RF-Based Fall Monitoring Using Convolutional Neural Networks

YONGLONG TIAN*, Massachusetts Institute of Technology
GUANG-HE LEE*, Massachusetts Institute of Technology
HAO HE*, Massachusetts Institute of Technology
CHEN-YU HSU, Massachusetts Institute of Technology
DINA KATABI, Massachusetts Institute of Technology

Falls are the top reason for fatal and non-fatal injuries among seniors. Existing solutions are based on wearable fall-alert sensors, but medical research has shown that they are ineffective, mostly because seniors do not wear them. These revelations have led to new passive sensors that infer falls by analyzing Radio Frequency (RF) signals in homes. Seniors can go about their lives as usual without the need to wear any device. While passive monitoring has made major advances, current approaches still cannot deal with the complexities of real-world scenarios. They typically train and test their classifiers on the same people in the same environments, and cannot generalize to new people or new environments. Further, they cannot separate motions from different people and can easily miss a fall in the presence of other motions.

To overcome these limitations, we introduce Aryokee, an RF-based fall detection system that uses convolutional neural networks governed by a state machine. Aryokee works with new people and environments unseen in the training set. It also separates different sources of motion to increase robustness. Results from testing Aryokee with over 140 people performing 40 types of activities in 57 different environments show a recall of 94% and a precision of 92% in detecting falls.

CCS Concepts: • **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**;

Additional Key Words and Phrases: Fall Detection, Device-free, Deep learning

ACM Reference Format:
Yonglong Tian, Guang-He Lee, Hao He, Chen-Yu Hsu, and Dina Katabi. 2018. RF-Based Fall Monitoring Using Convolutional Neural Networks. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 137 (September 2018), 24 pages. https://doi.org/10.1145/3264947

1 INTRODUCTION

Falls are the leading cause of accidental death and injury in older adults [46]. One in three adults over the age of 65 experiences a fall each year, with a significant fraction suffering an injury requiring a hospital visit [27]. Falls account for nearly 90% of fractures in seniors, and result in $34B of direct medical costs

*Co-primary Authors.

Correspondense to: Yonglong Tian <yonglong@mit.edu>, Guang-He Lee <guanghe@mit.edu>, and Hao He <haohe@mit.edu>
Authors' addresses: Yonglong Tian, Guang-He Lee, Hao He, Chen-Yu Hsu and Dina Katabi, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

2474-9567/2018/9-ART137 $15.00
https://doi.org/10.1145/3264947

annually [40]. The problem is aggravated by the fact that 12 million seniors in the US live alone [50]. Those people are particularly vulnerable to falls, which can go undetected for many hours [14]. Existing solutions require seniors to wear a pendant or other sensors. Yet, years of medical research has shown that wearable devices do not work well for the elderly. Seniors are typically encumbered by wearable technologies, and many of them suffer from memory problems and hence may forget to wear or charge their devices. Furthermore, those sensors can be dangerous; recently an elderly woman got strangled with her fall detection pendant [18].

The importance of the problem and the limitations of existing products have motivated researchers to seek passive fall monitoring solutions. In particular, researchers have proposed systems that transmit a low power RF signal and analyze its reflections off people's bodies to infer falls [4, 15, 35, 41, 45, 56, 58, 59]. Such RF-based fall detection systems work without any body contact. Hence, they can relieve seniors from the burden of wearing and charging devices and deliver safety at zero overhead.

While the research community has made significant advances toward RF-based fall monitoring, past proposals have significant limitations. State-of-the-art RF-based fall detection systems can be divided into two categories: The first category is based on Doppler radar [15, 22, 45]. These solutions exploit the relationship between the Doppler frequency and motion velocity. They associate falls with a spike in the Doppler frequency due to a fast fall motion. The second category is based on WiFi channel state information (CSI) [41, 56, 58, 59]. While this category differs in its input signal, it typically relies on the same basic principle. Specifically, it uses the Fourier transform to compute changes in the WiFi channel. It associates a burst of fast changes –i.e., a spike in high frequencies – with the fast motion that occurs during falls. Thus, whether they use Doppler or CSI, past solutions look at variations in the speed of the moving body, as captured by the RF signal. This approach leaves past work vulnerable to several practical challenges:

- *Dealing with complex falls and fast non-fall movements:* Not all falls have to be fast. In many cases, people catch a piece of furniture or lean against a wall as they fall, which softens the speed of falling. Also, it is common for elderly to fall off their wheelchair, which exhibits a slower speed than falling from a standing position. Such scenarios reduce the velocity of falling and the power in high frequencies, hence undermining the basic principle underlying past work. Furthermore, many non-fall activities exhibit fast motion and a burst of high frequencies. This includes sitting abruptly on a chair, or quickly opening a door. Since Doppler radar and CSI-based schemes characterize only motion velocity, they cannot distinguish such movements from falls.
- *Generalization to new homes and new people:* It is unrealistic to require a fall detection system to be retrained every time it is deployed in a new home or with new people. However, past RF-based fall detection systems are trained and tested in the same environment and typically with the same people. This problem is intrinsic to past techniques because they cannot separate reflections from different sources in space. Thus, the received signal is a combination of all the reflections in the environment. When the environment changes (e.g., a new home), the signals combine in a different way compared to the training environment, hence changing how fall and non-fall patterns look.
- *Detect falls in the presence of other motion:* A fall detection system should detect a fall even if the neighbor is moving next door. Past systems have no mechanisms to separate reflections from different people. As a result, RF reflections off the body of a moving neighbor can smear the frequency patterns associated with falls and prevent detection (as shown in §4).

In this paper, we aim to address the above challenges and deliver a reliable RF-based fall detection system that generalizes across people and environments, works in the presence of other sources of motion, and deals with complex activities and fall patterns.

We introduce Aryokee, a new design for RF-based fall detection. Aryokee departs from past work along two axes. Aryokee uses an FMCW radio equipped with two antenna arrays: a vertical array and a horizontal array. The advantage of such radio is that it provides spatial separability. FMCW can separate RF reflections based on distance from the reflecting body, and the vertical and horizontal arrays separate reflections based on their elevation and azimuthal angles. By characterizing reflections as a function of space, the system can differentiate reflections coming from the floor level from reflections coming from higher elevations.

However, it is not enough to use a different radio. It might seem that once we have spatial information, we can simply use the elevation of a person's reflection to detect falls. Such a design does not work well in practice. It is hard to set a threshold on the elevation of a fall because people fall at places with different elevations (e.g., stairs). Also, people can sit and lie on the floor to exercise. To design a robust fall detection system, we need to capture not only the location information but also the diverse spatio-temporal fall patterns. Such patterns are quite complex due to multiple reasons. As a person moves, at each point in time, the dominant reflection comes from different body parts and changes frequently over time [2]. If we only keep track of the locations of those dominant reflections, the radio may receive at some point strong reflections from the feet but weaker reflections from the upper body, causing it to incorrectly infer that the person is on the floor. This is further complicated by the fact that the human body is specular at the frequencies of interest (a few GHz) because the wavelength is larger than the roughness of the reflection surface [2, 8]. In this case, only signals that fall close to the normal on the surface are reflected back towards the radio. As a result, the radio senses only a few body parts at any point in time.

To deal with this complexity, Aryokee leverages convolutional neural networks (CNNs) [31], which have demonstrated the ability to extract complex patterns from various types of signals, such as images and videos [16, 20, 30, 51, 52, 57, 60, 62, 63]. Our CNN design is customized for fall detection using RF signals. Specifically, it uses spatio-temporal convolutions which allow the network to aggregate information across space and time and abstract complex spatio-temporal patterns. Further, we combine two CNNs: the first detects a fall event while the second detects a stand-up event. The two networks are coordinated via a state machine that tracks the transition of a person from a normal state to a fall state, and potentially back to a normal state. This combination enables Aryokee to estimate whether the person is able to stand up on their own after a fall, and the duration of fall events. Such information is a predictor of the person's overall health and ability to live independently [36]. Further, the amount of time on the floor is indicative of fall complications including muscle damage, pneumonia, pressure sores, dehydration, and hypothermia [21].

We have implemented Aryokee and evaluated its performance empirically. Results from testing the system with over 140 people performing 40 types of activities, including complex falls and fast non-fall motions, in 57 different environments reveal the following findings:

- Aryokee is highly accurate, even when trained and tested on different people and environments. Specifically, Aryokee has a recall of 94% and a precision of 92% in detecting a fall. In comparison, the only prior work that trains and tests on different environments and people shows a recall and precision in the range of 70% to 76% [41]. [1]
- Baselines based on Support Vector Machine (SVM) and Long Short-Term Memory (LSTM) that are trained and tested on the same dataset as Aryokee cannot generalize to new environments and people unseen in the training set, and exhibited a precision of 17% and 21%, and a recall of

[1]We note that this prior work is evaluated on a different dataset. However, as shown in Table 1, our dataset is more diverse and challenging.

16% and 15% respectively. This result emphasizes the importance of Aryokee's design in ensuring generalizability.
- Aryokee accurately detects whether the person is able to stand up on their own and the time spent on the floor. Its mean error in estimating the time spent on the floor is 1.254 seconds.
- Finally, Aryokee maintains its high accuracy in the presence of other people in the environment and other sources of motion.

Contributions: This paper makes the following contributions:

- It introduces the first convolutional neural network architecture for RF-based fall detection. Our CNN design extracts complex spatio-temporal information about body motion from RF signals. As a result, it can characterize complex falls and fast non-fall motions, separate a fall from other motions in the environment, and generalize to new environments and people.
- It presents a multi-function design that combines fall detection with the ability to infer stand-up events and fall duration.
- It presents an extensive empirical evaluation with multiple sources of motion, over 140 people, 57 environments, 40 types of activities, and complex fall and non-fall events.

2 RELATED WORK

Past work on fall detection can be divided into two categories: wearable and non-wearable technologies. Notable examples of wearable devices include accelerometers [12, 33], smart phones [1, 12], RFID [10], etc. However, prior work has shown that older people feel encumbered by wearable devices, might experience skin irritation, and tend to forget to wear or charge their sensors [48, 49, 61]. Non-wearable technologies solve these limitations and enable continuous fall monitoring without the need to wear devices. Among non-wearable technologies, cameras [32, 39] are accurate but they invade people's privacy, suffer from occlusions, and have a narrow field of view. Audio and vibration based sensors [5, 34] have a relatively low accuracy due to interference from the environment [38]. Pressure mats and pulling cords work only when the fall occurs near the installed device [37].

Our work is most related to past papers on RF-based fall monitoring [15, 35, 41, 45, 56, 58, 59]. Early systems use an array of RF sensors deployed around the perimeter of a room [35]. They measure the received signal strength indicator (RSSI) between all sensor pairs and infer falls based on changes in the RSSI of various links. However, such RSSI based solutions are difficult to deploy, require detailed fingerprinting of the environment, and have a relatively low accuracy. Also, some early systems use FMCW radios [4] but they focus on localization and present fall as a motivating example. They cannot deal with complex motion and are limited to four actions: fall, walking, sitting on a chair and sitting on the floor. They do not generalize to new environments or new people. They also cannot deal with other sources of motion in the environment.

State-of-the-art solutions can be divided into two categories: Doppler radar [15, 45] and CSI-based solutions [41, 56, 58, 59]. As explained in §1, both methods try to measure motion velocity as a function of time. Doppler-based approaches directly measure motion velocity by leveraging its relationship with the Doppler frequency. CSI-based approaches use the Short Time Fourier Transform (STFT) or the wavelet transform to estimate fast changes in the RF signal. They expect falls to generate a burst of high frequencies due to the high velocity of falling [41, 59]. They both use traditional machine learning classifiers (e.g., SVM, random forest) to distinguish high-frequency fall patterns from non-fall events.

Whether they use CSI or Doppler radar, state-of-the-art RF-based fall detection methods have important limitations. These cannot separate RF reflections from different people or objects in the environment.[2] As a result, they do not work well when a fall occurs in the presence of other sources of motion, and do not generalize to homes and people that do not appear in the training set.

Finally, one group has recently investigated the use of deep learning for fall detection [25]. Their approach, however, is intrinsically different from ours. Instead of CNNs, they use fully-connected auto-encoders with one hidden layer. Because their networks are fully-connected, they cannot scale to a deep architecture with many layers, making it difficult to learn complex patterns of diverse activities. As a result, their approach has a lower performance than Doppler-radar or CSI-based schemes. In contrast, our CNNs with spatio-temporal convolutions leverage locality in space and time to scale to many layers (more than 10), providing a highly accurate classifier [20, 30]. Further, they do not scale to new environments, do not work in the presence of more than one person, and are limited to 4 actions (fall, walk, sit, and bend) that have to be performed in a specific way with a particular orientation with respect to the radio [25].

3 CNN BACKGROUND

Convolutional Neural Networks (CNN) have been the main workhorse of recent breakthroughs in understanding images [30], videos [28, 55] and audio signals [7, 53]. Unlike ordinary neural networks where each neuron is connected to all neurons in the previous layer, in CNNs, each neuron is only locally connected to a few neurons in the previous layer. Thus, CNNs leverage local dependencies in the data to reduce the total number of weights that the network needs to learn. This allows them to build deeper networks and hence learn more complex features.[3]

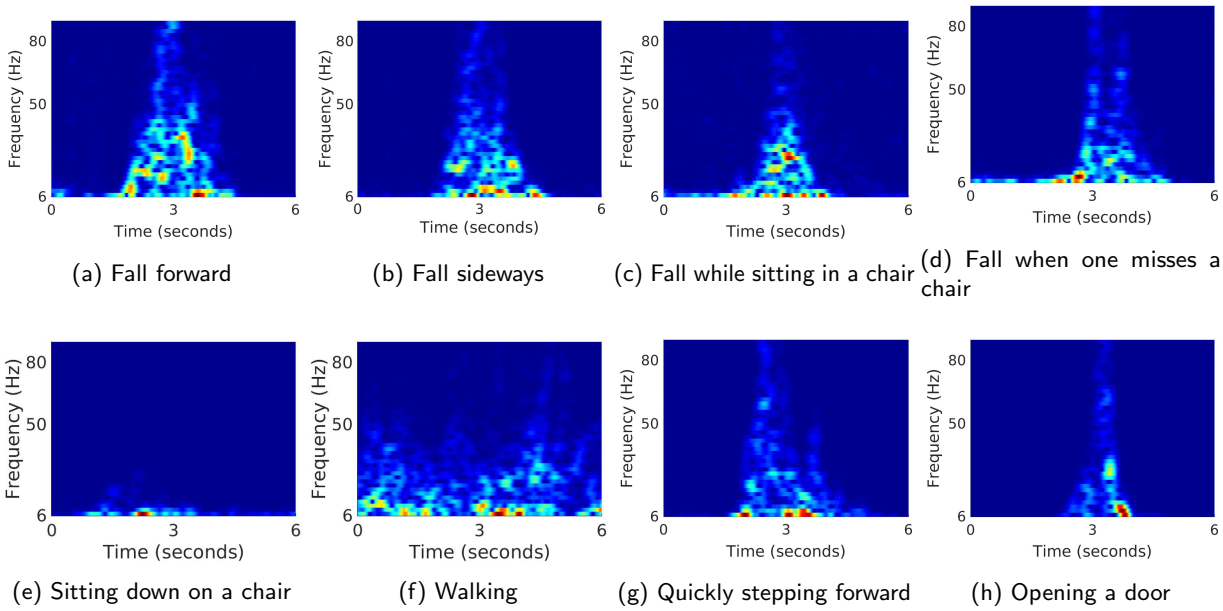
CNNs, like other neural networks, are trained by back-propagating the gradient. However, training a deep CNN is hard due to the notorious gradient vanishing problem. As the gradient is back-propagated to lower layers, repeated multiplication of small values may make the gradient infinitively small. The ResNet architecture addresses this problem and allows for efficient training even when the network is very deep [20]. Instead of trying to learn a direct mapping $H(x)$, ResNet make the layers learn a residual mapping $F(x) = H(x) - x$. Then the underlying mapping is recast into $F(x) + x$, where the first component is implemented by stacked layers and the second component by shortcut connections between layers. Such shortcut connections enhance and diversify the flows of gradients, and therefore make the optimization of very deep CNNs easier. With residual learning, CNNs can be scaled up to hundreds of layers without causing the gradient to vanish. ResNet has been highly successful and is adopted by the best-performing models in many tasks including object detection [30], image segmentation [43], speech synthesis [53], machine translation [26], and AlphaGo [47]. In this paper, we adopt the ResNet architecture for our CNNs.

4 PRACTICAL CHALLENGES FACED BY PAST RF-BASED FALL DETECTION

We start by examining the main technique underlying state-of-art RF-based fall detection systems. Most RF-based fall detection systems use either Doppler radar or WiFi CSI. In both cases, it is quite common to use a technique called time-frequency analysis [6, 41]. This technique applies a STFT to the RF signal. It produces a spectrogram where high frequencies are associated with fast motions. The intuition that

[2]Unlike, multi-antenna FMCW radios which separate signal reflections based on the location of the reflector, in these methods signals reflected from different bodies are superimposed and not separated (see section 9.1 in [23]).

[3]Of course, CNNs like other deep learning models typically require more training data than conventional methods, e.g., SVM or random forest. However, given the right amount of data, they tend to outperform such conventional approaches.



(a) Fall forward (b) Fall sideways (c) Fall while sitting in a chair (d) Fall when one misses a chair

(e) Sitting down on a chair (f) Walking (g) Quickly stepping forward (h) Opening a door

Fig. 1. Spectrograms for falls and non-falls based on time-frequency analysis used in past work. The top row shows fall events. The bottom row shows non-fall events. The figure shows the limitations of the time-frequency analysis used in past work, where fast movements such as opening or closing a door or taking a quick step forward can look like fall events.

those systems rely on is that the motion of falling is faster than sitting or walking. Hence falls tend to appear in a spectrogram as a burst of high frequencies [41].

To better understand the performance of time-frequency analysis, we run a few fall and non-fall experiments. We use a software radio to implement a Doppler radar and take the STFT of the wireless signal with a window size of 0.5 seconds and a time step of 5 milliseconds in a manner similar to [6, 41]. We plot the time-frequency spectrograms in Figure 1.[4] The figure shows that the time-frequency technique is highly effective at differentiating falls (Figure 1a - Figure 1d) from actions like sitting down on a chair (Figure 1e) and walking (Figure 1f). However, fast non-fall movements can be problematic. For example, a quick step forward and the action of opening or closing a door (Figure 1g - Figure 1h) can both produce a burst of high frequencies and can look like a fall event.

Another limitation of time-frequency analysis is its inability to separate different sources of motion. For example, if a neighbor walks during a fall event, the spectrogram of the walking motion super-imposes on the spectrogram of the fall, and can mask the fall event. Figure 2 shows an example of a fall in the presence of a walking motion. The fall occurs in the 27th second of the experiment but is masked by the walking motion.

In the following sections, we introduce a new design that can address the above limitations. We also demonstrate in the evaluation section that our system correctly classifies fast non-fall movements (*e.g.*, Figure 1g - Figure 1h) and detects falls in the presence of walking or other motion.

[4]For clarity, we plot frequencies higher than 5 Hz

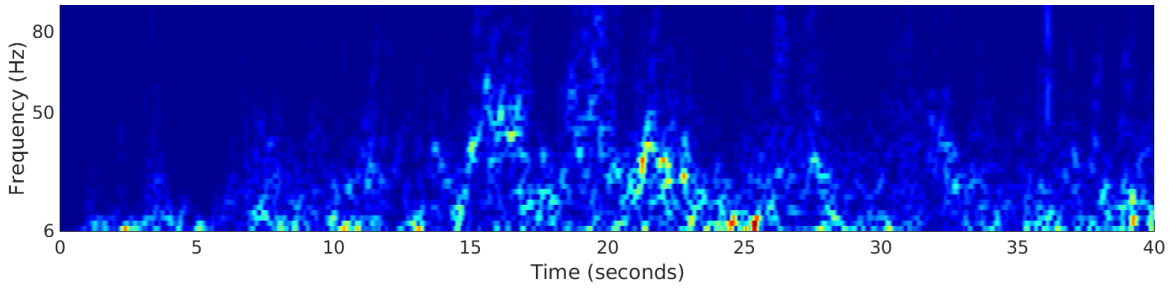


Fig. 2. Time-frequency analysis of a fall in the presence of walking motion of another person in the environment. The fall happened on the 27th second but is masked by the walking motion.

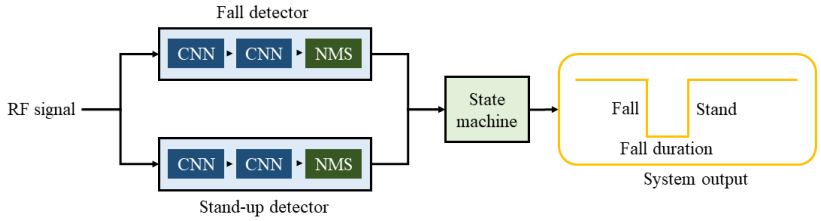


Fig. 3. An overview of our system. The fall and stand-up detectors detect falls and standing up events based on RF signals in a time window. The finite state machine integrates the results from these two detectors and generates a sequence of states ("normal" or "fall") indicating when a person falls and stands up, and the duration of a fall.

5 ARYOKEE SYSTEM OVERVIEW

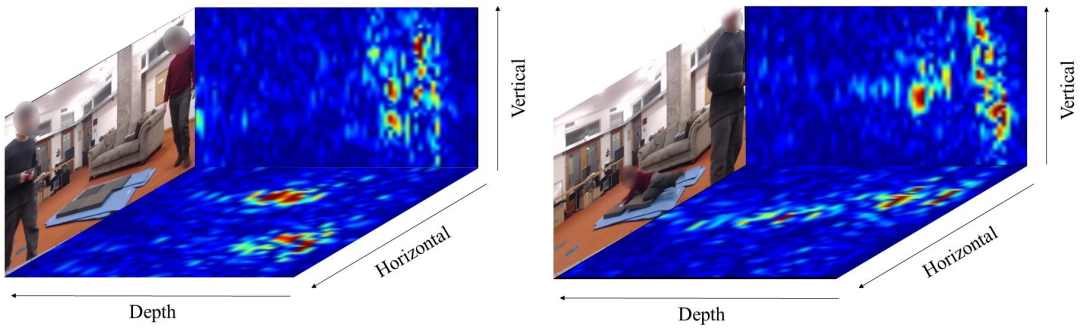
Aryokee is a passive fall detection system based on RF signals. It is built on a multi-antenna FMCW radio and takes as input the RF reflections from the environment. The system is integrated with three modules: a fall detector, a stand-up detector, and a finite state machine to deliver continuous fall monitoring. Figure 3 illustrates the design of Aryokee.

- The fall detector takes a short time window of RF signals and classifies if a fall occurs in the time window. The fall detector consists of two cascading CNNs and a none-maximum suppression (NMS) step. The first CNN filters out easier non-fall examples while the second one refines its predictions on harder non-fall examples. As the window slides over time, one prediction is made at each time step. The NMS step removes duplicated detections to pinpoint the precise time of a fall.
- The stand-up detector detects if a person stands up and operates similarly to the fall detector.
- The state machine integrates the window-based predictions from the fall and stand-up detectors to infer the exact fall time and fall duration.

The next few sections describe Aryokee in detail. We start with a description of the RF signal captured by the radio then elaborate on the various components of Aryokee.

6 CAPTURING RF SIGNALS IN 3D

Aryokee uses an FMCW radio equipped with a horizontal and a vertical antenna array to separate reflections from different locations in the 3D space. The FMCW radio separates RF reflections based on



(a) RF heatmaps when two people are walking.

(b) RF heatmaps when one person is falling.

Fig. 4. Visualization of our system's RF data. The figure shows two people walking in 4a, and one person walking while the other person is on the floor 4b. We represent RF reflections using a combination of horizontal and vertical heatmaps. The heatmaps show the reflection power as a function of space where red refers to higher power and dark blue refers to low power.

the distance of the reflector, whereas the vertical and horizontal arrays separate reflections based on their elevation and azimuthal angles. Thus, we can represent RF reflections as a function of the 3D space.

We project the 3D reflections on two perpendicular planes: a horizontal plane and a vertical plane. This generates two heatmaps at each time step. The first heatmap shows reflection powers as a function of the horizontal angle and depth, while the second heatmap shows reflection power as a function of the vertical angle and depth. Figure 4 provides two examples of these heatmaps. The figure shows that the horizontal heatmap separates people based on their location. The vertical heatmap captures elevation and hence shows a smaller expansion when the person is on the floor, as in Fig. 4b. The combination of these heatmaps allows Aryokee to separate motion from different sources and hence detect a fall in the presence of other activities.

7 ARYOKEE'S FALL MONITORING SYSTEM

In this section, we elaborate on each component of Aryokee, and how they are integrated to deliver the complete multi-function system that can infer falls, the ability to stand-up, and fall durations.

7.1 Fall Detection with Convolutional Neural Network

We formulate fall detection as a binary classification problem, *i.e.*, fall versus non-fall. Such classification is traditionally dealt with by manually designing hand-crafted features followed by a classifier such as SVM or Random Forest. However, it is hard to take into account all the complex fall patterns as well as the diverse non-fall activities in designing such system manually. In this paper, we resort to convolutional neural networks, which have demonstrated their advantages to capture complex mappings by learning from labeled examples.

We take into account the property of RF signals in designing the CNN. Specifically, as a person moves, the dominant reflection come from different body parts and changes frequently over time. Moreover, given that the human body is specular at the RF frequencies of interest, at each point in time only a small subset of body parts reflect signals back to the radio. To deal with these issues, we make our CNN model aggregate temporal information to capture the dynamics of falls as well as the diverse non-fall patterns. Therefore, our CNN takes a sequence of RF heatmaps as input and applies convolutions over time and

space to extract complex spatio-temporal patterns. In particular, we take as input a sequence of 112 horizontal and vertical heatmaps, which span 2.5 seconds.

In addition, the CNN model needs to fuse the information from the horizontal and vertical heatmaps. At the high level, the horizontal heatmap gives us the location of each person while the vertical heatmap reveals changes in elevations. Therefore, such fusion enables our system to detect fall of one person in the presence of other people's motions.

A naive way to fuse the two streams would be stacking the two heatmaps and treating them as two channels of same image stream before feeding them to the convolutional layers. However, the same pixel on the two heatmaps corresponds to different 3D positions, which violates the locality assumption of convolution layer that feature map in each channel should have the same coordinates.

In our design, we adopt two branches to extract features from the horizontal and vertical heatmaps separately. The weights are not shared across the two branches, since the movement patterns in horizontal and vertical directions are different. We use a convolutional stride of 2 to reduce the spatial and temporal dimensions of our data every two layers of convolutions. After 10 layers of spatio-temporal convolutions, we conduct max pooling along both spatial and temporal axes for each channel to derive a vector representation for the vertical branch and the horizontal branch. After concatenating the vector representations, two fully connected layers are employed to fuse the information from the two branches. Finally, we use a softmax function to obtain probability prediction for each class.

Our final model is illustrated in Figure 5. We trained our model end-to-end to minimize a loss function that captures the differences between network predictions and the correct labels (i.e., fall or non-fall). We denote x_i as the RF signals in a 2.5 second window around the ith frame, y_i as the fall label at timestamp i, and $\hat{p}(x_i, \theta)$ as the prediction of our model with parameters θ. All the parameters of the CNN model, θ, are optimized with a cross-entropy loss:

$$\min_{\theta} \sum_{i=1}^{N} -y_i \log \hat{p}(x_i, \theta) - (1 - y_i) \log(1 - \hat{p}(x_i, \theta)) \tag{1}$$

7.2 Combat Sparsity with Cascading Classifiers

Falls are sparse during continuous monitoring and usually overwhelmed by non-falls. For example, the ratio of falls over non-fall activities is typically close to 0 in people's daily life. Such unbalance is disadvantageous for training a fall classifier. This is because most non-falls are easy samples but they distract the classifier from recognizing hard samples. Therefore, we use cascading classifiers, a popular method that deals with sample unbalance for object detection in computer vision [54]. Basically, cascading is a multistage framework. The lower stages filter out easy negatives, while the higher stages only focus on samples that are not rejected by lower stages and then recognize hard negatives. For example, our cascading classifier consists of two stages and each stage is a convolutional neural network. As illustrated in Figure 6, the first CNN rejects easy non-falls, *i.e.*, green crosses, but retains the hard non-falls, *i.e.*, red crosses. The positives and remaining hard negatives are passed to the second classifier where most of the hard negatives can be further rejected. Increasing the stages of cascade can further improve the performance but will also bring more computational overhead. In Aryokee two-stage cascading works in real time and increasing to 3 stages can only bring marginal performance gain.

The benefits of cascading are twofold. First, it improves the overall performance as it tackles with easy and hard negative samples with two CNNs. Second, it enhances the robustness of Aryokee to environmental changes. Without cascade, the score boundary – i.e., the threshold – for identifying falls from non-fall is sensitive to the environment. In the first stage, the confidence scores of positives and hard negative

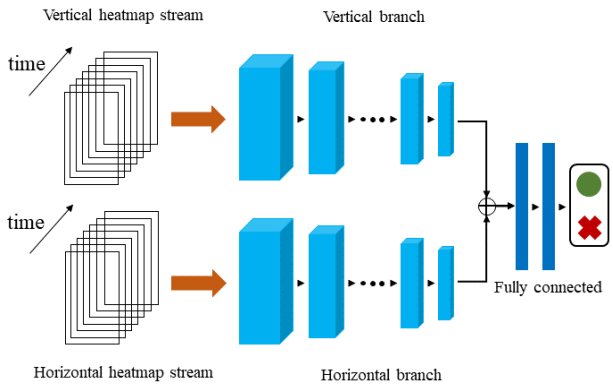


Fig. 5. Our 3D convolutional neural network design. This figure shows the network structure of one of the CNN blocks in Figure 3. The network consists of ten 3D convolution layers and two fully connected layers, followed by a softmax classifier. The features used for classification have 256 dimensions. The input RF signals have both vertical and horizontal heatmaps, which are convolved separately in each branch and fused right before the first fully connected layer.

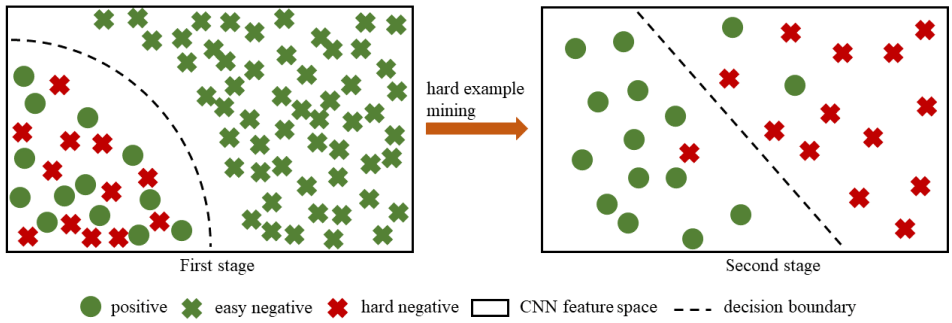


Fig. 6. An illustration of our cascading classifiers. The first stage rejects easy negative samples and passes hard ones to the second stage, which further rejects most of the hard negative samples. The negative and positive examples look redistributed between the two stages because they are projected in different spaces.

samples are very close, and therefore the boundary between them is susceptible to environmental changes. However, after the second stage, the score gap between positives and hard negatives is significantly enlarged. Then all environments can share the same threshold, which enhances the generalization ability.

During training, we start by training the first stage CNN with all training samples. Then we use this CNN to filter our easy negatives in the training set and train the second stage CNN. During inference, only samples that pass both stages are recognized as falls by our system.

7.3 Precise Monitoring with Non-Maximum Suppresion

Most past works expect the input to be a set of fall and non-fall examples, and run a binary classifier on each example. In practice, however, a fall detection device has to perform continuous monitoring –i.e., it

has to process a continuous stream of RF signals with all types of motion, and identify instances of falls automatically. It cannot expect that each activity is already segmented into a specific time window.

The objective of Aryokee is to monitor falls continuously and precisely, which requires the CNN to use a sliding window over the input RF signal. This creates some complications since a single fall can partially appear in multiple sliding windows. For example, given a 3-second sliding window with a stride of 0.2 seconds, a fall lasting 3 seconds can partially appear in about 30 windows. This can cause the same fall to be detected multiple times. Further, it becomes hard to pinpoint the exact time when the fall happens due to multiple plausible answers from overlapped prediction windows. Therefore, directly using the output of CNN as detection results is inadequate for precise monitoring.

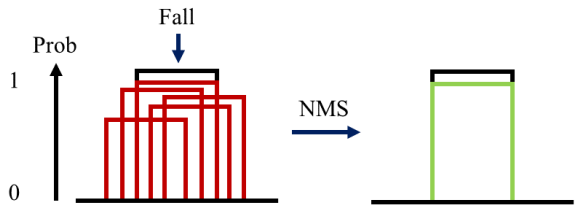


Fig. 7. An illustration of the NMS algorithm. Overlapped windows with lower probability will be suppressed by the local optimum.

To solve this problem, we borrow the idea of non-maximum suppression (NMS), which is widely used in computer vision for edge detection [9] and object detection [13]. At a high level, NMS preserves the detection window that is a local optimum in terms of its prediction score compared to its overlapped neighbors. Figure 7 illustrates the concept, showing multiple overlapping time windows with different probabilities being a fall event. Only the window with the highest probability is preserved, while other overlapping windows are ignored. Intuitively, the best-aligned window is expected to have higher prediction score than its neighbors which are aligned less accurately. Thus, NMS helps output one window for each fall of interest, each window tightly enclosing the fall.

Computationally NMS is performed as follows. Each window is associated with a score estimated by the CNN. We first sort all the detected windows by their prediction score in a descending order. Afterwards, we greedily select the highest scoring windows while skipping windows that are at least 50% covered by a previously selected window. Such operations are implemented in a matrix format. Therefore, NMS is efficient and the computational overhead is negligible.

7.4 State Machine and Stand-up Detection

In addition to detecting and pinpointing the time of a fall, it is helpful to also keep track of the status of the person, *i.e.*, whether the person is able to stand up on their own after a fall. This information is an important predictor of a person's overall health and is indicative of potential fall complications. In principle, the longer the person stays on the ground, the more alarming the situation is. It either indicates the person is unconscious or incapable of getting up on their own.

To estimate if the person is able to stand up on their own, we build a stand-up detector using a similar design as the fall detector described in Section 7.1. In addition, a state machine is designed to track the state of a fall event by incorporating the predictions of the fall and stand-up detectors. The state machine includes two states: a normal state and a fall state. Figure 8 illustrates the operation of the proposed state machine. When a fall is detected, the state is transitioned from the normal state to the fall state.

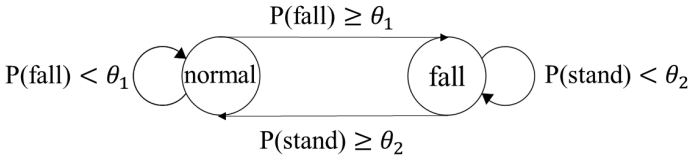


Fig. 8. An illustration of the proposed state machine. We maintain a normal state and a fall state, where state transition is coordinated by fall and stand-up detectors, which are built based on the techniques in previous sections. P(fall) and P(stand) are predictions from the detectors, and the two parameters θ_1 and θ_2 determine the thresholds that induce state transitions.

Once in the fall state, the state will not transition back to the normal until the stand-up detector detects a stand-up event. We estimate the person's ability to stand up and the duration of a fall by keeping track of the state transitions.

Combining the predictions of both detectors with a state machine also helps improve the performance of our system. For example, many activities involve movements similar to standing up. If we simply use the predictions of the stand-up detector to estimate the fall duration, we can easily over-estimate it because of many falsely predicted stand up events. With the regulation of our state machine, we simplified the problem to predict stand-up events conditioned on a fall happened. This design helps Aryokee estimate fall duration accurately.

8 IMPLEMENTATION AND DATASET

In this section, we describe our implementation and dataset.

8.1 Implementation

Neural Network Architecture. Our CNN architecture is based on the design of ResNet [20]. ResNet has been shown to perform well in a wide range of different tasks in computer vision. We incorporate its architecture with various design choices discussed in Section 7.1. As shown in Figure 5, both the vertical branch and horizontal branch have the same structure (10 layers) and independent parameters. All convolutional layers have a kernel size of 5. These two branches are merged and fed into two fully connected layers before the final softmax classification.

Training Details. All CNN models are trained with ADAM optimizer [29] with a learning rate of 0.002. Weight decay is not used. Batch Normalization [24] is adopted to accelerate training. To stabilize the learning process for unbalanced dataset, we sample 40% positives (falls) and 60% negatives (non-falls) for each batch. We augment positive samples by jittering windows around each fall. We subtract the median over time for each sample to remove reflections from static objects.

8.2 Radio Hardware

We built an FMCW radio on a printed circuit board (PCB) using off-the-shelf circuit components, and based on the design in [3]. The resulting radio can be operated from a computer via the USB port. It generates a frequency chirp that repeatedly sweeps the band 5.46 – 7.24 GHz. The radio has an average power of 0.08 μ Watts, which complies with the FCC regulations for consumer electronics in that

band [11]. At this power the radio has a typical coverage that is about 30 to 40 feet, which is similar to past work with FMCW radios [3, 4]. The radio is equipped with a horizontal and vertical antenna arrays. Each array has 12 antennas. The output of the antennas is processed using standard FMCW and antenna array equations [44] to produce the vertical and horizontal heatmaps. The frame rate of the heatmaps is 45 frames per seconds. Finally, while we built our radio to provide better compatibility with past FMCW-based research, a variety of FMCW radios with antenna arrays are available on the market [19, 42].

8.3 Dataset

We collected a large scale dataset of RF signals with fall labels from both male and female subjects with age between 20 and 50 year old. The subjects are faculty, students, and their partners. In our experiments, the radio device is either located on the wall or on a cart. As explained in the hardware section, the radio can sense people up to 30 to 40 feet away.

(a) Scale. The collected dataset is both large and diverse. To demonstrate the scale of our dataset, we compare its statistics with that of past work in Table 1. As shown in the table, our dataset contains 541 falls and around 0.5 million non-fall samples.

Table 1. Dataset statistics and comparison with past work.

	number of falls	number of non-falls	number of fall patterns	number of non-fall patterns	number of people	number of environments
Ours dataset	541	550,000	18	40	145	57
Palipana *et al.* [41]	326	744	4	8	3	5
Jokanović *et al.* [25]	117	291	4 (different angles)	3	3	2

(b) Environment diversity. Our dataset is collected from 57 different environments, including typical scenarios in homes, offices, classrooms, and modern buildings. Past work contains at most 5 different environments. Examples of our testing environments can be seen in Figure 9 and Figure 10. Further, our dataset covers scenarios with different levels of occlusion, including monitoring people who are occluded from the radio by a wall.

(c) Fall diversity; We collect 18 different patterns of falls, covering both basic falls and more challenging ones. To collect realistic fall patterns, we show each subject videos of real falls on YouTube and ask the subject to emulate those falls as realistically as possible. Subjects can fall in any orientation, i.e., forward, backward, lateral and on-position. Figure 9 shows some of the fall patterns. About 40% percent of the falls are relatively difficult to classify, such as falls from chairs, falls onto chairs, falls backward, falls in the presence of other moving people, slow or gradual falls, falls while trying to catch an object, etc. The remaining 60% are relatively easier falls and include slipping, tripping, sprained ankle, falling in place, etc. The full list of collected fall patterns is available in Appendix A.1.

(d) Non-fall diversity: We included 40 indoor non-fall activities. Examples non-falls are shown in Figure 10. Activities involving small or medium motion and small changes in elevation are covered in Figure 9i-9n. Activities with both medium motion and medium changes in elevation are shown in Figure 9o-9v. Activities with large or fast motion with drastic changes in elevation are shown in Figure 9w-9. We can divide the non-fall examples to relatively easy or relatively hard. The relatively easy negative examples include walking, siting, and standing. they occupy about 75% of the data. The remaining 25% are relatively harder samples. They include squatting, bending to pick up something, going down stairs,

sitting on the floor, jumping, lying down, etc. These examples are harder since they either have low elevation or fast motion similar to a fall.[5] By including all of these activities, the robustness of our system can be thoroughly assessed. We attach the full list of collected non-fall patterns in Appendix A.2.

(e) Human subject diversity: Our dataset involves more than 140 subjects, while past work includes up to 3 participants. 18 out of all the subjects contribute to falls and all subjects contribute to non-fall activities. All experiments are approved by our institutional review board (IRB).

9 EVALUATION

We first describe our experimental set-up and evaluation metrics, then evaluate Aryokee and compare it with a few baselines.

9.1 Evaluation Setup

The performance of the fall detector is evaluated by comparing its output with the human labeled ground truth. We ensure that data used for training and testing are from different environments except for the comparison study in Section 9.2.2.

Train/Test Split. We split our dataset into 5 and 16.5 hours for training and testing, respectively. We train our model in 5 environments and evaluate it in the remaining 52 environments. The total number of falls and non-fall examples used in training are 248 and 100,000, respectively. This leaves 293 fall examples, and 450,000 non-fall examples for testing.

Metrics. To emulate a realistic fall detection scenario, the ratio between fall and non-fall examples in our dataset is extremely unbalanced. In such a case, classification accuracy is not an ideal metric. A naive classifier can classify every window as non-fall and get an accuracy greater than 99% (*i.e.*, nearly 300 falls vs. 450,000 negative examples in the testing set), but be meaningless. Thus, we use the following metrics that can express the sensitivity and specificity of fall detection in the presence of unbalanced data. To explain the metrics, for simplicity, we abbreviate True Positives, False Negatives, True Negatives and False Positives as TP, FN, TN, and FP, respectively.

- *Accuracy:* The ratio of correctly classified samples over all samples, *i.e.*, $a = \frac{TP+TN}{TP+FN+TN+FP}$. When negative samples (*i.e.*, non-falls) dominate, $a \approx \frac{TN}{TN+FP}$ and measures mainly the accuracy of negative samples, ignoring the accuracy of detecting falls.
- *Precision:* The fraction of correctly detected falls over all detected falls, *i.e.*, $p = \frac{TP}{TP+FP}$.
- *Recall:* The fraction of correctly detected falls over the total number of falls, *i.e.*, $r = \frac{TP}{TP+FN}$.
- *F1 score:* The harmonic mean of precision and recall, *i.e.*, $\frac{2 \cdot p \cdot r}{p+r}$.

9.2 Evaluation of Aryokee's Fall Detection

We evaluate Aryokee under various conditions starting with the most general case where it is tested on new environments and new people unseen during training.

9.2.1 New People and New Environments. It is essential for a successful fall detection system to work in new environments and with new people without having to retrain the model. Thus, we test Aryokee's ability to work in such scenarios. We report the results in Table 2 below.

[5]The definition of relatively easy or hard negatives is subjective. Another definition might follow our cascaded detector. The first stage detector detects 97% non-falls as easy negatives and filters them out. The remaining 3% relatively hard ones are fed into the second stage.

(a) slip (b) trip (c) lose consciousness (d) lose balance

(e) fall from chair (f) fall with chair (g) w/ sb. cooking behind (h) w/ sb. jogging behind

Fig. 9. Sampled fall patterns in our dataset.

(i) write (j) use computer (k) water flower (l) ukulele (m) walk

(n) hulahoop (o) tai chi (p) stretch (q) yoga (r) pick up

(s) squat (t) sit (u) stairs (v) crawl (w) soccer

(x) pingpong (y) juggle (z) pool () dance () street dance

Fig. 10. Sampled non-fall patterns in our dataset.

Table 2. Aryokee's evaluation in *cross-people and cross-environment* settings. The table demonstrates Aryokee's ability to work in new environments and with people that it did not see during training.

	Precision	Recall	F1 Score	Accuracy
Aryokee	**0.919**	**0.938**	**0.929**	**99.96%**
FallDeFi [41]	0.76	0.70	0.73	74%

Continuous monitoring requires Aryokee to detect a few hundred falls from half a million examples, which is very challenging. Nonetheless, the table shows that Aryokee successfully detects 93.8% of the falls (*recall*) delivering a *precision* as high as 91.9%, even though Aryokee never sees the same people and environments during training. This demonstrates Aryokee's ability to generalize beyond the training set. In comparison, the previous best-performing method FallDeFi [41] achieves 76% *recall* and 70% *precision* in cross people and cross environment setting. We note that the performance of FallDeFi is reported on another dataset. The comparison between their dataset and ours is presented in Table 1, which shows that our dataset is significantly larger and more diverse. We also note the benefits of our cascaded design. In particular, around 97% out of the 450,000 negative samples are detected as easy negatives and filtered out by the first stage CNN. Most of the remaining 15,000 hard negatives are rejected by the second stage CNN.

Finally, the table shows that the classification accuracy is 99.96%. However as explained earlier, accuracy is not a useful metric when the data is unbalanced; simply classifying every window as a non-fall achieves 99% accuracy. Therefore, we only report *precision, recall* and *F1 score* in the following sections.

9.2.2 Comparison of Same vs. Different Environments and People. We also analyze the performance gap of Aryokee when tested on the same vs. cross environments and people. In this case, we keep the training dataset as before. Furthermore, we use the same trained model as before (i.e., we do not train a new model). Recall that the model was trained on 5 hours from 5 environments. We collect 3 additional hours of test data only from the people who appeared in the training set. This test data is divided into data from the 5 environments included in training, and data from 6 new environments that do not appear in training. Table 3 presents the results and compare them against testing on new people and new environments.

Table 3. Aryokee's Performance under Different Settings.

	Precision	Recall	F1 Score
Cross People, Cross Environment	0.919	0.938	0.929
Same People, Cross Environment	0.934	0.951	0.942
Same People, Same Environment	0.960	0.955	0.958

As reported in the second row of Table 3, Aryokee achieves 93.4%, 95.1% and 94.2% of *precision, recall* and *F1*, respectively, under the *same* people but in a cross-environment setting. If we further change *cross* environment to *same* environment, *precision, recall* and *F1* obtain a marginal improvement of by 2.6%, 0.4% and 1.6%, respectively. The results confirm expectations that since Aryokee works well across environments and people, it works also well on the same environments and people, and its performance is even slightly higher.

9.2.3 Through Wall Performance. We evaluate our fall detector in through-wall scenarios, where the radio and monitored people are separated by a wall, and compare the results with the line of sight scenarios. Again, we evaluate the same fall detection model as before without any new training. Only the test data is different. In particular, we partition the test data used for testing cross-environment and cross-people performance into two groups based on whether the monitored person was separated by a wall from the radio. We then evaluate fall detection in the two cases separately. The results are reported in Table 4. The table shows that Aryokee's fall detector works well even when the person is behind a wall. In particular, for the through-wall case, the *precision, recall* and *F1* are 93.2%, 94.5% and 93.9%, respectively. This is close to the line of sight setting, of which the *precision, recall* and *F1* are 95.6%, 96.1% and 95.8%.

Table 4. Aryokee's Through-Wall Performance.

	Precision	Recall	F1 Score
Line of Sight	0.956	0.961	0.958
Through Wall	0.932	0.945	0.939

9.2.4 Fall with Other Moving People. We investigate the robustness of our system when other people are moving or walking in the environment. Again, we evaluate the same fall detection model as before without any new training. Only the test data is different. In this case, we collect a dataset of people falling in the presence of nearby moving people. We collect this data in three new environments with three people. We use this data to test falling in the presence of nearby moving people. To test the case of no other moving people around, we use the test data from the cross-environment cross-people scenario but we exclude all examples when there were multiple people in the environment. The results are reported in Table 5. The *precision* is similar in both settings, while the *recall* drops slightly by 3.2% in the presence of other moving people. Finally, one may notice a small improvement in precision in the case of "with moving people" in comparison to the performance results for the case of cross-people cross-environment in Table 2. This is because the test data collected for scenarios with moving people include some people in the training set, while the results in Table 2 exclude such scenarios.

Table 5. With vs. without other moving people, under the *Same People Cross Environment* setting.

	Precision	Recall	F1 Score
w/ other moving people	0.941	0.914	0.928
w/o other moving people	0.936	0.946	0.941

9.2.5 Ablation Study. Our fall detector has a cascading classifier composed of two CNNs and a Non-Maximum Suppression module. To further understand the effectiveness of each design, we conduct an ablation study under *cross people and cross environment setting*.

To fairly evaluate each design, we compare *precision* under a similar *recall* of 94%. Results are illustrated in Table 6. Only a single CNN suffers a low precision of 0.552%, this is because non-falls dominates and it is hard for a single CNN model to handle all types of non-falls. By adding another cascade CNN, the precision is improved by 29.1%, because the first CNN can filter out easy non-falls while the second CNN focuses on hard non-falls. Our full model is obtained by adding the NMS module on top of the cascading classifier. This further improves the precision by 7.6%.

Table 6. Ablation study on each module of Aryokee's fall detector, under the *Cross People Cross Environment* setting.

Model	Precision	Recall	F1 Score
SingleCNN	0.552	0.938	0.695
SingleCNN+NMS	0.758	0.938	0.839
SingleCNN+cascade	0.843	0.938	0.888
full model (SingleCNN+cascade+NMS)	**0.919**	**0.938**	**0.929**

9.3 Comparison with Baselines

In this section, we compare Aryokee with three baselines: Linear SVM, RBF Kernel SVM and LSTM. All models are trained and tested on the same dataset. For SVM and Kernel SVM, since the input dimension of each sample is very high, we reduce the dimension using principle component analysis (PCA). In particular, we keep the top most PCAs that capture at least 98% of the variance. Since the amount of training data is too large, we cannot feed all of the samples into memory. Hence, we perform standard online hard negative mining [13, 17]. For the LSTM model, we use the same input as our CNN model, but construct a fully connected layer to reduce the dimensions before we feed the data into the LSTM core. Otherwise, an LSTM with high dimensional core size will require extremely heavy computation even with GPUs. This is not an issue for a CNN since it has only local connections as explained in the background section. Note that the fully connected layer and the LSTM are jointly trained in an end-to-end fashion.

We test these models under two different scenarios: (1) same-people same-environment, and (2) cross-people cross environment. In both cases, the training and testing datasets are the same as those used in the corresponding evaluation of Aryokee. The results are shown in Table 7. The table shows that for the same-people same-environment setting, Linear SVM, Kernel SVM and LSTM achieve F1 score of 0.658, 0.790 and 0.864, respectively, while our Aryokee achieves an F1 score of 0.958. However, when generalizing to the cross-people cross-environment setting, the F1 scores of the three baselines quickly drop below 0.2, while Aryokee keeps an F1 score of 0.929. These results show that while the baselines work well for simple scenarios, they cannot generalize. This emphasizes the benefits of Aryokee specific CNN-based model which captures complex spatio-temporal patterns in high dimensional data input. In contrast, traditional classifier and LSTM cannot automatically extract complex features from the data input.

Table 7. Comparison with Linear SVM, Kernel SVM and LSTM.

Method	Same People Same Environment			Cross People Cross Environment		
	Precision	Recall	F1	Precision	Recall	F1
Linear SVM	0.639	0.679	0.658	0.080	0.299	0.127
Kernel SVM	0.785	0.795	0.790	0.173	0.161	0.167
LSTM	0.843	0.886	0.864	0.207	0.148	0.172
Aryokee	**0.960**	**0.955**	**0.958**	**0.919**	**0.938**	**0.929**

9.4 Evaluation of Fall State Machine

Aryokee's state machine coordinates the fall and stand-up predictions to generate a state transition curve. Based on the curve, the ability to stand up after a fall and the fall duration can be inferred. Here we first explain our evaluation metrics and then present our quantitative and qualitative results. Note that we

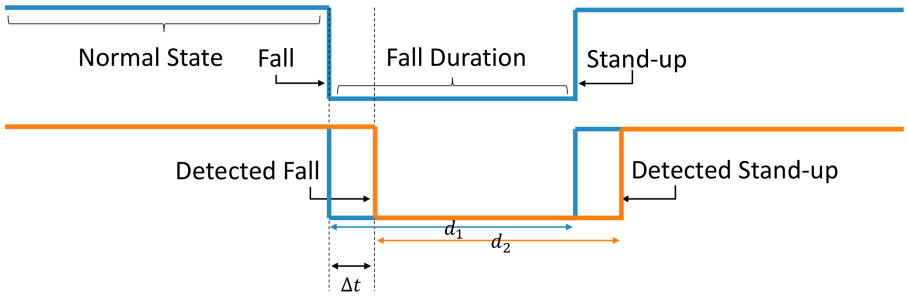
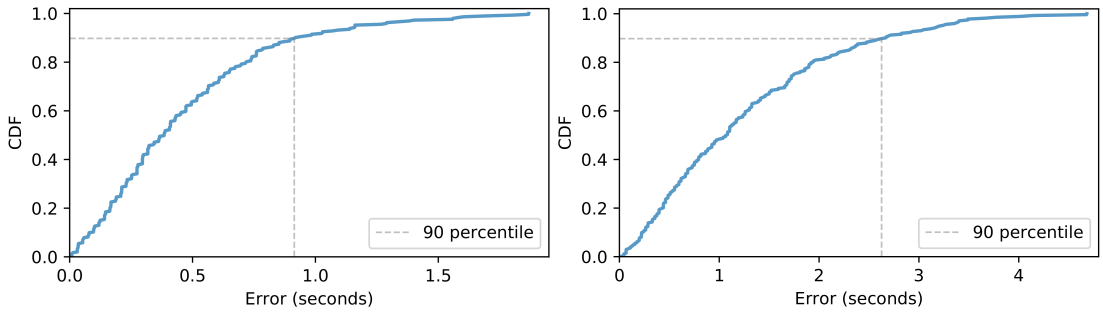


Fig. 11. State transition curve of a fall and the evaluation metrics. The blue curve shows the ground truth state transition and the yellow curve is Aryokee's prediction. Each falling edge and rising edge of the curve represent a fall and a stand-up event respectively. We measure the absolute offset in time Δt in detecting a fall and the error in estimating the fall duration $\Delta d = |d_1 - d_2|$.

use the same dataset for cross-people and cross-environment, but now evaluate our model on it for state monitoring.

Metrics. We evaluate two aspects of fall state monitoring: fall detection offset and duration of a fall. The idea is illustrated in Figure 11. We compute the absolute difference between the timestamps of the detected fall and corresponding ground truth as detection offset error Δt. Then we compute the duration of detected fall period d_2 and corresponding ground truth d_1. The duration estimation error is defined as the absolute error $\Delta d = |d_1 - d_2|$.



(a) CDF of the fall detection offset error.

(b) CDF of the fall duration estimation error.

Fig. 12. CDFs of the fall detection offset error and the duration estimation error.

9.4.1 Quantative Result: We evaluate Aryokee based on 293 falls in the testing set from the setting in Section 9.2.1, and the fall duration time ranges from a few seconds to more than 10 minutes. For the detection offset error Δt, Aryokee achieved an averaged error of 0.464 seconds and the 90 percentile error is 0.914 seconds. To show the distribution of errors, we plot the cumulative distribution function (CDF) over the detection offset errors in the test dataset in Figure 12a. For the duration estimation error Δd, the averaged error is 1.254 seconds and 90 percentile error is 2.626 seconds. The CDF of the duration estimation error is also shown in Figure 12b. The results show that Aryokee can monitor the state of falls accurately and reliability.

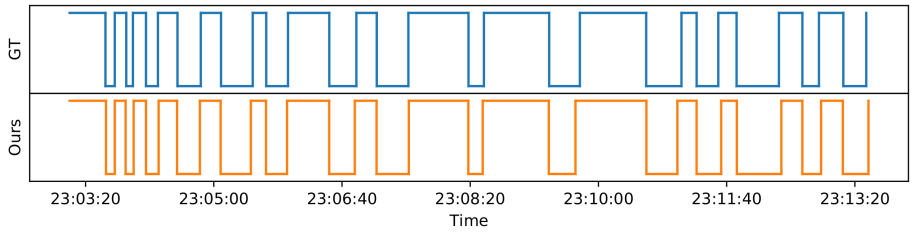


Fig. 13. An example of our fall state monitoring. The blue curve shows the ground truth while the yellow curve is Aryokee's prediction. Similar to Figure 11, the curve goes down every time a fall occurs and goes up when the person stands up. Our system monitors the states of all falls accurately.

9.4.2 Qualitative Results: For qualitative analysis, we ask a participant to perform 15 falls with random fall durations within a 10-minute experiment. Figure 13 shows the result of our state monitoring along with the ground truth state sequence. Aryokee accurately detects all state transitions, and the detected fall times and durations are highly aligned with the ground truth.

10 CONCLUSIONS

In this paper, we introduce Aryokee, a multi-function fall detection system that can detect falls, stand-up events, and fall duration using RF signals. The system uses an FMCW radio equipped with two antenna arrays to separate reflections from different locations in space. Our design extracts complex spatio-temporal information about body motion from RF signals using multiple convolutional neural networks governed by a state machine. The system is evaluated in a large-scale dataset involving 140 people performing 40 different activities and complex fall events in 57 different environments. Aryokee is highly accurate even when trained and tested on *different people and different environments* with a recall of 94% and a precision of 92%. The results show that our system generalizes across people and environments, works in the presence of other sources of motion, and deals with complex activities and fall patterns.

A COMPLETE LIST OF ACTIVITIES

A.1 Fall Activities

We list different types of fall activities collected in Table 8. We note that most activities involve basic variants as falling forward, backward, on position, and sideways.

Table 8. Fall patterns in our dataset.

slip	trip	loss of consciousness
loss of balance	from chair	with chair
from standing on chair	from standing on table	due to sprained ankle
due to kicking things	due to insufficient power	due to catching things
due to slippery walker	due to slippery crutch	due to failure jump
due to getting stricken	due to reaching out for things	due to lifting heavy furniture

A.2 Non-fall Activities

We enumerate the collected non-fall activities in Table 9.

Table 9. Non-fall patterns in our dataset.

eating	drinking	writing	using computer
watering flowers	ukulele	brushing teeth	standing
clapping	stretching body	walking	jogging
hula hooping	tai chi	yoga	sleeping on ground
stretching legs	squatting	picking up stuff	benching back
stairs	sitting on ground	tying shoelaces	lying on couch
air hockey	pingpong	pool	standing on hands
juggle	soccer	dancing	street dance
crawling	object falling	opening door	moving furniture
cooking	swinging leg	rolling on chair	taking a bow

ACKNOWLEDGMENTS

The authors thank the members of the NETMIT group for their insightful discussions, anonymous reviewers for their valuable comments and all the subjects for their substantial contribution to the data collection.

REFERENCES

[1] Stefano Abbate, Marco Avvenuti, Francesco Bonatesta, Guglielmo Cola, Paolo Corsini, and Alessio Vecchio. 2012. A smartphone-based fall detection system. *Pervasive and Mobile Computing* 8, 6 (2012), 883–899.

[2] Fadel Adib, Chen-Yu Hsu, Hongzi Mao, Dina Katabi, and Frédo Durand. 2015. Capturing the human figure through a wall. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 219.

[3] Fadel Adib, Zachary Kabelac, and Dina Katabi. 2015. Multi-Person Localization via RF Body Reflections.. In *NSDI*.

[4] Fadel Adib, Zachary Kabelac, Dina Katabi, and Robert C Miller. 2014. 3D Tracking via Body Radio Reflections.. In *NSDI*.

[5] Majd Alwan, Prabhu Jude Rajendran, Steve Kell, David Mack, Siddharth Dalal, Matt Wolfe, and Robin Felder. 2006. A smart and passive floor-vibration based fall detector for elderly. In *Information and Communication Technologies, 2006. ICTTA'06. 2nd*, Vol. 1. IEEE, 1003–1007.

[6] Moeness G Amin, Yimin D Zhang, Fauzia Ahmad, and KC Dominic Ho. 2016. Radar signal processing for elderly fall detection: The future for in-home monitoring. *IEEE Signal Processing Magazine* 33, 2 (2016), 71–80.

[7] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. 2016. Soundnet: Learning sound representations from unlabeled video. In *Advances in Neural Information Processing Systems*. 892–900.

[8] Petr Beckmann and Andre Spizzichino. 1987. The scattering of electromagnetic waves from rough surfaces. *Norwood, MA, Artech House, Inc., 1987, 511 p.* (1987).

[9] John Canny. 1987. A computational approach to edge detection. In *Readings in Computer Vision*. Elsevier, 184–203.

[10] Yung-Chin Chen and Yi-Wen Lin. 2010. Indoor RFID gait monitoring system for fall detection. In *Aware Computing (ISAC), 2010 2nd International Symposium on*. IEEE, 207–212.

[11] Federal Communications Commission. [n. d.]. Rules and Regulations for Title 47. *Federal Communications Commission* ([n. d.]).

[12] Jiangpeng Dai, Xiaole Bai, Zhimin Yang, Zhaohui Shen, and Dong Xuan. 2010. PerFallD: A pervasive fall detection system using mobile phones. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*. IEEE, 292–297.

[13] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. 2010. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence* 32, 9 (2010), 1627–1645.

[14] Jane Fleming and Carol Brayne. 2008. Inability to get up after falling, subsequent time on floor, and summoning help: prospective cohort study in people over 90. *Bmj* 337 (2008), a2227.
[15] Ajay Gadde, Moeness G Amin, Yimin D Zhang, and Fauzia Ahmad. 2014. Fall detection and classifications based on time-scale radar signal characteristics. In *Radar Sensor Technology XVIII*, Vol. 9077. International Society for Optics and Photonics, 907712.
[16] Ross Girshick. 2015. Fast r-cnn. *arXiv preprint arXiv:1504.08083* (2015).
[17] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.
[18] Nina Golgowski. 2017. Elderly Woman Strangled By Medical Alert Necklace After Fall. https://www.huffingtonpost.com/entry/medical-necklace-strangles-woman_us_56d75817e4b0871f60edbb47.
[19] Nicolas Grein and Hermann Winner. 1993. FMCW radar system with linear frequency modulation. US Patent 5,252,981.
[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[21] Yi He, Ye Li, and Chuan Yin. 2012. Falling-incident detection and alarm by smartphone with Multimedia Messaging Service (MMS). *E-Health Telecommunication Systems and Networks* 1, 01 (2012), 1.
[22] Jihoon Hong, Shoichiro Tomii, and Tomoaki Ohtsuki. 2013. Cooperative fall detection using doppler radar and array sensor. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on*. IEEE, 3492–3496.
[23] Chen-Yu Hsu, Aayush Ahuja, Shichao Yue, Rumen Hristov, Zachary Kabelac, and Dina Katabi. 2017. Zero-Effort In-Home Sleep and Insomnia Monitoring using Radio Signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 59.
[24] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*.
[25] Branka Jokanović and Moeness Amin. 2017. Fall Detection Using Deep Learning in Range-Doppler Radars. *IEEE Trans. Aerospace Electron. Systems* (2017).
[26] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099* (2016).
[27] Pekka Kannus, Harri Sievänen, Mika Palvanen, Teppo Järvinen, and Jari Parkkari. 2005. Prevention of falls and consequent injuries in elderly people. *The Lancet* 366, 9500 (2005), 1885–1893.
[28] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 1725–1732.
[29] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *The International Conference on Learning Representations*.
[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
[31] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[32] Tracy Lee and Alex Mihailidis. 2005. An intelligent emergency response system: preliminary development and testing of automated fall detection. *Journal of telemedicine and telecare* 11, 4 (2005), 194–198.
[33] Qiang Li, John A Stankovic, Mark A Hanson, Adam T Barth, John Lach, and Gang Zhou. 2009. Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. In *Wearable and Implantable Body Sensor Networks, 2009. BSN 2009. Sixth International Workshop on*. IEEE, 138–143.
[34] Yun Li, KC Ho, and Mihail Popescu. 2012. A microphone array system for automatic fall detection. *IEEE Transactions on Biomedical Engineering* 59, 5 (2012), 1291–1301.
[35] Brad Mager, Neal Patwari, and Maurizio Bocca. 2013. Fall detection using RF sensor networks. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on*. IEEE, 3472–3476.
[36] Roderick J McClure, Cathy Turner, Nancye Peel, Anneliese Spinks, Elizabeth Eakin, and Karen Hughes. 2005. Population-based interventions for the prevention of fall-related injuries in older people. *The Cochrane Library* (2005).
[37] Frank G Miskelly. 2001. Assistive technology in elderly care. *Age and ageing* 30, 6 (2001), 455–458.
[38] Muhammad Mubashir, Ling Shao, and Luke Seed. 2013. A survey on fall detection: Principles and approaches. *Neurocomputing* 100 (2013), 144–152.

[39] Hammadi Nait-Charif and Stephen J McKenna. 2004. Activity summarisation and fall detection in a supportive home environment. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, Vol. 4. IEEE, 323–326.
[40] John Hopkins Newsletters. 2015. Falls Cost U.S. Hospitals 34 billion in Direct Medical Costs. 100 (2015).
[41] Sameera Palipana, David Rojas, Piyush Agrawal, and Dirk Pesch. 2018. FallDeFi: Ubiquitous Fall Detection using Commodity Wi-Fi Devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 155.
[42] Alan Pierce. 2017. Walabot DIY Can See Into Walls. *Tech Directions* 76, 5 (2017), 8.
[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence* 39, 6 (2017), 1137–1149.
[44] Mark A Richards. 2005. *Fundamentals of radar signal processing*. Tata McGraw-Hill Education.
[45] Luis Ramirez Rivera, Eric Ulmer, Yimin D Zhang, Wenbing Tao, and Moeness G Amin. 2014. Radar-based fall detection exploiting time-frequency features. In *Signal and Information Processing (ChinaSIP), 2014 IEEE China Summit & International Conference on*. IEEE, 713–717.
[46] Laurence Z Rubenstein. 2006. Falls in older people: epidemiology, risk factors and strategies for prevention. *Age and ageing* 35, suppl_2 (2006), ii37–ii41.
[47] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.
[48] Frank Sposaro and Gary Tyson. 2009. iFall: an Android application for fall monitoring and response. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 6119–6122.
[49] Robert Steele, Amanda Lo, Chris Secombe, and Yuk Kuen Wong. 2009. Elderly persons perception and acceptance of using wireless sensor networks to assist healthcare. *International journal of medical informatics* 78, 12 (2009), 788–801.
[50] Renee Stepler. 2016. *Smaller share of women ages 65 and older are living alone: More are living with spouse or children*. Pew Research Center.
[51] Yonglong Tian, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Strong Parts for Pedestrian Detection. In *The IEEE International Conference on Computer Vision (ICCV)*.
[52] Yonglong Tian, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Pedestrian Detection Aided by Deep Learning Semantic Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[53] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).
[54] Paul Viola and Michael Jones. 2001. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. IEEE.
[55] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. 2016. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*. 613–621.
[56] Hao Wang, Daqing Zhang, Yasha Wang, Junyi Ma, Yuxiang Wang, and Shengjie Li. 2017. RT-Fall: A real-time and contactless fall detection system with commodity WiFi devices. *IEEE Transactions on Mobile Computing* 16, 2 (2017), 511–526.
[57] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*.
[58] 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*. ACM, 65–76.
[59] Yuxi Wang, Kaishun Wu, and Lionel M Ni. 2017. Wifall: Device-free fall detection by wireless networks. *IEEE Transactions on Mobile Computing* 16, 2 (2017), 581–594.
[60] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. 2015. From Facial Parts Responses to Face Detection: A Deep Learning Approach. In *The IEEE International Conference on Computer Vision (ICCV)*.
[61] Tong Zhang, Jue Wang, Ping Liu, and Jing Hou. 2006. Fall detection by embedding an accelerometer in cellphone and using KFD algorithm. *International Journal of Computer Science and Network Security* 6, 10 (2006), 277–284.
[62] Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. 2018. Through-Wall Human Pose Estimation Using Radio Signals. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[63] Mingmin Zhao, Yonglong Tian, Hang Zhao, Mohammad Abu Alsheikh, Tianhong Li, Rumen Hristov, Zachary Kabelac, Dina Katabi, and Antonio Torralba. 2018. RF-Based 3D Skeletons. In *Proceedings of the 2018 ACM SIGCOMM Conference*. ACM.

Received February 2018; revised May 2018; accepted September 2018