Multimodal deep learning for the classification of human activity

R. J. de Jong

Technische Universiteit Delft

TU Delft

Multimodal deep learning for the classification of human activity

Radar and Video data fusion for the classification of human activity

by

Richard de Jong

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday January 14, 2019 at 1:00 PM.

Student number:	4238575	
Project duration:	May 5, 2018 – January 14, 2019	
Thesis committee:	Prof. dr. A. Yarovoy,	TU Delft, responsible Professor
	Dr. F. Uysal,	TU Delft, supervisor
	Dr. D. M. J. Tax,	TU Delft
	Dr. Ir. J. J. M. de Wit,	TNO

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft TNO

Abstract

Persistent surveillance is an urgent proficiency. For security, surveillance cameras are a strong asset as they support the automatic tracking of people and are directly interpretable by a human operator. Radar on the other hand can be used under a broad range of circumstances: radar can penetrate mediums such as clouds, fogs, mist and snow, and it can be used when it gets dark. However radar data, compared to an optical sensor as video, is not as easily interpretable by a human operator. This thesis explores the potential of multimodal deep learning with a radar and video sensor to improve the classification accuracy of human activity.

A recorded and labelled dataset is created that contains three different human activities: walking, walking with a metal pole and walking with a backpack (10 kg). A Single Shot Detector is used to process the video data. The cropped frames are then associated with the start of a radar micro-Doppler signature with a duration of 1.28 seconds. The dataset is split in a training (80 %) and validation (20 %) set such that no data from a person in the training set is in the validation set.

Implementations of convolutional neural networks for the video frames and micro-Doppler signatures obtain classification accuracies of 85.78 % and 63.12 % respectively for previously mentioned activities. It was not possible to distinguish a person walking and walking carrying a backpack on basis of the micro-Doppler signatures.

The synchronised dataset is used to investigate different fusion methods. Both early and late fusion methods show an improvement in classification accuracy. The best obtained early fusion model achieves a classification accuracy of 90.60 %. Omitting the radar data however shows a drop in classification accuracy of just 0.9 %, identifying the video data as the dominant modality in this particular setup.

Contents

1

Introduction

Detecting and classifying human activity has a number of potential applications including physical security and law enforcement. Radar based activity recognition is also of great interest due to its applications in border control and security, pedestrian identification for automotive safety and remote health monitoring [1]. Furthermore, for the security in urban and built environment of large industrial sites, seaports and airports and major events and summits, persistent surveillance is an urgent proficiency.

For security, surveillance cameras are a strong asset as they support the automatic tracking of people and are directly interpretable by a human operator. However, at night or in unfavourable weather conditions such as rain and fog the performance of cameras degrades severely. For some applications cameras can also become an issue where privacy is concerned, as persons are easily recognisable.

Radar on the other hand can be used under a broad range of circumstances: radar can penetrate mediums such as clouds, fogs, mist and snow, and it can be used when it gets dark. Although micro-Doppler signatures have been identified as a potential biometric feature, humans are not easily recognizable and privacy concerns are therefore much less an issue. However radar data, compared to an optical sensor as video, are not as easily interpretable by a human operator.

Considering the recent advances in the field of machine learning, more specifically deep learning, both radar and video data can be used successfully with deep learning methods for the classification of human activity. One of the key enablers for deep learning are faster computers and optimised Graphics Processing Units (GPU's), as deep learning research often involves the use of large networks and datasets and optimisation algorithms are computationally intensive.

In this work Micro-Doppler signatures in combination with single frames from a video camera are used for classification of human activity. The possibility to improve the classification of human activity using both video, optical sensor, and radar at different fusion depths is investigated and a comparison is made with single modality trained neural networks. To validate the results and gain insight in the capabilities of deep neural networks saliency maps are used.

1.1. State of the art human activity classification

A review of existing methods for human activity classification and the relevance of human activity detection and classification is split in the following two sections: the relevance of human activity classification in video/surveillance is reviewed and the relevance and state of the art in human activity classification with radar is discussed. State of the art in multimodal machine learning is reviewed as well.

1.1.1. Video

A review of available human (activity) detection algorithms in surveillance videos and its applications is done in [2]. Applications include:

- Abnormal event detection
- Human gait characterisation
- Person detection in dense crowds and people counting
- Person tracking and identification

- Gender classification
- Pedestrian detection
- Fall detection for elderly people

Data from video surveillance is also widely used in research to identify/track and classify human behaviour [3]. How to use video for human activity recognition was shown in [4]. They make an analysis of the possible features that can be extracted from the video and then use a part of these features for the classification stage with a Bayesian classifier.

Gait along with body structure has been recognized as a potential biometric feature for identifying human beings. A multiclass support vector machine (SVM) has been used to classify human activity based on a sequence of frames (video). The spatial and temporal shape of motion of an individual is usually the same for all gait cycles and is considered to be unique to that individual [5].

1.1.2. Radar

That the human gait is a potential biometric feature is emphasized by [6]. Human identification based on micro-Doppler signatures using deep convolutional neural networks, has been used to identify humans, from a small group up to 20 people, up to 68.9% accuracy. Previously human gait classification based on doppler spectrograms was done in [7] without neural networks. This research focused on a person walking when moving both arms, a single arm or no arms. As the latter two can be indicative of a person carrying objects or a person in stressed situations.

Human detection based on (Doppler) radar is described in multiple sources [8–11]. However human detection based on micro-Doppler signatures from FMCW radar using artificial neural networks was first done in [12]. It proved to be an effective method to classify human activity. A distinction was made between 7 human activities: running, walking, walking while holding a stick, crawling, boxing while moving forward, boxing while standing in place and sitting still. Other work proposed a 14 layer convolutional neural network for multi-target human gait classification using micro-Doppler spectrograms [13]. A deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities was shown to be effective in [1]. The pre-training of a convolutional neural network by forcing the network to reconstruct its input, an autoencoder, showed an improvement in the classification results with a limited dataset.

1.1.3. Multimodal

Multimodal deep learning was first used for speech recognition [14]. Since then multiple papers have been written on this topic. Recent work in multimodal speech recognition used a video camera and an ultrasound probe to achieve visual-only speech recognition. In [15] a multimodal convolutional neural network for visual speech recognition was proposed. Multimodal speech recognition using mouth images from a depth camera together with the audio signal is shown to be viable in [16].

Multimodal learning involves relating information from multiple sources. In [14] different multimodal learning methods e.g. Multimodal fusion, Cross Modality Learning and Shared Representation Learning are identified. An audio-visual model for speech separation was proposed in [17]. The integration of visual information along with audio enables the isolation of a single speech signal from a mixture of sounds. This problem is also known as the cocktail party effect, the ability of the brain to focus on a particular stimulus while filtering out other stimuli.

A survey in [18] on multimodal fusion discusses amongst others when and how to fuse modalities. They state that one of the earliest considerations is to decide what strategy to follow when fusing multiple modalities. The most common method is to fuse information at a feature level, also known as early fusion. Combinations of these approaches are however also possible and are referred to as hybrid fusion.

The taxonomy in [19] on multimodal machine learning defines 5 challenges instead of the typical early and late split in data fusion, these challenges are defined as:

- Representation
- Alignment
- Fusion

- Translation
- Co-Learning

The fusion challenge is to join information from multiple modalities to perform a prediction. Information coming from different modalities may have varying noise powers and the predictive power of modalities can differ.

1.2. Problem Statement

Overall, the literature review shows that human detection and activity recognition is an active research field in both the radar and video domain. However no source addresses the issue of human activity classification using deep learning with both radar and video data.

The challenge that is addressed in this work is the challenge of fusion for prediction. The main goal of this thesis is to investigate the possibility to improve the classification of human activity using multimodal (radar and video) deep learning. The following research questions have been formulated to elaborate this challenge:

- Why use deep learning?
- What features need to be extracted from the radar data for human activity classification?
- What features need to be extracted from the video data for human activity classification?
- What multimodal deep learning topology using radar and video data is best suited for classifying human activity?
- What is the impact of using a single modality (radar) on a multimodal (radar and video) trained network?

1.3. Thesis Outline

This Thesis is structured as follows. Chapter 2 discusses the necessary radar signal processing. Chapter 3 introduces the reader to neural networks, consequently followed by an explanation of convolutional neural networks in Chapter 4. Chapter 5 clarifies the generation of a recorded and labelled dataset that was used for training and validation of the convolutional neural networks which are implemented for the single modalities in Chapter 6. The multimodal topologies and results are discussed in Chapter 7 and summarised in Chapter 8. The work is concluded with a discussion in Chapter 9 and conclusions in Chapter 10.

2

Radar Signal Processing

Radar, an acronym for RAdio Detection and Ranging, refers to an electrical system that detects the presence of objects by transmitting radio frequency (RF) electromagnetic (EM) waves towards a region of interest. The transmitted electromagnetic wave is reflected by the object and the radar receives the reflected signal some time later. The RF spectrum refers roughly to frequencies ranging from 3 kHz up to 300 GHz [20]. As in Chapter 5 Frequency Modulated Continuous Wave (FMCW) radar is used to generate a set of outputs this chapter introduces the concept of FMCW radar and the associated signal processing.

2.1. Doppler effect

As the Doppler effect is at the basis of Continuous Wave radars it will be addressed first. The Doppler effect is a change in frequency of a wave caused by the relative motion of the source to the observer. In radar an electromagnetic signal is transmitted to detect an object. In case the object (or transmitter) is moving, the frequency of the received signal will be shifted. Based on the frequency change of the reflected signal, radar can measure the radial velocity of the moving object.

Considering a stationary observer and a target moving away in the radial direction of the radar with a velocity v_r the received frequency can be determined according to [21]:

$$f_r = f_t \frac{1 + \frac{v_r}{c}}{1 - \frac{v_r}{c}} \tag{2.1}$$

where f_r is the received frequency in Hz, f_t the transmitted/carrier frequency in Hz and v_r the radial velocity of the target in m/s. The Doppler frequency shift is the difference of the transmitted and received frequency: $f_d = f_t - f_r$. Considering $v_r \ll c$ in Equation (2.1) the Doppler frequency shift is obtained [20]:

$$f_d = -\frac{2v_r}{\lambda} \tag{2.2}$$

where f_d is the Doppler frequency shift in Hz, v_r is the radial velocity in m/s and λ the wavelength of f_t in m. The velocity v_r is defined positive for objects moving away from the radar. As an object moves away from the radar the distance between the wavefronts of the reflected signal is larger than the transmitted signal, the reflected signal has a lower frequency, this corresponds with a negative Doppler shift .

2.2. FMCW radar

FMCW radar transmits a continuous signal modulated with a low frequency waveform. Figure 2.1 shows the transmitted signal of a linear chirp FMCW radar. The radar transmits consecutive chirp waveforms, which is a sinusoidal signal increasing (or decreasing) in frequency.

2.2.1. Block Diagram

Figure 2.2 gives an overview of the principle of FMCW radar. The Digital to Analog Converter (D/A) generates a chirp signal at baseband, this signal is then upconverted to the carrier frequency f_c and transmitted (after

amplification) by the transmitting antenna T_x. A small part of the transmitted signal s_t is fed back (through the coupler) to a mixer.

The transmitted signal is reflected by an object and an attenuated signal is received a short time later by the receiving antenna R_x. This echo is received τ seconds later, as the echo travels twice the distance to the object the delay is equal to:

$$\tau = \frac{2R}{c} \tag{2.3}$$

where τ is the delay in seconds, R the distance to the object in meters and c the speed of light in m/s. The received signal has a different (instantaneous) frequency at the time of arrival. This is shown in Figure 2.1, the frequency of the received chirp is different (lower) at the time of arrival compared to the chirp being transmitted. The difference in frequency is proportional to the time delay and is related to the object range. The transmitted and received signals are mixed to obtain the sum and difference frequency of the transmitted and reflected signal. As the difference frequency is the parameter of interest a low pass filter is used to filter the sum component. The difference in frequency between the transmitted and received signal is referred to as the beat frequency.

The beat frequency of the transmitted signal compared to the received signal depends on the delay τ and the rate of change of the chirp waveform. As the frequency of the chirp changes B Hertz in T_c seconds the beat frequency can be determined as:

$$f_{b_r} = \frac{\tau}{T_c} B \tag{2.4}$$

where f_{b_r} is the frequency of the beat signal in Hz, τ the time delay in seconds, T_c the chirp duration in seconds and B the bandwidth in Hz. Then then relation with the range can be obtained with the use of Equation (2.3) resulting in an expression for the expected beat signal.

$$f_{b_r} = \frac{2B}{cT_c} R \tag{2.5}$$

However this only holds for a stationary target. If a target is moving it will introduce a Doppler shift Equation (2.2) which results in the received chirp signal being shifted. The Doppler shift results in a slightly different beat signal:

$$f_b = \frac{2B}{cT_c} R - \frac{2}{\lambda} v_r \tag{2.6}$$

2.2.2. Signal Model

This section describes the principle of FMCW radar on the basis of a signal model, the corresponding signals (s_t,s_r and s_b) are also indicated in Figure 2.2 The transmitted signal can be modelled as [22]:

$$s_t(t) = A_t \cos(2\pi f_c t + 2\pi \int_0^t f_i(t_f) d t_f) \tag{2.7}$$

where A_t is the transmitted signal amplitude, f_c the carrier frequency, $f_i(t_f) = \frac{B}{T_c} t_f$ is the instantaneous frequency linearly increasing with time during a single chirp with duration T_c. Performing the integration leads to:

$$s_t(t) = A_t \cos(2\pi f_c t + \frac{\pi B}{T_c} t^2) \tag{2.8}$$

where the initial phase is assumed to be zero. The echo signal can be modelled as a time-delayed and attenuated replica of the transmitted signal.

$$s_r(t) = A_r \cos(2\pi f_c (t-\tau) + \frac{\pi B}{T_c} (t-\tau)^2) \tag{2.9}$$

where $s_r(t)$ is the received signal and A_r is the received (attenuated) signal amplitude. Performing the multiplication $s_b = s_t \cdot s_r$ and filtering the up conversion component leads to an expression for the beat signal [23]:

$$s_b(t) = A_b \cos(2\pi f_c \tau + \frac{\pi B}{T_c} (2\tau t - \tau^2)) \tag{2.10}$$

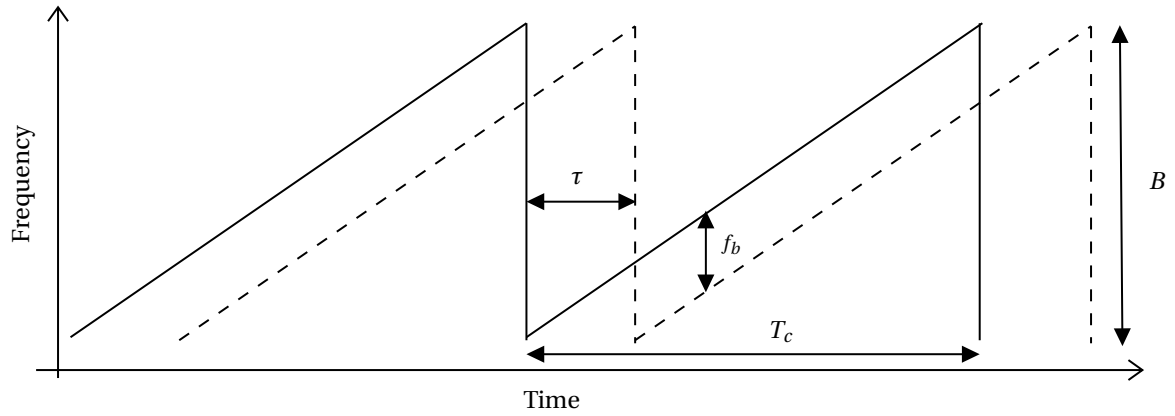


Figure 2.1: The instantaneous frequency of the transmitted signal (solid line) and the received signal (dashed line) for the sawtooth frequency modulation in case of a stationary target. The Bandwidth B, time delay τ, beat frequency f_b and chirp duration T_c are indicated. During a single chirp of duration T_c the frequency is 'swept' over B Hz.

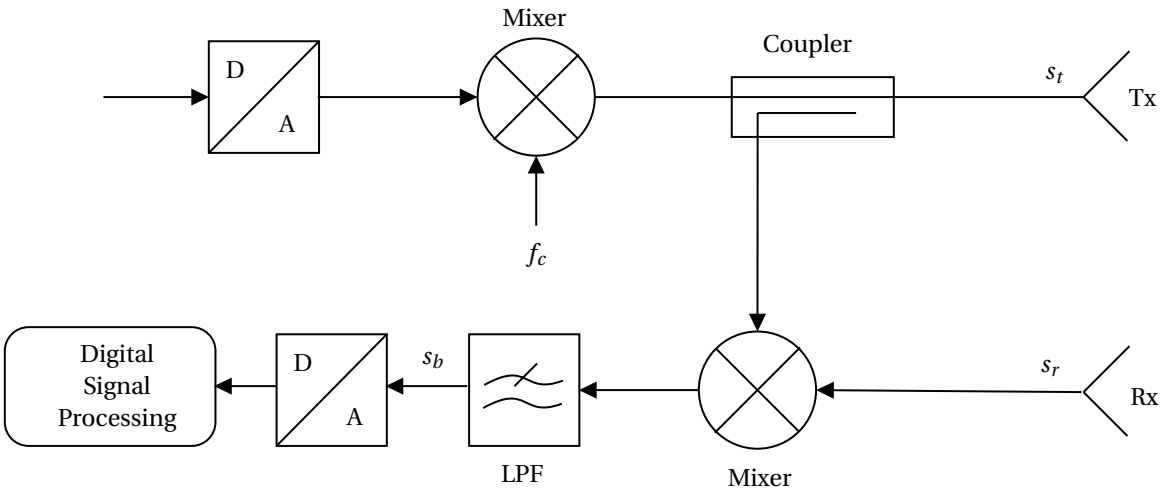


Figure 2.2: Basic overview of a FMCW radar. The Digital to Analog Converter and mixer are used to generate the chirp signal at the carrier frequency. This signal is transmitted and a small part is coupled to a mixer. The mixer mixes the transmitted signal with the received signal. As the mixing process produces a difference and sum component the sum component is filtered. The resulting signal is digitized for further processing. Tx stands for the transmitter and Rx for the receiver

where $A_b = \frac{1}{2} A_r A_t$.
For a stationary target at a distance R_0 Equation (2.3) can be substituted in Equation (2.10), dropping insignificant quadratic $1/(c^2)$ components leads to:

$$s_b(t) = A_b \cos(2\pi f_c \frac{2R_0}{c} + 2\pi \frac{2B}{cT_c} R_0 t) \tag{2.11}$$

it can be observed that this equation is composed of a frequency and a phase term, the frequency term is referred to as the beat frequency (see Equation (2.5)).

2.2.3. 2-Dimensional Signal Processing

If an object moves with some radial velocity v_r the delay will change over time. As the change in the delay over consecutive chirps is relatively slow it can only be noticed in the phase change of the beat signal. For an object with an initial range R_0 moving with a radial velocity v_r the delay τ will almost be a linear function of time [24]:

$$\tau(t) = 2\frac{R_0 + v_r t}{c} \tag{2.12}$$

where R_0 is the range at t = 0 s. Consider now the chirp in Figure 2.1 and following the derivation according to [24]:

$$f_i(t) = \frac{1}{2\pi}\frac{d\phi(t)}{dt} = f_c + \alpha(t - kT_c) \qquad kT_c - \frac{T_c}{2} < t < kT_c + \frac{T_c}{2} \tag{2.13}$$

for (k=0, ± 1, ...) indicating consecutive pulses and $\alpha = \frac{B}{T_c}$.

Substituting Equation (2.12) in Equation (2.10), considering the interval in Equation (2.13) and dropping quadratic components leads to:

$$s_b(t) \approx A_b \cos(2\pi(f_c\tau_0 + k f_d T_c + (f_{b_r} + f_d)t_k)) \qquad -T_c/2 + \tau < t_k < T_c/2 \tag{2.14}$$

where $f_d = \frac{2v}{c} f_c$, $f_{b_r} = \alpha\tau$ (the beat frequency in Equation (2.4)) and τ_0 is the delay for the object at R_0 (at t=0). From this equation it can be observed that the frequency of the beat signal is not only influenced by the delay but also by the Doppler effect: $f_b = f_{b_r} + f_d$. However for a short chirp duration the influence of the Doppler shift (on the observed beat frequency) will be minimal. The range information of Equation (2.14) is contained in the frequency spectrum of $s_b(t)$ [24]:

$$S_r(\omega, k) = \int_{-T/2+\tau}^{T/2} s_b(t_k) e^{-j\omega t_k} dt_k \tag{2.15}$$

where S_r is the range spectrum of s_b. The Fourier spectrum is determined for each chirp, k-index, separately. The maximum of the absolute value of this spectrum is obtained at $\omega = \pm 2\pi(f_{b_r} + f_d)$.

As Equation (2.14) is a function of k, the obtained spectrum S_r is a function of k as well, the angle $2\pi k f_d T_c$ varies with k. Sampling the range spectrum S_r at a fixed interval T_c the object movement, Doppler velocity, can be determined over K consecutive chirps [24]:

$$S_v(\omega, \theta) = \sum_{k=0}^{K-1} S_r(\omega, k) e^{-jk\theta} \tag{2.16}$$

where $\theta = \omega_v T_c$ and ω_v is the velocity frequency. The Doppler response of a single target is then located at $\theta_d = 2\pi f_d T_c$.

2.2.3.1. Overview

The beat frequency is estimated with the Fourier transform based on a single chirp of the received signal, the result of this transform is the range spectrum. Doppler can then be determined by measuring the phase difference between the short chirps. Measuring object range and velocity with FMCW radar involves the use of two-dimensional signal processing as the first Fourier transform is used to obtain the 'range' spectrum and the second Fourier transform is used to extract the Doppler information.

2.3. Micro-Doppler effect

In previous derivation a single target moving with a constant velocity was considered. In case the object has additional oscillatory motion on top of the bulk motion of the object the oscillations will create additional frequency modulation on the returned signal, the response in Equation (2.16) will be a superposition of the individual responses. This additional Doppler modulation is called the micro-Doppler effect [21]. The micro-Doppler signatures associated with the micro-Doppler effect form a distinctive characteristic of a target [25], an illustration of the micro-Doppler effect for a human walk is shown in Section 2.5.

2.4. Time-Frequency Analysis

The time-varying nature of micro-Doppler signals makes the direct application of the Fourier transform infeasible. To obtain information of how the velocity of a signal changes over time other techniques must be used. To evaluate how a function changes over time the most straightforward approach is to window the data perform a fourier transform and slide the window across the entire duration of the signal, this is also know as the Short-Time Fourier transform (STFT). In FMCW radar the beat signal is digitised and the Fast Fourier Transform (FFT) is used. The STFT for the discrete case [21]:

$$STFT(m,\omega) = \sum_{n=\infty}^{\infty} x[n]\, w[n-m] e^{-j\omega n} \tag{2.17}$$

where $x[n]$ is the digitised range response from Equation (2.15) and $w[m]$ is the window function. The width of the window function $w[m]$ determines the frequency and time resolution. Given a window of T seconds the minimum resolvable frequency is $1/T$ Hz. So a very wide window will give a good frequency resolution but a poorer time resolution.

As the frequency of the signal varies over time the Short-Time Fourier transform (STFT) can be used to analyse the time-varying behaviour of the signal. The spectrogram, which shows how the spectral content of the signal varies over time, is then defined as the squared absolute value of the STFT [21]:.

$$\text{Spectrogram}(m,\omega) = |\text{STFT}(m,\omega)|^2 \tag{2.18}$$

In fact the range-Doppler response is obtained by using the STFT in Equation (2.16). Overlapping windows are used when determining the range-Doppler response. The data is divided into overlapping frames, a window is applied and for each frame the Fourier Transform (FFT) is performed. Different window functions exist, in this work the Blackmann window is used.

2.5. Human Gait Analysis

Figure 2.3 shows a simulation of the human gait obtained with software from [21] and serves to highlight the different micro-Doppler components. The radar echo from a walking human contains different Doppler signatures which originate from the different body parts. The leg, arm and torso motion are indicated in the figure as well as the stance of the person at two time instances. The figure contains 3 gait cycles, a typical human gait cycle takes 1 second [26]. The simulation shows a human walking towards the radar (along the radial direction).

As people differ in size and length the micro-Doppler signatures vary from person to person. The micro-Doppler signature of a person is therefore a distinctive feature of a human being. Depending on the activity a human is performing the micro-Doppler signature is expected to change. The features that need to be extracted for the classification of activity therefore differ for different activities.

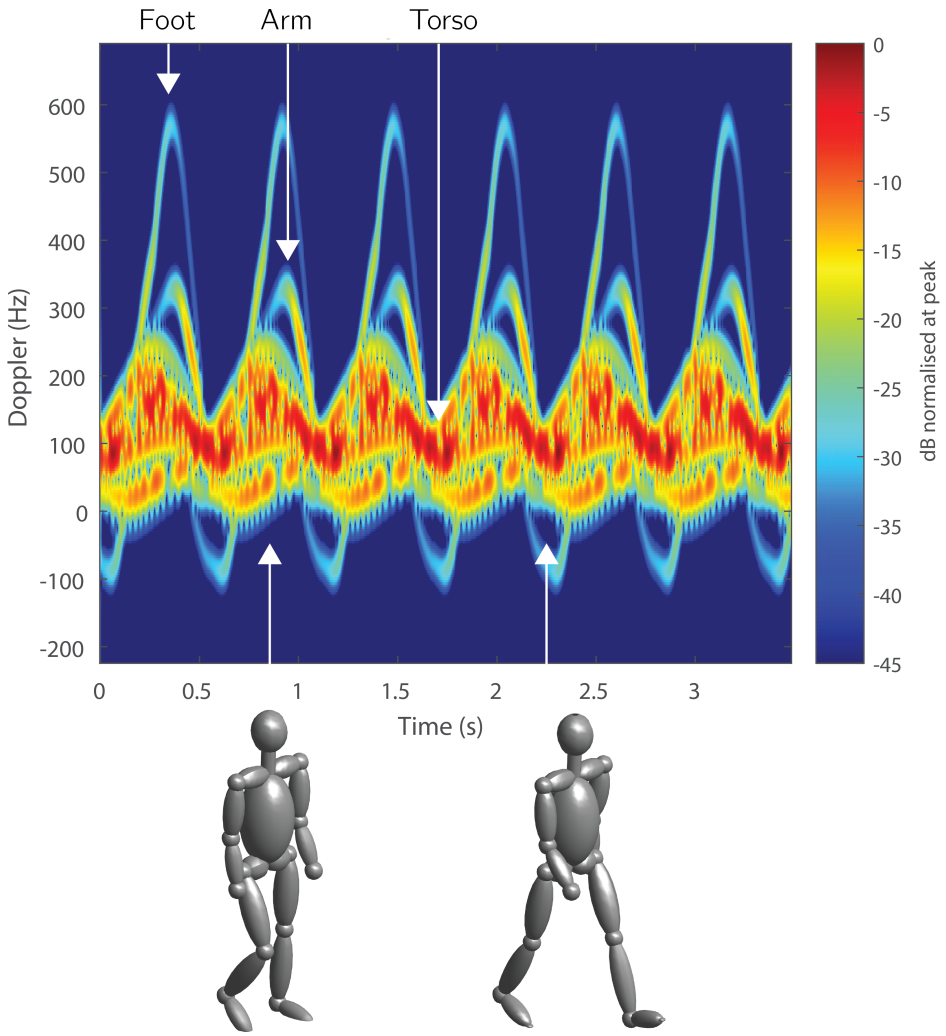


Figure 2.3: Example the micro-Doppler signature of a human walk. The foot, arm and torso motion responses are indicated. The stance of the person during the gait is also shown. The spectrogram is simulated using software from [21].

3

Neural Networks

The principle of Artificial Neural Networks (ANN) began as a (simplistic) model of how neurons in the brain function. Today's artificial neural networks are therefore loosely inspired by biological neurons. This chapter introduces artificial neural networks for classification purposes.

3.1. Neuron model

Figure 3.1 shows the mathematical model of a single neuron. The inputs **x** are multiplied with the corresponding weights **w**. The weighted inputs are summed and an activation function (non linearity) is applied. Traditionally a sigmoid or hyperbolic tangent (tanh) activation function was used as activation function to simulate the neuron response. However more recent research uses the rectified linear unit (ReLU), see Section 3.7.

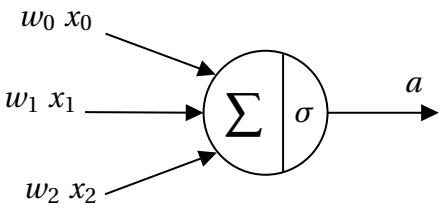


Figure 3.1: Mathematical model for a single neuron visualized, this specific neuron has 3 inputs x_1, x_2 and x_3 which are weighted with the corresponding weights w. The weighted inputs are summed and an activation function is applied. A bias can also be added, which is not shown in the figure.

Mathematically the neuron model can be summarised as [27]:

$$y = \sigma(\sum_i w_i x_i + b) \tag{3.1}$$

where y is the output of the neuron, σ is the (non-linear) activation function, w_i are the weights, x_i are the inputs and b is the bias of the neuron. However a single neuron is only able to solve (binary) linear classification problems, it is not able to simulate more complex (non linear) functions.

3.2. Neural network

The neural network shown in Figure 3.2 contains multiple neurons arranged in layers where the nodes of adjacent layers are all connected. It is a type of feedforward neural network, the information flows in a single direction from input to output. The a superscripts are used to indicate the output of the respective layers.

3.2.1. Forward propagation

During the forward propagation the output of the neural network is determined for a given sample or feature vector **x**. In vector form the outputs of each hidden layer can be determined according to Equation (3.2) where the input layer $\mathbf{a}^{(0)} = \mathbf{x}$.

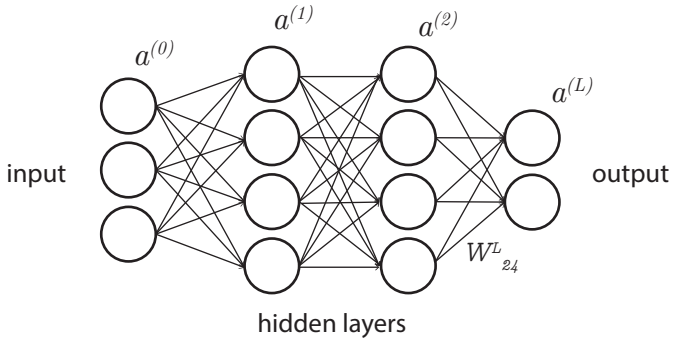


Figure 3.2: Example network of an ANN [27]. Each node in the figure is a model of the neuron as described in Figure 3.1. The arrows indicate connections (weights) from each neuron to all neurons in the next layer. The input layer is shown on the left, this specific example has 3 input nodes. The hidden layers are shown in the middle and the output layer is shown on the right. The amount of neurons in this example are chosen arbitrarily.

$$\mathbf{a}^{(l)} = \sigma(\mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}) \tag{3.2}$$

where $\mathbf{a}^{(l)}$ is a vector containing the output of the neurons in the lth layer (for $l = 0$ up to $L-1$), σ is the element wise activation function and $\mathbf{W}^{(l)}$ is a matrix containing the weights corresponding with layer $(l-1)$ and (l).

When the neural network is used for classification of a dataset into C classes, there will be C output neurons where each output neuron is associated with a specific class. The entire model can be seen as a mapping of a given sample/feature vector $\mathbf{x}$ to a score s for each class depending on all the weights $\mathbf{W}$ [27]. This is illustrated by Equation (3.3):

$$s = f(\mathbf{x}; \mathbf{W}) \tag{3.3}$$

where f is the (unknown) operation performed by the model. However one would like to interpret the scores of each neuron in the final layer as a probability. This can be achieved by applying the softmax function to the output of the final layer [27]:

$$a_k^{(L)} = P(Y = k|X = x) = \frac{e^{s_k}}{\sum_j e^{s_j}} \tag{3.4}$$

where $P(Y = k|X = x)$ is the probability of the sample x belonging to class k and s_j is the score for class j. The score s_k would be the output of neuron k in the last layer without softmax. In this equation $a_k^{(L)}$ is used to indicate the output for class k of the last layer. The class with the highest probability is the predicted class.

3.3. Loss function

To define how well the model is performing a loss function needs to be defined that indicates how well the predicted class corresponds with the actual class. The cross entropy loss or negative log loss is used in combination with the softmax function Equation (3.4) which, for a single sample, is defined as:

$$L = -\mathbf{y} \cdot \log(\hat{\mathbf{y}}) = -\sum_{i}^{n_{class}} y_i \log \hat{y}_i \tag{3.5}$$

where $\mathbf{y}$ is the target vector, a vector of length C (total classes) which contains a 1 at the position of the correct class, corresponding with the input sample x. The vector $\hat{\mathbf{y}}$ contains the predicted probabilities for all the classes from Equation (3.4). This equation reduces to computing the negative log for just the correct class.

Figure 3.3 shows the loss versus the predicted probability. As the predicted probability for the class is close to 1 the loss is low as the sample is already classified correctly. In case the predicted probability is low it means another class is most likely wrongly classified as the correct class, the resulting cost is therefore much higher.

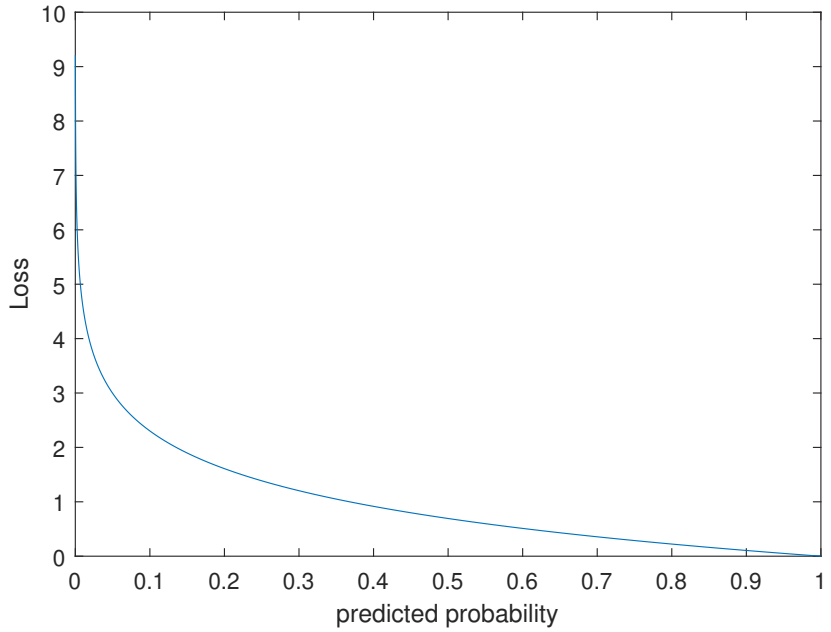


Figure 3.3: The cross entropy loss versus the probability (Softmax output for the correct class).

3.4. Backpropagation

Backpropagation together with gradient descent optimisation is the backbone of deep learning. The goal of the backpropagation algorithm is to determine the influence of each of the weights and biases in the network on the loss function. This allows to adjust all the weights and biases using stochastic gradient descent (or another learning algorithm). More formally the goal of backpropagation is to find:

$$\frac{\partial L}{\partial W_{jk}^{(l)}} \tag{3.6}$$

where $W_{jk}^{(l)}$ is the weight from the k^{th} neuron in the $(l-1)^{th}$ layer to the j^{th} neuron in the l^{th} layer [27]. These values can be found by recursively applying the chain rule from the output of the network to the input.

3.4.1. Stochastic Gradient Descent

The stochastic gradient descent algorithm can then be used to update all the weights and biases in the neural network. Consider a batch of m samples from the training set $\{\mathbf{x^{(i)}}, ..., \mathbf{x^{(m)}}\}$ with the labels $\mathbf{y^{(i)}}$. The weights can then be updated [27]:

$$\mathbf{W} = \mathbf{W} - \alpha \frac{\mathbf{1}}{\mathrm{m}} \sum_{\mathbf{i=1}}^{\mathrm{m}} \nabla_{\mathbf{W}} \mathbf{L}(\mathbf{x^{(i)}}, \mathbf{y^{(i)}}, \mathbf{W}) \tag{3.7}$$

where α is the learning rate, ∇ the gradient and $L(x^{(i)}, y^{(i)}, \mathbf{W})$ the loss for a specific sample and label. The gradient is determined over a small batch of training samples before the weights are updated, this results in more noise like behaviour compared to batch gradient descent but a faster convergence.

The two extreme cases are to use the entire dataset as a batch however as this results in a slow convergence as the algorithm first needs to compute the derivative over all training samples before making an update and this is computationally expensive most often the mini-batch gradient descent algorithm is used. The batch could only contain a single sample this is however very noise like as the gradient is estimated from a single sample. In practice the batch size is a hyperparameter which can be tuned to optimise the weights of a neural network.

3.4.2. Optimisation Algorithms

Instead of using a fixed learning rate for each parameter, adaptive algorithms have been invented which adapt the learning rate over time or accumulate the gradient (momentum) as learning progresses. Some popular

techniques, as stated in [27], for optimisation are SGD (Stochastic Gradient Descent), SGD with momentum, RMSProp, RMSProp with momentum, AdaDelta [28] and Adam [29]. An analysis of different optimisation techniques is done in [30]. There is however no single best method to optimise the weights in a neural network. In this work besides stochastic gradient descent the Adam [29] optimisation algorithm is used.

3.5. Initialisation

To start training a model all the weights need to be initialised. All the weights are randomly initialised according to a specified distribution [31]. The weight initialisation takes into account the number of input and output nodes as this was shown to improve the convergence of the gradient descent based optimisation techniques. More recently a slight adaptation to the weight initialisation in [31] was suggested by [32] in combination with the ReLU activations. The proposed weight initialisation uses a zero mean Gaussian distribution with standard deviation:

$$\sigma_{he} = \sqrt{\frac{2}{n_{in}}} \tag{3.8}$$

where σ_{he} is the standard deviation of the normal distribution and n_{in} is the amount of weights flowing into the layer.

The biases of the neural network are usually initialised at zero as the random initialisation of the weights takes care of the symmetry breaking.

3.6. Regularisation

As the training of the model involves minimising the loss function based on a training set it is possible, most likely, that the model starts overfitting on the training data. It might be able to perfectly classify the training data but it generalises poorly to test data. The network might start to memorise peculiarities, like noise, of the training set. To prevent this several regularisation strategies exist to improve the generalisation of the model.

The following list summarises some of the regularisation methods found in Deep Learning [27]:

- Data augmentation
- L2 regularisation
- L1 regularisation
- Dropout [33]
- Early stopping

The used methods in Chapter 6 are explained below.

3.6.1. Dropout

The key idea of Dropout [33] as discussed by the authors is to randomly drop units with their connections during the training phase, it was shown to be an effective method to improve the generalisation of the model. The random dropping of these nodes can be seen as training a lot of smaller (thinned) models in an ensemble [33]. Overall the random dropping of nodes introduces noise into the system and prevents the model to focus on a single feature and forces the model to learn to recognize/extract more global features.

3.6.2. Early stopping

A simple method to prevent overfitting is to keep track of the model performance during training and stop the training before the validation accuracy starts dropping. Although this method does not improve the validation accuracy during training it allows to obtain the best performing model.

3.7. Activation Functions

The Rectified Linear Unit (ReLU), see Figure 3.4, as activation function was empirically found [34] to improve the training speed and classification accuracy of deep neural networks compared to previously used activation functions (hyperbolic tangent and sigmoid function). The rectified linear unit is given by:

$$\sigma(x) = max(0, x) \tag{3.9}$$

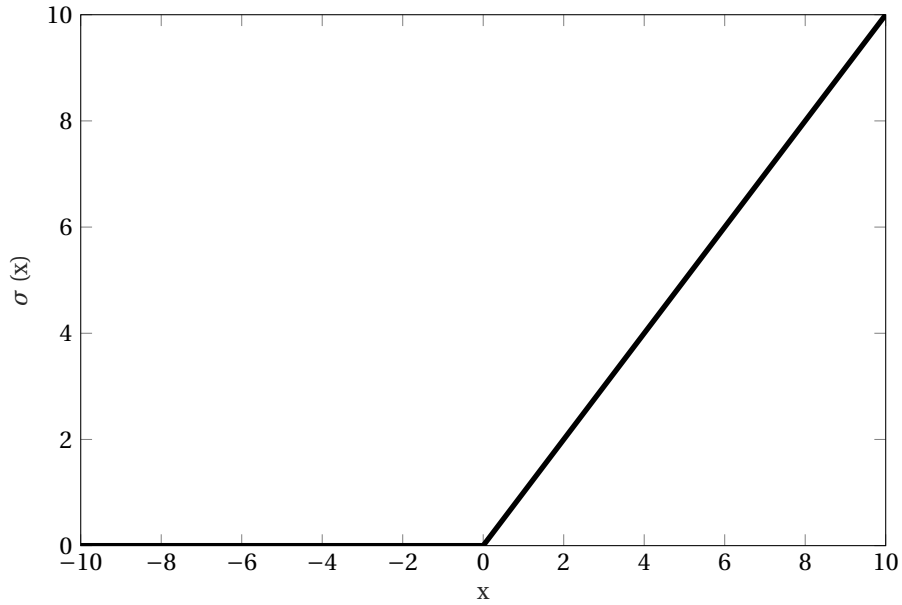


Figure 3.4: Rectified linear unit (ReLU). For negative inputs the activation is zero. For positive inputs the output is a linear function of the input.

Some advantages of the ReLU are that it is computationally fast (No exponent compared to a sigmoid or hyperbolic tangent function) and it deals better with the vanishing gradient problem (The derivative for x>0 is equal to 1). As in the back-propagation algorithm the derivatives are multiplied consecutively this can lead to very small weight updates for the hyperbolic tangent and sigmoid functions. This problem is alleviated with the ReLU.

A disadvantage of the ReLU is that any inputs x that become smaller than 0 result in a 'dead' ReLU as it will not be activated for that input. Some adaptations to the ReLU exist to adress this issue, Parameteric rectified linear unit (PReLU) [32] or Exponential Linear Unit (ELU) [35] however these methods do not necessarily improve the performance. What activation function to use and what the influence of the activation function on the model is still an open research topic.

3.8. Deep learning

Where classical classification algorithms involved a careful feature extraction and selection process and the selection of the correct algorithm for the classification, deep learning tends to approach these problems simultaneously. The 'deep' in deep learning refers to the multilayered structure of artificial neural networks. These models tend to improve classification accuracy over more conventional methods significantly when sufficient data is used for training. Deep learning involves multilayered neural networks with the purpose of 'automatic' feature extraction, selection and classification. This allows for computational models to learn representations of data with multiple levels of abstraction [36]. State of the art is significantly improved in speech recognition (e.g. [37], [17]) and image recognition problems amongst others. The state of the art performance of such systems is one of the main reasons to investigate this field. Convolutional Neural Networks (CNNs) especially have been established as a powerful class of models for image recognition problems [38] and will be discussed in the next Chapter.

3.9. Conclusion

This chapter introduced neural networks for classification purposes. How to determine the output of a neural network and how to optimise the weights of such a network was addressed. Furthermore the ReLU in combination with the so called 'He' [32] weight initialisation (see paper for details) will be used as this should improve the training speed and accuracy of the neural networks. As regularisation strategies Dropout and early stopping will be applied.

4

Convolutional Neural Networks

One of the first implementations of Convolutional Neural Networks (CNN) was done by [39], which at the time scored the highest on the MNIST database of handwritten digits. At the time such networks were difficult to train due to the computational intensity. In 2012 'AlexNet' [40] was introduced. This convolutional neural network was written with CUDA to run with GPU support, this network achieved the highest classification accuracy on the ImageNet Large Scale Visual Recognition Challenge [41]. They also used the ReLU (Rectified linear unit) as activation function, which had been shown previously to speed up training of deep neural networks [34]. One of the key results of this paper was that the depth of the model was essential for its high performance, which was possible due to the usage of GPUs during training. Since the classification error on the ImageNet challenge has dropped to a few percent due to the usage of even larger and deeper convolutional neural networks.

One of the disadvantages of (fully connected) neural networks for image recognition is the scalability, a single fully connected neuron in the first hidden layer of a neural network would need $W \times H \times 3$ (for a RGB colour image) weights, where W is the width in pixels and H is the height in pixels of the image. The number of weights increases significantly when larger images are considered. Convolutional neural networks are a special type of feedforward neural network, this chapter is therefore a direct extension of the previous chapter. Compared to a (fully connected) neural network the convolutional neural network is sparsely connected and shares parameters between connections, a specific neuron in the input layer for example is only connected with a small portion of neurons in the succeeding layer, this greatly reduces the amount of parameters that need to be optimized during training and overall benefits the generalisation capabilities [27].

4.1. Layers

As in Chapter 6 the optimisation of hyperparameters is discussed in terms of existing layers, this section discusses the different components used in Chapter 6. Typical deep convolutional neural networks are composed of at least some or all of the following layers: Input layer, Convolutional layer, pooling layer, fully connected layer and activation (ReLU) layer.

4.1.1. Input

The input layer holds the input data, in the case of a convolutional neural network this layer often holds a 2-Dimensional image.

4.1.2. Convolutional layer

The core element of deep convolutional networks is, as the name suggests, the convolutional layer. It consists of a set of learnable filters or kernels. Each filter or kernel often has small dimensions (e.g. 3x3). A 2-dimensional convolutional operation is applied on the input volume producing an activation map which shows the spatial response of the filter to the input image.

For a two-dimensional image I the convolution uses a two-dimensional kernel [27]:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{4.1}$$

where I is the image, K is the kernel and S is the output of the convolution. As the weights of the filters are learned through the backpropagation algorithm most machine learning libraries implement a cross correlation. It turns out that these filters learn to recognize edges or lines in the first layers and as the network becomes deeper the filters learn to recognize more complex shapes [40]. Directly applying the convolution however results in the input pixels near the border being influenced by less pixels than the pixels near the center as the convolutions are applied in consecutive layers. Zero padding can be used to pad the input images such that the output dimensions of the convolution operation are the same as the input image dimensions.

Figure 4.1 shows an example of the convolutional layer with a single depth slice and a single filter. The Figure shows the input dimensions $H_i \times W_i \times D_i = 4 \times 4 \times 1$, zero padding P of 1, the spatial extent of the filter F=3 and the output dimensions. $H_0 \times W_0 \times D_0 = 4 \times 4 \times 1$.

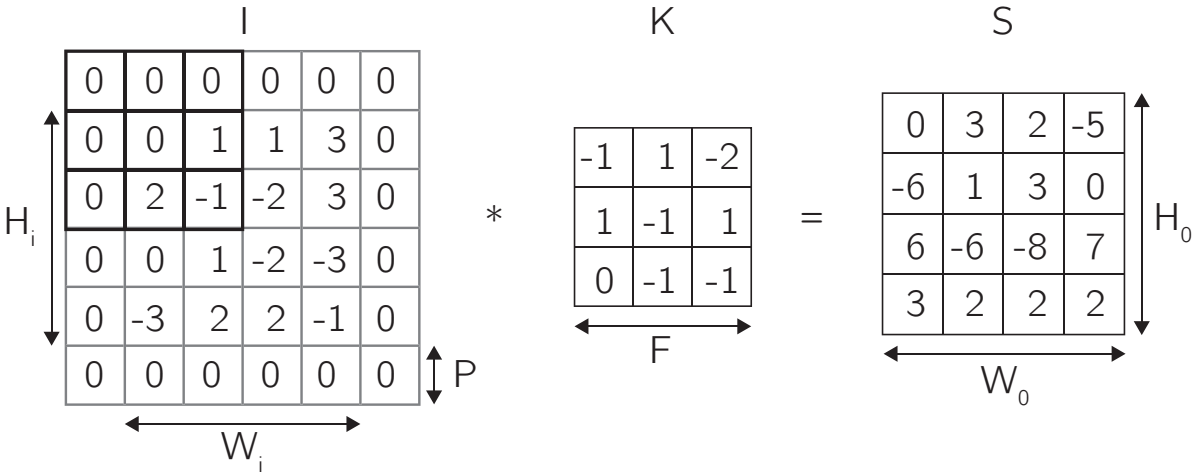


Figure 4.1: Example of the input and output of a convolutional layer with a depth D of 1. This example illustrates the relevant parameters. The example shows a cross correlation between the input and the kernel with a stride S of 1.

Overall the convolutional layer accepts a volume of size $W_i \times H_i \times D_i$. Where D is the depth, the number of channels of the input image/feature maps. Each convolutional layer is defined by four hyperparameters, which are [27]:

- Number of filters K
- spatial extent of the filter F
- stride S
- amount of zero padding P

Here the assumption is made that only square filters are used. The stride S is not shown in Figure 4.1 but is defined as the step size the filter takes as it slides over the input image. The output dimensions of each convolutional layer is then defined as: $W_o \times H_o \times D_o$ where [27]:

- $W_o = (W_i - F + 2P)/S + 1$
- $H_o = (H_i - F + 2P)/S + 1$
- $D_o = K$

4.1.3. Pooling layer

The purpose of the pooling layer is to reduce the spatial size of the representation and reduce the number of parameters in the network. It is a downsampling operation. The pooling operation downsizes the network spatially at each depth slice. The most successful downsampling strategy is the max pooling operation, see Figure 4.2, the filter only passes the highest value on to the next layer. The most common form of this layer consists of filters with a size of 2x2 applied with a stride of 2, therefore halving the input spatial dimensions. This in turn results in a translation invariance of the input image as small distortions in the input result in the same neurons being activated in deeper layers.

Figure 4.2 illustrates the max pooling operation.

Max pooling layers downsample the activation maps. A max pooling layers accepts a volume of size $W_i \times H_i \times D_i$. The max pooling layer is characterised by the following hyperparameters:

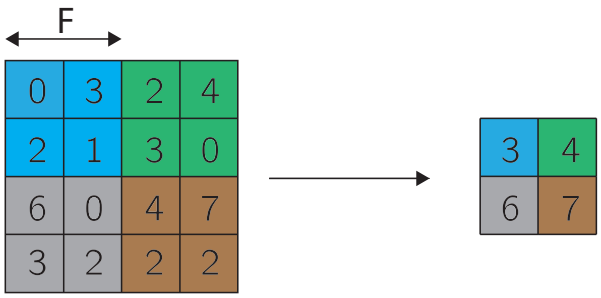


Figure 4.2: Example of a max pooling operation with spatial extent $F = 2 \times 2$ and stride S=2, with these settings the input dimensions are halved. The maximum value is passed on to the next layer. Only positive values are shown as it is often applied after a ReLU. The coloured areas illustrate the regions the max pooling operation is applied. The figure was inspired by [27].

- spatial extent F
- stride S

The max pooling layer produces a volume of size $W_o \times H_o \times D_o$ where:

- $W_o = (W_i - F)/S + 1$
- $H_o = (H_i - F)/S + 1$
- $D_o = D_i$

Common settings are $F = 2, S = 2$ or $F = 3, S = 2$

4.1.4. Activation layer (ReLU)

As discussed previously the activation function is an essential element of neural networks to be able to make more complex decisions. In the context of convolutional neural networks, they are applied after a convolutional layer (and after fully connected layers), the resulting activation map highlights different regions of its input which in turn might be passed on to another convolutional layer, a pooling layer or a fully connected layer.

4.1.5. Fully connected

Most convolutional neural networks use fully connected layers at the final layers of the network. The chain of convolutional layers, max pooling layers and activation layers can be seen as a feature extraction process. The input image is transformed to a compressed representation depending on the desired classification task. A network like Figure 3.2 might be applied to the final layer of the convolutional stages of the network. The output of the convolutional neural network is then composed of C neurons where each neuron corresponds with a class.

4.2. Overview

A common strategy in deep convolutional neural networks is to chain a convolutional, ReLU and pooling layer and then cascade multiple of these blocks until the final representation is sufficiently small to perform the classification with a 'regular' (fully connected) neural network. This way the convolutional neural network maps the original image from the original pixels to the final classification scores. The final design of a convolutional neural network is a careful task which involves a lot of experimentation.

Figure 4.3 shows an example of a typical CNN. Depending on the amount of filters in the first convolutional layer the input image is transformed in N feature or activation maps where N is the number of filters in the first convolutional layer. The pooling operation is then applied to reduce the spatial dimensions. The second convolutional layer is applied across the entire depth of the previous activation maps. Each kernel in the second layer will therefore have a size of $F \times F \times N$, where F is the spatial extent of the filter, which in turn results in a number of activation maps depending on the amount of kernels that were used in the second convolutional layer. It is common to increase the amount of filters as the depth in the network increases, which is shown by the increasing depth of the feature maps in Figure 4.3, as these architectures learn to recognise simple shapes in the earlier layers and more complex shapes in deeper layers.

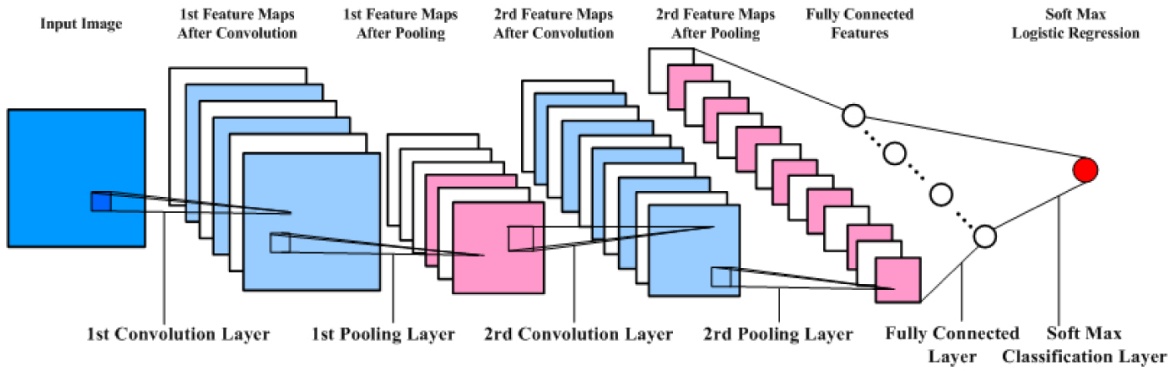


Figure 4.3: An example of a convolutional neural network including 2 convoluttional layers. The activation maps are also called feature maps. The ReLU layer is not explicitly shown in the figure but this operation takes place after each convolutional layer and fully connected layer in the figure. Taken from [42]

.

4.3. State of the art models

The availability of large (labelled) datasets for training and comparing architectures is essential in the field of deep learning and has led to several advances. Most of the state of the art models are trained and validated on the ImageNet database [41], which contains over 14 million images. As one of the conclusions of [40] was that the depth of the models improved the classification, current state of the art models for image classification contain several hundreds of layers, e.g. [43], and are not just limited to previously mentioned typical convolutional neural network layers. Another paper showed the effectiveness of using very deep neural networks with small filter dimension (3×3) [44].

4.4. Hyperparameters

Choosing appropriate hyperparameters is essential for the performance of deep learning models. Besides the type and parameters of the optimization algorithm, the model parameters itself needs to be optimized. The amount of layers, size of each layer, the depth of the model and the parameters of each layer all influence the performance of the final model. Besides the optimization of the weights the model itself also needs to be optimized. Possibilities to optimise these (and other) hyperparameters include a grid search or a random search over all the parameters of interest.

4.5. GradCam

The section is based on a method derived in 'GradCam' [45]. As the interpretation of features learned by a convolutional neural network is not always trivial different methods have been invented to get a better understanding of what a convolutional neural networks learns. One of these methods is 'GradCam', it aims to obtain a class specific saliency map which gives an indication of the pixels associated with a specific class:

$$L^c_{ij} = \sum_k w^c_k A^k_{ij} \tag{4.2}$$

where A^k_{ij} is the k-th feature map in the last convolutional layer (the '2nd Feature maps after Convolution' in Figure 4.3), the ij indexes are used to enumerate over the spatial location (pixels) in the feature map. The weights w_k in 'GradCam' are then defined as:

$$w^c_k = \frac{1}{Z} \sum_i \sum_j \frac{\partial Y^c}{\partial A^k_{ij}} \tag{4.3}$$

where Y_c is the classification score before the Softmax layer for the class of interest (which was defined as s_k in Section 3.2), Z is a normalisation constant (summation over all pixels i and j). The gradient $\frac{\partial Y^c}{\partial A^k_{ij}}$ is easily obtained using existing machine learning tools. The gradients obtained are averaged for each feature map, this average then forms the weight w^c_k.

4.6. Conclusion

Convolutional neural networks were established as state of the art image recognition capabilities. To achieve state of the art capabilities however a careful optimisation of all the model parameters needs to be done. One of the methods to do so is a grid search of the desired parameters. As the features learned from a convolutional neural network are not always easily interpretable the GradCam method is adopted to get a better understanding of the learned features.

5

Dataset

As this is novel research and deep learning requires a dataset to train and validate the neural networks a dataset was recorded. This chapter describes the creation of the recorded and labelled dataset. Section 5.1 explains the measurement setup. The recorded radar data and processing is explained in Section 5.2. The video data is reviewed in Section 5.3.

5.1. Measurement Setup

Radar and video data was collected from 35 different people with the goal to classify the 'activity' the person is performing. One of the applications of human activity classification is the detection of suspicious behaviour, determining whether a person is holding an object or carrying a heavy object can give an indication of such behaviour. To address this, three different scenarios or activities are considered. A person walking normally without carrying an object (*N*), a person walking with a rifle like (metal pole) object (*R*), or a person walking with a relatively heavy (10 kg) backpack (*B*).

Throughout this work the different activities can be identified by the following indicators:

- Nothing (N)
- Rifle (R)
- Backpack (B)

Figure 5.1 illustrates the measurement setup. The test subject walks at the start of the measurement, at t = 0 s, from approximately 40 meters towards the radar and video setup. The radar was located at an height of ≈ 1 m and the video camera about a height of 2.5 m. Each person performed each activity twice, resulting in a total of 210 measurements with a duration of 20 seconds each.

The radar data is processed and for each recording a spectrogram is generated, see Section 5.2. The video data is processed using a Single Shot Detector (SSD) [46] on each frame to extract the person with a bounding box from the frames in Section 5.3.

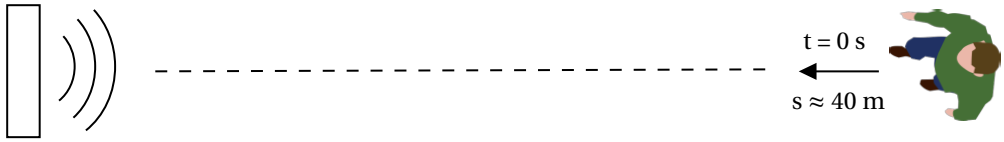


Figure 5.1: Simplified schematic of the measurement setup. At the start of the measurement at t = 0 s a person walks from approximately 40 meters towards the radar and video setup (the symbol on the left) along the radial direction of the radar. The video camera and radar recordings have a duration of approximately 20 seconds.

5.2. Radar Data

FMCW X-band radar is used with a carrier frequency f_c of 9.8 Ghz and a bandwidth B of 100 MHz. Resulting in a range resolution ΔR of 1.5 m. The radar parameters used are summarised in Table 5.1.

Table 5.1: The radar parameter and the parameters used for the creation of the spectrograms. PRF is the pulse repetition frequency, B is the bandwidth, CPI is the coherent processing interval (the size of the window used in STFT), 'No. of sweeps CPI' is the number of sweeps contained in a single Blackmann window. An overlap of 80% (0.08s) was used.

PRF (Hz)	B (MHz)	f_c (Ghz)	CPI (s)	No. of sweeps CPI	CPI Overlap (%)
2500	100	9.8	0.1	250	80

Figure 5.2 shows a radar measurement. A person walks from 40 meters towards the radar. At the end of the measurement the person is at approximately 10 meters from the radar. Some other reflections are visible as well, between 20 and 30 meters a tree was located which shows a clear response. At approximately 55 meters a car was located at the end of the measurement range which shows a strong response as well.

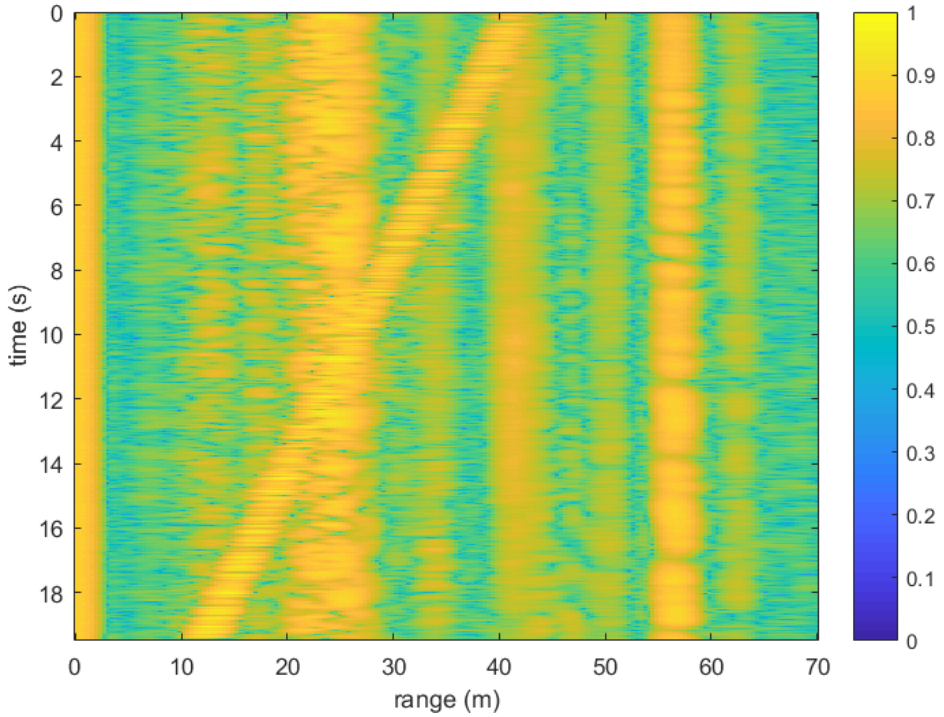


Figure 5.2: Example of a radar measurement, the image shows the normalised intensity. A person walks from about 40 meters at t = 0 s towards the radar. The response from a tree is visible in the range from 20 to 30 meters, a car was located at 55 meters.

5.2.1. Spectrogram

The parameters in Table 5.1 are used to generate a spectrogram for each measurement, to do so first a range tracker is implemented which tracks the human motion. This range tracker determines the maximum response in each Range-Doppler window. As discussed in Section 2.4 the Range-Doppler response is computed using a blackman window of 0.1 s containing 250 sweeps. Figure 5.3 shows the range Doppler response for a single window. The target is visible at 40 meters. After filtering the zero Doppler component the maximum response is determined in a small region (± 1.5 m) around the expected position. This range slice is then extracted and added to the spectrogram.

The Doppler interval was cropped to 64 bins corresponding with Doppler velocity -8.57 m/s up to 1.07 m/s. This was done to obtain samples with a duration of 1.28 s as this resulted in images with dimensions of 64 × 64 (height × width), this cropping was possible as all the persons in the dataset walked towards the radar with a common walking speed of about 2 m/s. Figure 5.4 shows a complete spectrogram. As 35 people

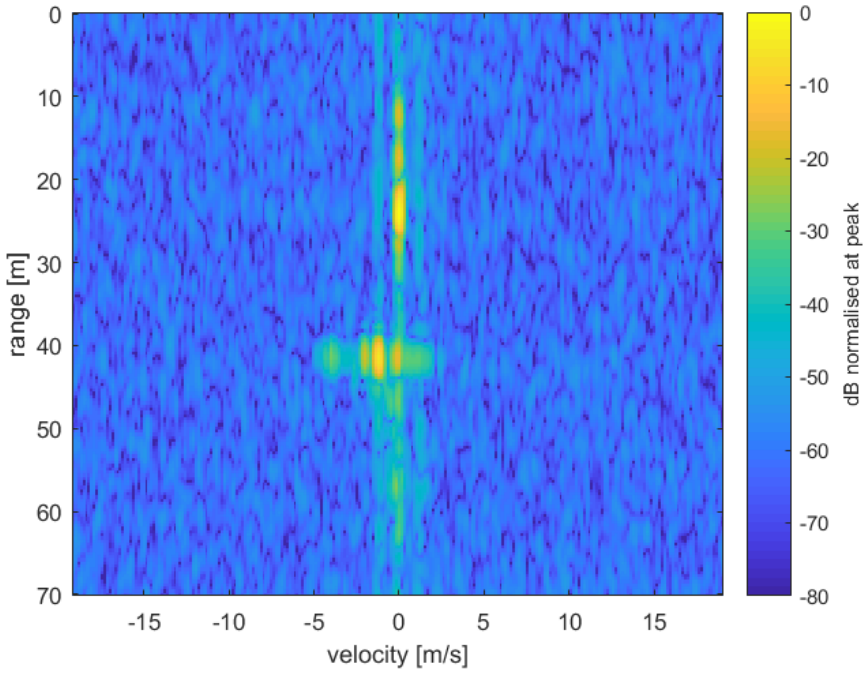


Figure 5.3: Normalised dB response of the windowed Range-Doppler response. Corresponding with a window size of 0.1 s containing 250 sweeps. The human motion is visible at a range of 40 m.

participated during the recording of the dataset and each person performed each activity twice, the total radar dataset consists of 210 such spectrograms.

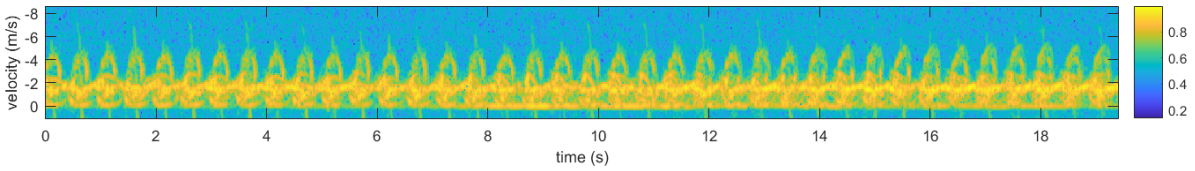


Figure 5.4: Example of a complete spectrogram. The Doppler interval is cropped to 64 bins, corresponding with Doppler velocity -8.57 m/s up to 1.07 m/s

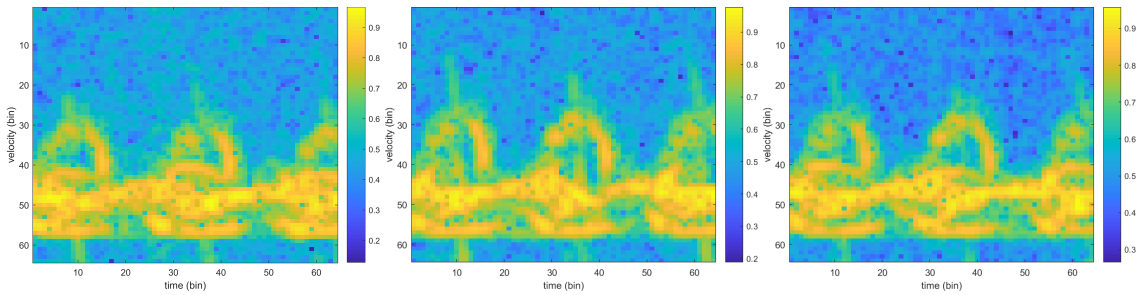
5.2.2. Classes

Figure 5.5a, 5.5b and 5.5c show respectively samples for the classes Nothing, Rifle and Backpack for a selected test subject. The difference in the spectrograms is most clearly visible between the metal bar (R) case and the other cases. When the person is holding a metal bar the person is not able to move his/her arms. Resulting in a different (absence) micro Doppler signature of the arm motion.

The motion of the legs, more specifically the foot as can be seen in Figure 2.3, results in the largest fluctuation in the micro-Doppler signature, the torso motion is relatively steady around 2 m/s. The motion of the arms is centered around the torso motion, oscillating between 1 m/s and 3 m/s.

5.2.3. Training and Validation

The dataset S_{rad} is split in a train and validation set for the implementation discussed in Chapter 6. Consider a set T_{rad} which consist of the training data and a set V_{rad} which contains the data used for validation. The sets are chosen randomly such that 80% of the data is used for training, set T_{rad}, and 20% is used for validation set V_{rad}. The sets are chosen such that spectrogram measurements belonging to a specific person are in one set and not in the other. This resulted in the measurements of 7 randomly chosen people being placed in the validation set and the measured spectrograms of 28 people being placed in the training set. Some other preprocessing steps, such as normalisation of the data, is discussed in Section 6.1.1.



(a) Nothing (b) Rifle (c) Backpack

Figure 5.5: Samples with dimensions 64 × 64 of the three classes. The difference between the sample between the Rifle class and the other classes is clearly visible, in case a person is holding a metal pole the arm motion is restrained.

5.3. Video Data

The video data recordings were made using a HD (1920x1080 pixels) camera recorded at ≈ 13.5 frames per second. A single shot detector (SSD) [46] was applied the frames in the video recording. This single shot detector is capable of detecting humans (and other objects) in images and draws a bounding box around the detected humans. The detected person is then extracted from the frame. The resulting image consists of the human only. Due to the persons walking from 40 meter towards the radar the extracted frames differ in size, increasing as the subjects approached the radar and video setup, and are cropped differently.

For training and testing the neural network all the images need to be of the same size, consequently all images were resized to have a height × width of 128 × 64.

5.3.1. Frames

An example of each class is shown in Figure 5.6. Due to the single shot detector that was used and the persons walking from 40 meters towards the radar and video setup the frames are all cropped slightly differently. The recordings were all made during daylight. As the measurements took multiple hours on different days the lighting in the frames differs slightly from measurement to measurement.

Although the single shot detector had a high detection rate (± 90 %) occasionally a person was not detected. This was mostly the case at the start of a measurement due to bad lighting. Furthermore this specific detector had a small false alarm rate, occasionally a random object is detected as a person. To make sure the dataset is clean, these frames were taken out by hand.

5.3.2. Training and Validation set

The validation and test set for the video data S_{vid} are split in the same way as the radar data. A set T_{vid} is made which consist of the training data and a set V_{vid} which contains the data used for validation. The sets are chosen randomly (with the same random seed as the radar data) such that 80% of the data is used for training, set T_{vid}, and 20% is used for validation, set V_{vid}. Again the frames belonging to a person in the training set are not used in the validation set.

5.4. Multimodal Dataset

The radar and video data are synchronised using the GPS times which were stored during the recordings of the measurements. For each available video frame extracted by the single shot detector the starting time of the closest radar frame (with a duration of 1.28 s) is found. Due to the processing of the radar data the time resolution of the spectrograms is Δt = 0.02 s (A 0.1 second window with 0.08 seconds overlap was used). This means the maximum deviation between the video frame and the radar frame is $\Delta t_{maxerror}$ = 0.01 s. The multimodal dataset will be denoted as S_{sync}. To emphasize, this synchronised dataset consists of the single frames from the video data (Figure 5.6) associated with the start of a 1.28 s radar spectrogram frame (Figure 5.5).

This dataset is in turn split in a training and validation set such that the synchronised measurements from 28 people (80%) are in the training set and the synchronised measurements from 7 people are in the validation set.

(a) Nothing (b) Rifle (c) Backpack

Figure 5.6: Samples with dimensions 128 × 64 of the three classes. (a) The Nothing (N) class, a person walking normally. (b) The Rifle (R) class, a person walking with a metal pole. (c) The Backpack (B) class, a person walking with a backpack on their back. The straps of the backpack are visible. The frames have been blurred and occluded deliberately for privacy reasons.

5.5. Overview

An overview of the complete dataset is given in Table 5.2. The total number of frames for the radar dataset is somewhat misleading as this includes the overlapping frames. The total number of synchronised frames (S_{sync}) is lower as the video frames at the end of the measurements can not be synchronised with the start of a 1.28 seconds radar frame as this data is not available.

Table 5.2: Overview of the total number of frames ('No. of frames') in the complete dataset. Although the radar dataset seemingly contains a high number of frames these frames are highly redundant due to the overlapping window. The 'Synchronised' column refers to the number of frames left after synchronisation is performed.

	Video (S_{vid})	Radar (S_{rad})	Synchronised (S_{sync})
No. of frames	47314	186394	43554

6

Single Modalitiy Implementation

One of the advantages of current machine learning research is the availability of highly optimised machine learning libraries. One of those libraries is Keras [47]. For the implementation of the convolutional neural networks Keras is used, which uses tensorflow as its tensor manipulation library [48].

As the Doppler information obtained using FMCW radar is often represented as an image, a spectrogram, it makes sense to apply convolutional neural networks on these images for the purpose of classification. In this sense the classification of human activity using time-Doppler spectrograms is posed as an image recognition problem [12]. The goal of the classification using CNNs is to classify human activity based on some sample $\mathbf{x}$, where $\mathbf{x}$ is an image of a predefined height and width as discussed in Chapter 5.

Section 6.1 discusses the optimisation of hyperparameters for the radar dataset as discussed in Chapter 5 and the implementation for the video data is discussed in Section 6.2.

6.1. Radar Implementation

This section discusses the implementation of a convolutional neural network for just the radar data. To make a comparison of the different architectures the train and validation set as described in Section 5.2.3 are fixed, the radar data from 7 persons (20%) is in the validation set and the rest of the data is used for training.

6.1.1. Preprocessing

For each sample drawn from the train and validation set the frames are first normalised in the range from 0 to 1, then the mean image from the *training* dataset is subtracted to center the dataset in the origin. This preprocessing step speeds up the training process and makes the final models more robust.

The mean image is determined as a preprocessing step by sliding a window over the entire training dataset, normalising each image in the range from 0 to 1 and then computing the mean image over this entire dataset. The mean image is then an image with dimension 64 × 64 × 1 (Height × Width × Channels). As the radar spectrograms only have 1 color channel .

6.1.2. Training

To train the models a window based approach is used on the spectrograms. During training at each epoch 64 frames with a time window of 64 bins (1.28s) are randomly drawn from each spectrogram in the train set, see the example in Figure 5.5. These frames are randomly shuffled and the model is trained for these samples, at the end of this epoch a new train set is drawn from the available train data, this process is repeated for 40 epochs.

This method also aims to use the full potential of the available data as the start of the gait cycle of a human walk in the randomly selected frames will differ from one frame to the other. The gait cycle of a human walking is rather consistent in the sense that the arm, length and torso size, i.e. the proportions of a specific person, do not change and therefore form a rather specific biometric feature of the human gait. People do however change their pace slightly which also impacts the shape of the spectrogram, to train a neural network one would like to have as much of this diversity in the dataset as possible.

6.1.3. Validation

To determine how well the model is performing a sliding window based approach is used on the validation set. After each epoch all the frames in the validation set are evaluated. The mean image from the *training* dataset is subtracted and the images are propagated through the models and the validation accuracy and loss is computed.

As a typical human gait cycle is about 1s a single gait cycle should already contain the information about the activity, the absence or presence of the arm motion can already be noticed in half a gait cycle so the available 1.28 seconds of data should be sufficient.

6.1.4. Grid Search

Table 6.1 summarises a grid search for several hyperparameters for the radar data with input image dimensions 64 × 64. After some initial trial and error the Adam optimisation algorithm was used with the default parameters (as proposed in [29]) except for the learning rate which was set at 0.0001. Furthermore a batch size of 16 was used. The last layers are two fully connected layers with 500 nodes and Dropout with a chance of 0.5 is applied after each fully connected layer, the last layer is a Softmax layer with 2 nodes, as for the optimisation of the hyperparameters only the Nothing and Rifle class were used (which will be discussed in Section 6.1.7). The grid search was done to classify just the Nothing (N) and Rifle (R) class. Early stopping is used to choose the model with the maximum classification score on the validation set. After each convolutional layer and fully connected layer a ReLU activation is applied.

The grid search shows the classification scores for a differing number of convolutional layers. The table clearly shows the impact of shallow architectures. For just two layers (16-32) in Table 6.1 the classification score is 82.8%. On the other hand increasing the complexity of the model, i.e. the number and size of the layers, can result in overfitting, as the model becomes more complex it can more easily 'memorize' the training data which in turn leads to bad generalization. In the end the model should be as complex as the data allows it to be, finding this optimum however is a time consuming process which involves a lot of experimentation. Overall an increase in the number of kernels as the depth increases seems to positively influence the classification score.

The classification scores in Table 6.1 are the result of training each model a single time (using the same test and validation set each time) for 40 epochs. The random initialisation of the weights causes the model to obtain a slightly different optimum each time it is trained, to get a better insight in this fluctuation several models were selected and trained for several times.

The models in Table 6.2 were trained using Stochastic Gradient Descent with nesterov momentum, it shows the results for training some of the selected models 5 times using the parameters in Table 6.3. The deviations in the accuracy are caused by the random initialisation, see Section 3.5, of the weights and the training process itself also influences the final classification accuracy.

Table 6.1: Classification accuracy for different hyperparameters. The 'Width' column refers to the number of kernels in each convolutional layer, where the depth increases from left to right, the first number mentioned corresponds to the amount of kernels in the first convolutional layer. After each convolutional layer max pooling with kernel size of 2 × 2 and stride 2 is applied. The highest obtained classification score is shown in bold face. Accuracy stands for classification score on the validation set. A "−' symbol is used to indicate a max pooling operation between convolutional layers. The ',' indicates consecutive convolutional layers were used.

Width	Kernel size	Accuracy (%)
16,16 - 32,32 - 64,64 - 128,128	3x3	87.45
16-32-64-128-256	3x3	86.6
20-30-40-50	3x3	87.06
	5x5	**87.97**
	7x7	87.94
8-16-32-64	3x3	85.05
	5x5	85.11
	7x7	84.34
10-10-10-10	3x3	82.94
20-20-20-20	3x3	85.15
	5x5	84.69
	7x7	86.60
16-32-64	3x3	85.63
	5x5	87.90
	7x7	87.50
30-40-50	3x3	87.04
	5x5	87.46
	7x7	86.23
16-32	3x3	82.8

Table 6.2: Classification accuracies for a selected number of models. The classification stage is composed of 2 fully connected layers with 500 neurons, Dropout with a chance of 0.5 is applied after the fully connected layers. The fluctuation of the classification score is visible for different runs. The average classification score C_{av} is shown in the most right column. The classification accuracy for run i is indicated by C_i. All classification accuracies are in percentages. The highest average classification accuracy is shown in bold face. The parameters used to train these models are shown in Table 6.3

Model	Width	Kernel size	C_1	C_2	C_3	C_4	C_5	C_{av}(%)
A	20-30-40-50	5x5	88.34	87.95	**88.42**	88.02	87.36	**88.03**
B	8-16-32-64	7x7	84.62	87.49	86.58	85.40	86.58	86.14
C	30-40-50	7x7	89.07	87.32	87.75	90.43	88.10	87.99
D	16-32-64	5x5	88.56	87.15	85.99	87.83	85.57	87.02
E	16-32-64-128	3x3	86.54	85.35	87.08	86.96	87.08	86.61
F	16-32-64-128-256	3x3	85.32	84.94	84.36	85.47	84.08	84.83
G	16-32-64-128	5x5	88.36	89.35	86.49	88.37	87.28	87.96

Table 6.3: Parameters used for training the models in Table 6.2. The definition of the Epoch was adjusted to contain 20 frames for each test person each epoch.

Optimiser	Learning rate	Nesterov Momentum	Batch size	Learning rate decay	Epochs
SGD	$1 * 10^{-4}$	0.9	16	$1 * 10^{-}6$	200

6.1.5. Final Model

The model in bold face (Model A - C_3 in Table 6.2) is used as the model for further analysis. Although not the best model in terms of classification accuracy it showed a rather consistent classification accuracy. This model has as input the cropped radar images of size 64 × 64. The model in bold face in Table 6.1 is summarised below, during training Dropout with a chance of 0.5 is added after the Fully connected layers:

- Input layer (size: 64 × 64 × 1)
- Convolutional layer (20 filters, size: 5 × 5)
- Max pooling (2 × 2)
- Convolutional layer (30 filters, size: 5 × 5)
- Max pooling (2 × 2)
- Convolutional layer (40 filters, size: 5 × 5)
- Max pooling (2 × 2)
- Convolutional layer (50 filters, size: 5 × 5)
- Max pooling (2 × 2)
- Fully connected: (500 Neurons)
- Fully connected: (500 Neurons)
- Output layer: (2 Neurons)

After the last Max pooling layer, before the fully connected layers the obtained feature maps are flattened. The 50 4 × 4 feature maps (which are the output of the last Max pooling layer) are converted to a single feature vector with a length of 800. This relatively small model contains 747,642 trainable parameters, an overview of this model is given in Figure 6.1. The convolutional and max pooling layers are summarised in the 'CNN' block.

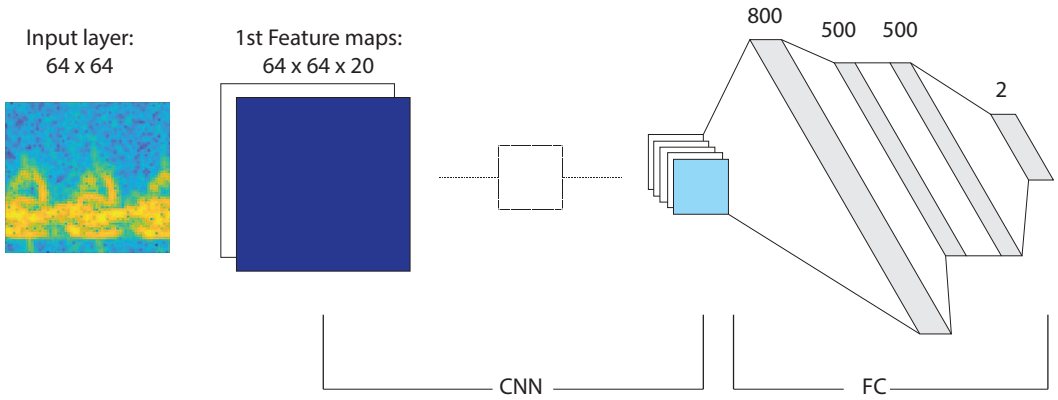


Figure 6.1: Overview of the convolutional neural network for classification of the radar data. The 'CNN' stage refers to the convolutional and max pooling layers, the first feature maps are also indicated. 'FC' refers to the fully connected layers used for classification. The numbers on the right indicate the size of the final layers.

6.1.6. Loss & Accuracy

Figure A.3 shows the loss for the training and validation set during training for the best model in Table 6.1, the validation accuracy during training is shown in Figure A.4. The model with the best performance is obtained at epoch 5. One of the issues that can be seen in Figure A.3 is the overfitting on the train data. Beyond epoch 5 the validation loss starts increasing.

The model was also trained with the stochastic gradient descent algorithm, the loss and accuracy on the train and validation set for model A - C_3 in Table 6.2 are shown in Figure 6.2 and Figure 6.3. This showed a slightly lower loss overall, however the results obtained with the different optimization algorithms are rather similar (both methods achieve a classification score of ≈ 88 %).

Figure 6.2: The training and validation loss for the Stochastic Gradient Descent Algorithm for model A - C_3 in Table 6.2. As training progresses the training loss keeps reducing until it almost hits 0 loss. The loss on the validation set however starts increasing after epoch 100. The validation accuracy remains roughly equal.

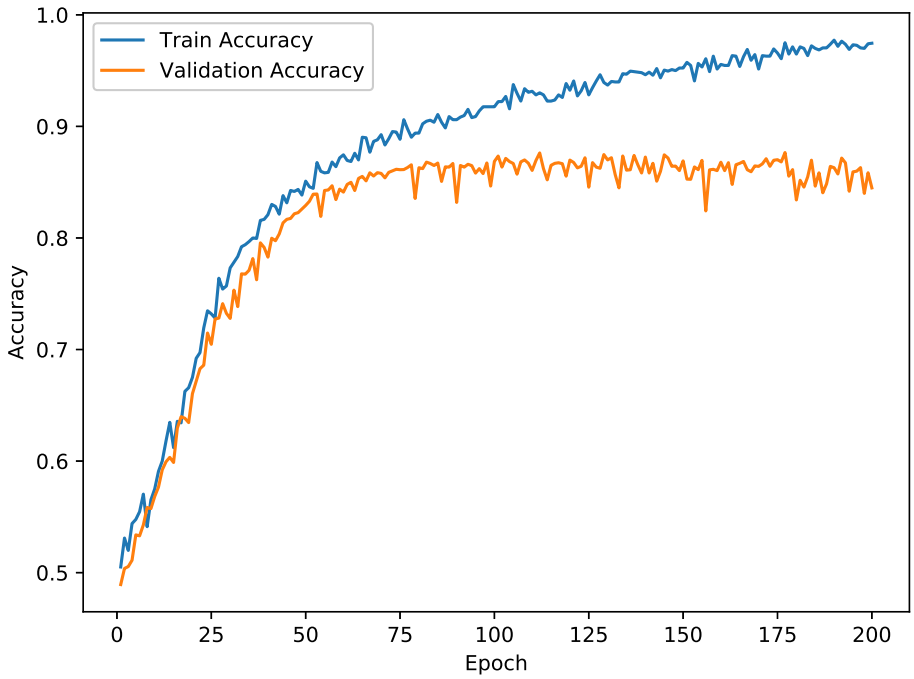
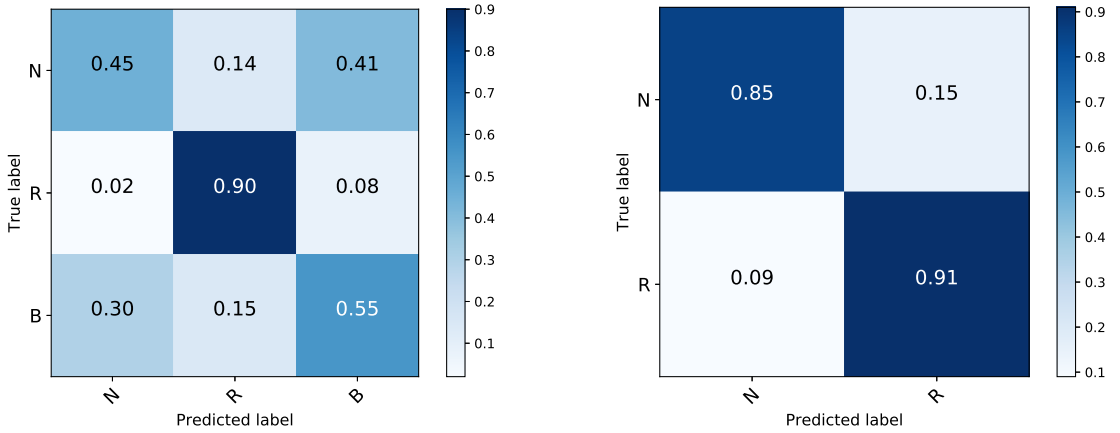


Figure 6.3: The accuracy during training for the train set and the validation set for model A - C_3 in Table 6.2. As training continues the training accuracy keeps increasing.

6.1.7. Results Final Model

Figure 6.4b shows a confusion matrix for the model implemented for the radar data (model A - C_3 in Table 6.2). The model develops a slight preference for the Rifle (R) class. This might be partially explained by the fact that during the measurements occasionally someone puts their hand in their pocket and therefore stops moving their arms, as the arm motion was identified as the main feature of interest.

In case the same model is trained on all three classes (the final softmax layer is extended to 3 neurons) the classification accuracy drops. Figure 6.4a shows the confusion matrix if the classifier is trained on all three classes. The model is not able to make a distinction between the Nothing (N) and the Backpack (B) class.



(a) Confusion matrix for all three classes. . Classification accuracy of 63.81 %

(b) Confusion matrix for Nothing and Rifle class. Classification accuracy of 88.42 %.

Figure 6.4: (a) Confusion matrix for the Nothing (N), Rifle (R) and Backpack (B) class. The model is not able to distinguish the Nothing and Backpack class. (b) Confusion matrix for training and validation for just the Nothing (N) and Rifle (R) class. Corresponding with model A - C_3 in Table 6.2.

6.1.8. Gait Phase Cycle Dependency

Figure 6.5 shows the dependency of the classification versus time. The plots were obtained using model A - C_3 in Table 6.2. The classification score on the bottom of the spectrogram shows the output of the softmax layer for the true/correct class. The red square in the figure is used to illustrate a frame drawn from the spectrograms and the arrow indicates the corresponding classification score. This spectrogram is classified correct all the time. The score on the left ranges from 0.5 to 1, as this model only classifies two classes it makes a correct prediction over the entire spectrogram.

Figure 6.6 shows a similar plot which is mostly falsely classified. After verifying the video data it seems that this particular person was walking with with his arms stiff besides his body, therefore not moving his arms. Which emphasizes the relevance of the arm motion for the correct classification of the Nothing and the Rifle class. Furthermore there does not seem to be a direct relation with the clutter of the tree, visible in the spectrogram in Figure 6.5 between index 500 and 650.

6.1.9. Validation

Figure 6.7 shows the results of the GradCam visualisation for the radar data for the Nothing (N) class. The obtained saliency maps have the dimensions of the spatial dimensions of the last convolutional layer. These saliency maps are then rescaled to the dimensions of the original input. This however causes the images to spread out and different interpolation techniques for this rescaling also affect the final saliency map slightly.

The figure on the right shows a frame of the spectrograms with the saliency map on top. The reddest pixels are most strongly associated with the correct class. On basis of this sample the Nothing class is most strongly associated with the arm motion.

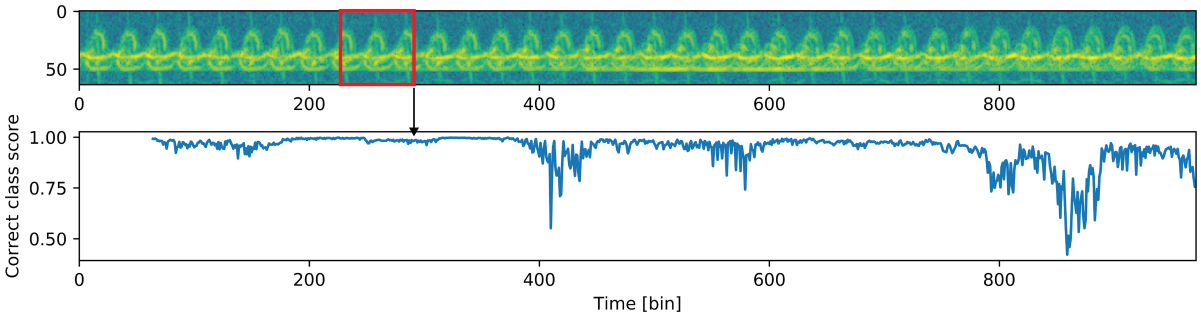


Figure 6.5: The spectrogram belongs to the Rifle class. The 'Correct class score' shows the output of the softmax layer for the correct class (Rifle) versus time. The red square shows a frame of 64 × 64 on which training and validation is done. Each frame in this spectrogram is classified correctly.

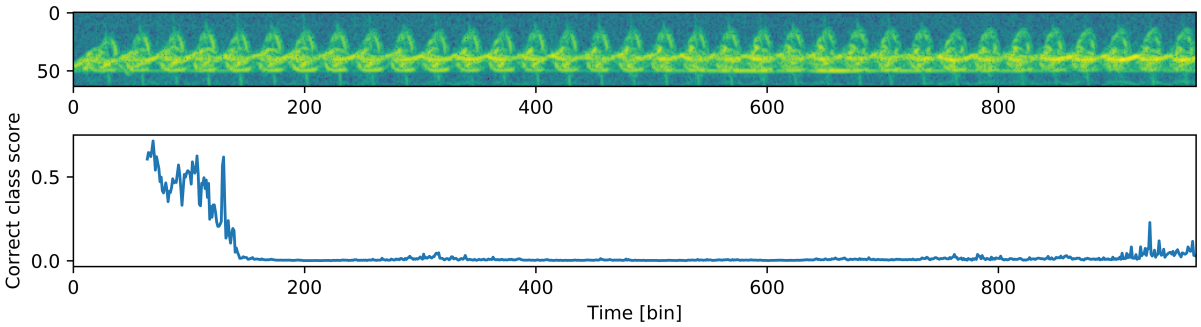
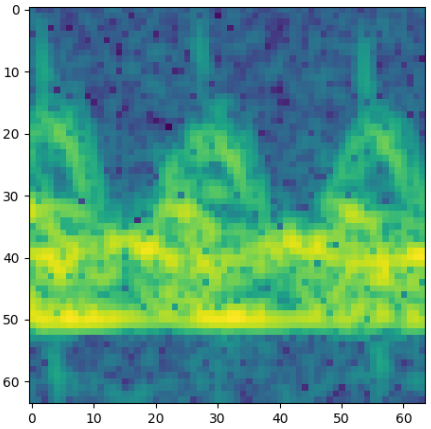
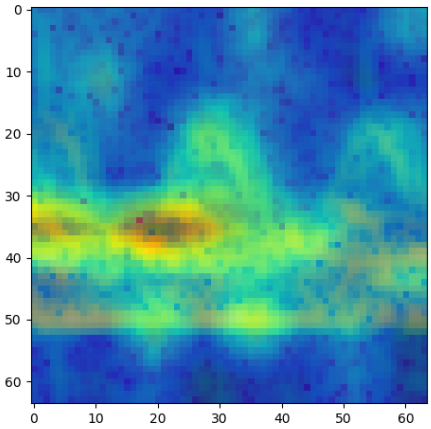


Figure 6.6: The spectrogram belongs to the Nothing class. The 'Correct class score' shows the output of the softmax layer for the correct class versus time. Almost each frame in this spectrogram is misclassified.



(a) Radar frame for the Nothing class.



(b) Radar frame with saliency map on top. The red region indicates the most relevant pixels for the classification of specified (Nothing) class.

Figure 6.7: Result of the gradcam visualisation for a selected sample. The left side shows a radar frame belonging to the Nothing class, on the right the saliency map from the GradCam method is rescaled and shown on top of the radar frame on the left. This method clearly associates the arm motion with the Nothing class. Although some other regions of this frame show some response as well.

6.2. Video Implementation

This section discusses the implementation of a convolutional neural network on the (single) frames from the video dataset as discussed in Chapter 5. First the training and validation procedure is discussed.

6.2.1. Training, Validation & Preprocessing

A similar approach as for the radar data was used to train and validate the models for the video data.

Preprocessing As preprocessing step the frames are divided by 255 to convert the RGB color channels in the range from 0 to 1 and the mean of this *training* dataset, the mean image with size $128 \times 64 \times 3$. is subtracted from each frame.

Training During training 20 random frames from each measurement in the training set are drawn, these frames are randomly shuffled and the model is trained for an epoch (until all frames in this set are used).

Validation After each epoch each frame in the validation set is propagated through the model and the loss and validation accuracy are computed. During the validation process the previously obtained *training* dataset mean image is subtracted.

6.2.2. Grid Search

Similarly as for the radar case a grid search for the video data is done to obtain an optimal model for classification of the video data. Table 6.4 shows the results of the grid search. After each convolutional layer and fully connected layer a ReLU activation function is applied. Zero padding is applied such that each convolutional layers input and output dimension are the same. The max pooling operation uses a 2×2 kernel with a stride of 2, therefore reducing the input dimensions by half.

Equivalently as for the radar implementation the final models were trained using stochastic gradient descent with the parameters in Table 6.3. Table 6.5 shows the average classification results for selected models. Based on these results 'Model A' from the table is selected. This data also shows the fluctuation of the classification accuracy over different runs.

Table 6.4: Grid search for the video data with input image dimensions 128 × 64. The results are obtained for the three class scenario. The depth refers to the amount of convolutional kernels in each layer. The 'Dense layers' column refers to the amount of nodes in the fully connected layers at the end of the model. The model with the highest classification score in this single run is shown in bold face.

Depth	Kernel size	Dense layers	Classification score (%)
16,16-32,32-64,64-128,128-256,256	3x3	1000, 1000	81.51
16,16-32,32-64,64-128,128-256,256	3x3	500, 500	83.52
16,16-32,32-64,64-128,128	**3x3**		**84.28**
16-32-64-128-256	3x3		81.03
10-10-10-10-10	3x3		63.32
20-40-60-80-80	3x3		81.10
30-30-30-30-30	3x3	100, 100	75.90
20-30-40-50	3x3	500, 500	81.10
	5x5		82.50
	7x7		80.95
8-16-32-64	3x3		75.75
	5x5		81.86
	7x7		82.29
20-20-20-20	3x3		76.74
	5x5		79.94
	7x7		77.15
20-40-60-80	3x3		78.74
30-40-50	3x3		78.27
	5x5		81.85
	7x7		83.16
16-32-64	3x3		80.10
	5x5		81.49
	7x7		81.95

Table 6.5: Average classification accuracy for selected models trained 5 times. Models were trained using the parameters in Table 6.3 with Dropout with a chance of 0.5 added after the last two fully connected layers. The columns C_i stand for the classification accuracy for run i. The average classification accuracy is shown on the right. All classifications accuracies are in %

Model	Width	Kernel size	C_1	C_2	C_3	C_4	C_5	C_{av}(%)
A	16,16-32,32-64,64-128,128	3x3	84.28	82.27	81.49	**85.46**	82.31	**83.16**
B	8-16-32-64	7x7	81.33	78.86	82.29	80.78	82.21	81.09

6.2.3. Final model

The final model mentioned in Table 6.4 (Model A - C_4) is shown in bold face. This model is summarized below:

- Input layer (size: 128 × 64 × 3)
- Convolutional layer (16 filters, size: 3 × 3)
- Convolutional layer (16 filters, size: 3 × 3)
- Max pooling (2 × 2)
- Convolutional layer (32 filters, size: 3 × 3)
- Convolutional layer (32 filters, size: 3 × 3)
- Max pooling (2 × 2)
- Convolutional layer (64 filters, size: 3 × 3)
- Convolutional layer (64 filters, size: 3 × 3)
- Max pooling (2 × 2)
- Convolutional layer (128 filters, size: 3 × 3)
- Convolutional layer (128 filters, size: 3 × 3)
- Max pooling (2 × 2)
- Fully connected: (500 Neurons)
- Fully connected: (500 Neurons)
- Output layer: (3 Neurons)

An overview of the model is given in Figure 6.8. 'CNN' refers to the convolutional and max pooling stages. The final feature maps, after the last max pooling layer are flattened and two fully connected layers are added as is shown in the figure.

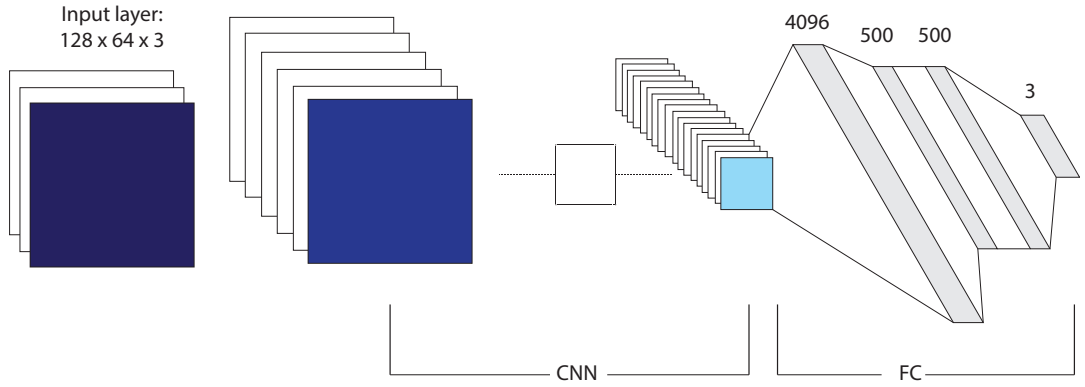


Figure 6.8: Overview of the convolutional neural network for classification of the video data. The depths of the feature maps in the figure are not the actual depth but are used to illustrate the increasing depth of the feature maps for deeper layers. The softmax layer is used on the last layer to obtain the probability for each class.

6.2.4. Loss & Accuracy

Figure 6.9 shows the training and validation loss for the model chosen in Table 6.4 (Model A - C_4. Figure 6.10 shows the training and validation accuracy. Although the classification score increases only slightly beyond epoch 75 the loss starts increasing which is an indication that the model is learning poor features.

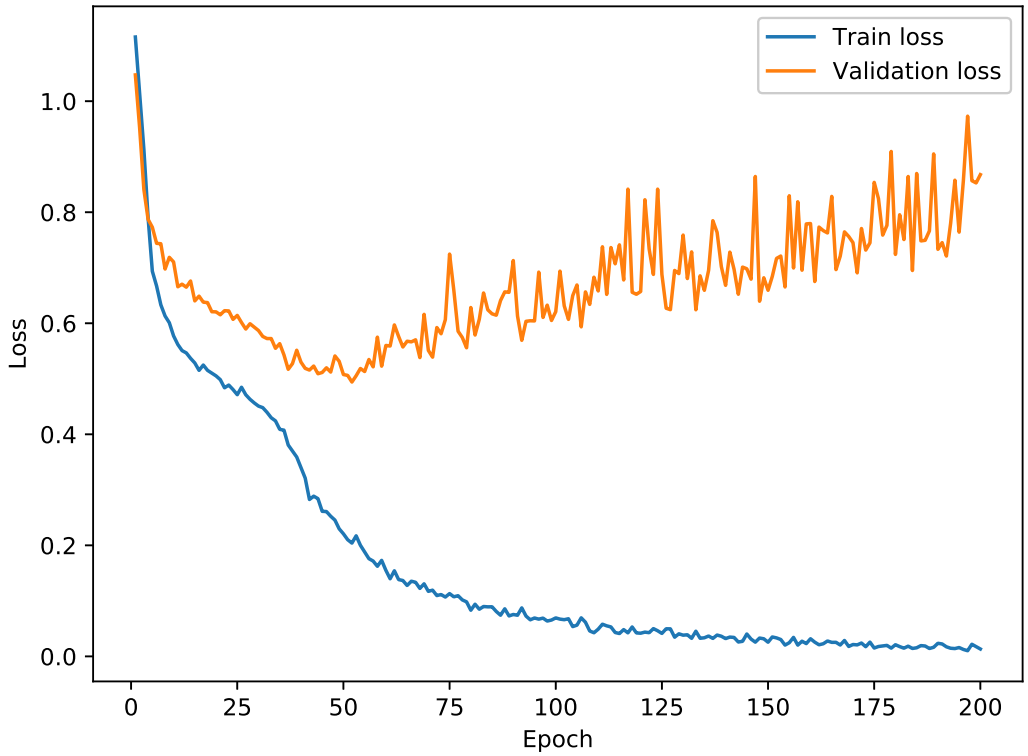


Figure 6.9: The training and validation loss for the best performing model obtained from Table 6.5.

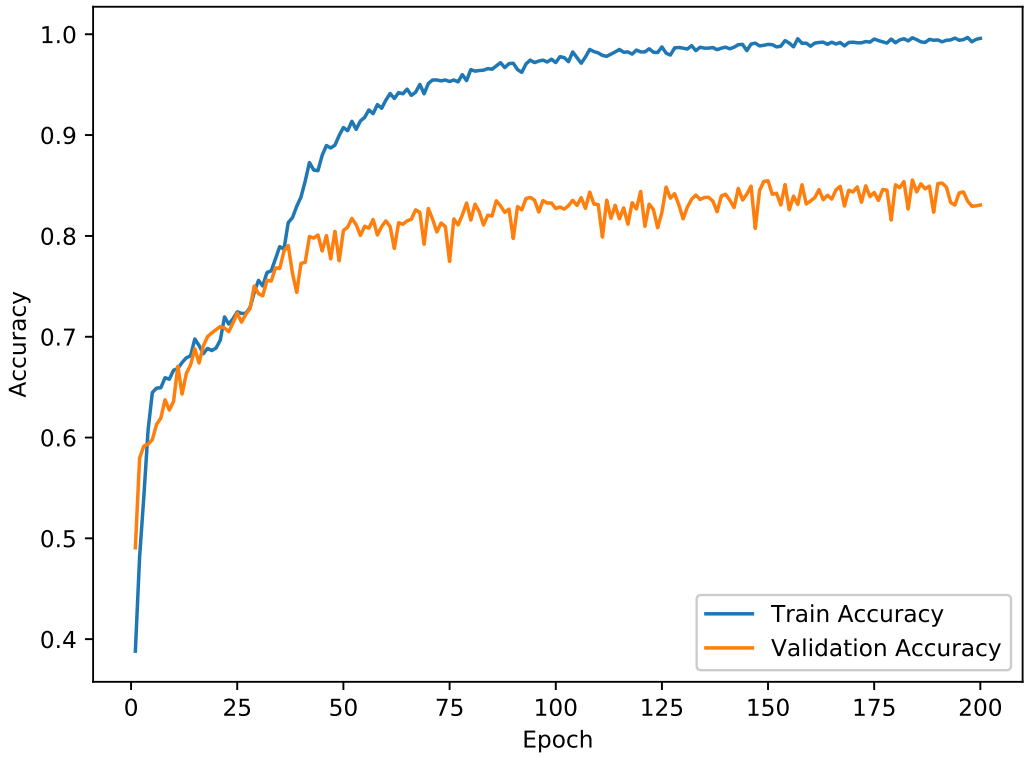
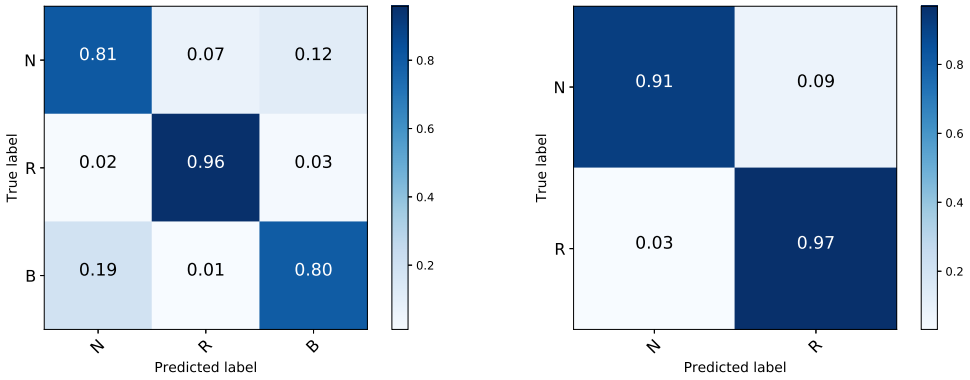


Figure 6.10: The training and validation accuracy for the best performing model obtained from Table 6.5. The best classification accuracy on the validation set is obtained at Epoch 150.

6.2.5. Results Final Model

Figure 6.11a shows the confusion matrix for the three classes. The video data is clearly able to detect the Rifle class. There is however some confusion between the backpack and Nothing class. The classification scores can be related to the visiblity of the backpack and the metal pole in the video frames. As the metal pole is clearly visible in the video data it makes sense that the classifier is able to detect this correctly. The backpack is most likely recognized by the visibility of the straps in the video frames, these straps are however much more difficult to see. Similarly a model was trained for just the Nothing and Rifle class in Figure 6.11b.



(a) Classification accuracy 85.46 %

(b) Classification accuracy 94.17 %

Figure 6.11: (a) Confusion matrix for model A - C_4 from Table 6.5. The video classifier is clearly able to distinguish the rifle (R) class but there is some confusion between the nothing (N) and the backpack (B) class.

6.2.6. Validation

Figure 6.12 shows the results of the GradCam method. The red pixels are the most important pixels for the correct classification of the corresponding class. The saliency map for the Rifle class (Figure 6.12b) is most intuitive. The metal pole is clearly associated with the Rifle class. Figure 6.12a shows the saliency map for the Nothing class, the model seems to associate the hands besides the body with the Nothing class. Figure 6.12c shows an example of the saliency map for the Backpack class, the backpack straps seems to be related to the backpack class and a bit optimistic the hands beside the body are also associated with this class, although the feature extraction process does not seem to be optimal as other regions are also associated with this class.

The saliency maps have been normalised for each image separately, so a direct comparison for the different frames can not be made.

6.3. Conclusion

Two convolutional neural networks were obtained in this chapter. One for the classification of the video data (Figure 6.8) and one for classification of the radar data (Figure 6.1). For each model a grid search was done to optimise the model parameters. For the final optimisation of the convolutional neural networks the stochastic gradient descent algorithm was used in combination with Nesterov momentum.

Validation with the GradCam method showed that convolutional neural networks are able to associate some of the expected elements with the correct class (the rifle in the video frames and the arm motion in the radar data). It also made clear that there is still room for improvement of the models as not all the saliency maps show a clear relation with the expected elements.

The convolutional neural networks in (Figure 6.8) and (Figure 6.1) will form the basis for the discussion on multimodal deep learning in the next chapter.

(a) Nothing (b) Rifle (c) Backpack

Figure 6.12: Gradcam results for the three classes. (a) The Nothing (N) class, a person walking normally. (b) The Rifle (R) class, is most clearly highlighted. (c) The Backpack (B) class, a person walking with a backpack on their back. The saliency maps are normalised for each image separately. The frames have been blurred and occluded deliberately for privacy reasons.

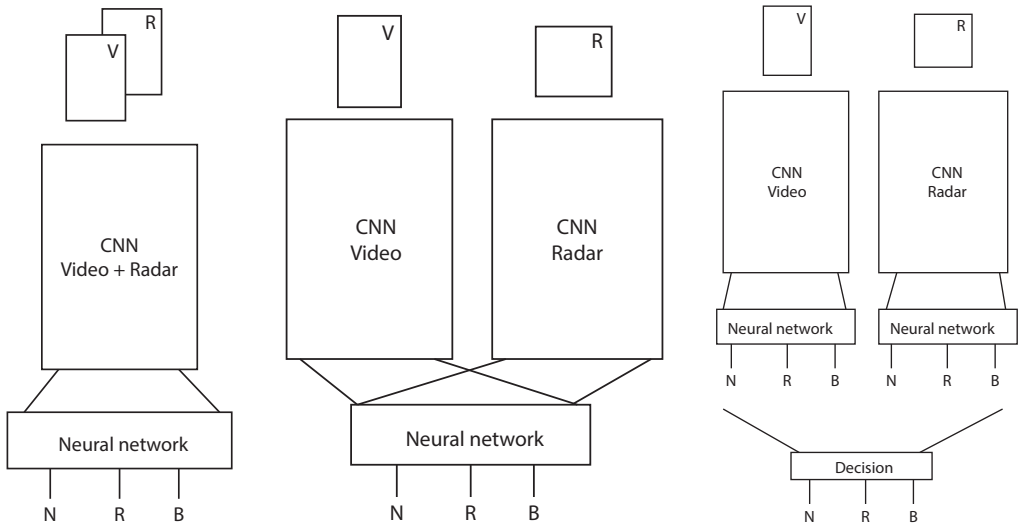
7

Multimodal Deep Learning

“Our experience of the world is multimodal - we see objects, hear sounds, feel texture, smell odors and taste flavors. Modality refers to the way in which somethings happens or is experienced” [19]. Research problems including multiple of such modalities are referred to as multimodal.

7.1. Data Fusion

In general data fusion can be done at different processing stages [49]. A general split that can be made in data fusion is in early, at feature level, and late fusion, at decision level [19]. The options in consideration in this work in the context of convolutional neural networks are referred to, as ‘Raw images’ Figure 7.1a, ‘Early’ Figure 7.1b and ’Late’ Figure 7.1c and relate to the fusion depth. To be able to take into account the possible correlation between the radar and video data a shared representation (Raw Images or Early fusion) has the preference.



(a) Raw Images (b) Early (c) Late

Figure 7.1: Simplified schematics of the proposed fusion schemes. The input is on the top, the input frames are indicated by V (Video) and R (Radar). The output, the classification scores for each class, are on the bottom indicated by (N, R and B). CNN refers to the convolutional stages of a convolutional neural network. ‘Neural network’ refers to the fully connected layers at the end of the neural network. The Late fusion scheme on the right requires some aggregation strategy to make a decision on the combined classification scores.

7.2. Raw Images

The 'Raw Image' fusion scheme is shown in Figure 7.1a. The input frames are shown above the figure. The CNN parts refers to the convolutional and max pooling stages as discussed in Chapter 4. As most fusion methods focus on early fusion to be able to capture the correlation of events the idea of the 'Raw images' fusion scheme is to fuse the data as early as possible.

As the dataset used for the training and validating the multimodal architectures is different than the video dataset the proposed model for the video data in Chapter 6 is trained and validated on the video frames in the multimodal dataset to make a fair comparison, this is indicated by the 'Video only' row in Table 7.1.

To implement the fusion scheme in Figure 7.1a the video data is rescaled to the same dimensions as the radar data. The video data which previously had dimensions $128 \times 64 \times 3$ (Height × Width × Depth) is resized to $64 \times 64 \times 3$. The synchronized radar frames are then added as a fourth channel. The combined radar and video input then has dimensions $64 \times 64 \times 4$. The model in Figure 6.1 with the adjusted input is used to obtain the results in Table 7.1. As the dimensions of the video data were reduced and a different model was used, this model was also trained and validated with just the video data in the multimodal dataset. These results are shown in the 'Video only' row in Table 7.1. Overall it seems that concatenating the frames reduces the classification accuracy. The radar frames might be just adding noise to the feature extraction process as the classification accuracy for the 'Video only' data is higher.

Table 7.1: The best models in terms of classification accuracy are shown over 5 runs for training and validation on the classes Nothing, Rifle and Backpack (NRB) and just the Nothing and Rifle (NR) class. The results are obtained with the Adam Optimization algorithm with a learning rate of 0.0001 over 70 epochs (Other parameters at default as proposed in [29]).

Classes:	NRB	NR
Radar only	63.18	87.51
Video only	82.76	94.87
Multimodal 'Raw Images'	79.58	91.12

Another problem with this method is that it requires the convolutional stages of the model to have the same size which is not necessarily the optimal feature extraction process for the different modalities. To obtain an (sub)optimal model for the new input a grid search as in Table 6.1 would need to be carried out. As this was not a very practical solution and the impact on the feature extraction process is not entirely clear this method was not further explored.

7.3. Early

The 'Early' fusion scheme is shown in Figure 7.1b. A convolutional neural network is used for feature extraction for each modality and the fusion is performed after feature extraction is done. A neural network is trained to perform classification on this shared feature space. Advantage of the Early fusion scheme, at feature level, is that it might be able to utilize the correlation between multiple features from different modalities [18], which might benefit the classification performance.

7.3.1. Implementation

Merging the models which were obtained previously, Figure 6.8 and Figure 6.1 results in a model with a final classification stage with a feature vector of 4896 neurons, two fully connected layers with 1000 neurons and the Softmax layer with 3 neurons. This model was trained with random initialization several times with the parameters in Table 7.2. Table 7.3 summarises the classification accuracies, based on the trained models no significant difference can be identified.

Table 7.2: Parameters used for training the models in Table 7.3. During training 20 synchronised radar and video frames are randomly drawn for each measurement in the training set.

Optimiser	Learning rate	Nesterov Momentum	Batch size	Learning rate decay	Epochs
SGD	$2 * 10^{-4}$	0.9	16	$1 * 10^{-6}$	300

Table 7.3: Classifications results obtained for training the video model on the same dataset as the multimodal model for the three classes (NRB). Early stopping was applied to obtain the best model in terms of classification accuracy on the validation set. The parameters in Table 7.2 were used for training. The classifications scores are shown in (%) for the different runs.

Run:	1	2	3	4	5
Radar only	63.63	63.10	62.89	**64.10**	62.08
Video only	85.38	84.47	82.33	84.43	**86.60**
Early fusion	85.59	**86.96**	86.16	84.17	86.49

7.3.2. Best Model

Directly concatenating the models obtained in Chapter 6 does not necessarily lead to an optimal multimodal model as the combined feature vector from both modalities is larger, additional fine-tuning of the last layers of the neural network is required to obtain the best performing model. With some experimentation of different parameters and optimizers a better performing model was found. This model is used for further analysis.

The final model still uses the same convolutional stages but uses two fully connected layers with a size of 500 as this seemed to improve the overall classification accuracy. The final model is illustrated in Figure 7.2. The accuracy and loss curves during training have been included in Figure A.5 and Figure A.6.

This model obtained a classification accuracy of 90.60 % for the three classes. However as this might also just be an improved classification on just the video part the impact of the radar data and the video data needs to be verified. To analyse the influence of the single modalities on the multimodal model the inputs are omitted one after the other and the model is validated on the validation set. In Figure 7.3c the video input is omitted, the mean video image is offered as input. The impact of removing the video input is significant. The overall classification accuracy is reduced to 44.34 %.

In Figure 7.3b the radar input is omitted, the mean radar image is offered as input as this results, due to the zero bias initialisation, in a zero input for the radar convolutional neural network stages. The classification accuracy reduces only slightly, a decrease of 0.9 % is observed.

The model trained on the three classes can also be used to classify just the Nothing and Rifle class. The Backpack and Nothing class are aggregated in the Nothing class in Figure 7.3d. Figure 7.3 summarises the results for each scenario.

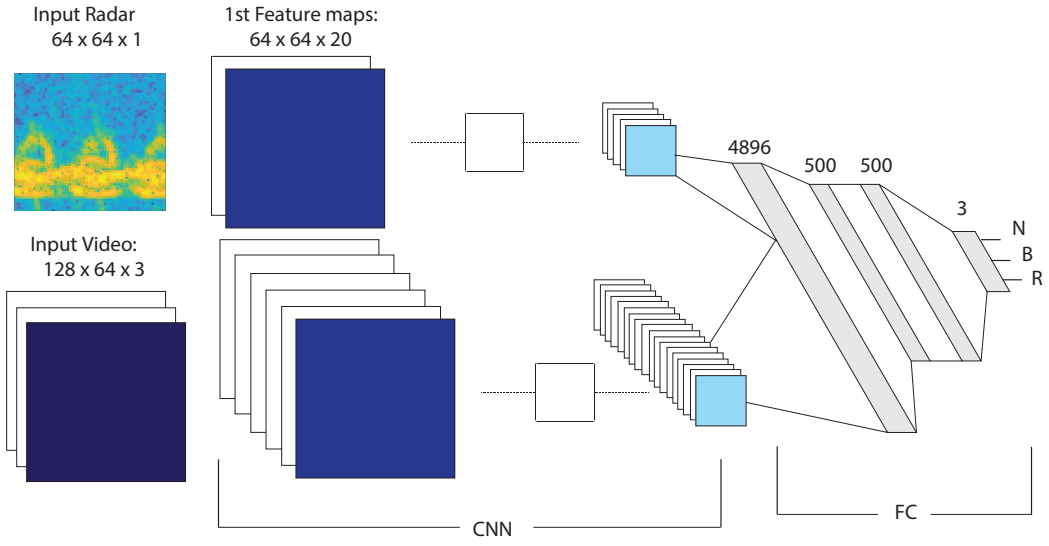
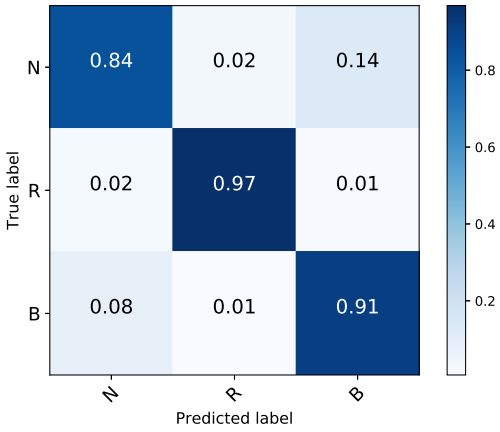


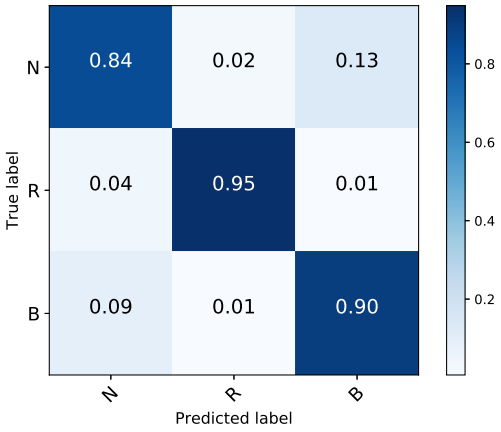
Figure 7.2: Overview of the convolutional neural network for classification of the multimodal model. The radar convolutional stages are shown on the top. The video convolutional stages are shown on the bottom. The input is shown on the left, the output classes on the right. The depth of the feature maps in the figure is not the actual depth but is used to illustrate the increasing depth of the feature maps for deeper layers. The softmax layer is used on the last layer to obtain the probability for each class. The figure also illustrates the size difference in the obtained feature vectors.

Table 7.4: The table shows the classification accuracies corresponding with Figure 7.3. The impact of the radar data on the multimodal neural network is small as removing the radar input only slightly reduces the classification accuracy. In case the same model is used to just classify the Nothing and Rifle class by aggregating the Nothing and Backpack class the overall score is also slightly improved. All accuracies are in (%) correct classification. The results were obtained using Adam with a learning rate of 0.0001.

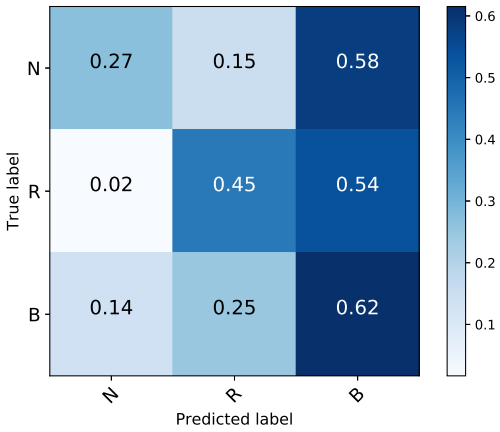
	Overall accuracy	Without radar input	Without video input	NR Classification
Accuracy (%)	90.60	89.70	44.34	97.64



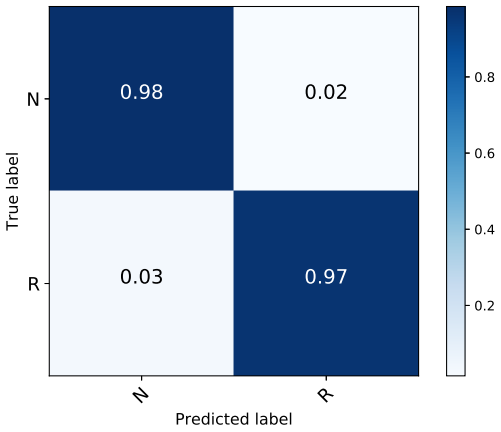
(a) Early fusion for all three classes. Classification accuracy: 90.60 %



(b) Without radar input. Classification accuracy: 89.70 %



(c) Without video input. Classification accuracy 44.34 %



(d) Model used to classify just the Nothing and Rifle class. Classification accuracy 97.64 %.

Figure 7.3: Confusion matrices for the early fusion scheme for different scenarios. (a) All three classes. (b) The radar input is omitted. (c) The video input is omitted. (d) The model is used to classify just the Nothing/Backpack and Rifle class by aggregating the Nothing and Backpack class.

7.4. Late

The 'Late' fusion scheme is illustrated in Figure 7.1c. The fusion is performed at a decision level based on some aggregation strategy. Taking the mean of the separate classifications or training a classifier to make a decision on the scores are possible strategies. In [50] methods and algorithms for combining classifiers are discussed, in general these methods relate to taking some sort of average. The softmax classification scores of models obtained in Figure 6.1 and Figure 6.8 for the single modalities are combined.

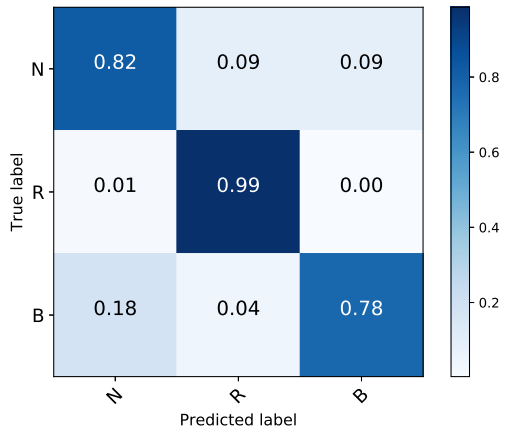
To combine the predictions the scores of the softmax outputs are averaged, the maximum score of the average is then used to predict the class. The results of this method are shown in Table 7.5. A slight improvement is visible for the three class classification (NRB column), the classification accuracy improves to 86.12 %, an improvement of 0.34 %. For the two class classification (NR column), the classification accuracy improves to 95.65 %, an improvement of 1.08 %.

This method is expected to improve the classification accuracy as one classifier miss classifies a class and produces a lower score than the classifier for the other modality. If there would be no disagreement between the classifiers, taking the average would not improve the classification accuracy.

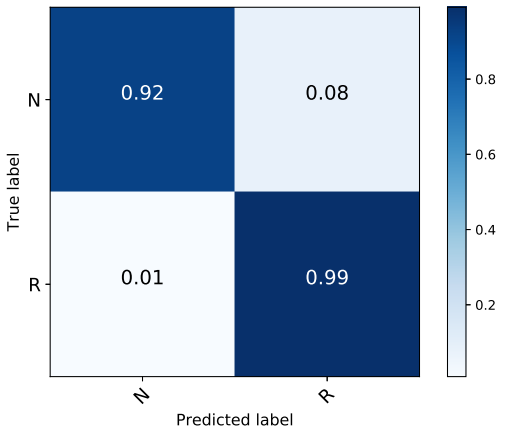
The confusion matrices, see Figure 7.4, also show the confidence of both classifiers in the Rifle (R) class, as after averaging the classification accuracy for this class reaches 99%.

Table 7.5: The results for combining the predictions of two models. The 'Video' row refers to the classification accuracy of the video model in Figure 6.8 validated on the multimodal validation set. Similarly the 'Radar' row shows the classification scores on the multimodal dataset. The average row indicates the classification accuracy when the scores of the individual classifiers are averaged. All accuracies are in (%).

Classes:	NRB	NR
Video	85.78	94.57
Radar	63.12	87.14
Average	86.12	95.65



(a) Confusion matrix for all three classes . Classification accuracy of 86.12 %



(b) Confusion matrix for Nothing and Rifle class. Classification accuracy of 95.65 %.

Figure 7.4: (a) Confusion matrix for the Nothing (N), Rifle (R) and Backpack (B) class for the averaged classification. The model is not able to distinguish the Nothing and Backpack class. (b) Confusion matrix for the Nothing (N) and Rifle (R) class for the averaged classification.

8

Results

To give an overview and make an comparison of the different methods this chapter summarises the results obtained in Chapter 6 and Chapter 7.

8.1. Results Single Modality Dataset

The results for the single modality implementations are summarised in Table 8.1. Both the radar and video classifiers are able to correctly identify the Rifle class in most of the cases. For the radar spectrogram data this classification was associated with the presence or absence of the arm motion.

Table 8.1: The classification results obtained in Chapter 6. All classification accuracies are in %. The column 'NRB' shows the classification accuracy obtained for training and validation of all three classes. The column 'NR' shows the classification accuracy for the two classes Nothing and Rifle.

Classes:	NRB	NR
Video only	85.46	94.17
Radar only	63.81	88.42

8.2. Results Multimodal Dataset

As the multimodal dataset is a subset of the datasets for the single modalities, the classification accuracy on the video part of the multimodal dataset is verified with the video classifier in Figure 6.8. Similarly the classifier for the radar data is used to validate the results on the radar data in the multimodal dataset. The results of this validation procedure are shown in Table 8.2. Overall the results differ only slightly with the classification accuracies in Table 8.1.

Table 8.2: The classifiers trained on the single modality dataset validated on the respective data of the multimodal dataset.

Classes:	NRB	NR
Video	85.78	94.57
Radar	63.12	87.14

The classification accuracies for the different multimodal fusion schemes are summarised in Table 8.3. Based on the best obtained classification accuracy the Early fusion method, fusion at feature level, looks the most promising. Although this model also showed that the impact of the radar data on the classification was minimal. Further optimisation can be done to improve the performance of the models and the impact of the single modalities on the multimodal models could be adjusted by choosing a slightly different topology or using some additional methods during training.

Table 8.3: The classification results for the multimodal fusion schemes obtained in Chapter 7. The column 'NRB' shows the classification accuracy obtained for training and validation of all three classes. The column 'NR' shows the classification accuracy for the two classes Nothing and Rifle. All accuracies are in %.

Classes:	NRB	NR
'Raw Images'	79.58	91.12
Early	90.60	97.64
Late	86.12	95.65

9

Discussion & Recommendations

9.1. Radar Neural Network

Based on the observations of the performance of the radar classifier the classes Nothing and Rifle are not necessarily what is detected, a more accurate description of the classes would be 'arm movement' and 'no arm movement'. In case this classifier would be used on somebody walking with his hands in his pockets it will most likely detect such an event as the Rifle class. To be able to investigate the possibility to distinguish multiple classes that involve carrying e.g. different objects they should be included in the dataset.

9.1.1. GradCam

The Convolutional Neural network is able to associate the presence or absence of the arm motion with the Nothing and Rifle class. Nonetheless the saliency maps also show room for improvement.

9.1.2. Radar Spectrogram Window

During the setup of the different neural networks different window sizes for the training and validation of the spectrograms were tested, although not included in this work the classifications seem rather robust as long as a gait cycle is included in this window (>1 second). Advantage of a smaller window however is the reduction in the window size, this lead to an easier optimization of the convolutional neural network due to reduced complexity which also lead to reduced training times.

9.1.3. Weight initialization

The initialization of the weights, either 'Glorot' [31] or 'He' [32] did not seem to influence the validation accuracy but the models converged faster when using the weight initialisation proposed by 'He'.

9.2. Video Neural Network

The neural network trained for the video data showed that during training the loss started to increase rather quickly. Although the classification accuracy remained rather steadily extracting the wrong features from the video data can complicate the optimisation of the multimodal fusion schemes. The video dataset also contained a slight bias due to the Single Shot Detector used, as this method does not always detect a person in each frame, this can increase the classification accuracies on the video data as it is likely that the frames were no person is detected would also have trouble during the classification of the activity.
The backpack scenario was included in the dataset with a weight of 10 ± 0.5 kg, although the video camera is able to classify this activity reasonably well (82%) Figure 6.11a it does not necessarily detect the weight of the backpack. To be able to make a distinction between different backpack weights they should be included in the dataset. A heavier backpack will also influence the micro-Doppler signatures more heavily.

9.2.1. GradCam

The validation of the video neural network using the GradCam method showed that the feature extraction was not optimal, especially for the backpack case one would expect a clear focus on the backpack straps as they are visible in the frames. Difficulty when using methods like GradCam for validation is that such

techniques are more heuristic than scientific. As it is not known what the output of such a method will be, the interpretation of the saliency maps is left to the observer.

9.3. Deep Learning

In this work only a single scenario was included in the dataset in which the activities were performed, persons walking from 40 meters towards the radar and camera setup. Although the dataset was relatively small this limited scenario made the use of Deep Learning a viable solution, on basis of the obtained classification accuracies. In case classification needs to be performed under different angles, for example walking across the radar such scenarios need to be included in the dataset. This would however require a significant amount of data.

9.4. Multimodal

As the models trained for the video and radar convolutional neural network required different training times (Compare Figure 6.10 and Figure 6.3), to obtain the maximum classification accuracies training the multimodal models with random initialization might not be the best solution. An alternative would be to pre-train the convolutional stages to perform good feature extraction and then train and fine-tune a neural network on the combined feature space, although this is not expected to improve the results significantly as the confusions matrices for the Early multimodal fusion scheme show a high classification accuracy for the Rifle class and the neural network for the radar data was not able to distinguish the nothing and the backpack class. In general however, the models trained in all the chapters could be further optimized, which makes a comparison based on just classification accuracies somewhat difficult as putting more effort in optimising one model might lead to an improvement in the classification accuracy for that model, this does not necessarily mean that another model could not achieve these higher classifications.

To verify the impact of the radar and video data on the multimodal 'Early' fusion scheme the input was omitted one after the other. This showed that the impact of the video data on the model is much larger. This is however also caused by the size of the feature vectors from the different modalities, as the feature vector obtained from the video data is much larger it is to be expected that it has a larger impact on the overall classification. Different model architectures or training methods might be used to balance the impact of both modalities if that is desired.

Due to the data dependency of deep learning on the performance, a general conclusion about what the best fusion method is for different activities is difficult to answer. Overall it is to be expected that the presence of multiple modalities can help classification as long as it is able to resolve some ambiguity. Extracting the right features to do so and optimizing the right neural network is however not a trivial task. In audio visual speech recognition the suggestion is made that a certain position or shape of the mouth in a video frame can help resolve ambiguity in classifying certain phrases. An equivalent scenario, detecting a person in a certain position in a video frame with the associated micro-Doppler signature, is likely to help the classification of human activity when using radar micro-Doppler signatures and video data. A larger more diverse dataset should give more insight in what activities can and cannot be resolved.

9.5. Dataset

Some of the scenarios described in the introduction were not included in the dataset, to verify the capabilities of a multimodal neural network with radar and video in differing weather and lighting (at night) conditions these scenarios should be included in the dataset.

A single frame from the video data was synchronised with the start of a 1.28 seconds frame from the radar data to create the multimodal dataset. During this time multiple frames from the video camera can be used for the classification.

Furthermore the generation and synchronisation of the data was a time consuming process. Due to an error in the camera setup the frame rate of the video recordings was not constant but fluctuating. This resulted in a rather arbitrary average frame rate of 13.5 fps. To synchronize the radar data with the video data more easily, an automated synchronisation would be beneficial. As the processing of the radar data in a spec-

trogram offers some flexibility in the time resolution of the obtained spectrograms they could more easily be synchronised with the frame rate of the video camera.

Initially the radar input size for the convolutional neural network was chosen as 90 × 100 pixels and the video input data was chosen as 100 × 40 pixels however due to the application of the max pooling operation with a kernel size of 2 × 2 the dimensions are halved each time this operation is applied. This lead to less well defined feature maps as the dimensions are halved each time at some point a row or column of these feature maps is either omitted or additional zero padding needs to be applied. One way to prevent this problem is to choose the input image dimensions as a power (or multitude) of 2.

9.6. Generalisation

In this work the dataset was split in a single training and validation set on which the results were based. To get a better impression of how well the models and different fusion methods perform over the entire dataset a k-fold cross validation can be done.

10

Conclusion

Convolutional neural networks are an effective tool for the classification of human activity in both radar and video data. In this work three different human activities were considered:

- Walking
- Walking with a metal pole
- Walking carrying a backpack

For classification of a person walking with a rifle like object (a metal pole) and a person walking with his hands free, a convolutional neural network trained and validated on single frames from video data showed a classification accuracy of 94.57 %. A convolutional neural network trained to classify 1.28 second windows of human micro-Doppler signatures corresponding with the same classes obtains a classification accuracy of 87.14 %.

Validation with saliency maps of the neural networks showed that the neural network trained for the radar data is able to associate the presence or absence of the arm motion with respectively the nothing and rifle class. The video model was clearly able to associate the rifle class with the carrying of a metal pole.

The aim of this work was to investigate the possibility to improve human activity classification using multimodal deep learning with radar and video data. Both early and late fusion methods show the possibility to improve the classification accuracy.

The best observed classification accuracy uses an early fusion method, fusion at feature level, and obtained a classification accuracy of 97.64 % when distinguishing people walking with a metal pole and without. For the classification of the three classes this model obtained an accuracy of 90.60 %. The impact of the radar data on the classification accuracy however was minimal as removing the radar data shows a drop in classification accuracy of just 0.9 %, identifying the video data as the dominant modality in this particular setup. In case the radar data is omitted during validation the accuracy drops significantly, this effect is probably also related to the large difference in obtained feature vectors for the different modalities.

Bibliography

[1] M. S. Seyfioğlu, A. M. Özbayğglu, and S. Z. Gurbuz, "Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities," *IEEE Transactions on Aerospace and Electronic Systems*, 2018.

[2] M. Paul, S. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications - a review," vol. 2013, p. 25, 11 2013.

[3] R. Bodor, B. Jackson, and N. Papanikolopoulos, "Vision-based human tracking and activity recognition," in *Proc. of the 11th Mediterranean Conf. on Control and Automation*, vol. 1, 2003.

[4] P. C. Ribeiro, J. Santos-Victor, and P. Lisboa, "Human activity recognition from video: modeling, feature selection and classification architecture," in *Proceedings of International Workshop on Human Activity Recognition and Modelling*. Citeseer, 2005, pp. 61–78.

[5] S. Srivastava and S. Sural, "Human gait recognition using temporal slices," in *Pattern Recognition and Machine Intelligence*, A. Ghosh, R. K. De, and S. K. Pal, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 592–599.

[6] P. Cao, W. Xia, M. Ye, J. Zhang, and J. Zhou, "Radar-id: human identification based on radar micro-doppler signatures using deep convolutional neural networks," *IET Radar, Sonar & Navigation*, 2018.

[7] F. H. C. Tivive, A. Bouzerdoum, and M. G. Amin, "A human gait classification method based on radar doppler spectrograms," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, p. 389716, 2010.

[8] J. Li, S. L. Phung, F. H. C. Tivive, and A. Bouzerdoum, "Automatic classification of human motions using doppler radar," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 2012, pp. 1–6.

[9] Y. Kim and H. Ling, "Human activity classification based on micro-doppler signatures using a support vector machine," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 5, pp. 1328–1337, 2009.

[10] ——, "Human activity classification based on micro-doppler signatures using an artificial neural network," in *Antennas and Propagation Society International Symposium, 2008. AP-S 2008. IEEE*. IEEE, 2008, pp. 1–4.

[11] Y. Kim, S. Ha, and J. Kwon, "Human detection using doppler radar based on physical characteristics of targets," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 2, pp. 289–293, 2015.

[12] Y. Kim and T. Moon, "Human detection and activity classification based on micro-doppler signatures using deep convolutional neural networks," *IEEE geoscience and remote sensing letters*, vol. 13, no. 1, pp. 8–12, 2016.

[13] R. Trommel, R. Harmanny, L. Cifola, and J. Driessen, "Multi-target human gait classification using deep convolutional neural networks on micro-doppler spectrograms," in *Radar Conference (EuRAD), 2016 European*. IEEE, 2016, pp. 81–84.

[14] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 689–696.

[15] E. Tatulli and T. Hueber, "Feature extraction using multimodal convolutional neural networks for visual speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2971–2975.

[16] Y. Yasui, N. Inoue, K. Iwano, and K. Shinoda, "Multimodal speech recognition using mouth images from depth camera," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2017*. IEEE, 2017, pp. 1233–1236.

[17] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein, "Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation," *arXiv preprint arXiv:1804.03619*, 2018.

[18] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli, "Multimodal fusion for multimedia analysis: a survey," *Multimedia systems*, vol. 16, no. 6, pp. 345–379, 2010.

[19] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[20] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin, *Principles of modern radar*. Citeseer, 2010.

[21] V. C. Chen, *The micro-Doppler effect in radar*. Artech House, 2011.

[22] J.-J. Lin, Y.-P. Li, W.-C. Hsu, and T.-S. Lee, "Design of an fmcw radar baseband signal processing system for automotive application," *SpringerPlus*, vol. 5, no. 1, p. 42, 2016.

[23] J. J. M. De Wit, "Development of an airborne ka-band fmcw synthetic aperture radar," Ph.D. dissertation, TU Delft, Delft University of Technology, 2005.

[24] A. Wojtkiewicz, J. Misiurewicz, M. Nalecz, K. Jedrzejewski, and K. Kulpa, "Two-dimensional signal processing in fmcw radars," *Proc. XX KKTOiUE*, pp. 475–480, 1997.

[25] V. C. Chen, W. J. Miceli, and D. Tahmoush, *Radar micro-Doppler signatures: processing and applications*. The Institution of Engineering and Technology, 2014.

[26] P. van Dorp *et al.*, *Human motion analysis with radar*. 9789090252407, 2010.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[28] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] T. Schaul, I. Antonoglou, and D. Silver, "Unit tests for stochastic optimization," *arXiv preprint arXiv:1312.6055*, 2013.

[31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *CoRR*, vol. abs/1502.01852, 2015. [Online]. Available: http://arxiv.org/abs/1502.01852

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[34] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[35] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *CoRR*, vol. abs/1511.07289, 2015. [Online]. Available: http://arxiv.org/abs/1511.07289

[36] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[37] J.-C. Hou, S.-S. Wang, Y.-H. Lai, Y. Tsao, H.-W. Chang, and H.-M. Wang, "Audio-visual speech enhancement using multimodal deep convolutional neural networks," *arXiv preprint arXiv:1703.10893*, 2017.

[38] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[39] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*. Springer, 1999, pp. 319–345.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[42] L. Fang, D. Cunefare, C. Wang, R. H. Guymer, S. Li, and S. Farsiu, "Automatic segmentation of nine retinal layer boundaries in oct images of non-exudative amd patients using deep learning and graph search," *Biomedical optics express*, vol. 8, no. 5, pp. 2732–2744, 2017.

[43] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016. [Online]. Available: http://arxiv.org/abs/1602.07261

[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[45] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016. [Online]. Available: http://arxiv.org/abs/1610.02391

[46] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015. [Online]. Available: http://arxiv.org/abs/1512.02325

[47] F. Chollet, *Deep learning with Python*. Manning Publications Co., 2017.

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

[49] F. Castanedo, "A review of data fusion techniques," *The Scientific World Journal*, vol. 2013, 2013.

[50] L. I. Kuncheva, *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

A

Appendix

A.1. Radar Implementation

A.1.1. Classes NRB

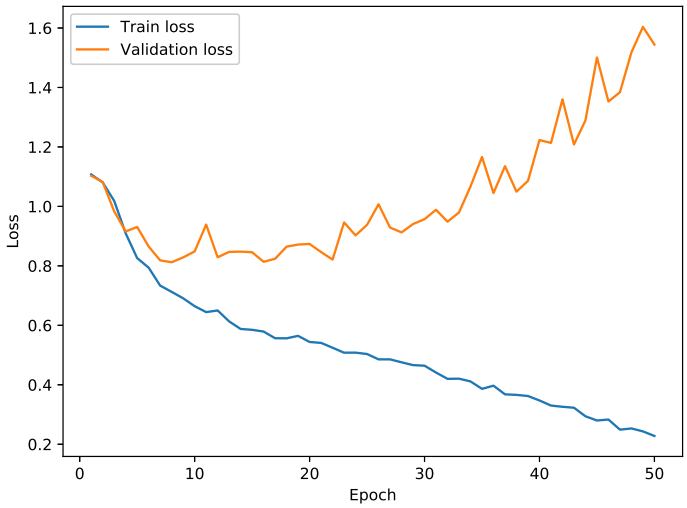


Figure A.1: The training and validation loss for the radar model trained on all three classes. The figure illustrates the inability of the model to learn useful features to distinguish the Nothing and Backpack class.

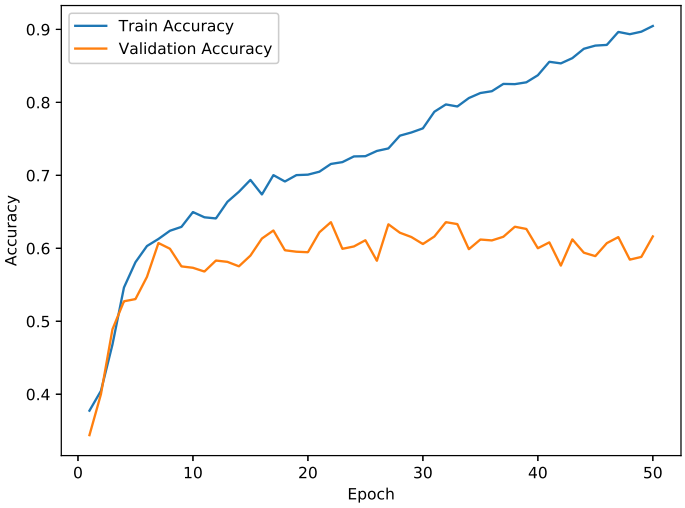


Figure A.2: Radar model trained for all three classes. The linear progression on the training accuracy indicates the model is overfitting/memorizing the training data.

A.1.2. Loss & Accuracy

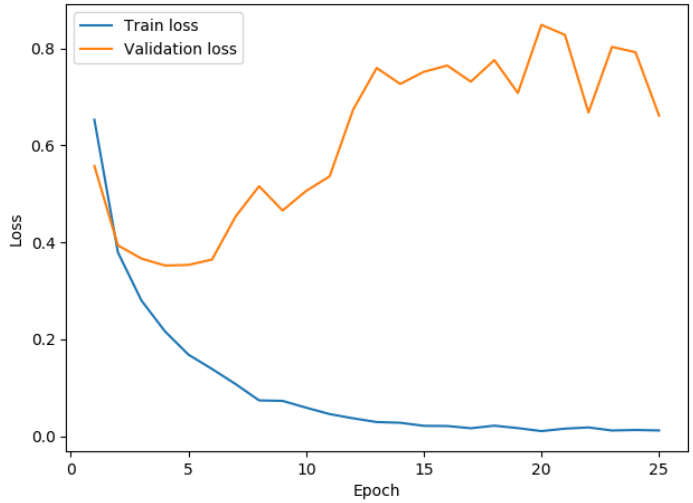


Figure A.3: The training and validation loss. As training progresses the training loss keeps reducing until it almost hits 0 loss. The loss on the validation set however starts increasing after epoch 5, the model starts to overfit on the training data.

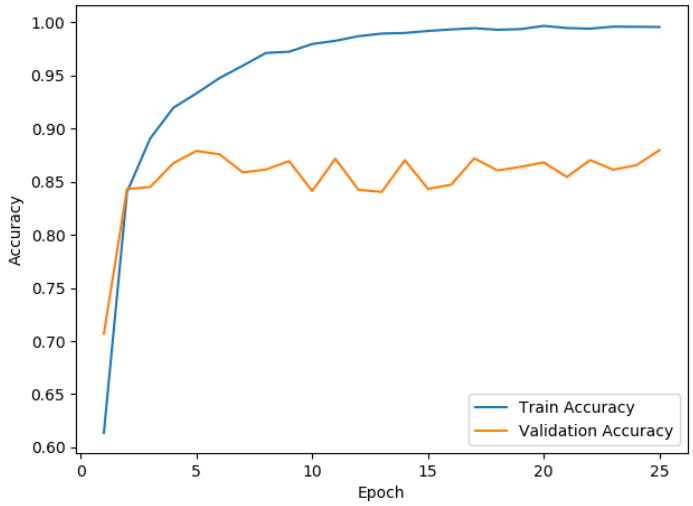


Figure A.4: The accuracy during training for the train set and the validation set. As training continues the training accuracy keeps increasing. The maximum score is reached at epoch 5.

A.2. Multimodal

A.2.1. Early Fusion

Figure A.5, Figure A.6 shows the validation and training accuracy for the best model obtained in Section 7.3.2.

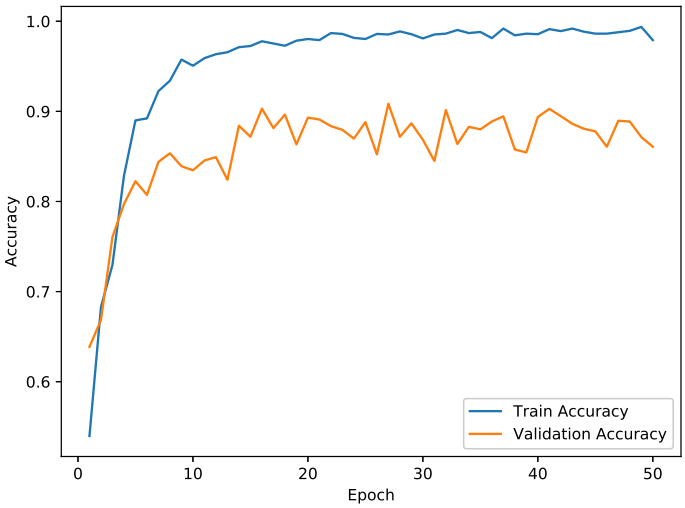


Figure A.5: The accuracy during training for the train set and the validation set. For the model obtained in Section 7.3.2.

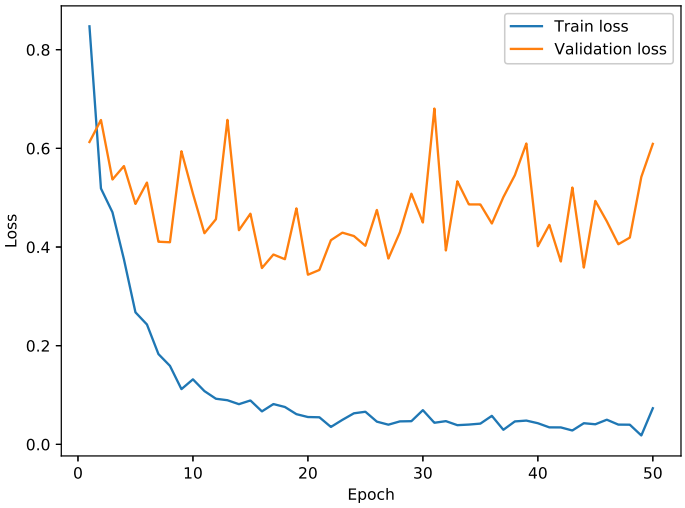


Figure A.6: The accuracy during training for the train set and the validation set for the model obtained in Section 7.3.2.