Application Note

Team: “Optimizers”

About this document

The current document is the application note for team: “Optimizers”

# Scope and purpose

The purpose of this document is to show our results of the given task and explain the detailed process of the project.

# Intended audience

The intended audience is everyone who participated or organized the final round of this event.

Table of contents

[Table of contents 1](#_Toc104482389)

[1 Abstract 2](#_Toc104482390)

[2 Hardware 3](#_Toc104482391)

[3 Preprocessing of Data 5](#_Toc104482393)

[4 Machine learning model 6](#_Toc104482394)

[5 Training 7](#_Toc104482395)

[6 Evaluation 8](#_Toc104482396)

[7 Complexity 9](#_Toc104482397)

[Revision history 10](#_Toc104482398)

# Abstract

Project Description: This project is focused on developing a live people counting solution using a short ranged FMCW Radar. Radar sensors offer a promising and effective sensing modality for human activity classification. Present radar-based activity recognition systems exploit micro-Doppler signature by generating Doppler spectrograms or video of range-Doppler images (RDIs). This data can then be passed through a deep neural network for classification. In this project, we propose a convolutional neural network architecture and analyze the problem of people counting.

# Hardware

## Project’s Components:

* XENSIV™ 60GHz radar sensor for advanced sensing by infineon.
* USB cable.
* Laptops.

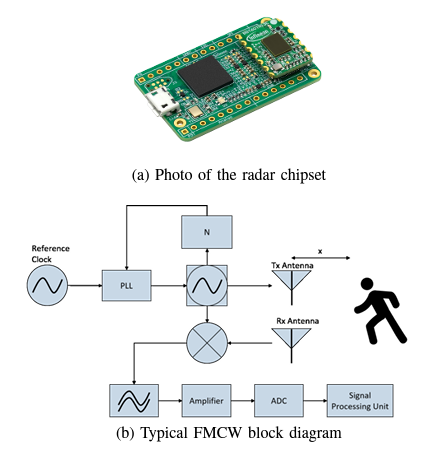


Fig. 1: (a) Infineon’s BGT60TR13C 60-GHz radar sensor. (b) Functional block diagram of FMCW radar RF signal chain depicting 1TX, 1RX channel

The work in this project is based on Infineon’s BGT60TR13C FMCW radar chipset. Its operating frequency ranges from 57GHz to 64GHz with an adjustable chirp duration. Its block diagram is shown in Fig. 1. The transmit path consists of a voltage controlled oscillator (VCO) that is regulated by a phase locked loop (PLL) to a reference frequency of fref = 80MHz. Highly linear frequency chirps between 57GHz and 64GHz are produced by adjusting the divider value and an additional tuning voltage ranging from 1V to 4.5V. In the receive path the echo returning from the target object is down-converted with a replica of the transmitted frequency chirp. Therewith the baseband frequency spectrum can be sampled by the 12bit analog-digital converter (ADC). Moreover the receive path contains an intermediate frequency buffer amplifier and an analog IF-filter that can be adjusted corresponding to the received frequency range. The radar chip is package in an embedded waver level ball grid array package including 4 integrated patch antennas realized by a metal redistribution layer. Three of them are receive antennas and one is the transmit antenna. Consequently, the radar sensor contains three identical receive paths and one transmit path. The radio frequency (RF) signal is distributed by an active RF distribution network to the receive paths. The transmitted up chirp from the FMCW radar’s ramp generator is reflected by a moving object and is received at the receiver after round trip delay caused by the target’s range from the radar and the velocity of the target. The received signal is mixed at the receiver with the transmitted signal and the resultant signal is low-pass filtered, thus performing the matched filtering operation.

# Preprocessing of Data

The Doppler information of the target is extracted by monitoring the change of target peak along slow-time, which is the inter-chirp time. One common approach is applying FFT along the fast time as well as slow-time dimension. The outcome of this operation is a two dimensional matrix representing the received power spectrum over range and velocity, also known as range - Doppler image (RDI). The received and deramped IF data is stored in matrices of size Nc×Ns, where Nc being the number of chirps considered in a frame and Ns is the number of transmit samples per chirp.

The radar chipset has three receiver antennas resulting in three matrixes of RDI data. In this project we experimented with many ways of handling the data.

Some of those methods include:

* Forming an extended matrix which consists of the concatenation of the three separate matrixes of the three receiving antennas.
* Calculating the mean value of each corresponding element of the already mentioned matrixes.

After our experimentation we decided on the following methodology.

We are calculating the RDIs and using high dimension matrixes which include all of the antennas information. Afterwards we calculate the difference in value between every four frames to extract information about the notion of movement.

After the preprocessing pipeline the data is then used as input for the training of a convolutional neural network.

# Machine learning model

We used a convolutional neural network as which are known for their adaptability and robustness against variability and noise.

We used the following network architecture.

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

max\_pooling2d (MaxPooling2D (None, 1, 32, 64) 0

)

conv1d (Conv1D) (None, 1, 30, 64) 12352

batch\_normalization (BatchN (None, 1, 30, 64) 256

ormalization)

conv1d\_1 (Conv1D) (None, 1, 30, 32) 6176

batch\_normalization\_1 (Batc (None, 1, 30, 32) 128

hNormalization)

conv1d\_2 (Conv1D) (None, 1, 30, 16) 1552

batch\_normalization\_2 (Batc (None, 1, 30, 16) 64

hNormalization)

flatten (Flatten) (None, 480) 0

dense (Dense) (None, 64) 30784

dense\_1 (Dense) (None, 4) 260

dense\_2 (Dense) (None, 4) 20

=================================================================

Total params: 51,592

Trainable params: 51,368

Non-trainable params: 224

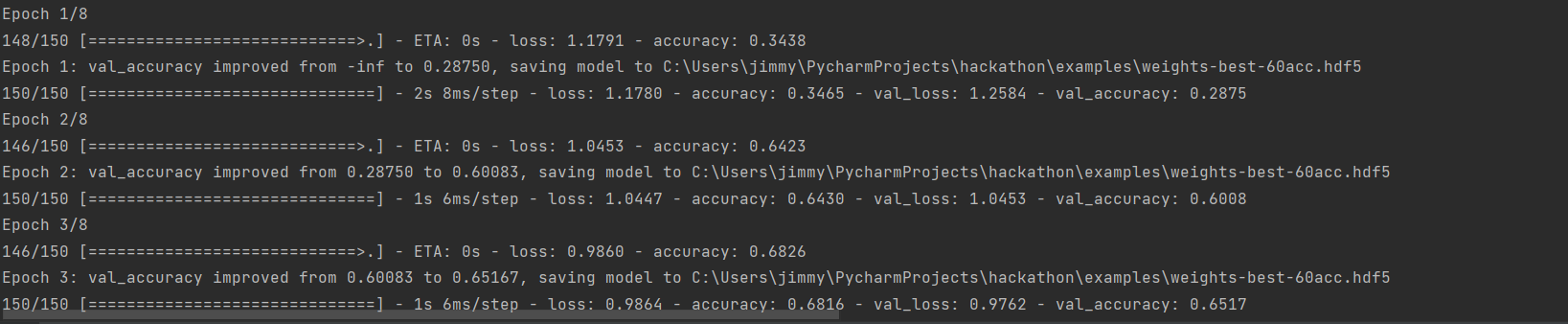
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

We are using a max pooling layer to drop the dimensionality of the data and then a number of 1D convolutional and dense layers.

# Training

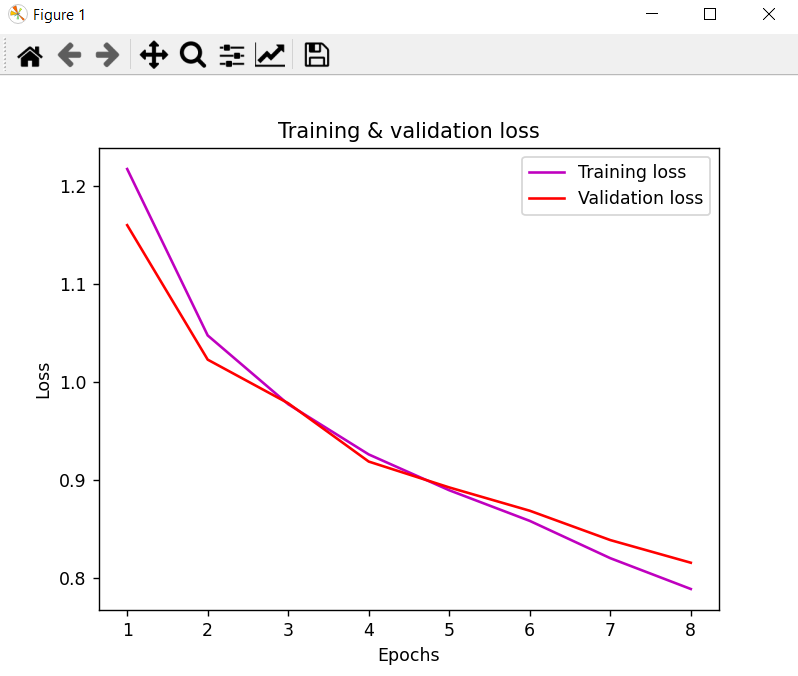
Before using the sensor data to train our network we fist split them up into two sets. The training and testing set. The training set includes the 80% of the whole data while the other one the rest 20%.

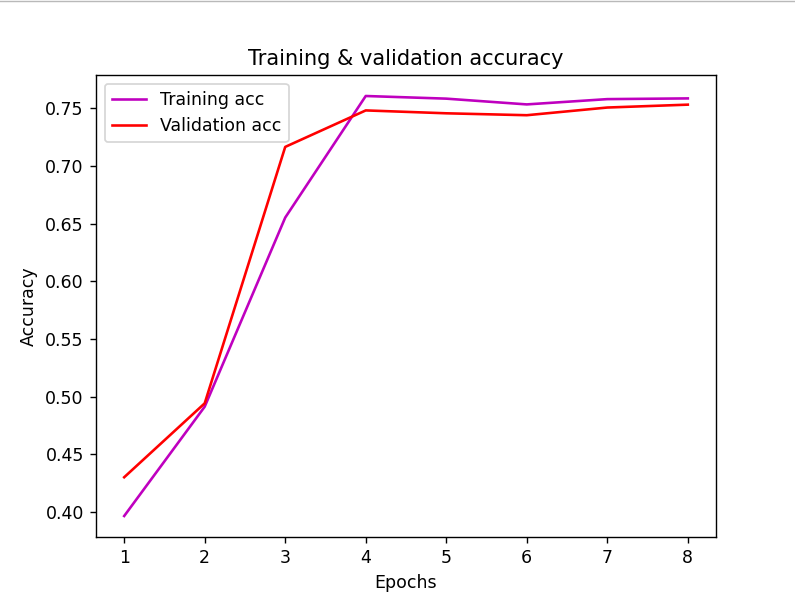
We trained the model for 8 epochs using the Adam optimizer with a set learning rate of 0.0003. In order to monitor the model we used callbacks. More specifically we used early stopping to stop the model earlier when improvement in validation score is not achieved and logs.



# Evaluation

We evaluated the model using the testing set and achieved an accuracy of around 75%.





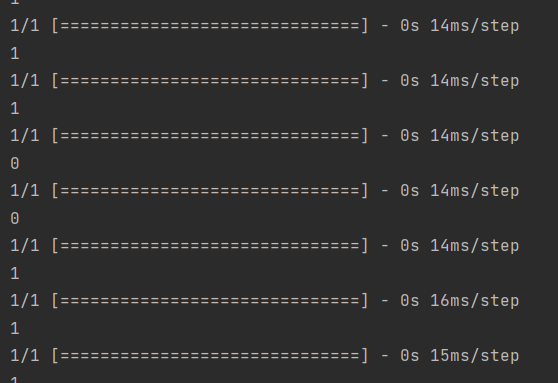
We also evaluated using real time data.

# Complexity

Compared to the training stage of the development the classification process takes significantly less time.

Bellow you can see the time spent per prediction during live time classification.

1 represents the amount of people in the space.



Revision history

| Document version | Date of release | Description of changes |
| --- | --- | --- |
| V.0.1 | 26/5 | First version |
|  |  |  |
|  |  |  |