WaveCounter

Passage detection and direction classification using CSI time-series

Henrik Ahlinder, Sebastian Andersson, Martin Dahl,
Christian Gustavsson, Joel Henneberg, Tiger Kylesten

Linköping University, Communication Systems

Introduction

This is a project in the course TSKS23, Signal Processing, Communication and Networking, CDIO, at the communication systems department at Linköping University. The goal of this project is to show if it is possible, using deep learning methods, to predict what directions a person is moving when a radio-based signal is broken by the person in question. We implemented this concept into a product, which purpose is to keep track of how many are residing in a room. This was brought up in conjunction with the COVID-19 pandemic were it became essential to keep track of the number of people in buildings.

Methodology

The system is setup in a doorway to classify the direction of the movement using machine learning algorithms and Software-Defined Radio (SDR). Two PLUTO SDR devices are used, where one acts as the transmitter (Tx) and the other as receiver (Rx). With this recordings could be made of the signals when a person passed by, see figure 1 to see how the environment looked. From the recorded data a CSI estimation is made. Each recording was 5 seconds with a frequency of 5500 Hz. This setup was used to collect the data to train the models used for the classification. The models task is to classify if the person went in or out from the room, thus resulting in this being a binary classification problem. All the members in the group had contributed to the collection of data, guaranteeing that some variety was in the data. In total around 4500 data points were collected were half of it was labeled as in and the rest as out. This was to ensure a balanced dataset. The training of the models were done on 80 % of the collected and the 20 % that was left was for testing the performance of the model. The metrics used for evaluating the model were the accuracy and the F1-score. A validation set was not used here since no hyperparameter tuning of the models was preformed, making a validation set unnecessary.

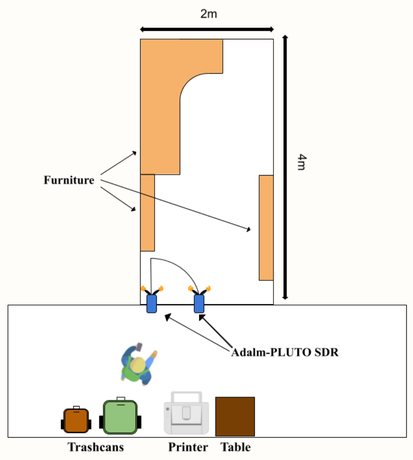

Figure 1: Sketch of the environment in which data collection took place.

CSI estimation

In the hardware, the CSI estimation was executed in the following way:

1. Transmitter sends $x[n] = 1$ for $n_0 \leq n \leq n_0 + 99$.
2. Every 100 samples are used to estimate $|h|$ (CSI absolute value), this gives an estimation frequency of $f_s/100 = 5500\text{Hz}$.
3. Explicitly, $|h|$ is estimated as in Equation 1 (note that $x[k] = 1$).

$$|\hat{h}|[n] = \frac{1}{100} \sum_{k=100n+1}^{100(n+1)} \left| \frac{y[k]}{x[k]} \right| \tag{1}$$

Pre-Processing

Before the CSI estimation can be feed into the models they need to be processed, see figure 2 for the raw CSI estimation. Several tools were are used for the preprocessing: Butterworth low-pass filter, Min-max normalization, Z-score normalization, downsampling and a shift-time discrete-wavelet transform (DWT). These tools are used in different combinations based on what format the input to the model has. In this case there are two types of inputs, 1 dimensional and 2 dimensional. The pipelines for them are the following:

1-dimensional input

1. Butterworth low-pass filter
2. Min-max normalization
3. Downsample

2-dimensional input

1. Butterworth low-pass filter
2. Min-max normalization
3. DWT
4. Z-score normalization

See figure 3 for 1-dimensional case and figure 4 for the 2-dimensional one.

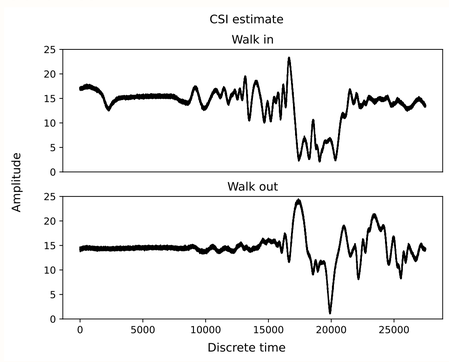

Figure 2: Output of hardware subsystem, 27500 estimates of the CSI.

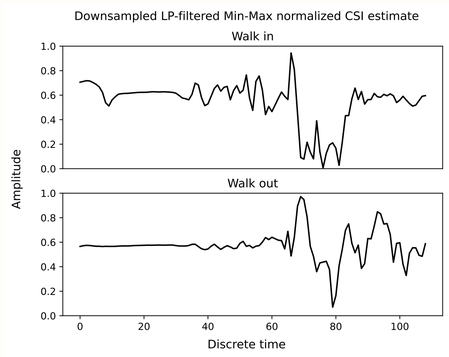

Figure 3: 1-dimensional input

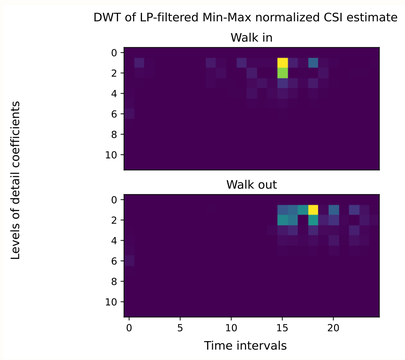

Figure 4: 2-dimensional input

Models

Several models were tested: fully connected neural network, LSTM neural network, SVM and Hidden Markov. These did not give satisfactory results. Two other models were tested instead, 1DCNN, 2DCNN. CNN stand for Convolutional Neural Network, and 1D is 1-dimensional and 2D is 2-dimensional. The reason for a convolutional network is to make the model time-invariant. The exact architecture is not covered here but can be found in the technical documentation for the project, see *Further Information*.

Result and discussions

In table 1 the performance of all the models are shown when they were trained on a subset that is 10 % of the original datatset.

Model name	Accuracy	F1 Score
1DCNN	80.4%	0.771
2DCNN	82.1 %	0.773
FCNN	58.0%	0.514
LSTM	60.7%	0.545
SVM	61.6%	0.538
HMM	65.2%	0.435

Table 1: Top performances when trained and tested on a smaller dataset with a 75/25% train/test split.

From table 1 it is apparent that 1DCNN and 2DCNN outperforms the other more standard methods. These two were therefore trained on the whole dataset and the performance can be seen in table 2.

Model name	Accuracy	F1 Score
1DCNN	91.1%	0.91
2DCNN	81.3%	0.81

Table 2: Performance when trained and tested on $\mathcal{D}$ with a 80/20% train/test split for 100 epochs.

In table 2 it is shown that 1DCNN is the one that got significantly better with a larger dataset. So for this particular use case of CSI sensing, the frequency content (which the DWT extracts) was not essential to get a satisfactory performance. A more "raw" version of the CSI estimate is sufficient. Although this result is promising, these models are very sensitive to changes in the environment. A slight change in the environment (environment in figure 1) changed the performance drastically, even for the best model, to less than 50 % accuracy, which is worse than guessing in this case. So the conclusion here is that it is possible to get a direction classification to work with CSI estimation, but for these models the environment needs to be static.

Further information

This project main goal was the direction classification, the passage detection was a extra task. If information on the passage detection, software, hardware, model architecture or more is sought after in for this project, visit `wavecounter.xyz` to see the technical documentation.