Determining the rate of small gestures unobtrusively

Koen Reefman
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
k.h.a.reefman@student.utwente.nl

Nikita Sharma
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
n.sharma@utwente.nl

ABSTRACT

The number of elderly people with Alzheimer's disease is projected to double in the next three decades, thus increasing the pressure on the available care system. The quality of healthcare could greatly be improved by using unobtrusive sensing to detect anomalies in regular behaviours, such as an agitated mood in elderly people. If a caregiver knows in advance that someone is agitated, the person could be approached in a different way. One of the more interesting unobtrusive sensing solutions is with the use of Channel State Information (CSI). This paper uses CSI data in order to first identify deviant behaviour and then estimate the rate at which a movement, in this case sitting and standing up, is performed. This paper will discuss signal denoising techniques as well as several machine learning techniques which are used to classify human movement and calculate the rate at which they occur. This paper finds that the best way to determine the rate of human movements is by first using a Savitzky-Golay filter, then feeding the data into a Support Vector Machine and finally count the number of peaks in the prediction graph to get the rate.

Keywords

Unobtrusive Sensing; Channel State Information; Human Activity Recognition; Machine Learning; Subtle Human Gestures

1. INTRODUCTION

The number of people with Alzheimer's disease is projected to more than double in the next 30 years, from 5.8 million today to around 13.8 million in 2050 (in the U.S.) [1]. This increase in the amount of people with Alzheimer's disease is mainly caused by the number of older people, which is also projected to more than double in the next three decades to a total of 1.5 billion worldwide. This is caused by increasing levels of life expectancy and a decreased level of fertility [7]. This massive increase in older people (especially with Alzheimer's disease), together with the current global pandemic which affects caregivers of these elderly people [11], leads to a massive increase in demand for caregivers for these elderly people.

To aid in the improved healthcare for elderly people, previous research in the field was conducted on the indication of an agitated mood by looking for human movements [3]. This study concluded that certain human movements such as throwing objects, kicking objects and repetitive motions (such as finger tapping) indicated a higher level of agitation in the involved person. These different types of human movement were later, in another study, combined into an agitation scale for elderly people with dementia [6]. This scale could be used by caregivers to assess someone before helping them and therefore approach them better and improve healthcare if they know in advance if a person is agitated. To ease the pressure on caregivers in this situation, the detection of agitation using small human gestures could be done using unobtrusive sensing.

Unobtrusive sensing is state of the art, especially in healthcare. Some implementations in healthcare range from using cameras, which are a potential breach of privacy for people involved, to using Channel State Information (CSI), which can detect humans using Wi-Fi signals [2]. CSI technology is relatively new and has primarily been used to detect larger movements such as a person falling down, walking or running. There are a few exceptions where CSI is used to detect smaller human movements such as detecting what a person is saying by analyzing the shape of their mouth [13].

This paper aims to take a small step towards implementing this remote sensing of human gestures in healthcare on a larger scale, to alleviate some of the work pressure for caregivers and improve the quality of healthcare for elderly people with Alzheimer's disease.

To achieve the aforementioned goal, the following research question will be answered: *How can the rate of small human gestures be determined by using unobtrusive sensing?* This main research question will be explored using the following sub-questions:

1. *How can actual human gestures and background noise be differentiated?*
2. *What is the best way to identify the rate of human gestures?*

The rest of the paper discusses related work and methodology before moving on to the results. The results section first explores the the first sub-question by using different methods to filter CSI data and comparing them. The second sub-question is researched by using different machine learning models on data for different receivers spread out across the test area. After the results section, the discussion section will discuss the results found in the results section and form answers to the sub-questions and research question. The paper is finished off by a conclusion with recommendations for further work.

2. RELATED WORK

Human movement detection using remote sensing has been researched extensively and have produced interesting results over the years. Most of the earlier solutions in this field were centered around using cameras to detect movement [4]. There were however several drawbacks to using a camera to detect movement, such as the need for a line of sight and privacy concerns for the users. To combat this issue, radar technology was used to remove the need for a line of sight and reduce privacy concerns. One example of this radar technology being used is a study where the radar was able to detect humans through a 10cm thick concrete wall [9].

A more recent invention in the field of remote sensing is using Wi-Fi signals to detect human movements using Channel State Information (CSI). This topic was shortly introduced in the introduction, most research using CSI is however focused on larger human movements. Most current research focuses on movements such as walking, running or falling down [2, 5, 15]. One example includes using CSI to determine the movement speeds of different body parts and relating them to a certain body movement [14]. Different parts of these research papers might be used in the healthcare for elderly people, for example to detect whether a person has fallen down or detect certain other body movements. CSI has also been researched on a smaller scale, one example of this is in a paper that details the detecting of mouth movements to "hear" what a person is saying by measuring the reflection of Wi-Fi signals [13]. Lastly, research has been done on the presence of multiple people in a room and the ability to detect gestures while there are more than one person in the room [12]. This research can also be implemented in an elderly home where multiple people are present in the same room.

3. METHODOLOGY

As mentioned before in the introduction, this paper aims to take a small step towards implementing remote sensing technology in healthcare to measure the level of agitation of elderly people with Alzheimer's disease. This paper focuses on processing and interpreting CSI data from two participants, using three different receivers, obtained in a realistic test environment. Each receiver has 9 channels with 30 subcarriers, which gives a large amount of data to process. The sender sends 100 packets per second, so that gives 12000 data points in a span of two minutes. This CSI data, together with video data for each participant, was gathered under a broader ongoing project ENTWINE by co-author Nikita Sharma, before the current global pandemic. Only a small part (two participants) of this data set was used for this preliminary research, the manuscript of making the full data publicly available is under process at the moment. Participants were asked to perform several activities on different locations for a certain amount of time, the location of the transmitter, receivers and locations where participants performed certain activities are shown on Figure 1. The blue circle "Tx1" is the transmitter, the green circles "Rx<>" are receivers and the red circles "L<>" are locations where participants performed activities. Only three receivers were used, those are Rx1, Rx2 and Rx3. The last receiver, Rx4, is not interesting for this research paper since it is located in another room.

To prepare the available data per receiver for use with machine learning algorithms, signal denoising methods are used. The first method that is used is a moving average filter, also known as a rolling mean, this method is implemented using the Pandas package in Python. The second method that is experimented with is a Savitzky-Golay Filter, which is implemented using the SciPy package in Python. These methods were chosen because of their popularity. To further prepare data for machine learning, the best channel was selected for each receiver to achieve the best possible signal. To obtain one signal, several options are explored such as taking the mean, the minimum and the maximum of the 30 subcarriers.

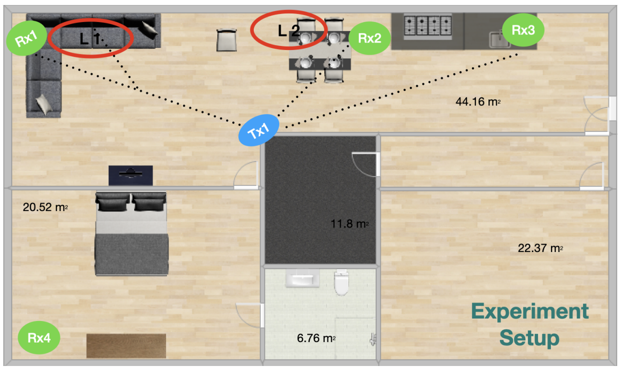

Figure 1. Experiment setup

After there is one CSI signal, the data is compared to video data gathered while participants were performing their activities. This video data is used to annotate the CSI data to indicate when a certain activity is performed. For example, peaks and valleys in a CSI signal, which correspond to video data, were annotated to indicate when a person was sitting or standing. This data was all annotated with the use of an online CSI annotation tool, found at https://trainset.geocene.com/.

After the data is prepared and annotated, three different machine learning algorithms are trained and compared to see which is the best at detecting human movements. The first algorithm, Long Short Term Memory (LSTM) is implemented in Python using the Keras and TensorFlow packages. The second one, Convolutional Neural Network (CNN) is also implemented using Keras and TensorFlow. The last algorithm that is looked at is a Support Vector Machine (SVM), which is implemented using the Scikit-learn package in Python.

At last, after the three machine learning algorithms have been trained and compared, the rate of a movement is calculated using the results from the machine learning algorithms.

4. RESULTS

4.1 Sub-Question 1

To answer the question "How can actual human gestures and background noise be differentiated?", some signal denoising and smoothing techniques are looked at. The main goal is to remove as much noise as possible from the signal and keep the peaks and valleys intact. However, before the data is smoothed, the 30 different subcarriers are combined into one signal by taking the mean value of each row, the minimum value of each row and by taking the maximum value of each row. Figure 2 shows the difference between these three. The minimum value has the largest amplitude of the three graphs, which makes it easier to detect when a person is performing an action. The minimum value graph, which is used in further experiments, does have some outliers that are 0, these are smoothed out in the next sections by different algorithms.

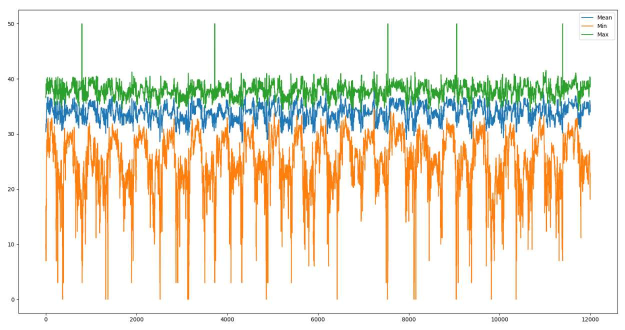

Figure 2. The mean (blue), min (orange) and max (green) values for all 30 subcarriers of one channel

4.1.1 Rolling Average Smoothing

The first smoothing method looked at is the Moving Average, also known as Rolling Average. This method works by taking a certain window size, summing all values in that window and then dividing it by the window size. This then results in a straight line from the first point in the window to the second point in the window. Several different window sizes between 50 and 200 are tested to see if rolling average is sufficient in denoising the signal while keeping the peaks and valleys intact.

Figure 3 shows the difference of one of the CSI signals between several window sizes. In this figure, each window size is compared with the previous window size (in increments of 50) on a total of 12000 data points. It is clear that a higher window size ensures a smoother signal with less random variations. Figure 3 a) shows the graph of window size 50, the general shape of the signal is intact; however, there are still too many fluctuations, especially at the peaks and in the valleys. Compare that to the result in Figure 3 d), where there are little to no random fluctuations left in the peaks and valleys of the signal while the general shape of the signal stays intact. The only problem that is left is that the width and height of the peaks and valleys are modified by a larger window size, Figure 4 shows the effects of a larger window size in more detail.

As mentioned before, the higher the window size for rolling average smoothing, the more detail is lost and the more condensed the peaks and valleys are. In Figure 4 it is very visible that window size 50 (blue line) has the width and height of the peaks very close to the original raw data, while window size 200 (red line) has a smoother line that can give a rough indication of when a person is performing a certain activity, yet lacks details about the width and height of the peaks, which means there is a lack of detail regarding the length of an activity. Comparing all four options, the graph with window size 150 looks most promising when it comes to preserving the shape of the graph as much as possible while also removing as much noise as possible. However, this is not the best solution to keep the shape of the signal as close as possible to the original signal, another option that was explored is the Savitzky-Golay Filter.

4.1.2 Savitzky-Golay Filter

The second method looked at is the Savitzky-Golay Filter [10], this method works similarly to the rolling average filter by taking a certain window size; however, for the Savitzky-Golay Filter one can also choose the polynomial degree. It works by taking the window size, a certain polynomial of degree n and consequently fitting that polynomial to the data points in the selected window. Several different window sizes and polynomial degrees were tested to see which one fit the original data the best and kept the width of the peaks intact, starting with a polynomial degree of 2 since polynomial degree 1 is the same as rolling average smoothing.

In all three cases with different polynomial degrees, the first two window sizes (101 and 201) are not sufficient, since there are still too many small details in the graphs. Take for example a look at the small peak at x ≈ 2400 in Figure 5, the blue line (window size 101) and the orange line (window size 201) still have a substantial peak there while the other two lines have smoothed them out more. In all cases, window size 401 is also insufficient, since it distorts the shape of the peak too much. Figures 5, 6 and 7 show at x ≈ 2700 that the peak for window size 401 is almost a triangle, which defeats the purpose of using a Savitzky-Golay Filter. Figure 8 plots different polynomial degrees for a window length of 301.

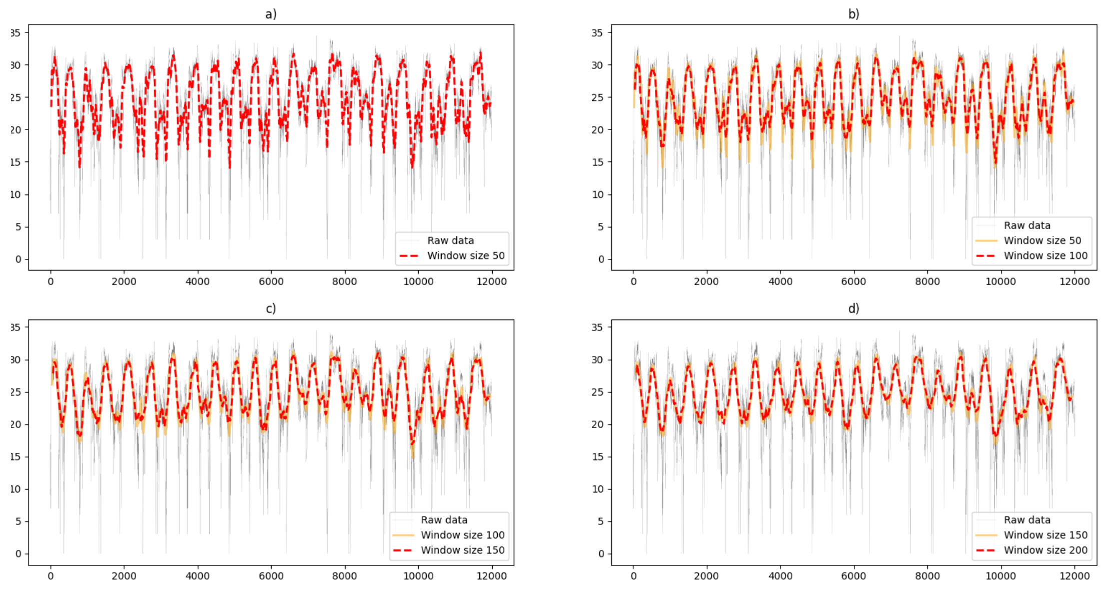

Figure 3. Different window sizes for rolling window denoising

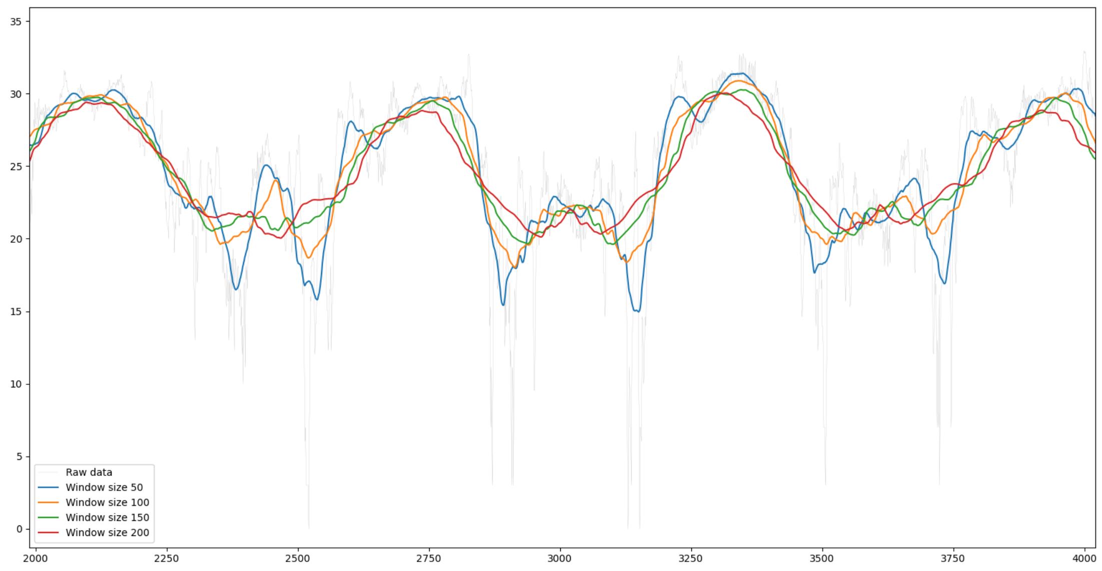

Figure 4. Different window sizes compared

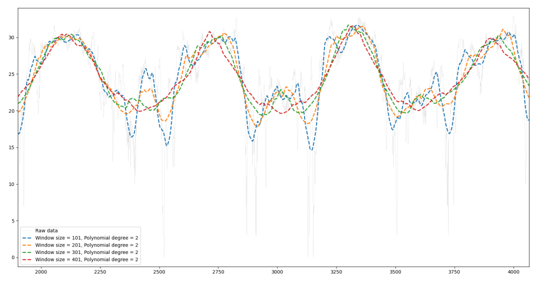

Figure 5. Comparison of Savitzky Golay filters with different window lengths, polynomial degree = 2

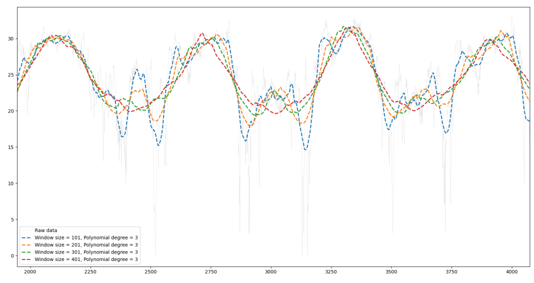

Figure 6. Comparison of Savitzky Golay filters with different window lengths, polynomial degree = 3

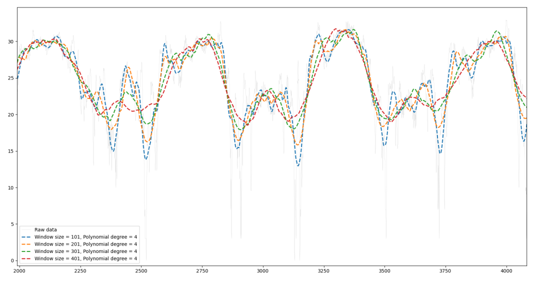

Figure 7. Comparison of Savitzky Golay filters with different window lengths, polynomial degree = 4

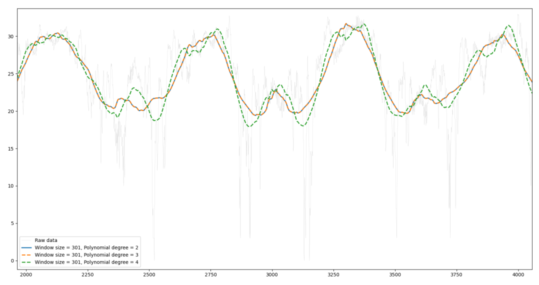

Figure 8. Comparison of Savitzky Golay filters with window length = 301, and different polynomial degrees

As can be seen in Figure 8, the graphs for polynomial degree 2 and 3 are the same and are quite good at smoothing the original data while retaining the shape of the original peaks. The same goes for polynomial degree 4, however there are still too many small peaks for the purpose of detecting movements consistently.

Figure 9 shows the best result from the rolling average method (window size 150) and the best result from the Savitzky-Golay Filter (window size 301, polynomial degree 3) and shows them side by side. From this figure it is clear that the Savitzky-Golay Filter is better at preserving the shape of the peaks while also filtering out most of the random noise from the original data. Therefore, this method is used to smooth all data.

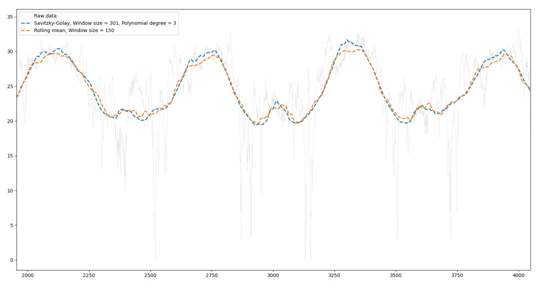

Figure 9. Savitzky Golay filter and rolling average comparison

4.2 Sub-question 2

Three different machine learning algorithms are used in order to classify whenever a person is sitting or standing, this information can then be used to calculate the rate of those human gestures. The three algorithms that are used are Long Short Term Memory (LSTM), Support Vector Machine (SVM) and a Convolutional Neural Network (CNN). These three algorithms will be compared to see which is the best at classifying human movements. For the purpose of clarity, all figures in this subsection will be on one specific signal from receiver 1, person 2. In the comparison section of this subsection, the performance of these algorithms will be shown on signals from different receivers and different persons.

4.2.1 LSTM

The first algorithm that is explored is the LSTM, which was implemented in python using TensorFlow. The model used for training has one input layer with an input dimension of TIME_STEPS, one LSTM layer with N units and one output layer which decides whether a person is sitting or standing. The TIME_STEPS and N variables are experimented with to find the best possible results. The available 12000 data points are split 60% test data, 20% validation data and 20% test data.

Both TIME_STEPS and N were experimented with, TIME-_STEPS influences the depth of the LSTM layer while N influences the width of the LSTM layer. To achieve the best possible results, while also having a reasonable computation time, both variables are experimented with separately.

Modifying TIME_STEPS.

To find out the best number for TIME_STEPS, N is set to 1000 to ensure the LSTM layer is wide enough for a good accuracy. Time steps were tested in increments of 5, starting at 5. They were each tested 5 times with an epoch of 10, since the accuracy already stabilizes at around epoch 5. Table 1 shows the results of this experiment, this table contains the average of 5 runs per time step. The individual runs for each time step can be found in Table 8 on appendix A.

Time steps	Overall Accuracy (%)	Loss (%)	Time (s)
5	93.914	0.051	21.864
10	95.142	0.037	29.21
15	95.246	0.039	35.086
20	94.604	0.45	44.12
25	95.45	0.036	48.834
30	95.514	0.037	55.094

Table 1. Impact of different time steps on a simple LSTM

The training time for the LSTM model appears to be linear with regards to the number of time steps, while the accuracy and loss of all models are relatively the same with a small increase of accuracy from 93.9% at 5 time steps to 95.5% at 30 time steps. The best looking time step is 15, since it has a very high accuracy of 95.2%, a low loss with 3.9% and a decent training time of 35.1 second for 12000 data points. If the need for a faster training time arises, one could go for 5 (or even less) time steps and still achieve an acceptable accuracy. For the purposes of comparing the LSTM to other machine learning methods, 15 time steps will be used since it has a good accuracy for an acceptable training time.

Modifying N.

To test the impact N has on the accuracy of the LSTM algorithm, the optimal number of TIME_STEPS which was discovered in the previous section is used in all cases. This experiment was also carried out with 10 epochs for each N, the values for N are 1 to 5 and 10. The results for individual runs for each value of N can be found in Table 9 on appendix A

As can be seen in Table 2, from N = 3 and onward the accuracy does not really change anymore. N = 2 did have some good results, while others were as poor as those in N = 1. In these first few N values, the training time is also not an issue, since it is a tiny increase from 13.29 seconds at N = 1 to 13.67 seconds at N = 10. With these results in mind, the chosen N value for the purposes of comparing this LSTM to other machine learning methods is N = 5. This N is chosen for having the smallest error margin for training time while having roughly the same accuracy, loss percentage and training time as other values, both higher as well as lower than 5.

N value	Overall Accuracy (%)	Loss	Time (s)
1	60.48	0.238	13.294
2	81.372	0.136	13.778
3	94.774	0.478	13.616
4	94.812	0.440	13.658
5	94.662	0.442	13.398
10	94.634	0.425	13.666

Table 2. Impact of different N size on a simple LSTM

4.2.2 CNN

The second algorithm that is explored is a Convolutional Neural Network (CNN). The model used in this section consists of one input layer, one hidden layer of size M and one output layer of size 2. The output layer of size 2 is because there are two final states, either a person is sitting down or a person is standing up. The size N of the hidden layer will also be experimented with, different values for the number of neurons in this layer will be tested.

As can be seen in Table 3, the average accuracy and training time of the CNN will stay roughly the same after N = 6 with the exception of N = 100000 which takes significantly longer to train than the previous values in the table (a 47.61% increase in training time compared to N = 10000). The values in the table are a result of running the CNN algorithm 5 times and taking the average over all 5 runs, the complete results can be found in Table 10 on appendix A. Table 3 also shows that the loss decreases the more neurons are in the hidden layer. Since the accuracy and training time stay roughly the same for N values under 100000, N = 10000 will be used for further experiments in this paper for the reason that it has the smallest standard deviation for accuracy and loss while also having the smallest loss value out of all different N values.

4.2.3 SVM

The last algorithm that is explored is the Support Vector Machine (SVM), which has advantages of fewer training parameters and higher stability in small-sample model training, thus being a better choice than a CNN [8]. For the SVM model, different kernels can be used. The most used kernels are ”linear”, ”poly” and ”rbf”, these will be tested to see which is the best at detecting when a person is sitting or standing. The ”poly” kernel can have different polynomial degrees, some different polynomial degrees are also tested. Table 4 show that, in terms of accuracy and loss, the different kernels give a similar result. The main difference is in the training time of the SVM model for different kernels, with the time being as low as 0.56

Size	Accuracy (%)	Loss	Time (s)
1	90.34	0.56454	6.158
2	82.488	0.54688	6.71
3	87.418	0.54508	6.236
4	92.672	0.50554	5.99
5	89.332	0.47638	5.784
6	94.93	0.47662	5.746
7	92.276	0.44632	5.74
8	92.036	0.43408	5.764
9	93.268	0.41962	5.606
100	94.008	0.18562	5.992
200	94.54	0.16042	6.058
1000	94.658	0.14976	5.946
10000	95.196	0.13996	5.638
20000	92.52	0.19136	5.462
100000	95.142	0.14574	8.322

Table 3. Impact of the number of neurons in the hidden layer of the CNN

seconds for "rbf" while going to a massive 130.99 seconds for the "poly" kernel with a polynomial degree of 4. For further experiments in this paper, the "rbf" kernel will be used since it has the lowest training time while having a good accuracy.

N value	Overall Accuracy (%)	Loss	Time (s)
linear	94.94	0.4366	0.65
poly, degree = 1	95.92	0.4368	0.79
poly, degree = 2	95.89	0.4371	0.59
poly, degree = 3	95.89	0.4371	11.72
poly, degree = 4	95.91	0.4369	130.99
rbf	95.87	0.4373	0.56

Table 4. Impact of different kernels on an SVM

4.2.4 Comparison

So far, the looked at machine learning algorithms have been trained on only one CSI signal from person 2, receiver 1. In this section, the three machine learning algorithms will be trained on all three receivers with data from person 1, person 2 and on data from both persons. This is done to see if certain traits of the CSI signal are present in different candidates. Tables 5, 6 and 7 show the results of comparing the three aforementioned machine learning algorithms on the signals of different receivers (0, 1 and 4) and different participants (P1 and P2).

For all three receivers, the SVM is the best performing machine learning algorithm with the highest overall accuracy for all receivers and all data, except for receiver 0 with data from both participants where it is in close second place with 81.13% behind the LSTM with 81.68%. The receiver with the overall highest accuracies is receiver 1 which is located next to the sofa (Rx1 on Figure 1). The accuracies for P2 were also significantly higher than the accuracies for P1 in all cases.

4.3 Calculating the rate of gestures

The rate of sitting down and standing up is counted from the available video data, person P1 stands up 18 times while P2 stands up a total of 21 times. Figures 10, 11 and 12 are all from receiver 1 and person 2, they show the smoothed Savitzky-Golay graph in blue, the actual annotations in orange and the machine learning algorithm's decision in red. As can be seen in the figures, all machine learning methods have some trouble with the peak at around 9000, this also causes all algorithms to count more times than the actual number of sitting down and standing up is. The rate of gestures is determined by counting the number of peaks for the prediction graph (red line) and subsequently dividing that number by 120, which gives the average rate over two minutes. Figure 10, the LSTM, counts 22 times standing up, Figure 11, the CNN, also counts 22 times, and Figure 12, the SVM, counts 23 times.

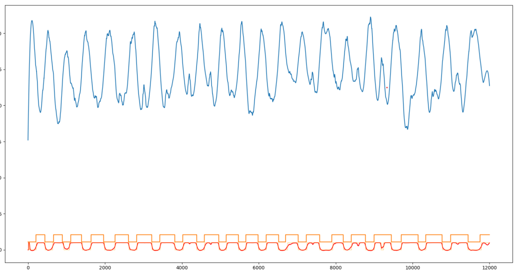

Figure 10. Receiver 1, P2 predictions LSTM

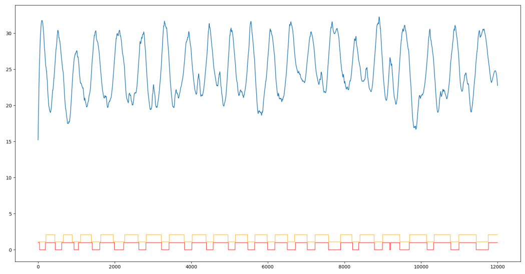

Figure 11. Receiver 1, P2 predictions CNN

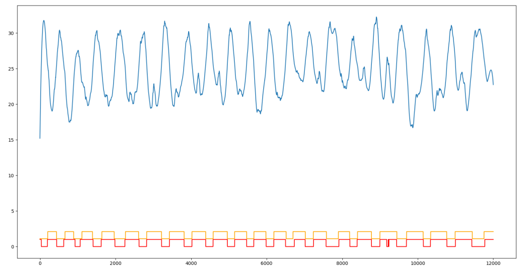

Figure 12. Receiver 1, P2 predictions SVM

4.4 Amplitudes for different receivers

The last findings looked at are the amplitudes of different receivers for the same person, Figure 13 shows those different graphs. The values of receiver 0 are between 22.3 and 29.4, for receiver 1 they are between 15.2 and 32.3, and for receiver 4 the values are between 11.8 and 26.2.

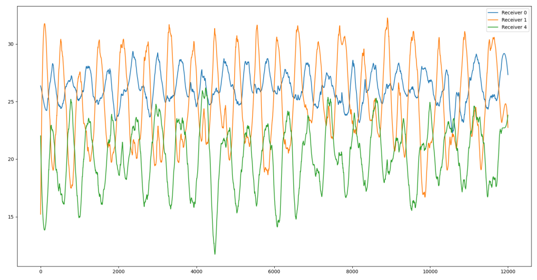

Figure 13. Amplitudes of different receivers for the same person

5. DISCUSSION

The results in the previous section indicate that receiver 1 is the best for detecting human movements, this can be seen in tables 5, 6 and 7. When combining these tables with regard to the overall accuracies, Table 6/ receiver 1

Loc1 R0	Train data (60%)	Vali data (20%)	Test data (20%)	Vali acc (%)	Overall acc (%)	Sitting acc (%)	Standing acc (%)
CNN	P1	P1	P1	73.04	72.22	78.33	69.33
	P2	P2	P2	89.33	89.13	84.69	97.92
	P1 + P2	P1 + P2	P1 + P2	62.73	62.57	60.24	67.58
LSTM	P1	P1	P1	76.65	72.42	76.67	70.18
	P2	P2	P2	83.4	89.85	92.09	87.04
	P1+P2	P1+P2	P1+P2	77.24	81.68	79.79	84.05
SVM	P1	P1	P1	73.92	72.87	76.01	71.09
	P2	P2	P2	90.25	89.85	87.64	93.48
	P1+P2	P1+P2	P1+P2	80.38	81.13	84.23	78.33

Table 5. Results for receiver 0

Loc1 R1	Train data (60%)	Vali data (20%)	Test data (20%)	Vali acc (%)	Overall acc (%)	Sitting acc (%)	Standing acc (%)
CNN	P1	P1	P1	72.17	71.59	88.98	61.59
	P2	P2	P2	94.54	94.61	92.57	98.32
	P1 + P2	P1 + P2	P1 + P2	69.52	69.1	73.64	62.66
LSTM	P1	P1	P1	74.63	75.43	88.03	66.16
	P2	P2	P2	93.63	94.86	96.31	92.7
	P1+P2	P1+P2	P1+P2	66.95	74.36	72.37	79.93
SVM	P1	P1	P1	75.08	75.95	85.55	67.72
	P2	P2	P2	96.21	95.92	95.87	95.98
	P1+P2	P1+P2	P1+P2	76.06	76.59	71.87	96.09

Table 6. Results for receiver 1

Loc1 R4	Train data (60%)	Vali data (20%)	Test data (20%)	Vali acc (%)	Overall acc (%)	Sitting acc (%)	Standing acc (%)
CNN	P1	P1	P1	74.12	73.17	71.84	74.98
	P2	P2	P2	94.83	94.33	95.79	92.99
	P1 + P2	P1 + P2	P1 + P2	60.92	59.85	65.66	56.94
LSTM	P1	P1	P1	82.35	74.38	74.82	73.89
	P2	P2	P2	88.47	93.48	96.65	90.81
	P1+P2	P1+P2	P1+P2	86.97	77.69	76.96	78.49
SVM	P1	P1	P1	74.42	74.62	76.89	71.33
	P2	P2	P2	95	95.05	93.68	96.46
	P1+P2	P1+P2	P1+P2	77.21	77.75	75.49	80.58

Table 7. Results for receiver 4

gives the highest overall scores. Further results that support the assumption that receiver 1 is the best for detecting human movements is found in Figure 13, which shows the different amplitudes for different receivers. In this figure it is shown that receiver 1 has the largest difference between the highest and lowest point of the graph. This can be explained by the participant being between the sender and receiver 1, as is seen on Figure 1. This causes receiver 1 to have the highest value, while simultaneously having the largest decreases in value. This is caused by the participant performing movements which block the direct signal between the sender and receiver.

However, this does not mean that a person has to be in between the sender and receiver. Figure 1 shows that other receivers are also able to pick up human movements, this is further indicated by tables 5 and 7 which have a slightly lower accuracy overall than 6. However, these accuracies are still sufficient enough to detect human movements.

Lastly, P2 has overall a much higher accuracy than either P1 or P1 and P2 combined. This can be explained by the two persons having a different body type/size, having different postures while doing the same movement and performing those movements at a different rate, thus the signal being fundamentally different.

6. CONCLUSION

The difference between background noise and actual gestures can be determined by looking at the available CSI data from 30 different subcarriers for one receiver. The minimum value of those subcarriers is then taken to obtain a graph with clear features, which is then smoothed using a Savitzky-Golay Filter to remove the noise.

Several different machine learning algorithms were used to identify the rate of human gestures, which all gave roughly the same answer. The accuracy of these machine learning algorithms were compared and it was determined that a Support Vector Machine was the best at predicting when a person was performing an activity in this case.

With the answers to the sub-questions, the research question "How can the rate of small human gestures be determined by using unobtrusive sensing?" can be answered. The rate of small human gestures can be determined by taking available CSI data, using a Savitzky-Golay Filter to remove noise from the data, feeding the filtered data into a Support Vector Machine and finally count how often the SVM predicts a certain movement.

Although this paper looked at several different ways to determine the rate of a movement, from using different smoothing techniques to different machine learning algorithms, there is still room for improvement in the future. One thing that comes to mind to improve this research is having more participants do the same tasks, since there were only two participants mentioned in this paper. This would increase the accuracy at which machine learning algorithms detect certain movements since there is more data to learn from. Secondly, other machine learning algorithms could be explored to see if accuracy can further be improved. Lastly, a product could be developed that detects the rate of human gestures in real time, with which caregivers could determine a patient's agitation level.

7. REFERENCES

[1] 2020 alzheimer's disease facts and figures. *Alzheimer's & Dementia*, 16(3):391–460, 2020.

[2] T. Z. Chowdhury. *Using Wi-Fi channel state information (CSI) for human activity recognition and fall detection*. PhD thesis, University of British Columbia, 2018.

[3] J. Cohen-mansfield, M. S. Marx, and A. S. Rosenthal. A Description of Agitation in a Nursing Home. *Journal of Gerontology*, 44(3):M77–M84, 05 1989.

[4] J. W. Davis and A. F. Bobick. Representation and recognition of human movement using temporal templates. pages 928–934, 1997. cited By 441.

[5] L. Guo, L. Wang, C. Lin, J. Liu, B. Lu, J. Fang, Z. Liu, Z. Shan, J. Yang, and S. Guo. Wiar: A public dataset for wifi-based activity recognition. *IEEE Access*, 7:154935–154945, 2019.

[6] A. Hurley, L. Volicer, L. Camberg, J. Ashley, P. Woods, G. Odenheimer, W. Ooi, K. McIntyre, and E. Mahoney. Measurement of observed agitation in patients with dementia of the alzheimer type. 5:117–132, 01 1999.

[7] U. N. D. of Economic and P. D. Social Affairs. World population ageing 2020 highlights: Living arrangements of older persons. 2020.

[8] E. Raczko and B. Zagajewski. Comparison of support vector machine, random forest and neural network classifiers for tree species classification on airborne hyperspectral apex images. *European Journal of Remote Sensing*, 50(1):144–154, 2017.

[9] T. S. Ralston, G. L. Charvat, and J. E. Peabody. Real-time through-wall imaging using an ultrawideband multiple-input multiple-output (mimo) phased array radar system. In *2010 IEEE International Symposium on Phased Array Systems and Technology*, pages 551–558, 2010.

[10] A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

[11] K. Sheth, K. Lorig, A. Stewart, J. F. Parodi, and P. L. Ritter. Effects of covid-19 on informal caregivers and the development and validation of a scale in english and spanish to measure the impact of covid-19 on caregivers. *Journal of Applied Gerontology*, 40(3):235–243, 2021. PMID: 33143545.

[12] R. H. Venkatnarayan, S. Mahmood, and M. Shahzad. Wifi based multi-user gesture recognition. *IEEE Transactions on Mobile Computing*, 20(3):1242–1256, 2021.

[13] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. Ni. We can hear you with wi-fi! *IEEE Transactions on Mobile Computing*, 15(11):2907–2920, 2016. cited By 114.

[14] W. Wang, A. Liu, M. Shahzad, K. Ling, and S. Lu. Understanding and modeling of wifi signal based human activity recognition. volume 2015-September, pages 65–76, 2015. cited By 453.

[15] S. Yousefi, H. Narui, S. Dayal, S. Ermon, and S. Valaee. A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine*, 55(10):98–104, 2017. cited By 91.

Appendix A

Time step 5				Time step 20			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	94.11	5.03	22.51	1	95.91	3.78	44.78
2	95.01	4.7	21.62	2	95.86	3.46	42.55
3	95.2	4.11	21.81	3	96.03	3.55	45.06
4	92.28	5.7	21.59	4	95.09	4.02	44.45
5	92.97	6.02	21.79	5	90.13	7.51	43.76
Average	93.914	5.112	21.864	Average	94.604	4.464	44.12
Time step 10				Time step 25			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	95.55	3.8	29.7	1	94.56	4.14	51.07
2	95.05	3.72	28.53	2	96	3.29	49.05
3	93.97	4.45	29.97	3	96.02	3.45	48.03
4	95.53	3.46	28.77	4	95.01	3.93	47.76
5	95.61	3.38	29.08	5	95.66	3.38	48.26
Average	95.142	3.762	29.21	Average	95.45	3.638	48.834
Time step 15				Time step 30			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	95.63	3.51	36.15	1	94.51	4.26	54.27
2	94.87	4.9	34.55	2	95.42	3.65	53.08
3	95.23	3.81	34.62	3	96.02	3.5	55.61
4	95.44	3.35	34.89	4	96.01	3.43	55.38
5	95.06	3.97	35.22	5	95.61	3.44	57.13
Average	95.246	3.908	35.086	Average	95.514	3.656	55.094

Table 8. Modifying time steps full results

N = 1				N = 4			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	60.48	24.05	14.45	1	95.46	3.84	14.08
2	60.48	23.93	13.06	2	94.47	4.15	13.31
3	60.48	23.09	12.86	3	94.66	4.26	13.08
4	60.48	24.07	12.72	4	95.19	5.13	12.91
5	60.48	23.93	13.38	5	94.28	4.62	14.91
Average	60.48	23.814	13.294	Average	94.812	4.4	13.658
N = 2				N = 5			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	60.48	23.94	14.57	1	94.9	4.46	14.14
2	94.24	5.55	12.94	2	94.94	4.29	13.21
3	60.48	21.27	13.02	3	94.27	4.48	13.47
4	95.82	6.33	13.75	4	94.81	3.96	12.75
5	95.84	10.7	14.61	5	94.39	4.91	13.42
Average	81.372	13.558	13.778	Average	94.662	4.42	13.398
N = 3				N = 10			
Run #	Acc (%)	Loss (%)	Time (s)	Run #	Acc (%)	Loss (%)	Time (s)
1	93.83	6.05	14.03	1	94.63	4.31	14.16
2	94.88	5.12	12.6	2	93.51	4.82	13.15
3	95.02	4.16	13.95	3	95.27	3.97	14.16
4	95	4.2	13.96	4	94.78	4.07	13.29
5	95.14	4.39	13.54	5	94.98	4.1	13.57
Average	94.774	4.784	13.616	Average	94.634	4.254	13.666

Table 9. Modifying N full results

Size 1				Size 2				Size 3			
Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)
1	90.66	0.6007	6.51	1	95.72	0.5413	6.15	1	91.66	0.5054	6.24
2	93.62	0.5986	6.17	2	82.32	0.53	7.21	2	93.83	0.5263	6.13
3	90.15	0.5579	5.96	3	94.17	0.4287	7.02	3	95.32	0.5245	5.61
4	89.59	0.5782	6.02	4	79.83	0.5569	7	4	95.88	0.5704	6.37
5	87.68	0.4873	6.13	5	60.4	0.6775	6.17	5	60.4	0.5988	6.83
Average	90.34	0.56454	6.158	Average	82.488	0.54688	6.71	Average	87.418	0.54508	6.236
Size 4				Size 5				Size 6			
Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)
1	94.1	0.5581	5.77	1	88.82	0.4758	6.04	1	92.61	0.5053	6.12
2	86.45	0.5552	5.64	2	95.92	0.4926	5.71	2	95.93	0.4242	6.02
3	95.43	0.4244	7.28	3	92.31	0.4761	5.14	3	95.92	0.4837	5.44
4	91.83	0.4699	5.71	4	73.98	0.5151	5.43	4	95.9	0.5095	5.69
5	95.55	0.5201	5.55	5	95.63	0.4223	6.6	5	94.29	0.4604	5.46
Average	92.672	0.50554	5.99	Average	89.332	0.47638	5.784	Average	94.93	0.47662	5.746
Size 7				Size 8				Size 9			
Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)
1	95.28	0.4604	6.04	1	91.89	0.4304	5.79	1	93.67	0.5052	6.21
2	90.91	0.4592	5.53	2	91.76	0.4528	5.62	2	95.37	0.3661	5.45
3	85.29	0.494	5.59	3	90.53	0.4057	5.59	3	93.45	0.3982	5.43
4	94.07	0.396	5.94	4	92.58	0.4973	5.78	4	89.42	0.4061	5.46
5	95.83	0.422	5.6	5	93.42	0.3842	6.04	5	94.43	0.4225	5.48
Average	92.276	0.44632	5.74	Average	92.036	0.43408	5.764	Average	93.268	0.41962	5.606
Size 100				Size 200				Size 1000			
Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)
1	95.92	0.1756	5.74	1	95.23	0.1472	6.28	1	94.67	0.1468	5.8
2	95.91	0.1699	5.51	2	93.99	0.1661	5.55	2	95.28	0.1373	6.5
3	94.54	0.1674	5.39	3	94.32	0.1621	6.16	3	92.33	0.193	5.69
4	90.79	0.2166	6.76	4	95.87	0.1453	6.68	4	95.48	0.1349	6.26
5	92.88	0.1986	6.56	5	93.29	0.1814	5.62	5	95.53	0.1368	5.48
Average	94.008	0.18562	5.992	Average	94.54	0.16042	6.058	Average	94.658	0.14976	5.946
Size 10000				Size 20000				Size 100000			
Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)	Run	Acc (%)	Loss	Time (s)
1	95.39	0.1356	6.25	1	95.92	0.1296	5.83	1	94.36	0.1528	9.2
2	95.25	0.1377	5.66	2	93.4	0.1682	5.5	2	95.65	0.1329	8
3	95.67	0.1321	5.47	3	94.63	0.1479	5.36	3	95.55	0.1552	8.19
4	94.05	0.1601	5.34	4	82.93	0.3796	5.28	4	94.44	0.1512	8.03
5	95.62	0.1343	5.47	5	95.72	0.1315	5.34	5	95.71	0.1366	8.19
Average	95.196	0.13996	5.638	Average	92.52	0.19136	5.462	Average	95.142	0.14574	8.322

Table 10. CNN full results