Human Fall Detection in Indoor Environments Using Channel State Information of Wi-Fi Signals

Sankalp Dayal, Hirokazu Narui, Paraskevas Deligiannis
{sankalpd, hirokaz2, pdelig}@stanford.edu

Abstract

The aim of this project is to detect the fall of an individual in indoor environments by monitoring Wi-Fi channel state information (CSI) including amplitude and phase and using this data to recognize the person's activity. In an indoor setting, where Wi-Fi access points are present, human movements change the propagation characteristics of the environment, thereby altering the effective channel and the CSI of the received signal. Under some conditions, we can recognize this change in CSI characteristics as a signature of the activity performed and thus, identify it. Our experimental setup consists of two Wi-Fi access points between which a subject performs various activities. We use an augmented PCA technique to extract CSI features and apply both traditional supervised (SVM, Decision Trees, Multinomial Logit Regression) and deep (Long Short-Term Memory) learning methods to our data. LSTM achieves the best accuracy in our validation dataset, approximately 80%.

Introduction & Related Work

Many individuals, especially elderly people, live alone despite facing, potentially undiagnosed, health problems. In such circumstances, a fall can indicate an immediate threat to their wellbeing and even, survival. Therefore, detecting falls as early and accurately as possible is critical to treating them in a timely and effective manner.

Several, currently implemented, systems rely on the individual pressing an emergency button or alternatively, on designated professionals checking on the individual at regular intervals. However, when an individual falls, they may already be unconscious, in which case the former fails. Additionally, in the case of a heart attack or severe stroke, the individual needs to be treated within a few minutes, which means that the latter method is either unsuccessful or highly impractical.

Interesting proposed solutions involve using wearable sensors measuring acceleration ([5]), radars ([6]) or cameras ([7]). Nevertheless, elderly people often complain about having to carry a sensor around, reasonably priced radars may have very limited range of operation on the order of decimeters ([6]) and cameras require line of sight and sufficient lighting to operate effectively, while also raising privacy concerns.

This is where fall detection based on Wi-Fi comes into play. It is unobtrusive, respects the individual's privacy, works any time of the day and only requires equipment regularly found in homes today, namely Wi-Fi access points (APs). Human bodies reflect and scatter Wi-Fi signals and thus, movements affect the channel between the transmitter and the receiver leaving distinguishing signatures in the CSI data. Hence, activity recognition and fall detection are possible.

Few papers have attempted to perform activity recognition using Wi-Fi CSI data. First, [8] recognizes spoken words based on the CSI variation caused by lip movement, but its main shortcoming is that it relies on directional antennas to lower the signal noise, whereas most commercially available APs have omni-directional ones. Then, [9] uses CSI histograms to identify activities, such as taking a shower or washing dishes. However, the activities recognized are location dependent, whereas falls are not and thus, the methods applied do not carry over to our task. Moreover, utilizing only the CSI histogram removes a lot of information from the dataset.

Moving closer to our goal, [1] performs relevant activity recognition using a model-based approach, in which CSI data are correlated with movement speeds and speeds are then correlated with activities. Hidden Markov Models are used to compute the parameters of these models and the accuracy achieved is exceptional, greater than 95%, but we are interested in model-free methods. The closest approach to ours is described in [2] and specializes in fall detection using Wi-Fi CSI data of a 3x3 MIMO channel. Anomaly detection, first, detects the start of an activity and an SVM recognizes falls among those activities achieving an accuracy greater than 85%. We try to investigate how different learning algorithms perform at this task without utilizing anomaly detection and by

reducing the channel to 1x3 SIMO, whereby our input data are effectively reduced by two thirds.

In our experiments, an AP with one antenna uses 30 OFDM bands to continuously transmit packets to a receiving AP having three antennas (1x3 SIMO). Our raw input data consist of the complex-valued CSI of the received signal for each of these packets at each of the frequency bands. Therefore, in total, we have 90 complex-valued streams (timeseries) of CSI data, which we break up in overlapping windows of a 1-sec duration. We will expand on the collection and handling of data in subsequent sections.

We use multi-class classification algorithms whose output for each of these windows is an activity in the set {Lying Down, Falling, Picking Up, Running, Sitting Down, Standing up, Walking, No Activity}. By grouping together all other activities except for Falling, we can achieve a Fall / No Fall dichotomy. In particular, we use both traditional supervised learning algorithms: SVMs with a Gaussian kernel, Decision Trees (DTs) and Multinomial Logit Regression (MLR), as well as the deep learning Long Short-Term Memory (LSTM) method.

Experimental Setup, Dataset & Features

Our equipment consisted of two Intel Wi-Fi Wireless Link 5300 802.11n MIMO APs, one with a single antenna functioning as the transmitter and one with three antennas functioning as the receiver, with an antenna spacing of 2.6 cm. They were placed at approx. 5 meters apart (Fig. 1) and the subjects performed their activities in the area between them (not necessarily in the line of sight).

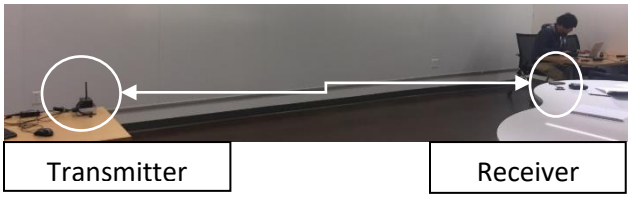

Fig. 1. Experimental Setup

The Wi-Fi transmitter transmitted packets at 30 different subcarrier frequencies (OFDM), each spaced 312.5 kHz apart with the center frequency at 5 GHz and at a rate of 50 packets per second. This gives rise to 90 complex-valued CSI streams, 1 for every combination of antenna pair and OFDM band. Data were collected for two subjects and multiple activities. We used camera recordings to identify the start and stop times of the activities performed and to annotate the timeseries accordingly.

Table 1 summarizes the duration of each activity in our dataset, whereas Fig. 2 shows the raw amplitude data that we received for each antenna pair while a subject was walking.

Activity	Duration	Activity	Duration
No Activity	84 sec	Sitting Down	55 sec
Walking	155 sec	Standing Up	43 sec
Running	256 sec	Lying Down	61 sec
Picking Up	55 sec	Falling	42 sec

Table 1. Duration of activities in dataset

We, now, move on to data preprocessing and feature extraction. Since CSI data are inherently very noisy, applying traditional filters such a low-pass Butterworth or a median filter is inadequate. Therefore, we use a special PCA-based technique proposed in [1]: First, we divide the data for each activity into 1-second intervals and calculate the corresponding PCA coefficients and eigenvectors. Then, a signal is reconstructed using the first six eigenvectors except for the first one (eigenvectors 2 through 5), since the first eigenvector corresponds to the line of sight component and is hardly helpful for activity detection. This reconstructed signal has reduced noise and can be, thus used for feature extraction.

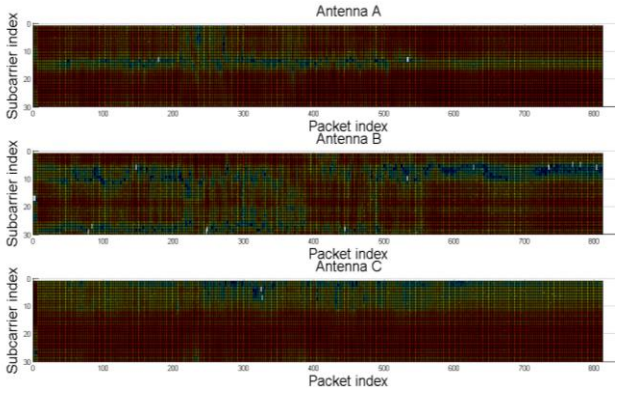

Fig. 2. Amplitude heatmap of received signal for each packet (packet index on the horizontal axis), for every receiver antenna (A, B, C) and for every subcarrier (subcarrier index on the vertical axis).

Fig. 3 shows the raw timeseries amplitude data for the first OFDM band and compares the result of applying a Low-Pass Butterworth filter with the resulting second PCA component using the method we followed.

To extract features we calculate the 50-point Short-Time Fourier Transform (STFT) of these filtered data over 1-sec windows with an overlap of 0.5 sec. Fig. 4 shows the spectrogram for 20 seconds of data while a subject was lying on the bed. This STFT gives the first 26 entries of our feature vector. In addition, we calculate the change in these coefficients with respect to the last set (indicating changes in the spectrum) and this gives an additional six entries for a feature vector of size 32.

Methods

We will very briefly describe the supervised learning algorithms used, namely SVMs, DTs and MLR, and then, offer a slightly more thorough explanation of how LSTM works.

We used Support Vector Machines (SVMs) with a Gaussian kernel. Since SVMs are naturally binary classifiers, we created multiple classifiers on a one vs. one basis following the Error – Correcting Output Codes (ECOC) multiclass model. This gave 21 classifiers, as we have seven classes. We used hinge loss as the binary loss function.

To apply multinomial logistic regression (MLR) to our multi-class problem we used the logistic function to generate six binary classifiers, which calculated the probability that a sample belonged in each of these classes. Of course, the probability for the last class is calculated as 1 minus the sum of probability for the other classes.

Decision Trees (DTs) were chosen because the do not assume any inherent relationship between the features. The cost function used for the splits in the construction of the binary decision tree is based on information entropy. In our model, we got 385 nodes for the decision tree.

The iterative algorithm for the construction process is loosely defined below:

1) Information Gain = Entropy (parent) - Weighted Sum of Entropy (Children)
2) Split so that you get the highest information gain.
3) Repeat 1, 2 until the information gain is zero.

LSTM (Long Short-Term Memory) is a relatively new type of a recurrent neural network (RNN), whose generic structure can be seen in Fig. 3. It has the ability to forget or remember values and given the right weights, it has the same computational power as any conventional computer. LSTM networks are particularly well suited at handling timeseries data. A simple basic building block of this network can be seen in Fig. 3, as well.

In our experiment, similar to how we would handle a speech recognition problem, we directly use the 1-sec intervals of the CSI timeseries data as an input to LSTM, without any preprocessing. As for the structure of the NN, we use 90 input units, 200 hidden units (1 hidden layer), and one of our seven classes as label for each of the inputs. To prevent local minima, we use the SGD (Stochastic Gradient Descent) with batch size 500 and learning rate 0.001 based on similar published work and some experimentation of our own. These values appear to work well for the small size of our dataset. Finally, we separate our original dataset into three subsets with respective durations: training set (1606 sec), validation set (544 sec) and test set (544 sec) and use them to train and evaluate the performance of this algorithm.

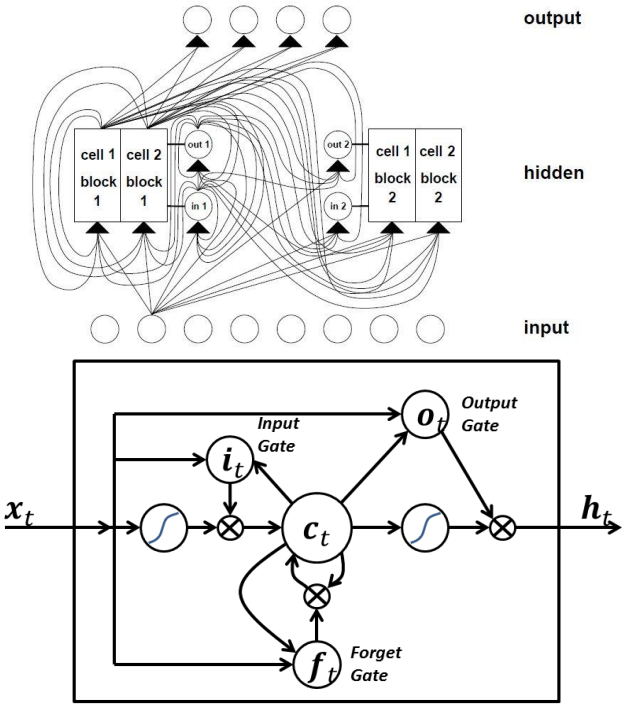

Fig. 3. Structure of a LSTM-based RNN and its building block. Sources: [4], [10].

Supervised Learning Results & Discussion

We, primarily, chose the supervised learning techniques as a simple benchmark to evaluate the performance of the state-of-the-art LSTM. Additionally, SVMs are powerful algorithms on their own and the closest previous approach to our

Actual Activity	Detected As							Accuracy (%)
	Bed	Fall	Pick	Run	Sit	Stand	Walk	
SVM								
Bed	**34**	0	9	40	7	4	28	**27.87**
Fall	2	**0**	3	42	2	1	34	**0.00**
Pick Up	5	0	**50**	31	7	4	14	**45.05**
Run	4	0	13	**206**	6	5	78	**66.03**
Sit Down	2	0	11	36	**30**	3	18	**30.00**
Stand Up	2	0	6	37	5	**15**	21	**17.44**
Walk	6	0	5	115	3	1	**200**	**60.61**
Preci-sion (%)	**62**	**NaN**	**52**	**40**	**50**	**45**	**51**	
MLR								
Bed	31	0	9	37	8	7	30	**25.41**
Fall	3	8	4	36	1	3	29	**9.52**
Pick Up	11	0	40	28	12	3	17	**36.04**
Run	10	5	11	171	10	6	99	**54.81**
Sit Down	9	1	9	36	25	2	18	**25.00**
Stand Up	3	1	5	31	6	17	23	**19.77**
Walk	8	3	14	95	6	6	198	**60.00**
Preci-sion (%)	**41**	**44**	**43**	**39**	**37**	**39**	**48**	
DTs								
Bed	**86**	5	7	10	2	6	6	**70.49**
Fall	10	**49**	4	11	1	4	5	**58.33**
Pick Up	5	2	**88**	5	4	2	5	**79.28**
Run	7	10	9	**275**	1	3	7	**88.14**
Sit Down	10	3	8	11	**57**	5	6	**57.00**
Stand Up	8	2	5	7	7	**51**	6	**59.30**
Walk	12	5	9	10	2	6	**286**	**86.67**
Preci-sion (%)	**62**	**64**	**68**	**84**	**77**	**66**	**89**	

Fig. 4. Confusion matrices, accuracy (recall) and precision for all activities for the three supervised techniques: SVM (top), MLR (middle) and DTs (bottom).

own utilized them to achieve remarkable accuracy ([2]). What is more, in choosing these techniques we tried to test different assumptions on the distribution of the feature vector. In particular, the SVM with Gaussian kernel assumes a Gaussian distribution of the feature vector, MLR assumes a linear relationship, whereas DTs only assume that the features are uncorrelated.

We evaluated the performance of the algorithms using 10-fold cross validation. Since this is a multi-class classification problem, we used confusion matrices with corresponding accuracy (recall) and precision metrics to visualize the performance of the algorithms. The results for each of the three algorithms are shown in Fig. 4 and a comparison based on accuracy follows in Fig. 5.

Activity	SVM	DTs	MLR
Lying Down	27%	70%	25%
Falling	0%	58%	9%
Picking Up	45%	79%	36%
Running	**66%**	**88%**	**54%**
Sitting Down	30%	57%	25%
Standing Up	17%	59%	19%
Walking	**60%**	**87%**	**60%**

Fig. 5. Comparative table of the accuracy of supervised methods.

Except for the generally low accuracy, we make two main observations. First, the Decision Trees clearly outperform the other two techniques and second, the more repetitive an action is and the more stable the body posture is for its duration, the more accurate its prediction is across all algorithms.

The first observation is somewhat counterintuitive and unexpected. Unless it is down to a peculiarity of our small dataset, it could indicate that the assumptions made by SVM and MLR about the prior distribution of the feature vector are poor, whereas the features are adequately uncorrelated for DTs to perform well.

The second observation comes as no surprise. Due to our relatively low sampling rate (which is tied to the packet transmission rate: 50 Hz) and the relatively large window size (1 sec) our data fail to accurately capture phenomena that happen very fast and involve abrupt changes in posture, e.g.

falling or standing up. On the other hand, walking or running are roughly periodic activities with less frequent posture changes and our data capture them quite accurately.

LSTM Results & Discussion

As seen below (Fig. 6), the optimization process converge in less than 1000 epochs (about 400) and our test result shows total accuracy of 77.5%. From the confusion matrix (Fig. 7) we can see that LSTM outperforms traditional supervised learning at fall detection. However, at "Sitting Down" and "Standing Up" it does not achieve good accuracy. It seems like the similar activities "Standing Up", "Falling Down" and "Laying Down" are difficult to discriminate using this model.

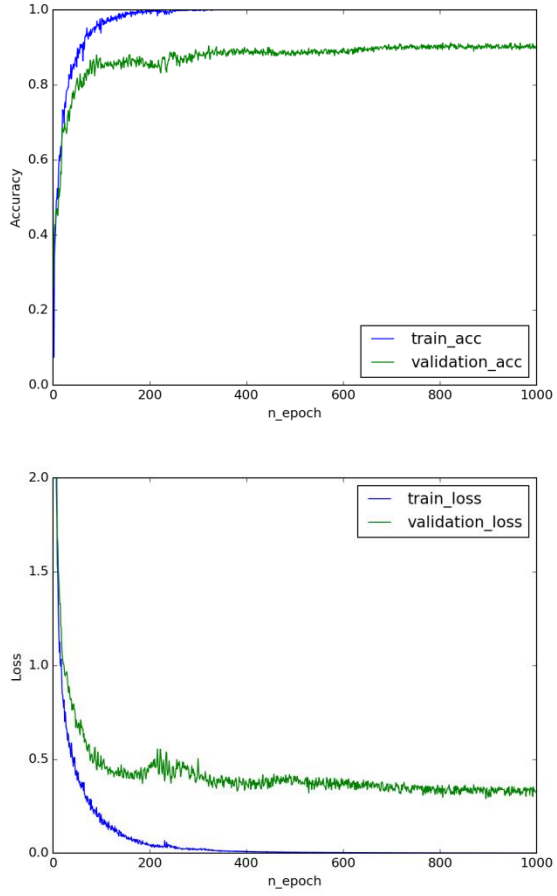

Fig. 6. Accuracy and Loss function as the number of epochs increases.

For this issue, we have some choices in annotation for fitting to applications. In Human Fall Detection, we can put together "Laying Down", "Sitting Down" and "Standing Up" to "Changing the Pose". In Activity Recognition, we can decompose the activity to more details (ex. "Laying Down" to "Sit down" and "Laying Down"). In any case, we believe how to annotate for the data is a one of difficult task.

		Prediction								
		Laying Down	Falling	Walking	Picking Up	Running	Sitting Donw	Standing Up	Total # of samples	Accuracy [%]
Label	Laying Down	35	2	0	3	1	5	3	49	71.4
	Falling	1	30	1	0	0	0	4	36	83.3
	Walking	0	3	125	0	8	0	1	137	91.2
	Picking Up	0	1	0	31	21	1	1	55	56.4
	Running	0	0	0	5	167	1	0	173	96.5
	Sitting Down	21	4	1	0	3	13	10	52	25.0
	Standing Up	1	0	5	1	1	12	15	35	42.9

Fig. 7. Confusion matrix for LSTM.

Conclusion & Future Work

On the one hand, the accuracy achieved by the traditional supervised methods is poor, but we have identified a number of ways in which we might improve it. On the other hand, LSTM gave very promising results, which have prompted us to continue working on this interesting and challenging problem. Based on our observations above, there are several directions towards which we plan to move in the future.

1) Our dataset is very small, especially for training deep learning algorithms; collecting more data should improve the accuracy of our algorithms' predictions.

2) The sampling frequency (packet transmission rate: 50 packets/sec) and the window size (1 sec) are probably too small and too large, respectively, to capture fast, transient phenomena like falls. We plan to test transmission rates of up to 0.6 or 1.2 kHz, since the posture changes during a fall might happen at frequencies as high as 300 Hz, as well as smaller windows.

3) For traditional learning techniques, preprocessing the data by applying the Discrete Wavelet Transform instead of the STFT will allow us to utilize time and frequency data simultaneously.

4) State based models, such as Hidden Markov Models, may be better at capturing transient phenomena than the constant-sized window ones that we used.

5) Extending our algorithms to utilize phase data in addition to amplitude seems particularly promising, since phase data perform well at discerning movements in a number of applications.

6) Evaluating additional learning algorithms such as Neural Nets may help us gain better insights into the nature and peculiarities of the problem.

References

[1] Wei Wang, Alex X. Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking* (MobiCom '15).

[2] Chunmei Han, Kaishun Wu, Yuxi Wang, and Lionel M. Ni, 2014 WiFall_Device-free Fall Detection by Wireless Networks. In IEEE Conference on Computer Communications, (IEEE INFOCOM ‘14).

[3] Linux 802.11n CSI Tool. Authors: Daniel, Halperin, Wenjun Hu, Anmol Sheth, David Wetherall Maintainers: Daniel Halperin, David Ward.

[4] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-178.

[5] E. Ertin, N. Stohs, S. Kumar, A. Raij, M. al’Absi, and S. Shah. AutoSense: unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field. In Proceedings of ACM Sensys, 2011.

[6] Google project Soli. https://www.youtube.com/watch?v=0QNiZfSsPc0. (Retrieved 2016-12-16.)

[7] J. K. Aggarwal and S. R. Michael. Human activity analysis: A review. ACM Computing Surveys, 43(3), 2011.

[8] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. M. Ni. We can hear you with Wi-Fi! In Proceedings of ACM MobiCom, 2014.

[9] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu. E-eyes: In-home device-free activity identification using fine-grained WiFi signatures. In Proceedings of ACM MobiCom, 2014.

[10] By BiObserver - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=43992484