).

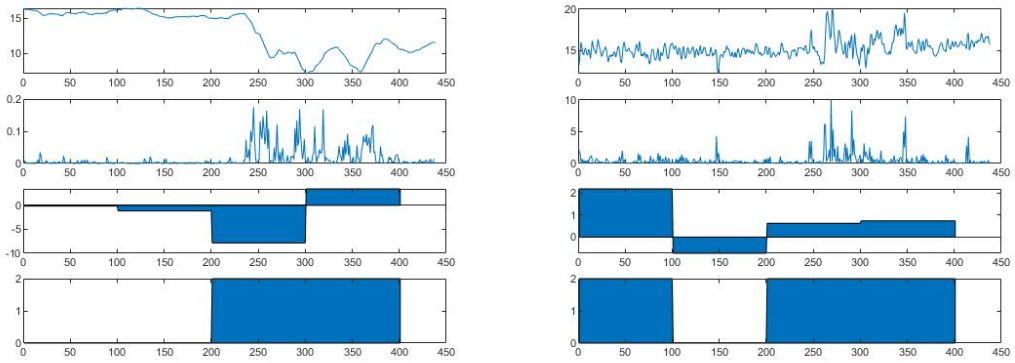
5.4 Own methodology

In order to try and improve the data set and results, the other approaches were implemented. This included working on a better activity detection system, minimize removing useful data or interpolating useless data and attempting different machine learning solutions or feature exctraction.

5.4.1 Detecting activities through variance and slope

Detecting the activities was done using two methods: by analysing the slope and viewing the slope as indicator for an activity, and by doing the same for the variance. This was done using a sliding window and looking at the averages. In order to use the slope or variance, either must be determined first for all data. This was done in `MATLAB`, by writing an algorithm that calculates the slope over an x number of frames. The reason this was used and not the `diff` or `gradient` functions in `MATLAB` has to do with the flexibility: the custom algorithm allows to skip frames and thus changing the sensitivity of the slope. Calculating the slope for each frame resulted in slopes very close to 0 and thus not actually showing a steep slope. An example can be seen below, with the slope and activity detection scaled over the entire duration of the activity to compare. It should be noted variance worked in a same manner.

The accuracy of the algorithm is not high enough to view individual parts of the activity reliably (e.g. each individual clap while clapping or strife in running). However, in some cases it is pretty close to. It should be noted that the system attempts to be dynamic, which could cause it to be close in some cases and be completely off in other cases. For example, due to being dynamic the algorithm can easily get confused at the start of new frames (Figure 5.4b). This was in most cases fixed by only considering either a) the longest activity or b) starting at the second activity. For research, it is enough for the algorithm to detect the start and end of the overall activity and for this it functions well enough.



(a) Showing a correctly detected activity (b) Example of dynamic detection falling short

Figure 5.4: Figures showing the detection of an activity using slope data. The top is the actual signal, the second is the actual slope change, the third is the sliding window and the fourth (bottom) is the actual detection

5.4.2 Equal lengths for the frames

The aforementioned techniques to determine the length of the activities for the different trials were used to create equal frame sizes. Note that the slope analysis was used to determine the frame sizes, but that the variance could be used in the same manner. This was done by implementing the theory discussed in section 5.3.4 (Algorithm 2). These equal frame lengths were required for training the SVM.

5.5 Multi-class SVM

In order to train the SVM using all antennas (instead of per antenna pair), the data needed to be concatenated. Luckily, SVM is oblivious to what the data means and thus this can be easily achieved by putting information in a huge array. All of these arrays have the same size, due to all the individual trials now having the same size. Data was concatenated by first having all the values for each antenna pair per subcarrier and by then putting all the subcarriers in a row.

Detecting the accuracies for the dynamic activities was still comparable to results found in previous research [49]. For the basic shape classification, results

Algorithm 2 Creating equal framesizes, functions names are self-explanatory

```
Import all trials as trials for activity a
longestEvent ← 0  ▷ First, one must determine the longest event in all trials
for all trials do
    eventLength ← getEventLength(trial)
    if eventLength > longestEvent then
        longestEvent ← eventLength
                                  ▷ Secondly, the actual framesizes are calculated
for all trials do
    totalFrames ← getTotalFrames(trial)
    eventLength ← getEventLength(trial)
    if totalFrames < longestEvent then
        Discard trial or interpolate
    else
        left ← getLeftFrames(trial)
        right ← getRightFrames(trial)
        event ← getEventFrames(trial)
        prior, sub ← (longestEvent − eventLength) / 2
        if prior < left & sub < right then
            trial ← createNewTrial(prior,event,sub)
        else if proir > left then
            sub ← sub + prior − left
            prior ← left
            trial ← createNewTrial(prior,event,sub)
        else if sub > right then
            prior ← prior + sub − right
            sub ← right
            trial ← createNewTrial(prior,event,sub)
    SaveNewTrial(trial)
```

were between 30% and 40%. For the static postures in the office, accuracies were between 70% and 85% depending on the day and for the anechoic chamber, these were also in this range. Therefore, no extensive research was continued into using SVM to classify the remaining data and a focus was put into convolutional neural networks. The activity detection algorithm was still used for convolutional networks to crop the signals for activities which were recorded for a long time (over 10.000 frames) in order to make the data more manageable.

5.6 Convolutional NN

In order to attempt to improve the performance is by using a convolutional neural network. The convolutional neural network in this research was a simple network, consisting of the regular layers (input, fully connected and output) and an extra convolutional layer. This convolutional layer slides over each of the graphs individually and filtering the features out of these. In total, this convolutional layer has 20 filter layers inside of the convolutional layer. The input layer defines the input size of the images, which in this case is an $1750x1313xD$ image, where the first two are the resolution of the image and the D is the depth of the images (RGB is 3 layers, monochrome is 1 layer). A visualization of this neural network can be found in Figure 5.5. he learning rate was 0.001. The batch size was incredibly low (16-32) in order to keep processing possible. No dropout was used.

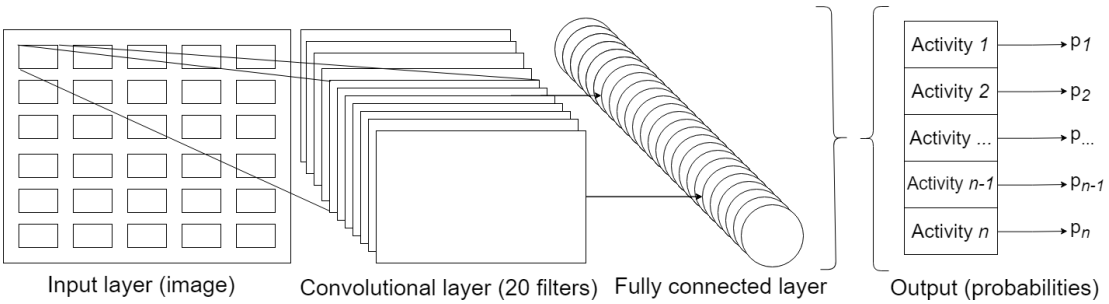
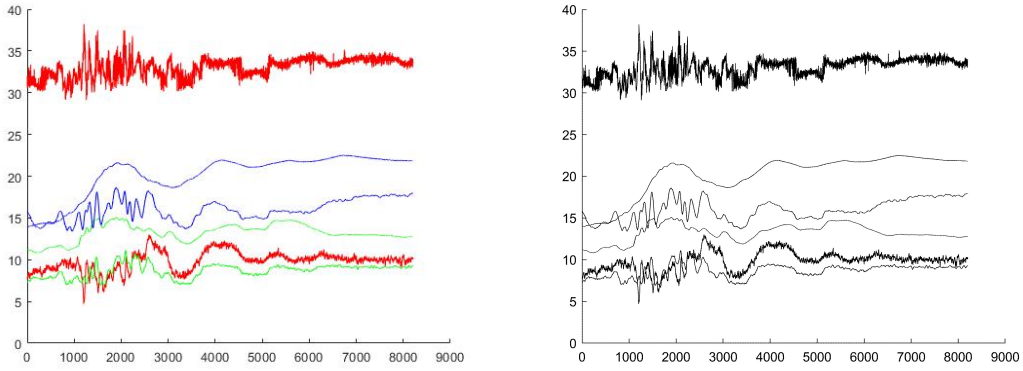


Figure 5.5: Visualization of the convolutional neural network used

It should be noted that for each learning progress, the number of iterations were kept to a minimum. This had to do with a fact that training the network took a long time due to the image size, number of layers and size of the data. For

example, the arrays containing the data in `MATLAB` grew to 31 GB in size, which had to be stored in memory. Therefore, a dedicated 128 GB SSD was wiped and enabled for virtual memory/paging.



(a) RGB (b) Monochrome

Figure 5.6: Different types of images rendered for this research

When it came to the input data, this was rendered and classified in two-fold: the original RGB images and as monochrome renders of these images, as can be seen in Figures 5.6a and 5.6b, respectively. The reason the data was classified for both RGB and monochrome had to do with the depth of the images: RGB has a depth of 3, whereas monochrome images only have a depth of 1. Thus, this decreases the complexity and memory size, while increasing the speed at which the neural network is trained and at which data is classified. It is interesting to see if the trade of between speed and amount of data is worth it in regard to the accuracy.

Furthermore, different types of images are fed into the deep learning network: the default image of how the network is viewed, as well as both the slope and variance data as generated by the algorithms described in the methodology. This is to compare how no feature extraction holds against some feature extraction. While neural networks learn better with more data and thus with no feature extraction, feature extraction once again decreases the complexity and memory size.

Given all of the above, for each of the activities in the data set, 6 pieces of data are generated: 3 types of images (normal, variance and slope) in both RGB and monochrome. This means that, given the 4 different activity sets, a total of

Table 5.2: Table showing the classification rates for the existing data set, regardless of days (from [**?**])

	RGB			**Mono**		
	Raw	**Slope**	**Variance**	**Raw**	**Slope**	**Variance**
Sit and stand	100%	87.9%	95.5%	100%	72.7%	81.8%
Hand shapes	93.0%	70.8%	65.5%	90.1%	64.9%	51.5%
Unused activities	91.0%	46.2%	50.0%	84.6%	37.2%	50.0%
Static (office)	60.9%	X	X	64.3%	X	X
Static (anechoic)	64.4%	X	X	67.8%	X	X
Average	81.87%	68.3%	70.33%	81.37%	58.27%	61.1%

24 different image sets were generated. This meant the total amount of images generated (based on the data collected for each activity), between 30,000 and 31,000 images were generated. The overall accuracies, regardless of the days, for all data can be found in Table 5.2.

5.6.1 Sit and stand

Sit and stand is an easy problem to solve for machine learning: it is a binary problem, that is, the answers are limited 0 to 1. This can be translated to it is $Aor!A$. Therefore, the expectation was that convolutional networks have no trouble differentiating between sitting and standing.

In the case of a raw RGB image, the classification accuracy was 100%. Looking at the same raw image in monochrome, the accuracy was also 100%. For the RGB slope and variance, the accuracy dropped to 87.9% and 95.5%, respectively. For monochrome, this dropped slightly more, namely to 72.7% and 81.8% for slope and variance, respectively.

5.6.2 Hand shapes

The handshapes are a more complex problem, as there are five classes. Therefore, it is likely that the accuracy drops slightly for such a problem. Data for this activity was gathered over two days, but was only classified as a single batch. In

	RGB			**Mono**		
	Raw	**Slope**	**Variance**	**Raw**	**Slope**	**Variance**
Day 1	86.8%	68.1%	70.3%	90.1%	62.6%	62.6%
Day 2	98.8%	90.0%	81.3%	97.5%	65.0%	58.8%
Day 1 **for Day** 2	25.3%	26.9%	32.7%	28.6%	38.4%	31.0%
Day 2 **for Day** 1	45.1%	42.7%	30.1%	48.4%	33.7%	48.8%

Table 5.3: Table showing the classification accuracies for individual days, as well as training with one day and classify with the other

order to view the ability to use data of a single day in order to classify it for another day, both instances are considered in this.

For the full classification, the accuracy of the raw RGB image was 93.0%. For the monochrome data, the accuracy of classifying the handshapes dropped to 90.1%. When it came to the variance, the RGB accuracy was 65.5% and the accuracy for the monochrome was 51.5%. When looking at the RGB and monochrome data for slopes, these accuracies were 70.8% and 64.9%, respectively.

Looking at the individual classification for each day (Table 5.3), individual days have high accuracies for the raw RGB and monochrome data. For the first day, this was 86.8% for RGB and 90.1% for the monochrome. For the second day, this was 98.8% and 97.5%, respectively. Looking at the slope data for the first day, this was 68.1% and 62.6% for the RGB and monochrome, respectively. For the second day, these were higher: 90.0% and 65.0%. The RGB variance classification accuracy was 70.3%, whereas the equivalent monochrome data was 62.6%. For the second day, the same images resulted in accuracies of 81.3% and 58.8%.

When training with one day and attempting to train the other day, accuracies dropped to the point they were not useful. Training with the first day and testing with the second day resulted in accuracies as low as 25.3% and as high as 38.4%. For training with the second day and training with the first day, accuracies were slightly higher: the lowest accuracy was 30.1% and the highest was 48.4%.

5.6.3 Unused data

No details regarding the collection of the unused data were provided. Therefore, it is not known whether or not the data was gathered over multiple days or when/how it was collected. However, the data was labeled, so it could be classified. As explained before, not a lot of data was provided for each activity (about 35 elements per activity).

The results can also be found in Table 5.2. For raw images rendered with RGB, the average accuracy was 91.0% and for monochrome this was 84.6%. Both slope and variance fell short in classifying this data set, with slope being the higher of the two (46.2% and 44.9% for RGB and monochrome, respectively). Variance was on the bottom, with 50.0% and 37.2% for RGB and monochrome, respectively.

5.6.4 Static postures

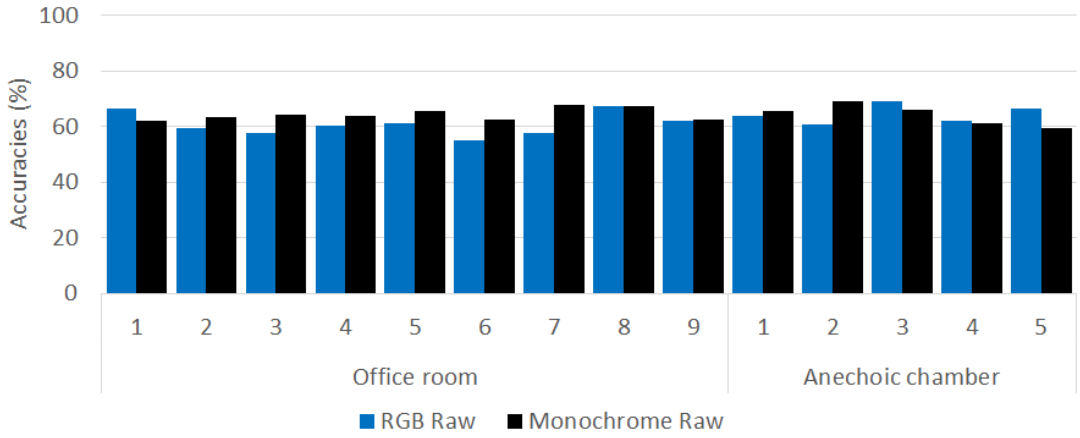


Figure 5.7: Figure showing the average accuracies per day in the office room and anechoic chamber setup

As mentioned before, the static postures are measured over multiple days in multiple locations. The daily accuracies for each location will be discussed in more detail in their sections. Furthermore, it should be noted that slope and variance data was no longer considered at this point, as they were always lower in terms of accuracy compared the raw data (in some cases 40%). Therefore, a focus was

put on the raw data, as calculating the slopes and variance actually takes more computation power and time.

The accuracies for each for both the office and anechoic are listed in Table 5.4 and shown in Figure 5.7. The average accuracies for the RGB and monochrome raw data for the office room were 60.89% and 64.33%, respectively. The highest accuracies measured were for seventh day, with an accuracy of 67.8% (monochrome). The lowest were measured on the third and fifth day, with 57.9% (RGB). For the anechoic chamber, the RGB accuracy was 64.44% and the monochrome accuracy was 64.32%.

When considering data between days, the accuracies are incredibly low when trying to train with a day and then test with the other (Table 5.5). For the office room data, when trained with 5 days and classified with 4, the accuracies were 11.6% and 17.6% for RGB and monochrome, respectively. When training with 8 days and classifying with 1, the accuracy for RGB slightly increased (to 16.3%), but decreased to 14.3% for the monochrome data. For the anechoic experiments, when training with 3 and classifying with 2, accuracies were 15.8% and 24.6% for RGB and monochrome, respectively. When classifying with 4 days and classifying 1 day, both RGB and monochrome increased: 22.8% and 25.6%, respectively.

5.7 Discussion

For sit and stand, the accuracy was extremely high. This is due to the aforementioned reason of it being a two-class problem. Furthermore, sitting down causes disturbances in the CSI, whereas standing still does nothing and thus causes a stable signal. As the CNN learns from the shape of the signal, it is understandable that it can easily detect stable systems and disturbed signals.

Interesting to note is that during the second day of hand movements, the accuracy was incredibly high (98.8%) compared to the first day (86.8%). Apparently, the higher accuracy also leads to a higher classification rate over the days. This could be because the activities were performed in a very similar matter during the second day, whereas in the first day they were not and only some represented the second day. For example, imagine 98% of the activities performed in a likely manner during the second day and that during the first day 40% of all activities were performed in said manner. Now, one can expect the second day to have a higher

accuracy compared to the first day. When trying to classify the second day based on the first day, only 40% will edge towards the correct activity and the other 60% will edge towards other activities, thus resulting in a lower accuracy (25%). When training with the second day, the 40% correctly performed activities during the first day will be classified and the others will not, which still leads to an accuracy rate of 40%. This explains the lower accuracy for the first day (different ways to perform the activity and a lower classification rate on day two), whereas the second day can more accurately classify the activities in the first day (thus having a higher accuracy) and still have an incredibly high accuracy rate.

For the static postures (both in the anechoic chamber and office room) the accuracies were lower (by 10% to 20%) when compared to results from the previous research [49]. This is likely because static postures do not impact the signal significantly, but instead, affect the CSI consistently or minimally. Using old (numerical) machine learning techniques, such as SVM or Forest Decision can detect these small values, as they are part of the input. However, these small changes cannot be detected using the current implementation of the convolutional network: each graph is an $1750x1313$ pixels image. When taking the padding out, the image is reduced to $1425x1020$, meaning that each of the 30 graphs is $285x170$. Now, looking at the range of values on each of the axes, these vary between 50 and 600 for the y-axis and are 1000 to 10000. This means one pixel contains multiple frames of information, and therefore the information of several frames is lost. The way convolutional neural networks analyse these, they cannot see the difference between certain values and thus lose out on a lot of information. This could be fixed by changing the images input into the CNN, but for this research this was not considered.

For static postures measured over multiple days, accuracies dropped between 10% and 15%. This is likely due to the same reason as previously described, and could therefore likely be fixed as well. However, interesting to note is that accuracies did seem to increase when more data was used. For the office room experiments, this was by 5% and for the anechoic chamber this was by 7%.

On the other hand, dynamic activities had a much higher accuracy rate when compared to the previous research [49]. While the sit-and-stand and the unused data were not mentioned in the research, the accuracies were extremely high in this research. When using the RGB raw data, accuracies were close to (if not)

100.0%. When comparing the basic hands movements, previous research reported to have accuracies of up to 50% to 60%. This research achieved accuracies of up to 93.0% when detecting the hand movements. When training with a single day and then tested with the other, accuracies dropped to 50.0% at maximum. However, interesting to note is that the day with the highest individual accuracy also achieved the higher accuracy when trained with (Table 5.3, *Day* 2 *for Day* 1).

For accuracies in general, it seemed slope and variance were decreasing the accuracy quite a bit at times (up to 40%). This is likely because CNN can extract its own features, and could extract these features on its own. Instead, by extracting features, the CNN is limited in its options to pick the features it can learn from, thus decreasing the overall accuracy. For monochrome data, the difference between raw, slope and variance in accuracies when compared to the RGB equivalents can be explained by the fact that monochrome already contains less information due to the change in image depth. Thus, the antenna pairs are anonymous and therefore, slopes and variance can belong to any pair. This eliminates slopes and variances belonging to a specific pair and it is likely that some activities share the same variance and slopes but for different antenna pairs, and thus looking at them anonymously makes them less vulnerable to slope and variance analysis.

5.8 Conclusion

The features extracted in this research (slope and variance) are not useful when classifying through a convolutional network. In most cases, the variance and slope decreased accuracies by at least 10% to 15%. Therefore, a focus was put on the raw images. This was done as computation times were high when it came to calculating slopes and variance and then rendering these additional images. It can also be explained when considering the CNN network: it contains 20 filters which extract its own features and by already performing feature extraction, one lowers the potential features to be extracted.

Convolutional neural networks perform better when attempting to detect dynamic movements compared to detecting static postures. Dynamic movements had accuracies up to 98%, with sit-and-stand even reaching a 100%. However, for the static postures, the accuracies were around 60% per day. This is likely because the current implementation does not support a high enough resolution to view the

small differences caused by static postures. If static postures are to be classified, it would be better to use a new way to input the images, such as plotting the average over all subcarriers or antenna pairs. However, by averaging signals, data could potentially be lost.

Over multiple days, accuracies decreased. As no ground truth was recorded, data could not be compared to what actually happened. As reported in the previous research [49], this is likely because signals change due to environmental influences that are not accounted for. However, it is also likely that activities were not performed at the exact same speed or in the exact same space. One node likely has troubles classifying this, as the rotation and location of the activities are also not accounted for. Even when performing the same activities at the same speed and in the same location, activities can be performed in a different manner: it could be higher from the ground, fingers in a different location or moving/talking during the experiments. For static postures, these could be that different postures were not exactly the same over different days. However, there is reason to believe that more data from multiple days could increase the accuracy.

	RGB raw	**Mono raw**
Office room		
Day 1	66.7%	62.0%
Day 2	59.6%	63.2%
Day 3	57.9%	64.3%
Day 4	60.2%	63.7%
Day 5	61.4%	65.5%
Day 6	55.0%	62.6%
Day 7	57.9%	67.8%
Day 8	67.3%	67.3%
Day 9	62.0%	62.6%
Average	60.89%	64.33%
Anechoic chamber		
Day 1	63.7%	65.5%
Day 2	60.8%	69.0%
Day 3	69.0%	66.1%
Day 4	62.0%	61.4%
Day 5	66.7%	59.6%
Average	64.44%	64.32%

Table 5.4: Table showing the accuracies per day and the average over all days for static activities

Trained with	**Tested with**	**RGB raw**	**Mono raw**
Office room			
1,2,3,4,5	**6,7,8,9**	11.6%	17.6%
1,2,3,4,5,6,7,8	**1**	16.3%	14.3%
Anechoic chamber			
1,2,3	**4,5**	15.8%	24.6%
1,2,3,4	**5**	22.8%	25.6%

Table 5.5: Table showing the accuracies between days for the static postures

Chapter 6

Experiments

After working with the existing data set, additional data was gathered through experiments. This data focuses more on scalability and larger movements, rather than single person and static postures or small, dynamic shapes. This chapter explains why other activities were considered, as well as the methodology of the experiment and the analysis of the gathered data. A conclusion is given in regards to the gathered data, correlations and discoveries.

6.1 Activities

(a) Clapping (b) Walking (c) Waving (d) Jumping (e) Sitting (f) Falling

Figure 6.1: Basic shapes drawn by hand

The activities performed during the experiments (Figure 6.1) for this research had some overlap with the activities from the existing data set, but it was also chosen to focus on some other activities. The overlapping activities are sitting, walking, clapping and jumping. It should be noted that while the activities originate from the existing data set, they are not discussed in the research [49] as they

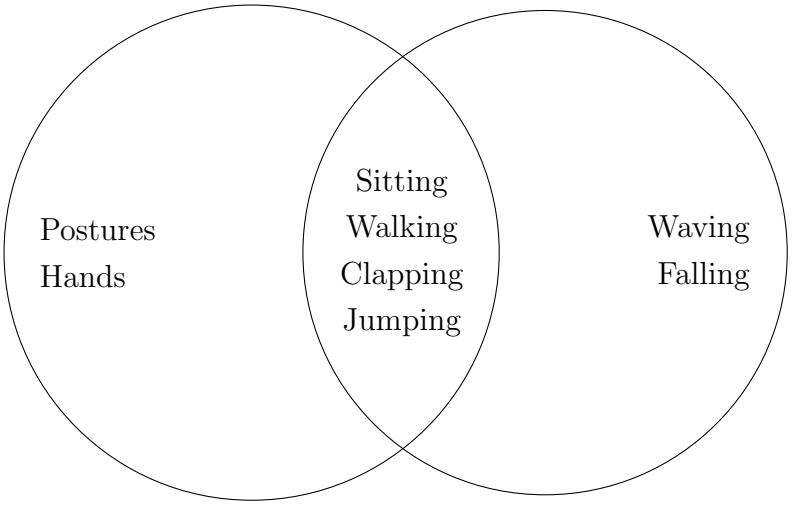


Figure 6.2: Difference between existing data set (left) and this research (right), also showing the overlap (middle).

were part of the unused data. The activities added in this research are falling and waving, both added as they can provide a more societal impact. The differences between the previous chapter and this chapter in terms of activities analysed are illustrated in Figure 6.2.

6.1.1 Waving and falling

Static postures are dropped as it is unlikely that people remain in a fixed position for more than a millisecond in the act of an activity. It could be relevant when trying to pass data through static postures (like an alphabet), but that is out of the scope for this research. In the same manner, drawing basic hand shapes in the air was dropped from the research as it is also only usable while trying to pass data (such as drawing letters in the air). However, as seen in chapter 4, CNNs do not really seem capable of analysing this.

Waving and falling offer a more societal impact (Figure 6.1c and 6.1f, respectively), as they both are more crude, dynamic activities which can convey a lot of information. Waving can be used as either a greeting, asking for attention or in case of an emergency. For example, imagine a nursery: an older family member drops to the ground and a person starts to wave in order to get the attention of a nurse, but fails - the system could potentially still pick this up and page a nurse.

On the other hand, waving is an indication of greeting, so it can also be used to know when people are around each other.

Falling has a big societal impact, as there are a (target) groups that are not able to move or function on their own. These people could include the elderly or handicapped people. Potentially, the lives of these people could be greatly improved when a more pervasive system can be developed that automatically notifies nearby family members or experts in case of an emergency.

6.1.2 Group information

While part of the activities is skipped in this research, a focus is added on group data: data is gathered from a broader audience consisting of people with different characteristics. The reason a broader audience was considered is to consider potential scalability of such a system. Data is compared among different people, in order to evaluate the possibilities of sharing data among people. This is important to know how possible it is to classify data from a person based on the data from other people, or to see a correlation between the amount of people and the accuracy on different classification techniques.

In order to come to a proper conclusion, it is important to consider data from people having different characteristics. Therefore, an effort was made to gather data from people not sharing such characteristics: height, weight, gender or hair. This data is logged anonymously and a consent form will be signed by participants. This will be discussed more comprehensively in the methodology (section 6.2.2).

6.1.3 Multiple days

Part of this research also focuses on looking at the scalability of the system for a given participant: data is gathered from a set of the same participants over several subsequent days. This data is compared only between the same participant over different days, in order to reach conclusions about the accuracies over different days for the same people. This will say something about the stability of CSI over different days, as well as how accurately participants can perform the same activity in the same manner over different days.

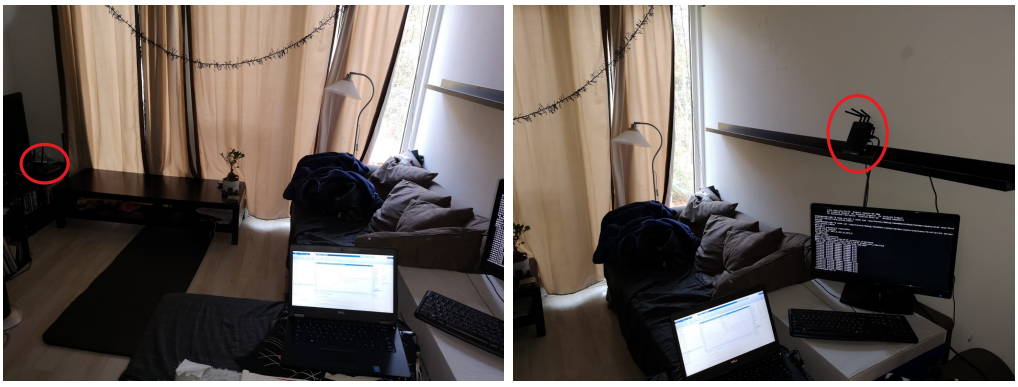
In order to keep the data consistent, participants were asked to wear the same clothes and hairstyle for the different days, as well as standing in the same position

and attempting to perform the activities in the same manner. This is useful in order to compare the data between the different participants performing activities randomly. By knowing the capabilities of CSI classification for the same participant over different days more can be said about how classifying over different participants affects the CSI signal.

6.2 Methodology

6.2.1 Experimental setup

The experimental setup was put in a small sized living room, in an actual studio (Figure 6.3). This was done to replicate an actual living situation: furniture, curtains, electronical devices and other wireless networks. The living room has a size of approximately 4x3 meters and is closed in by two full concrete walls, essentially one glass wall covered by curtains and by "open space", which leads into the rest of the studio. The setup consisted of the modified Gigabyte Brix IoT



(a) Left side of the room (transmitter in red)

(b) Right side of the room (receiver in red)

Figure 6.3: Experimental setup in the studio

(Figure 6.3b) from section 4.3, a TP-LINK AC1750 (Figure 6.3a) and a laptop for notes and a monitor connected to the Brix IoT for information. This screen provided information to the participants, e.g. telling them the activity to perform or how far they were into a specific activity. A mat was located on the floor to indicate the perfect location to perform the activities. Furthermore, a sofa, table

with a plant, TV furniture, chair, bookcase with books and a working desk were located within the immediate vicinity of the receiver and transmitter.

While the participants were performing the activities in the room, analysis of the data was done immediately and right next to the participants (at the two monitors). This was done to both monitor the participants, as well as adding some "living external factors". While no sudden or impactful movements were made, it would still be interesting to see if this would greatly affect the accuracy of the participants. This was done for both experiments regarding different participants and same participants.

6.2.2 Participants

As part of this experiment focuses more on the influence of different people on the accuracy of the classification, repeated data was captured from different people. An attempt was made to have people with strongly different characteristics when it comes to height and weight. While this information remains anonymous, different people of different heights and weights were asked to help. In total, 9 people participated in the experiment over the course of three days: three people on Day 1, three people on Day 2 and three people on Day 3. People were welcomed over the course of the day, depending on their time and schedule, so there was no consistency during experiment times (Table 6.1).

The second part of this research focuses on looking into the stability of data over the course of multiple days for a given participant. This means that over three subsequent days additional data was gathered for two participants performing the same activities (excluding jumping). The two participants always performed their experiments immediately after the other and in the same order, but at different times during the evening (Table 6.1).

The experiments were conducted under informed consent. Each participant was required to read and sign the consent form found in Appendix E. By signing the form, participants agreed to understanding the rules, their data being collected and potentially shared with other researchers and only published in its aggregated form. However, data is anonymous in general and no connection can be made back to each participant. Furthermore, participation was on voluntary basis.

6.2.3 Data gathering

For all experiments, three receiving and three transmitting antennas were used. Packets were transmitted at 48 Mbps using 64QAM (1/2) modulation. For each trial, the receiver recorded 100 packets at a rate of 20Hz (one packet every 0.05 seconds), thus meaning that every trial took 5 seconds. After these 5 seconds, the program was halted for a single second to allow it to flush the data. However, even this single second seemed to be short, as data could still be written into the file of the next trial. This means some trials were a bit shorter than 100 frames (95-99), and some were a bit longer (101-105). Fortunately, this is not a huge percentage of the packets and therefore no attempts were made to fix this. In total, every activity took between 9 and 10 minutes to complete, except for falling, which took 15 to 20 minutes due to participants having to get up again.

Different participants

The introduction to the experimental setup was done in a similar way for every participant. First, participants were welcomed to the location and the consent form was given. Meanwhile, any general questions were answered. After signing the consent form, a quick and global explanation of the global and setup was given. Then, it was explained to the participants which activities they were supposed to do and in what manner. Lastly, the participants were shown what happens to their data and they were ensured that no degree of personal information could be extracted, thus ensuring nothing could be led back to them.

During the experiments, participants were asked to perform the activities. However, participants were allowed to perform the activities as they pleased. This means that each participant was allowed (and encouraged) to switch the way they performed activities between or during different trials. Furthermore, participants were allowed to face whichever direction they pleased. However, they were asked to stay on the mat during clapping and waving. For doing nothing, participants were asked to sit on the sofa underneath the receiver and to pretend they were watching television or reading a paper. During walking, jumping and falling participants were allowed to move around the room, but to not venture too deep into the room (past the laptop).

Activities were performed in the same order for every participant, except for

the first. For the first participant, the order was i) doing nothing, ii) clapping, iii) waving, iv) walking, v) jumping and vi) falling. However, it was deemed that this was too exhausting: performing walking, jumping and falling in a row was physically very demanding. Therefore, it was decided to switch the order up to the following for the remaining participants: i) clapping, ii) walking, iii) waving, iv) jumping, v) doing nothing and vi) falling. Between each activity there was 30 seconds break, in which the participants could take a small break and get ready for the next exercise. It is important to note that all wireless devices within the room were turned off: participants were asked to turn their phones on airplane mode or off and all computers and laptops were disconnected from the wireless network. On the other hand, participants were allowed to talk and were encouraged to have a conversation while performing their tasks.

Same participants

As the two participants were already familiar with the setup, no introductions were given. This time, participants were instructed on where to stand and how to perform the activities, as well as reminding them that the clothing style and hairstyle should be consistent throughout the experiments of the different days. Participants were shown what happens to their data once again and more specific questions were answered this time.

The same activities were performed excluding jumping, as this was deemed too physically demanding. However, activities were made consistently. For clapping, it was now required to clap at chest-level at a consistent pace for all three days. Walking had to be done in at a constant pace in a square throughout the room. Waving was a big wave next to the body and every 10 trials, the arm was switched. Doing nothing was now sitting still. Finally, falling was done by falling on a pillow with the knees and then fall with the chest area on the sofa.

Like the experiments for different participants, devices were disconnected from the wireless networks and other electronic devices (like the TV and computer) were turned off. Only one laptop was on, which is the laptop on the side at which data analysis happened. Participants were also encouraged to talk, in order to keep a consistency between the experiments with different participants and same participants.

6.2.4 Analysis

The analysis for these experiments is done in the same as was done in the previous chapter: classification was done through a convolutional network based on both the RGB and monochrome images containing the regular image, slope image and variance image. The slope and variance were both based on 10% of the frame size and with a 50% overlap. The data is compared in order to establish which solution provides the highest accuracy to classify all the data.

Furthermore, the data is considered per person. This first includes classifying the images per person, to look in the stability of the data for a specific person. When the stability has been verified, the focus will be put on the scalability of the system: different amounts of participants of the group are used to train the neural network and the other part of the group is used for validating the convolutional network. The combinations of participants for training and testing are random, but all combinations for each day are considered.

Data is also considered per day as previous research reported that CSI is not reliable over multiple days [49]. This is done by classifying all data over the days combined and then classifying. Data over days is also considered in combination of different days: two different days are trained with and the third one is used for testing. This is done to look at data consistency over the days. However, a lack of consistency is expected multiple people perform activities differently over time. Therefore, two participants were asked to perform the same activities (excluding jumping) over three subsequent days in order to look at the stability of data for the same person. For each of these participants, their individual activities are classified in order to look at data stability, but it is expected this achieves high accuracies. Furthermore, data for each participant is trained with each day and then tested with every other day (resulting in 6 classifications per participant) and trained with two out of three days and classifying with the remaining day (resulting in another 3 classifications per participant).

Like the previous chapter, the neural network consists of one convolutional layer, consisting of 40 filters and a window that scans every graph in the input image. The learning rate was 0.001. The batch size was incredibly low (16-32) in order to keep processing possible. No dropout was used. Classification for the individuals, over a given day and all days was done by using 66% of the data for

training and using the remaining data for testing.

6.3 Results

The results of all the experiments will be discussed below, and the mentioned averages of the accuracies are based on generated confusion matrices. The tables mentioned in this chapter can be found at the end of the chapter (Table 6.2-6.16).

6.3.1 Different participants

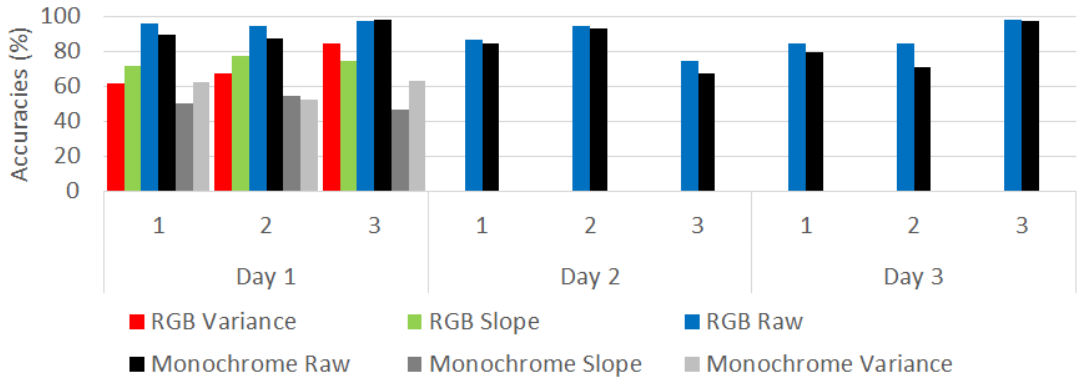


Figure 6.4: Figure showing the accuracies per participants per day. *Note that the first day also offers information regarding slope and variance data*

First day

For the first day, for each participant (3) the data was individually classified, then the data the entire day was classified (1) and then combinations of participants were trained with and classified (6). This resulted in 10 classifications. For each classification, 6 confusion matrices were plotted. Thus, this resulted in 60 confusion matrices.

For the individual analysis on the first day (Table 6.2), accuracies were quite high per individual when using the raw RGB data, with 95.73% on average. This dropped to an average of 91.43% for the monochrome raw data. Interesting to note is that for a certain participant, the monochrome was more accurate than

the RGB: 98.0% against 97.0%. For the slope data, the RGB rendered images had a reached a higher accuracy for each participant compared to the monochrome images, with average accuracies of 74.37% and 50.33%, respectively. The same could be said for variance, with accuracies for the RGB and monochrome of both 71.07% and 59.53%, respectively. It is interesting to note that in RGB, slope analysis is more accurate than variance, but that this is the other way around in monochrome.

When combining the data of all participants for training and classifying, the system reached an accuracy of 89.10% for the RGB and 84.9% for the monochrome raw data. Looking at the slope analysis, RGB images scored 62.5%. The monochrome equivalent scored 47.7%. Looking at the variance, the accuracies were 61.5% and 59.5% for RGB and monochrome images, respectively.

Classifying for two different participants and testing with the other resulted in lower accuracies (Table 6.3). As all possible combinations between the three participants are considered, this resulted in three different scenarios. On average, the highest accuracies were still reached with RGB and Monochrome raw data: 41.87% and 37.40%, respectively. For the variance, this was 36.67% and 34.27% for RGB and monochrome on average, respectively. Lastly, for the slope analysis the average accuracies were 40.37% for RGB and 33.37% for monochrome.

Lastly, a single person was classified and then tested with the other two (Table 6.4). This resulted in another three scenarios to be considered. For the raw RGB and monochrome images, the accuracies were still the highest: 36.30% and 34.80%, respectively. For RGB, slope analysis resulted in a slightly higher accuracy than variance on average: 31.00% and 30.13%, respectively. However, this was different for the monochrome data: 28.67% and 33.40% for the slope analysis and variance, respectively.

Second day

As the slope and variance accuracies were always lower compared to the raw data, they were removed from the classification. This was done in order to save time: as the accuracies were lower and computation costs and time high, it was more convenient to focus on the raw data. Therefore, only 20 confusion matrices needed to be created: 2 for each classification (RGB raw and monochrome raw). Further-

more, as the results for the first day were really high and stable, participants were encouraged to be even more inventive when it came to the way they performed activities and to slowly move around the room as they performed them.

The fact participants were more encouraged to perform activities differently could be seen in the accuracies, as most were lower for individual participants. For the RGB data, the accuracy was 84.97% and for the monochrome data it was 81.67%. These lower accuracies were mainly caused by a participant that performed every trial for each activity differently and moved around quite a bit, with accuracies of 74.5% and 67.6% for the RGB and monochrome, respectively. When classifying all the participants over the day, the accuracies were 82.3% and 74.7% for RGB and monochrome, respectively.

When training with two people and testing with the other, the accuracies were 47.10% and 39.87% for RGB and monochrome raw data, respectively. For Combination 1 and 3, the accuracies of the RGB data were even over 50%, with 58.3% and 50.3%, respectively. When trained with only a single person, the accuracies were 34.13% and 32.17% for RGB and monochrome, respectively. The lowest accuracies were 29.2% and 28.5% for RGB and monochrome.

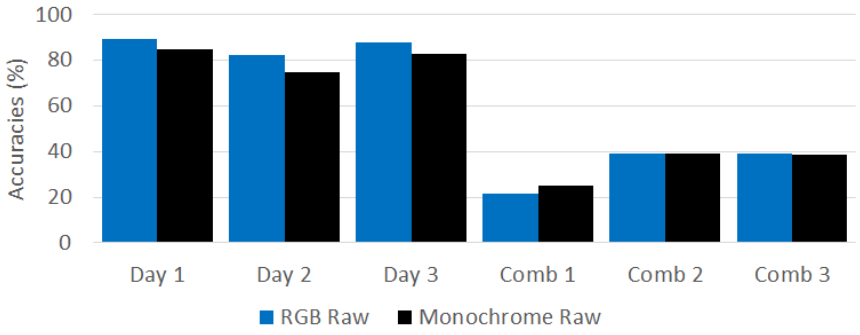


Figure 6.5: Figure showing the accuracies of each day overall and combinations between days (training and testing)

Third day

With the lower accuracies on second day, participants were asked to perform activities a bit stricter again: movement became more limited again and activities had to be performed more defined once again. Once again, these effects could be

seen in both the RGB and monochrome data: 88.87% and 82.37%, respectively. This time, the lower accuracy was 84.3% for two of the three participants, with the third one having an accuracy of 98.0% (for the RGB data). When training with all participants and classifying over all participants (67% for training and 33% for testing), the accuracies were 87.7% and 83.0%, for RGB and monochrome, respectively.

When classifying for two people and training for the other, average accuracies were down to 35.23% and 33.67% for RGB and monochrome, respectively. This time, the highest accuracies for the RGB and monochrome were 44.0% and 39.7%, respectively. For training with one person and classifying the other two, the average accuracies were 32.17% and 32.83% for RGB and monochrome, respectively. The highest in this case were 38.0% and 34.5%, in the order of RGB and monochrome.

Over the days

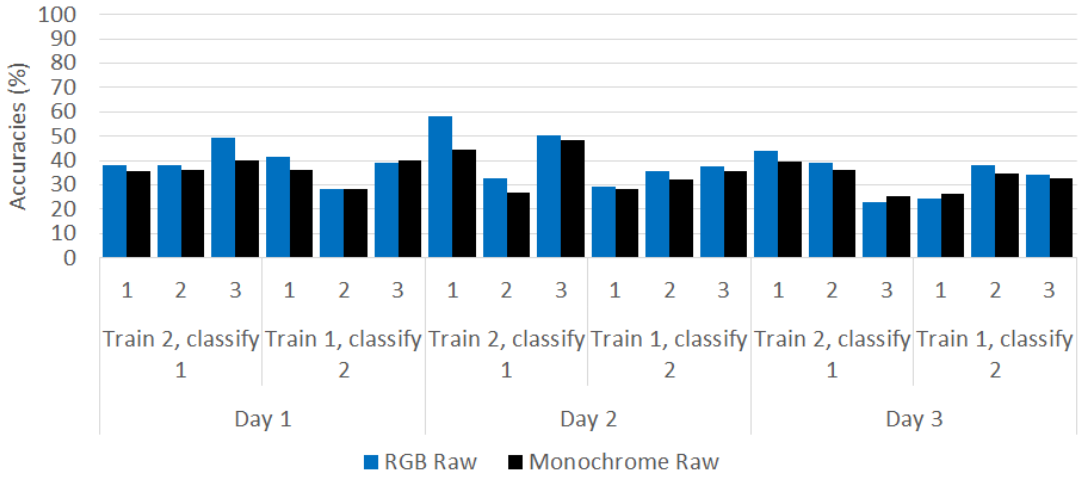


Figure 6.6: Figure showing the accuracies for combinations of training and classifying with different numbers of participants per day

When classifying for the full days, the accuracies of all participants together were quite low compared to the individual days: 78.5% and 67.6% for RGB and monochrome, respectively. When combining two days for testing and testing with

the other, accuracies were once again low (Table 6.12). On average, the accuracies for these experiments were 33.36% and 34.33% for RGB and monochrome, respectively. The lowest accuracy here was 21.6% for RGB images and 25.2% for monochrome images. The highest were 39.3% for RGB and for monochrome it was 39.0%. The final experiment was done by training all data from all days, except for a single participant, and then classifying said participant with the trained network. This increased the acccuracy to 56.3% for both the RGB and monochrome.

6.3.2 Same participants

Analysis for the same participants is done differently than for the different participants: whereas for different participants data was analysed per day, for the same participant this is done by looking into training with different days. First, the accuracies per day are discussed for both participants, then training with one day and testing with the others and lastly, training with two days and testing with the remaining day.

Individual day accuracies

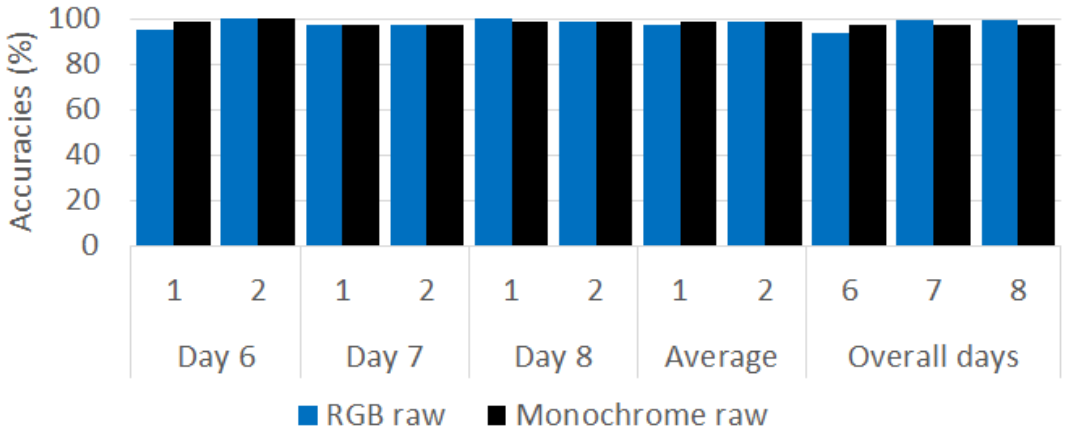


Figure 6.7: Accuracies for individual days per participant and overall for same participants over multiple days

The individual day accuracies were extremely high, even reaching 100% in three cases: twice in RGB and once in monochrome. On average, the accuracies for both

participants in RGB were 98.22% and for monochrome 98.60%, which is only a minor difference. The lowest accuracy was measured for Participant 1 on Day 6, namely 95.3% and 98.8% for RGB and monochrome, respectively. Participant 2 had an accuracy of 100% on Day 6 for both RGB and monochrome classification. The classification accuracies can be found in Figure 6.7 and Table 6.13.

Training with one day

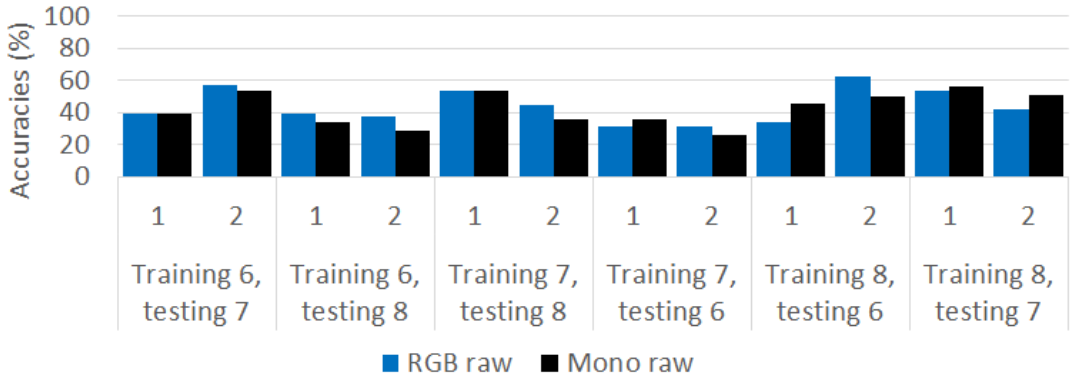


Figure 6.8: Accuracies for training with one day and testing with another day

Accuracies dropped when training with one day and testing with another. On average, accuracies were for both participants were 39.70% and 39.50% for RGB and monochrome, respectively. However, certain combinations between days reached accuracies over 50%, up to a maximum of 62.4% (RGB). On the other hand, some accuracies went as low as 28.4% (monochrome). On average, Participant 2 (45.67%) had a higher accuracy compared to Participant 1 (42.00%). However, these differences are only minor when considering the full scale of $0 - 100\%$. These accuracies are visualized in Figure 6.8 and all values are shown in 6.15.

Figure 6.8 shows that the higher accuracy for Participant 1 can be contributed two instances: *Training 6, testing 7* and *Training 8, testing 6*, in which the difference is $20 - 25\%$. In other instances, it is common for Participant 1 to have a 5% higher accuracy than Participant 2, the biggest difference being 10% (for *Training 7, testing 8*). Looking at the accuracies, if the combination of a specific training and specific testing day achieves a high accuracy, then the opposite combination

is either i) equally good (*Training 7/8, Testing 8/7* for Participant 2) or ii) a big decrease in accuracy (*Training 6/7, Testing 7/6* for Participant 1).

Training with two days

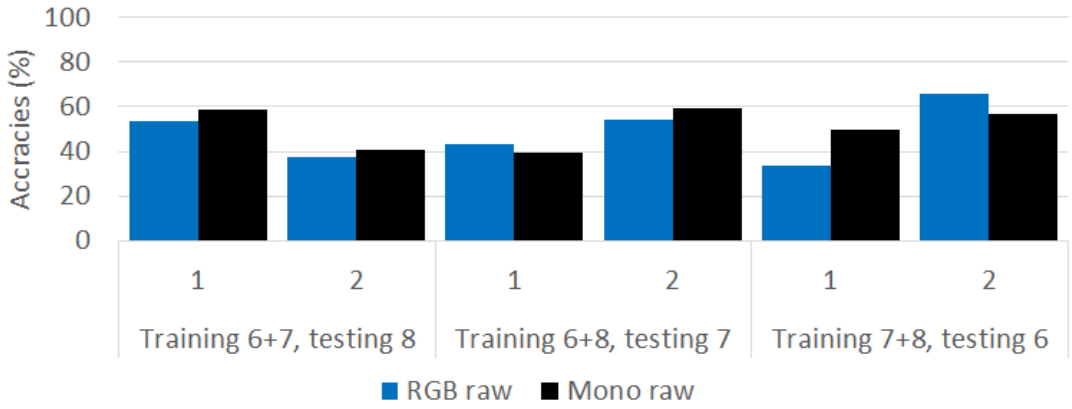


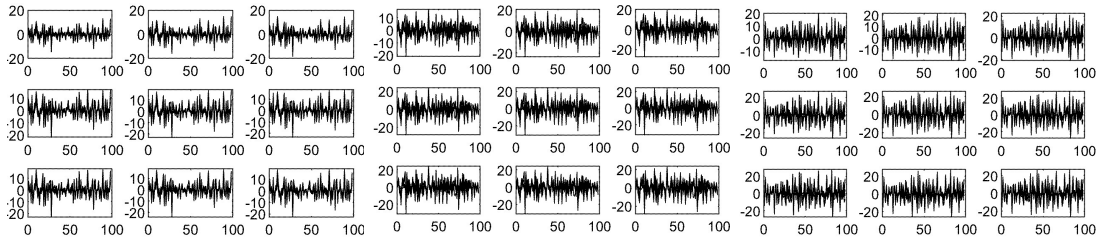
Figure 6.9: Accuracies for training with two days and testing with another day

Accuracies increased slightly when training with two days and classifying with the remaining day (Figure 6.9, Table 6.16). On average, accuracies were 47.93% and 50.73% for the RGB and monochrome classifications, respectively. The highest accuracy was measured for Participant 2 when training with Day 7 and 8 and testing with Day 6, namely 65.6%. The lowest accuracy measured was 33.6% using RGB for Participant 1. On average, Participant 2 once again reached a higher accuracy than Participant 2: 52.6% against 43.6%, respectively.

Looking at Figure 6.9, Participant 2 has higher accuracies for *Training 6+8, testing 7* and *Training 7+8, testing 6*: the accuracies are $20-30\%$ higher. Participant 1 has a higher accuracy in the case of *Training 6+7, Day* 8: the accuracy is 15% higher. This explains why the average for Participant 1 is lower than Participant 2. When looking at the highest accuracies of each day, the average of these are 57.33%.

6.4 Discussion

When classifying activities of an individual performing activities freely, the accuracies were high ($> 90\%$) when considering the raw RGB data and even for the monochrome data the accuracy was right below 90% for the first day, in which activities were performed quite similarly and with little movement. This decreased for the second day, as activities were more inconsistent and there was more movement. On day three, these activities were once again a bit higher, as the movement during trials was minimized again. For individuals performing the activities consistently (Day 6 to 8), accuracies were extremely high ($> 95\%$) and in some cases even 100% for all days. This implies that consistent activities achieve higher accuracies than inconsistent activities for limited data. Furthermore, throughout all activities, the participants were talking and this did not seem to influence the accuracies. This could be as it is the same influence over all trials (and activities) and the convolutional network thus ignores this.

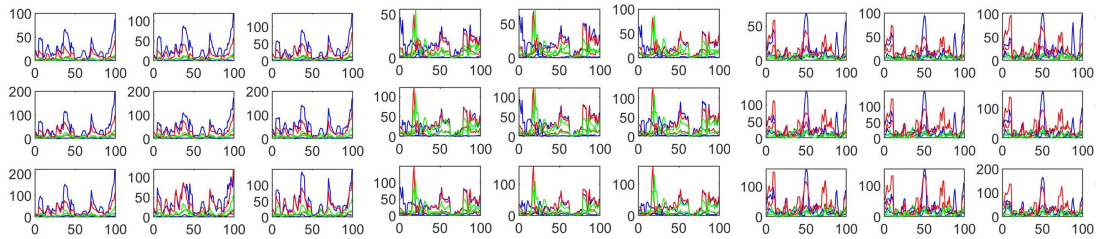


(a) Clapping (b) Falling (c) Jumping

Figure 6.10: Slope analysis in monochrome for three activities

Looking at feature extraction, variance and slope scored much lower than the raw data. The reason they did has to do with that they indicate when things start, but fall short when detecting what happens when an activity stabalizes. Thus, they essentially only detect the peaks - and peaks from different activities can look alike. This effect is shown in Figure 6.11. It is likely that monochrome slope analysis loses most of its accuracy due to the signals being all black and cluttered together and therefore multiple activities look alike (Figure 6.10). It seemed that during feature extraction, the convolutional network lost features to select from and was thus limited in what it could learn from. Therefore, the network started to confuse waving and clapping, as well as jumping and falling.

It is likely that waving and clapping are alike, both being hand movements in the air. The main difference is that clapping causes an explosion of air and sound which echoes off the walls for a longer period of time. This can often be seen in signals by seeing an individual peak and then some smaller ones. On the other hand, waving causes an immediate effect as it affects the propagation, but there are no to little disturbances after the actual wave. whereas jumping and falling also cause a big impact on the signal. For jumping and falling, both are full body activities and therefore both cause a huge impact on the CSI. The main difference would be that jumping could happen more frequently within a single trial and that the body still affects the signal afterwards. For falling, the entire body suddenly disappears. Depending on the fall, this could be a huge peak in the signal, or just a permanent change in the remaining trial.



(a) Clapping (b) Falling (c) Jumping

Figure 6.11: Variance analysis in RGB for three activities

When classifying all the different participants on a single day, accuracies still remained quite high, with 89.10% for the raw data and 84.9% for the monochrome. However, they did drop for variance (62.5% and 47.7% for RGB and monochrome, respectively) and slope (61.5% and 59.5% for RGB and monochrome, respectively). This is likely due to the aforementioned reasons: slope analysis is too cluttered (especially in monochrome) and variance only shows the impact peaks, which can be shared between activities. When combining data over multiple days (training with two and testing with one) for different participants, accuracies were around 35% on average. This is comparable to the accuracies seen when training with two participants and testing with a single participant, which resulted in 35% to 40%.

Accuracies were higher for the same participant performing the activities over multiple days. For training with one day and training with the other, accuracies were between 40% and 50%, sometimes going up to 60%. For training with two

days and classifying with the other, the average accuracies were around 50%. The lowest accuracies were increased from 28% to 33%, whereas the highest went from around 60% to 65%. This implies that classifying with different people has a negative effect on the accuracies when limited data is available. However, in both cases, it is strongly implied that more data results in higher accuracies.

Interestingly, it seems that participants performing activities inconsistently leads to a lower individual accuracy, but to higher accuracy rate when combining with other participants. This is likely because it becomes harder to classify their own activities as they are inherently different, but when combined with a participant with a higher individual accuracy in order to classify another participant, this inconsistency leads to a broader learning set and thus a higher accuracy. This can be seen during second day, where a single participant had quite a low accuracy for its own data (74.5% and 67.6% for RGB and monochrome, respectively), but when using this participant for classification with another participant, the accuracy when classifying a third participant resulted in higher accuracies compared to the other days (58.3% and 44.3%) (Table 6.6, Combination 1 and 3). Thus, higher individual accuracies imply that participants were more consistent with their activities between trials, but this makes it harder to classify among different people as the data is really personalized.

Furthermore, when classifying all different participants for a single day, the average accuracies were quite high with 88.87% and 82.37% for RGB and monochrome images. When classifying all the participants at the same time, this dropped to 78.5% and 67.6%, respectively. This verifies the claims made during the experiment: while participants were encouraged to perform activities differently, for each day the amount of freedom in performing the activity was different: day one was quite strict, day two allowed a lot of movement and day three allowed minimum movement, but more freedom in the activities themselves. This leads to higher accuracies for the days themselves, but to a lower accuracy when classifying all the days together, as the data from each day is potentially inherently different.

It is also likely that the stable CSI was slightly different for every participant. As mentioned in the methodology, effort was made to replicate a living room. Therefore, the environment was always different: chairs were moved around, curtains moved or windows opened. These all affect the CSI as propagation between the router and the node. This also causes the data to be slightly different for each

participant.

When looking at previous experiments done over days [49], accuracies for unique participants over days were as low as results reported there: between 20% and 40%. The main difference is that it was reported more data leads to less accurate classification, but with convolutional networks it seems that this is not the case: training with 8 pieces of data over different days and then classifying another participant resulted in a higher accuracy (close to 60%).

However, it should be noted that previous research [49] used the same person to perform the activities over days: this should thus be compared to the experiments related to the same participants over multiple days. Looking at the accuracies for training a single day and testing for a single day for both participants, these were already higher than the previously found results: 43.83%. Some of these results were as high as $55 - 60\%$ based on a single day of data. Comparing this to the experiments for different participants, this was as low as $30 - 40\%$ on average. For training with two days and testing with the other, the average accuracy was slightly higher once again, being 50.73% when considering the monochrome data. This implies that data is scalable over multiple days for a specific participant, but more data from the participants should be gathered.

For the two participants performing the same activities over multiple days, one specific participant often reached higher accuracies. This could be explained by one participant performing activities more consistently than the other over multiple days, or the clothing styles of both participants. The participant who had the higher accuracy was wearing clothes that tighter around the body, whereas the other participant was wearing lose clothing. This could cause different effects on the propagation of the signal when these are moving.

Interesting to note is that in some cases, certain combinations of training and testing days achieve high accuracies, but the opposite combination would have a low accuracy (6.8, (*Training 6/7, Testing 7/6* for Participant 2)). This can be explained by the training day consisting of slightly different interpretations of the same activity (a broader spectrum of activities), as this would create a larger overlap when trying to classify the testing day. Then, when turning this around, the training day is now much shallower compared to the broader spectrum, and therefore having much lower accuracies. On the other hand, some combinations achieved high accuracies in both directions (6.8, (*Training 7/8, Testing 8/7* for

Participant 2)). This could be explained by these two days sharing the exact same interpretation of the same event.

6.5 Conclusion

In general, raw data is the most promising to look at, reaching accuracies of 98% when classifying for a specific individual and 90% when classifying for multiple participants. In case of group training and testing on specific participants, the accuracies dropped to 50%. Raw RGB data outperformed the monochrome data on average by 5% to 10%. Variance and slope data should not be used for classification, as the average accuracies for individuals were between 50% and 75%, and for group analysis this was 30% to 50%, when considering both RGB and monochrome. Therefore, it can be concluded that a focus should be put on the analysis of raw data (containing all the features) in RGB and potentially monochrome, in order to reduce the complexity of images and thus decrease computation time.

Wen it comes to the lower accuracies for different participants, it is likely that this is due to some moving around more than others. The reasoning behind this is that every participant added a lot of randomness, but only those that moved around more actually had lower accuracies. This could be seen on the second day, in which movement was encouraged, but on day three the movement was more restricted once again. However, participants with lower accuracies seem to be more useful for classifying other participants. This is likely due the fact they offer a wider range of possible ways to perform the activity, especially when combined with people other people.

It seems that data between different participants and days results in lower accuracies. However, this is likely due to the fact that there is simply not enough data available. When looking at using a single participant to classify two participants, the accuracy was 35% on average. When adding more data in the form of another participant and classifying a single participant, the average accuracy was increased to close to 45%. When using eight participants to classify a single one, the accuracy was close to 60%. When we continue this line to potentially get 85 and 95 accuracy, this would result in needing a lot of participants in order to gather data. On the other hand, this could be decreased by making distinctions in activities itself: e.g. waving with left hand, waving with right hand, waving above

head, waving with two hands. It is likely this would greatly increase the accuracy, as well, as this is the case. This could be done by recording activities so there is a ground truth to consider.

Accuracies for the same participant over multiple days were higher compared to different participants over multiple days. This is likely due to the fact that the same person can perform the same activity more consistently, but the CSI still changes on a daily base due to environmental effects. However, with limited data and instable CSI, accuracies of $50 - 60\%$ were still reached for multiple day classification, which is higher than the current state-of-the-art. Also, individual analysis of the same participants was higher than current state-of-the-art, with accuracies reaching as high as 100%. The difference between two participants seems to be dependent on either a) the physique of the participants or b) the appearance (e.g. hair and clothes) of the participant.

Overall, convolutional neural networks can be used to accurately estimate activities for individuals based on both the RGB and monochrome data. The accuracy decreases when classifying over multiple days and different people to as low as 25%. Accuracies for the same participant over multiple days was closer to 40% when training on one day and 50% when training for two days. These low accuracies are likely due to a combination of single node not being able to properly detect everything that is happening and a lack of data.

Participant	**Time**
Day 1	
P1	15:00
P2	16:00
P3	20:00
Day 2	
P4	11:30
P5	13:00
P6	17:30
Day 3	
P7	14:00
P8	17:30
P9	19:00
Day 6	
P1	20:00
P2	21:00
Day 7	
P1	21:00
P2	22:00
Day 8	
P1	19:00
P2	20:00

Table 6.1: Table showing the times at which the experiments were conducted for a given participant. Note that participant 1 and 2 are different for Day 1-3 and Day 6-8.

	RGB			**Mono**		
	Raw	**Slope**	**Variance**	**Raw**	**Slope**	**Variance**
Participant 1	96.1%	71.6%	61.8%	89.2%	50.0%	62.7%
Participant 2	94.1%	77.2%	67.3%	87.1%	54.5%	52.5%
Participant 3	97.0%	74.3%	84.2%	98.0%	46.5%	63.4%
Average	95.73%	74.37%	71.10%	91.43%	50.33%	59.53%

Table 6.2: Table showing the classification rates when classifying for a single participant (randomly ordered) for the first day

	RGB			**Mono**		
	Raw	**Slope**	**Variance**	**Raw**	**Slope**	**Variance**
Combination 1	38.1%	40.5%	36.5%	35.8%	32.1%	33.1%
Combination 2	38.3%	34.3%	35.7%	36.3%	36.7%	36.3%
Combination 3	49.2%	46.2%	37.8%	40.1%	31.3%	33.4%
Average	41.87%	40.37%	36.67%	37.40%	33.37%	34.27%

Table 6.3: Table showing the classification rates when training with two participants and testing with the other for the first day

	RGB			**Mono**		
	Raw	**Slope**	**Variance**	**Raw**	**Slope**	**Variance**
Combination 1	41.4%	28.5%	27.9%	36.2%	25.9%	24.5%
Combination 2	28.4%	33.2%	31.4%	28.2%	27.2%	30.4%
Combination 3	39.1%	31.3%	31.1%	40.0%	32.9%	32.6%
Average	36.30%	31.00%	30.13%	34.80%	28.67%	33.4%

Table 6.4: Table showing the classification rates when training with one participant and testing with the other two for the first day

	RGB raw	**Mono raw**
Participant 1	86.3%	84.3%
Participant 2	94.1%	93.1%
Participant 3	74.5%	67.6%
Average	84.97%	81.67%

Table 6.5: Table showing the classification rates when classifying for a single participant (randomly ordered) for the second day

	RGB raw	**Mono raw**
Combination 1	58.3%	44.3%
Combination 2	32.7%	27.0%
Combination 3	50.3%	48.3%
Average	47.10%	39.87%

Table 6.6: Table showing the classification rates when training with two participants and testing with the other for the second day

	RGB raw	**Mono raw**
Combination 1	29.2%	28.5%
Combination 2	35.5%	32.3%
Combination 3	37.7%	35.7%
Average	34.13%	32.17%

Table 6.7: Table showing the classification rates when training with one participant and testing with the other two for the second day

	RGB raw	**Mono raw**
Participant 1	84.3%	79.4%
Participant 2	84.3%	70.6%
Participant 3	98.0%	97.1%
Average	88.87%	82.37%

Table 6.8: Table showing the classification rates when classifying for a single participant (randomly ordered) for the third day

	RGB raw	**Mono raw**
Combination 1	44.0%	39.7%
Combination 2	39.0%	36.0%
Combination 3	22.7%	25.3%
Average	35.23%	33.67%

Table 6.9: Table showing the classification rates when training with two participants and testing with the other for the third day

	RGB raw	**Mono raw**
Combination 1	24.2%	26.5%
Combination 2	38.0%	34.5%
Combination 3	34.3%	37.5%
Average	32.17%	32.83%

Table 6.10: Table showing the classification rates when training with one participant and testing with the other two for the third day

	RGB raw	**Mono raw**
Day 1	89.1%	84.9%
Day 2	82.3%	74.7%
Day 3	87.7%	83.0%
Average	86.37%	80.87%

Table 6.11: Table showing the classification rates when classifying for full days

	RGB raw	**Mono raw**
Combination 1	21.6%	25.2%
Combination 2	39.3%	39.0%
Combination 3	39.2%	38.8%
Average	33.36%	34.33%

Table 6.12: Table showing the classification rates when training with two days and testing with the other

	RGB raw	**Mono raw**
	Day 6	
Participant 1	95.3%	98.8%
Participant 2	100.0%	100.0%
	Day 7	
Participant 1	97.6%	97.6%
Participant 2	97.6%	97.6%
	Day 8	
Participant 1	100%	98.8%
Participant 2	98.8%	98.8%
Average	98.22%	98.60%

Table 6.13: Table showing the individual classifications per participant per participant for strict data

	RGB raw	**Mono raw**
	Day 6	
Day 6	93.9%	97.6%
Day 7	99.4%	97.6%
Day 8	99.4%	97.6%
Average	97.57%	97.60%

Table 6.14: Table showing the average accuracies per day (overall) for strict data

	RGB raw	**Mono raw**
	Participant 1	
Day 6, **Day** 7	39.6%	39.2%
Day 7, **Day** 6	31.6%	36.0%
Day 6, **Day** 8	39.6%	34.4%
Day 8, **Day** 6	34.4%	45.2%
Day 7, **Day** 8	53.2%	53.2%
Day 8, **Day** 7	53.6%	56.0%
	Participant 2	
Day 6, **Day** 7	56.8%	53.6%
Day 7, **Day** 6	31.2%	26.0%
Day 6, **Day** 8	37.2%	28.4%
Day 8, **Day** 6	62.4%	49.6%
Day 7, **Day** 8	44.4%	36.0%
Day 8, **Day** 7	42.0%	50.8%
Average	43.83%	42.67%

Table 6.15: Table showing the average accuracies training for training and testing between different days (Day 6 and Day 7)

	RGB raw	**Mono raw**
	Participant 1	
Day 6+**7, Day** 8	53.6%	58.8%
Day 6+**8, Day** 7	43.6%	39.2%
Day 7+**8, Day** 6	33.6%	50.0%
	Participant 2	
Day 6+**7, Day** 8	37.2%	40.4%
Day 6+**8, Day** 7	54.0%	59.2%
Day 7+**8, Day** 6	65.6%	56.8%
Average	47.93%	50.73%

Table 6.16: Table showing the average accuracies training for training and testing between different days (Day 6, Day 7 and Day 8)

Chapter 7

Conclusion

Part of this thesis was to create a small node and test this node in practice. However, as a reminder, the main research question for this research is: *What is the influence of multiple people and days on CSI when classifying human activity through deep learning*, and in order to answer this question, the following research questions were considered:

RQ1 What is the correlation between feature extraction and the accuracy of CNNs when classifying human activities using a single node?

RQ2 What is the correlation between combining data of different days for training/classification and the accuracy of the system per day?

RQ3 What is the correlation between combining data of participants for training/classification and the accuracy of the system?

7.1 Node creation

Creating a small node was challenging and a lot of hardware and software issues came to light while developing a node. While a solution smaller than the current state-of-the-art has been created using the Gigabyte Brix IoT, it is likely that there are smaller solutions. However, using this solution, different nodes have been constructed that can be put to use and the software has been written for the nodes to function together in a network. The final dimensions of these nodes are $165x105x27$mm and the node weighs around $738g$. This node was used to collect

data from different people in a real-life setting during the experiments in Chapter 6.

7.2 Feature extraction

The first research question was to consider the need of feature extraction for the convolutional networks in human activity analysis. Through the experiments in both Chapter 5 and 6, it became clear that the features considered performed worse than using the raw, unfiltered data. For convolutional neural networks this makes sense: feature extraction limits the amount of features the network can learn from. This is likely what caused the slope and variance analysis to be 10% to 15% lower than the raw analysis data.

Therefore, the correlation between feature extraction and the accuracy on a convolutional network is that it lowers the accuracy (**RQ1**). While this holds true for the current implementation for inputting images, this could be different for other implementations. However, it is overall likely that convolutional networks can potentially learn more from the raw data.

7.3 Day-dependent data

When comparing data between different days, accuracies drop massively, as was seen in Chapters 5 and 6. Looking at the dynamic activities (basic handshapes), the individual day accuracies were between 86.8% and 98.8%, and combining both days and using cross-validation had an accuracy of 93.0%. However, when considering training for one day and then testing on the other resulted in accuracies as low as 25% and at maximum 45%. For the static postures, accuracies dropped greatly (as low as 10%) over multiple days of data, but it did once again increase slightly when using more days. It could be that this is because the data is unreliable over multiple days, but this could not be verified due to a lack of ground truth data. It should be noted that one day held more information than the other, which means that accuracies can be increased when activities are performed in exactly the same fashion. This also becomes apparent when looking at multiple days for the same participant in chapter 6: accuracies were around 40% for training with

one day and around 50% when training with two days of data.

Looking into the possibility of using data from different days for classification and training, accuracies were high when combining all the information within a day: on average 88.87% and 82.37% for raw RGB and monochrome data, respectively. However, when training a network using two days and verifying with the other, accuracies dropped to 33.36% and 34.33% for RGB and monochrome, respectively. Therefore, it seems that multiple participants in a day does not greatly affect the accuracy, but when combining the data, the accuracy drops greatly.

Therefore, it is heavily implied that multiple days could be used to classify other days, but one would require a lot of days with comparable data. This is not the case for data in Chapter 5 as there is no ground truth data available. However, data for both different participants and same participants accuracies do indicate that it is possible to classify over multiple days, but only when enough data is available. Accuracies were $30-40\%$ and $50-60\%$ for different and same participant data, respectively, when training with two days and training for the other. Thus, there seems to be both a negative and positive correlation between day-dependent data and the accuracy: few days of data result in lower accuracies, but it is implied that more days would increase the accuracy (**RQ2**).

7.4 People-dependent data

It is hard to share data between both different and same participants. This became clear in Chapter 6. Accuracies were high for the individuals ($90-100\%$), but became lower when the model was trained with data from several participant and then tested on another participant. Accuracies were as low as 20% to 40% for different participants and 40% to 50% for the same participant (over multiple days), and these accuracies are too low to classify with. For the different participants, this is likely because they were encouraged to perform activities differently and move around while performing these activities. Furthermore, as the activities were conducted in an actual studio, the environment was different and alive through all these experiments. Interesting to note is that continuous influence from external factors does not seem to influence the accuracies: participants were engaged in conversation most of the time, but individual accuracies were still high (up to 100%).

However, between different people the accuracies dropped to values below 30% for both groups of participants. Even different participants performing right after each other had a low classification accuracy when training for one and testing with the other. The only except for this was one participant who had an accuracy below average, but did increase the average accuracies when tested on other participants. This is most likely due to this participant performing a lot of activities extremely inconsistently, but therefore mimicking activities performed by others. For different participants, no comparison was made between either.

During the days, there were massive inconsistencies between participants: day one was quite strict compared to the other days, day two consisted of people being allowed to move around more and on day three movement was stricter again. This could be seen by the fact that the accuracy dropped when considering all participants at once. However, it should be noted that when the model was trained with eight participants and tested with only one, the accuracies closed up to 60%. This could imply that with more data, it is possible to classify behaviour of new participants.

At this point, the conclusion to what the correlation between multiple people and the accuracy of a convolutional network is that data from multiple participants caused the accuracies in the system to decrease and thus make the system unreliable. However, it is heavily implied that more data from different participants would create a more stable system again. Therefore, there is both a negative and positive correlation, depending on the amount of data available, much like the day-dependent data (**RQ3**).

7.5 Possible applications

The current, single-node system with the results found in this research could have a few use-cases. First of all, the accuracies are high for a given individual and it is implied that accuracies could be carried over when activities are performed in a similar way and in a small room. Therefore, applications of this system could be to monitor prisoners in their cells without video cameras, or to monitor patients that are hospitalized in a specific room on their own.

Chapter 8

Limitations and future work

8.1 Custom device or drivers

In order to minimize the size of the nodes, one could consider to create a custom device. By selecting the correct circuitry and components, it is likely that it is possible to create a device of the Raspberry Pi or Hummingboard Edge i2eX with a working PCIe slot and architecture able of running the correct operating system and thus the Linux CSI Tool.

On the other hand, creating custom drivers could also solve this issue. In the case of the Hummingboard and with the correct knowledge, it seems possible to modify the drivers of other network cards or the software to make it work. This would allow the creation of an even smaller node and thus in turn an even less unobtrusive node.

8.2 More complex convolutional networks

In general, deep learning solutions are very complex and cost a lot of computational power. The potential of the convolutional networks was bottlenecked by the hardware used in this research. In total, two systems were used for everything described in this thesis. The specifications of both can be found in the Table 8.1.

Using more powerful hardware would allow the deployment of more complex neural networks and this could in turn lead to more accurate solutions. Furthermore, it would also increase the speed at which networks train and classify and

	System 1	**System 2**
Processor	Intel i7-4770k	Intel i5-5200U
RAM	8 GB 1866 MHz	8 GB 1600 MHz
Hard drive	SSD	SSD
Graphics card	GTX 770 2 GB	Intel HD Graphics 5500
Operating System	Windows 10	Windows 10

Table 8.1: Used hardware for training, testing and validating neural networks

thus decrease the computation time. While these systems are likely to be quite expensive, cloud computations could be useful here: multiple systems of nodes use the same, powerful computer for their calculations.

This can also be achieved by implementing the convolutional network differently. Towards the end of the research, attempts were made to use `tensorflow` instead of `MATLAB` and this greatly improved the performance. This can be explained by `MATLAB` having several layers on top of the deep learning layer and outdated, whereas `tensorflow` is a low-level `Python` library, which is regularly updated. However, not enough time was left to convert all existing `MATLAB` code into a working `tensorflow` equivalent.

8.3 External factors

External factors, or environmental factors, affect the propagation of wireless signals as explained in the background. Different external factors affect the signal differently (e.g. scattering, refracting, absorption) and little to no research has gone into detecting and dealing with these factors. Once it is known how the signals are affected by these, it may be possible to adapt systems by compensating for these factors and thus create a more stable and reliable system. Therefore, it is likely that the accuracy can be greatly improved when these factors are countered.

This (and other current research) eliminates the external factors, which makes the systems all but reliable in dynamic environments.

8.4 Multiple nodes

While multiple nodes were created for this research, due to time constraints it was not possible to test different solutions with a variable number of nodes and their locations (and thus their density). Multiple nodes could collect the CSI from different angles and could potentially track and analyse multiple humans at once more accurately. The angle of arrival (AoA) could be used in these cases more accurately, as it could tell something about the orientation of the person. Multiple nodes could also be used in a voter like system, where each node predicts the own activities and then vote together on which activity is more likely: this will create resistance to the system. Furthermore, sensor fusion with other sensors or nodes could be used to have a more stable and reliable result.

As aforementioned, multiple nodes could potentially keep track (of the activities) of multiple humans. Current solutions all focus on a single person, perhaps multiple single persons, but not the activities of groups or people within a group. Multiple nodes would make this easier: a node could try to only classify people close to it, or look at the global picture. Especially in estimating how a group is moving or distributed around a room, multiple nodes could provide more useful information.

8.5 Activity detection

The algorithms could be used to more precisely define activities within the gathered frames. By more precisely detecting activities in the data prior to the training phase, training could be focused on more important features of the activity. Furthermore, the frames could potentially be made smaller and this would increase the speed at which training and classification takes place.

Recognizing different parts of the activity could be another important part. For example, one could more accurately extract features from within the activity and view activities as the sum over different movements. This would make it possible to have a more scalable system, as not every activity has its signature on the CSI, but rather, every movement has a different CSI signature, but activities have a different movement signature. This would result in systems detecting the movements and from this classify the activities. This could potentially learn to

better real-time systems: whereas it is now unknown how long CSI entries should be, but should be at least include a large part of the activity, for movements the entries can be shorter and an estimation can be made based on combining the movements.

8.6 More and stricter data

While this research looked into both data from different people and different days, this is by far not enough data to draw any conclusions in regards to accuracies. In order to draw some definite conclusions, data should be gathered over multiple days consistently and for a larger, but the same group. This would greatly increase the scope of any project, but only then it would be possible to safely state something about how scalable such a system truly is. However, activities should be performed more strictly (e.g. only clapping in front of the body at the same height, or waving with the left hand while facing the node) or more sub activities should be considered (e.g. "Waving with left hand; waving with right hand" or "Falling on knees; falling on the side"). It is also important to note that activities should be recorded in order to achieve a ground truth, so activities can be reviewed and data can be verified.

Recommended Reading

[1] G. A. Oguntala, R. A. Abd-Alhameed, S. M. R. Jones, and J. M. Noras. Unobtrusive mobile approach to patient location and orientation recognition for elderly care homes. In *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*, pages 1517–1521, 2017.

[2] A. Murad and J. . Pyun. Deep recurrent neural networks for human activity recognition. *Sensors (Switzerland)*, 17(11), 2017.

[3] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *Proceedings of the 2014 6th International Conference on Mobile Computing, Applications and Services, MobiCASE 2014*, pages 197–205, 2015. Cited By :55.

[4] F. Haider and G. Shaker. Wearable-free wireless fall detection system. In *2017 IEEE Antennas and Propagation Society International Symposium, Proceedings*, volume 2017-January, pages 2457–2458, 2017.

[5] X. Huang and M. Dai. Indoor device-free activity recognition based on radio signal. *IEEE Transactions on Vehicular Technology*, 66(6):5316–5329, 2017.

[6] F. Li, M. A. A. Al-Qaness, Y. Zhang, B. Zhao, and X. Luan. A robust and device-free system for the recognition and classification of elderly activities. *Sensors (Switzerland)*, 16(12), 2016.

[7] M. Dai and X. Huang. *Radio signal based device-free velocity recognition*, volume 9204 of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2015. Cited By :1.

[8] Y. Gu, L. Quan, and F. Ren. Wifi-assisted human activity recognition. In *Proceedings, APWiMob 2014: IEEE Asia Pacific Conference on Wireless and Mobile 2014*, pages 60–65, 2014. Cited By :5.

[9] Y. Gu, J. Tian, L. Zhang, Z. Liu, F. Ren, and X. Wang. Activity recognition via channel response: From theoretical analysis to real-world experiments. In *IEEE Vehicular Technology Conference*, volume 2017-June, 2017.

[10] X. Dang, Y. Huang, Z. Hao, and X. Si. Pca-kalman: device-free indoor human behavior detection with commodity wi-fi. *Eurasip Journal on Wireless Communications and Networking*, 2018(1), 2018.

[11] H. Zou, Y. Zhou, J. Yang, and C. J. Spanos. Towards occupant activity driven smart buildings via wifi-enabled iot devices and deep learning. *Energy and Buildings*, 177:12–22, 2018.

[12] C. Wang, S. Chen, Y. Yang, F. Hu, F. Liu, and J. Wu. Literature review on wireless sensing-wi-fi signal-based recognition of human activities. *Tsinghua Science and Technology*, 23(2):203–222, 2018.

[13] J. Choi, W. Lee, J. Lee, J. Lee, and S. Kim. Deep learning based nlos identification with commodity wlan devices. *IEEE Transactions on Vehicular Technology*, 2017. Article in Press.

[14] C. Han, Q. Tan, L. Sun, H. Zhu, and J. Guo. Csi frequency domain fingerprint-based passive indoor human detection. *Information (Switzerland)*, 9(4), 2018.

[15] S. A. Shah, A. Ren, D. Fan, Z. Zhang, N. Zhao, X. Yang, M. Luo, W. Wang, F. Hu, M. Ur Rehman, O. S. Badarneh, and Q. H. Abbasi. Internet of things for sensing: A case study in the healthcare system. *Applied Sciences (Switzerland)*, 8(4), 2018. Cited By :1.

[16] T. Xin, B. Guo, Z. Wang, P. Wang, and Z. Yu. Freesense: human-behavior understanding using wi-fi signals. *Journal of Ambient Intelligence and Humanized Computing*, 9(5):1611–1622, 2018. Cited By :1.

[17] N. Jing, Y. Sun, L. Wang, and J. Shan. Fine-grained wireless propagation ambience sensing. *International Journal of Distributed Sensor Networks*, 14(10), 2018.

[18] Y. . Cheng and R. Y. Chang. Device-free indoor people counting using wi-fi channel state information for internet of things. In *2017 IEEE Global Communications Conference, GLOBECOM 2017 - Proceedings*, volume 2018-January, pages 1–6, 2018.

[19] X. Wang, X. Wang, and S. Mao. Deep convolutional neural networks for indoor localization with csi images. *IEEE Transactions on Network Science and Engineering*, 2018. Article in Press.

[20] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang. Device-free wireless localization and activity recognition: A deep learning approach. *IEEE Transactions on Vehicular Technology*, 66(7):6258–6267, 2017. Cited By :6.

[21] G. Rescio, A. Leone, and P. Siciliano. Supervised machine learning scheme for electromyography-based pre-fall detection system. *Expert Systems with Applications*, 100:95–105, 2018.

[22] M. C. Rushambwa, M. Gezimati, and J. B. Jeeva. Development of a wearable wireless body area network for health monitoring of the elderly and disabled. In *IOP Conference Series: Materials Science and Engineering*, volume 263, 2017.

[23] M. Ahmed, N. Mehmood, A. Nadeem, A. Mehmood, and K. Rizwan. Fall detection system for the elderly based on the classification of shimmer sensor prototype data. *Healthcare Informatics Research*, 23(3):147–158, 2017.

[24] P. V. Er and K. K. Tan. Non-intrusive fall detection monitoring for the elderly based on fuzzy logic. *Measurement: Journal of the International Measurement Confederation*, 124:91–102, 2018.

[25] C. . Hsieh, C. . Huang, K. . Liu, W. . Chu, and C. . Chan. A machine learning approach to fall detection algorithm using wearable sensor. In *Proceedings of the IEEE International Conference on Advanced Materials for Science and Engineering: Innovation, Science and Engineering, IEEE-ICAMSE 2016*, pages 707–710, 2017. Cited By :1.

[26] J. A. Santoyo-Ramn, E. Casilari, and J. M. Cano-Garca. Analysis of a smartphone-based architecture with multiple mobility sensors for fall detection with supervised learning. *Sensors (Switzerland)*, 18(4), 2018.

[27] B. Jansen and R. Deklerck. Context aware inactivity recognition for visual fall detection. In *2006 Pervasive Health Conference and Workshops, PervasiveHealth*, 2007. Cited By :63.

[28] L. Hazelhoff, J. Han, and P. H. N. De With. *Video-based fall detection in the home using principal component analysis*, volume 5259 LNCS of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2008. Cited By :33.

[29] M. Humenberger, S. Schraml, C. Sulzbachner, A. N. Belbachir, . Srp, and F. Vajda. Embedded fall detection with a neural network and bio-inspired stereo vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 60–67, 2012. Cited By :12.

[30] T. Banerjee, J. M. Keller, M. Skubic, and E. Stone. Day or night activity recognition from video using fuzzy clustering techniques. *IEEE Transactions on Fuzzy Systems*, 22(3):483–493, 2014. Cited By :18.

[31] F. Attal, S. Mohammed, M. Dedabrishvili, F. Chamroukhi, L. Oukhellou, and Y. Amirat. Physical human activity recognition using wearable sensors. *Sensors (Switzerland)*, 15(12):31314–31338, 2015. Cited By :69.

[32] J. Chawla and M. Wagner. Using machine learning techniques for user specific activity recognition. In *Proceedings of the 11th International Network Conference, INC 2016*, pages 25–29, 2016.

[33] H. Fang and C. Hu. Recognizing human activity in smart home using deep learning algorithm. In *Proceedings of the 33rd Chinese Control Conference, CCC 2014*, pages 4716–4720, 2014. Cited By :7.

[34] Y. Chen and Y. Xue. A deep learning approach to human activity recognition based on single accelerometer. In *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pages 1488–1492, 2016. Cited By :13.

[35] M. Kulin, T. Kazaz, I. Moerman, and E. de Poorter. End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications. *IEEE Access*, 2018. Article in Press.

[36] C. Beard and W. Stallings. *Wireless Communcation Networks and Systems*. Pearson, Pearson Education Limited, Edinburgh Gate, Harlow, Essex GM20 2JE, England, 5 edition, 2016. Global edition.

[37] Y. Chapre, P. Mohapatra, S. Jha, and A. Seneviratne. Received signal strength indicator and its analysis in a typical wlan system (short paper). In *Proceedings - Conference on Local Computer Networks, LCN*, pages 304–307, 2013. Cited By :23.

[38] IEEE Computer Society. Wireless lan medium access control mac and physical layer (phy) specifications. Technical Report 2, IEEE, IEEE, 3 Park Avenue, New York, NY 10016-5997, USA, 2016. Last accessed: May 1, 2018.

[39] Linux CSI Tool. https://dhalperi.github.io/linux-80211n-csitool/. Accessed: November 26, 2018.

[40] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[41] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. 2012. Cited By: 959.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Paper presented at the Advances in Neural Information Processing Systems*, 2:1097–1105, 2012.

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

[44] A. Booranawong, N. Jindapetch, and H. Saito. A system for detection and tracking of human movements using rssi signals. *IEEE Sensors Journal*, 18(6):2531–2544, 2018.

[45] L. Guo, L. Wang, J. Liu, W. Zhou, and B. Lu. Huac: Human activity recognition using crowdsourced wifi signals and skeleton data. *Wireless Communications and Mobile Computing*, 2018, 2018.

[46] L. Liu, B. Yin, S. Zhang, X. Cao, and Y. Cheng. Deep learning meets wireless network optimization: Identify critical links. *IEEE Transactions on Network Science and Engineering*, 2018. Article in Press.

[47] J. Wang, L. Zhang, Q. Gao, M. Pan, and H. Wang. Device-free wireless sensing in complex scenarios using spatial structural information. *IEEE Transactions on Wireless Communications*, 17(4):2432–2442, 2018.

[48] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, and D. J. Inman. Wireless and real-time structural damage detection: A novel decentralized method for wireless sensor networks. *Journal of Sound and Vibration*, 424:158–172, 2018.

[49] P. Bagave. Unobtrusive sensing using wi-fi signals. *Master thesis*, 2018.

Appendices

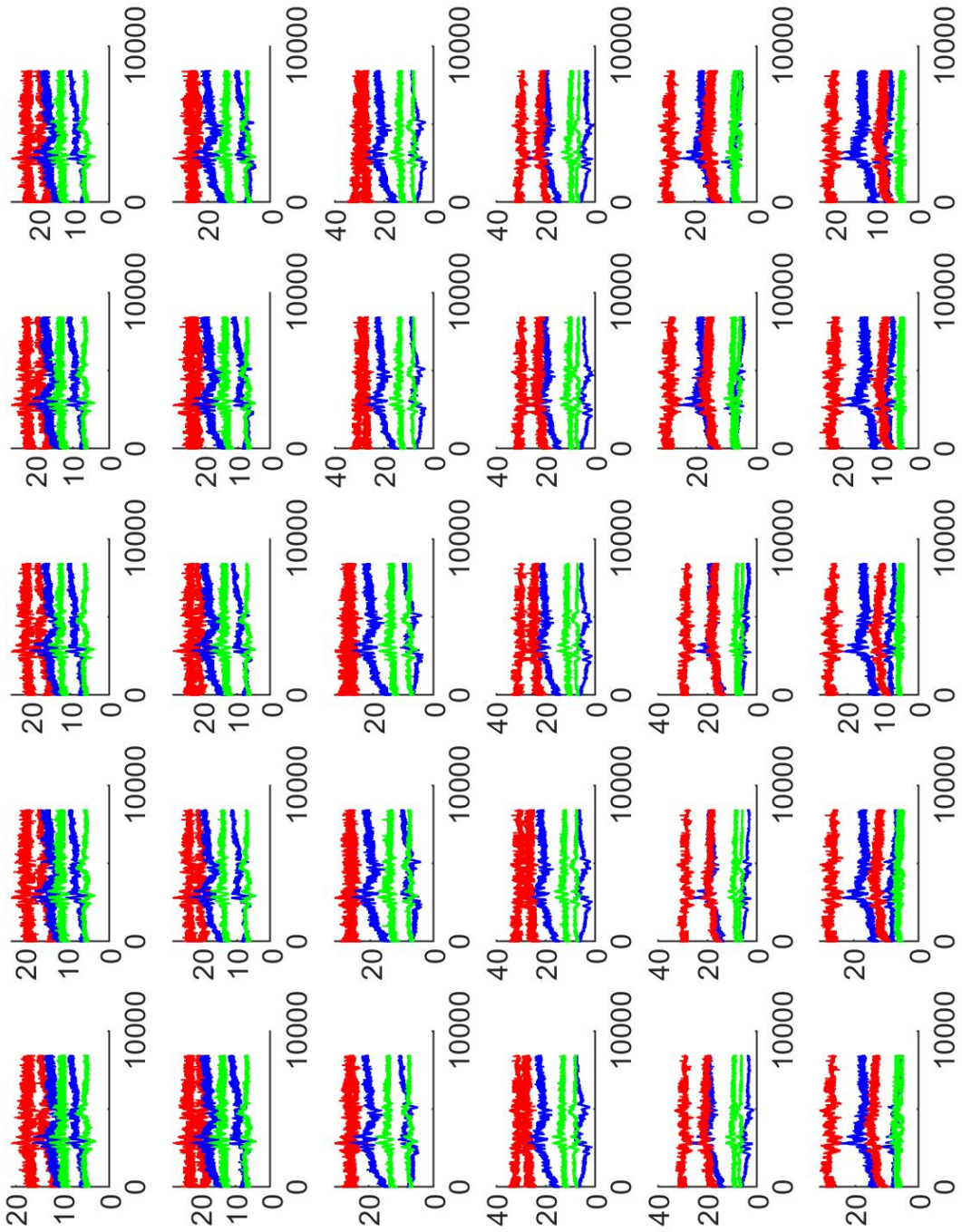
Appendix A

Amplitude for all subcarriers over frame duration

On the next page, an example can be found where all 30 subcarriers are plotted. What is plotted is the absolute (`abs()`) and `get_scaled_csi()` values, meaning this is essentially the SNR.

This example is mentioned in the background, to compare all different subcarriers for the same frames, as well as in Chapter 4 to explain what kind of images are fed into the convolutional network.

The actual image can be found on the next page.



Appendix B

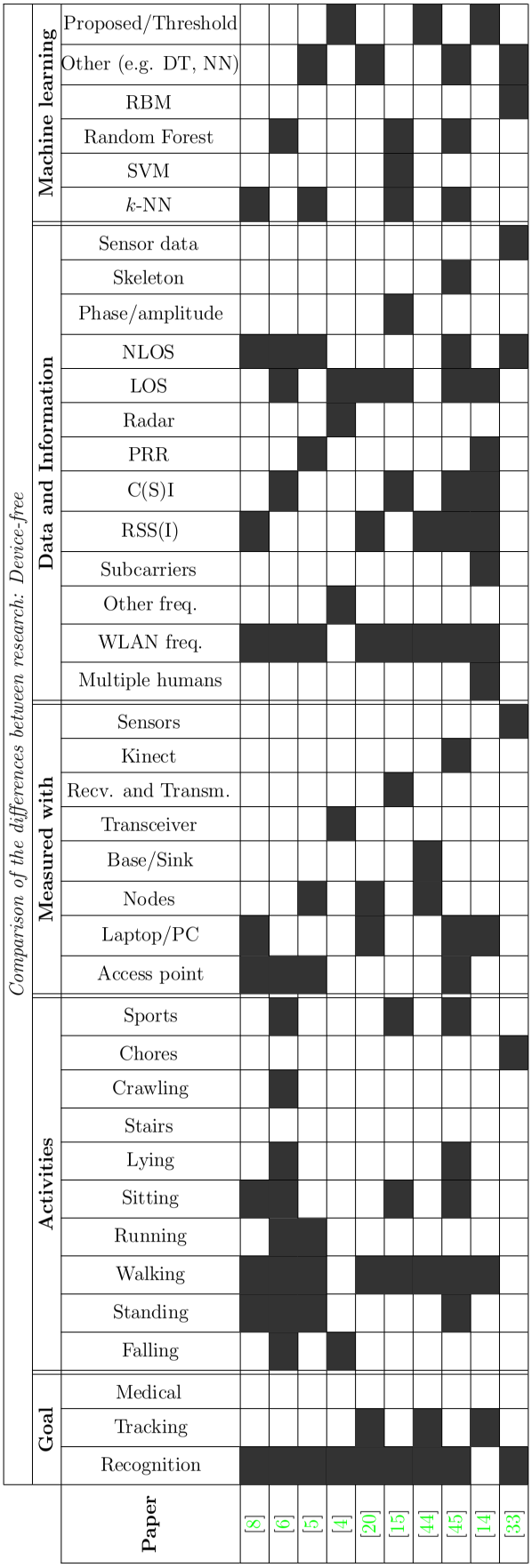
Literature guide

This literature guide shows the key differences between the related works. First, it contains a table (spread over multiple pages) containing the keywords of each research in order to quickly read up on the mentioned literature. Another table is shown which attempts to visualize the differences between each research.

Unobtrusive human activity recognition and classification (i)

Paper	Goal	Technique	Activities	Measured with	Data and information	Machine learning
Fang, 2014 [33]	Recognizing human activity in a smart home	Device-free	Bed to toilet, breakfast, bed, computer, dinner, laundry, leave home, lunch, night, medicine	Sensors in the smart home: motion sensors, temperature sensor, water, door, burner and item sensor	Sensor values (undefined, examples given)	Restricted Boltzmann Machine, HMM, Naive Bayes
Gu, 2014 [8]	Unobtrusive recognition of human activity	Device-free	Standing, sitting, walking	Access point, laptop	2.4 GHz (802.11b), received signal strength (RSS), ambient WiFi signals, NLOS	k-NN algorithm
Li, 2016 [6]	Recognize and classify human activity of the elderly	Device-free	Walk, sit down, lie, stand up, squat down, fall, and crawl	Access point	2.4 GHz, Channel state information (CSI), wireless local area networks (WLAN), (N)LOS	Random Forest
Huang, 2017 [5]	Track and identify humans based on their height	Device-free	Standing, walking, running	Four corner nodes, Access point	Different frequences (2.41 to 2.49 GHz), NLOS, packet receive rate (PRR)	SMO, k-NN
Haider, 2017 [4]	Device-free fall detection in elderly homes	Device-free	Fall	Radio transceiver	Electromagnetic wave, 3.3 to 10.3 GHz, NLOS	? (proposed)
Wang, 2017 [20]	Device-free human activity recognition and classification	Device-free	Localization (17), activities (4), gestures (4)	Eight nodes, laptop	2.4 GHz, LOS, RSS	Deep learning (supervised learning)

Shah, 2018 [15]	Detect sleepiness and sleep attacks in narcroleptic patients	Device-free	Walking, sitting on a chair, push-ups, and narcolepsy sleep episodes	Receiver and transmitter	2.4 GHz, Channel information (CI): phase and amplitude information, S-band sensing technique	mutli-class SVM, k-NN, Random Forest
Booranawong, 2018 [44]	Detection and tracking of human movements	Device-free	Walking, moving	One base, one receiver and three transmitting nodes	2.4 GHz, RSSI, threshold consideration RSSI, same altitude	Not defined (propsed algorithm)
Guo, 2018 [45]	Human activity recognition through crowdsourced WiFi signals and skeleton data	Device-free, Imaging	Sixteen activities	Transmitting node (Access point), Receiving node (laptop), Kinect	2.4 GHz (IEEE 802.11n), RSSI, CSI, skeleton data, activity segmentation	k-NN, Random Forest, Decision Tree
Han, 2018 [14]	CSI fingerprint-based human detection	Device-free	Human detection	Laptop (as receiver)	CSI, RSSI, subcarrier matrix, packet rate, multiple antennas, varied, number of packets, offline training, online matching	None, self-developed algorithm for online traning and voting


Comparison of the differences between research: Device-free
Machine learning
Proposed/Threshold
Other (e.g. DT, NN)
RBM
Random Forest
SVM
k-NN
Data and Information
Sensor data
Skeleton
Phase/amplitude
NLOS
LOS
Radar
PRR
C(S)I
RSS(I)
Subcarriers
Other freq.
WLAN freq.
Multiple humans
Measured with
Sensors
Kinect
Recv. and Transm.
Transceiver
Base/Sink
Nodes
Laptop/PC
Access point
Activities
Sports
Chores
Crawling
Stairs
Lying
Sitting
Running
Walking
Standing
Falling
Goal
Medical
Tracking
Recognition
Paper
[8]
[6]
[5]
[4]
[20]
[15]
[44]
[45]
[14]
[33]

Appendix C

Requirements explained

Functional requirements	
Must have	
Number	**Requirement explained**
FR1	Nodes must be able to collect CSI. The goal of this research is to look into activity recognition through CSI analysis. Therefore, each node must be able to capture its own CSI.
FR2	In order to have a more robust system, nodes must be able to store their data locally for at least a day. This is so that in case of a server crash, the data could still be manually extracted from a node and saved somewhere.
Should have	
FR3	In order to have a modular system, in which nodes can synchronize and need to reach decisions on activities, the nodes should be able to communicate with each other. In this case, the system is more easily deployed later in more complex scenarios.
FR4	Nodes should be free of maintenance in order for them to collect data continuously. Therefore, nodes should function in a headless manner. However, they should be reachable from a distant computer, so that they can be fixed and monitored from a distance.
FR5	Nodes should collect their own status, as this is important to increase the reliability and robustness of a system. Nodes should also be able to communicate this with one another, or tell the server or other nodes when a node has been quiet for a while.
FR6	In order to allow analysis of all the nodes, a server should collect all data and classify this. In this case, computation does not need to be done on the node, but all data is still classified. It is likely that the server is more powerful than a single node.
Won't have	
FR7	The nodes will not have a lot of computational power, and every node working on its own deep learning network seems redundant. Therefore, none of the nodes will perform any heavy computations on its own.

Non-functional requirements	
Number	**Requirement explained**
Must have	
NR1	In order for the system to be deployed, nodes must be smaller than the current state-of-the-art. Nodes are currently the size of routers or laptops, and these cannot easily be put in a situation, due to their size. Therefore, a smaller node should be developed which can more easily be put in a location.
NR2	The Linux CSI Tool is an established tool and has been proven to work. Furthermore, previous research on the subject at the University of Twente has worked with this, so therefore it was decided to work with the Linux CSI Tool.
NR3,NR4	These requirements are caused by the Linux CSI Tool, as this requires a specific kernel version and a specific piece of hardware.

Appendix D

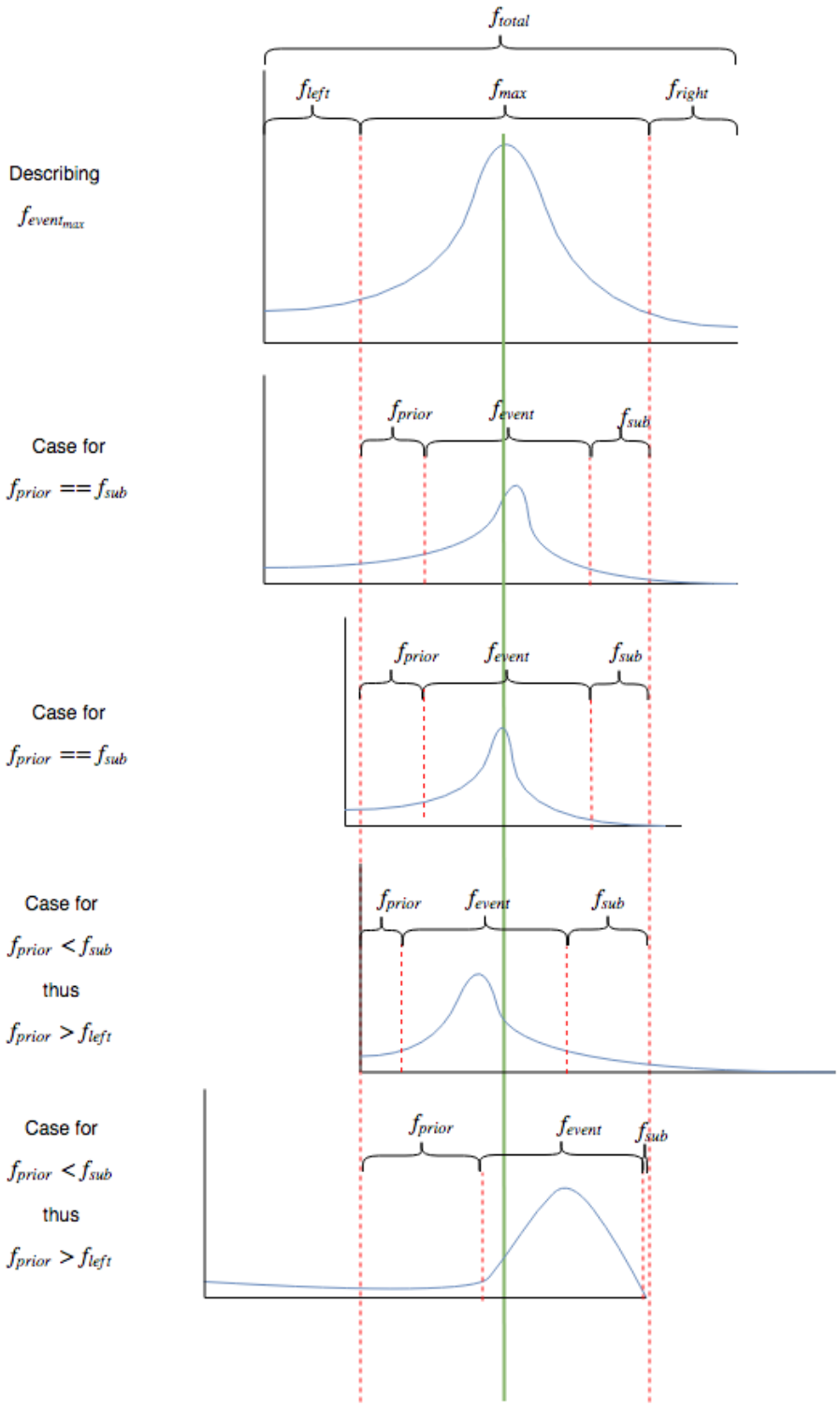
Visual representation of less interpolation

The figure on the next page can be used to better visualise the solution to use less (to no) interpolation while trying to create data of equal lengths. It is mentioned in both chapters 5, first as a theory and then as an implementation, and chapter 6, where it is once again used as an implementation.

In the figure, five examples of the frame length can be found. This is done by creating frames of different sizes (such as the third and fourth case), as well as having the event happening at different times (shown by shifting the overall graph). The situation of each of the graphs is summarized on the left.

The first graph shows the overall idea and definition of f_{total}, f_{left}, f_{max} and f_{right}. The green light indicates the center of the activity and the red running all the way down indicate the edges of the maximum frame length (f_{max}). Each graph below that contains an additional two red lines, indicating f_{prior} and f_{sub}. The first three cases are pretty normal, but the last two show cases in which there is not enough data to try to achieve the fair balance of $f_{prior} = f_{sub}$, in which case one is larger/smaller than the other.

Note that this only works for cases $f_{total} \geq f_{max}$.


f_{total}
f_{left}
f_{max}
f_{right}
Describing
$f_{event_{max}}$
f_{prior}
f_{event}
f_{sub}
Case for
$f_{prior} == f_{sub}$
f_{prior}
f_{event}
f_{sub}
Case for
$f_{prior} == f_{sub}$
f_{prior}
f_{event}
f_{sub}
Case for
$f_{prior} < f_{sub}$
thus
$f_{prior} > f_{left}$
f_{prior}
f_{event}
f_{sub}
Case for
$f_{prior} < f_{sub}$
thus
$f_{prior} > f_{left}$

Appendix E

Consent form

Informed consent form
"Device-free sensing & Deep learning"

Collecting data through experiments needed for the research on device-free sensing and deep learning for the Computer Science graduation project (MSc level) at the University of Twente

---

Purpose of the study

The goal of this study is to collect data required to continue the research. This research focusses on device-free sensing and deep learning. Device-free sensing is a new method to monitor human activity without the need of wearable sensors. This research focusses on combining device-free sensing with deep learning by looking at the accuracies. This research could be fundamental to future detecting systems.

Procedures

Participants will first be introduced to the system and experimental setup and asked some questions, as well as perform a test run of the activities. Then, the participants will be asked to perform these activities at least 50 times (up to 60), but breaks can be taken when desired. Between every activity (50-60 runs), a 5 minute break is offered. At the end, the particpant is thanked for the participation and questions can be asked.

Duration

The total duration will be 3060 minutes for one person; ± 20-30 minutes per additional person

Risk/discomfort

During the course of this experiment, participants may become fatigued. After each activity (50-60 runs), a five minute break will be offered. Furthermore, the particpant is free to take several small breaks (at request) between runs, as some runs may require more energy than othters. Also, the usual risks of performing activities in a living room are present. If a participant feels unwell, the experiment is terminated immediately.

Alternative to participation

Participation in this study is voluntary; feel free to withdraw or discontinue participation at any time.

Cost and compensation

Participation in this study will cost nothing. You will not be paid for your participation.

Confidentially

All information collected during this experiment will be anonymized. Any personal data collected during these experiments will not be published in any way, only the processed and aggregated data will be published. Even so, none of this will be linked back to the particpant. In its anonymized form, the collected data (not the personal data) may become available to other researchers.

Consent

☐ I have read and understood the information on this form.

☐ I accept the information on this form and agree to join this study.

☐ I will not discuss the details of this experiment/material with anyone.

Participant's signature _____ Date _____

Researcher's signature _____ Date _____