New Techniques for Real-Time Fall Prediction and Development of an Injury Prevention System

Emily Kamienski 1,1

1Massachusetts Institute of Technology

October 31, 2023

Abstract

This work describes new techniques for improving the performance of real-time fall prediction models
New Techniques for Real-Time Fall Prediction and Development of an Injury Prevention System

EMILY A. KAMIENSKI1, PAOLO BONATO2(Senior Member, IEEE), and H. HARRY ASADA1(Life Fellow, IEEE)

1Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: emilyk@mit.edu; asada@mit.edu)
2Department of Physical Medicine and Rehabilitation, Harvard Medical School, at Spaulding Rehabilitation Hospital, Charlestown, MA 02129 USA (e-mail: pbonato@mgh.harvard.edu)

Corresponding author: Emily A. Kamienski (e-mail: emilyk@mit.edu).

This work was supported in part by NSK Co. and the National Robotics Initiative under Grant 2133072.

ABSTRACT Falls may cause serious injuries to older adults, leading to a deteriorated quality of life. Real-time fall injury prevention devices are lacking, especially for the balance impaired population who rely on mobility aids. A key challenge to the reliability of such devices is real-time fall prediction, which is a time-critical decision making process. A fall must be predicted preemptively so that the system has sufficient time to deploy the injury prevention mechanism. Furthermore, the fall predictor must have a sufficiently low false positive rate for practical use. This paper presents three techniques for improved fall prediction and a fall injury prevention device for older adults with declining balance. We first collected a novel fall dataset from human subjects undergoing diverse loss-of-balance situations while using a mobility aid. Data obtained from these human subject tests contain diverse patterns of fall cases, yet both false negative and false positive rates must be very low; otherwise, the system cannot be adopted reliably. The first modeling technique uses multiple Long-Short Term Memory networks in parallel, each tuned to an individual pattern of fall data, yielding a higher sensitivity than a baseline model trained on all the fall data. Second, to reduce the false positive rate, another Long-Short Term Memory network is constructed to predict the time remaining before the fall prevention mechanism must be activated, so that if time allows, prediction may be delayed and additional data collected. Third, confounding cases are further analyzed using a metric of data deficiency, called the Lipschitz quotient. Additional data features that lower the Lipschitz quotients, increasing data predictability, are sought and incorporated into the original signals. Separately learning individual fall patterns, and delaying fall predictions using the time remaining prediction successfully increased the number of identified falls and lowered the false positive rate. Augmenting the dataset further improved performance, and the best model had a 97% identified falls rate at a 0.17% false positive rate. An LSTM based predictive model is implemented on a novel walker-type fall prediction and prevention prototype. The walker has a small footprint for improved maneuverability, and becomes untippable when it’s expandable legs are deployed in the event of a predicted fall. Thus, the older adult tethered to the untippable walker is protected from a fall. This promises immense benefits for future research on improving older adult wellbeing through real-time fall protection.

INDEX TERMS Fall prediction, Machine Learning, Mobility Aid, Fall Injury Prevention, Lipschitz Analysis, Eldercare, Assistive robots, Time-to-Event Prediction

I. INTRODUCTION

A. MOTIVATION

FALLS are the leading cause of injuries and hospital admissions among the elderly population. In the United States, approximately 3 million emergency room visits and over 800,000 hospitalizations of elderly people annually are due to injuries sustained from a fall [1]. These injuries can have deleterious long-term effects on the individual, as many suffer a decreased quality of life and reduction in independence following a fall [2]. Besides the individual hardships imposed, there is a significant economic impact as well since Medicare pays $29 billion annually in relation to non-fatal falls in the elderly population [3]. Also note that many fall injuries occur while using a walking aid - estimated at over 47,000 annually [4].

The overarching goal of the current work is to provide a fall-injury prevention system for those older adults with a balance impairment who need to use mobility aids in daily activities. This requires multi-disciplinary research, including mechanical design for developing an effective physical means
to prevent a hard ground-hit from occurring and advanced signal processing for monitoring the older adult, predicting a fall in real time, and controlling the injury prevention device.

B. INJURY PREVENTION DEVICES
There are few existing options for devices that respond in real-time to prevent a fall injury. One of the few available methods that are commercially available includes airbags packaged into wearable vests or belts, many of which were translated from similar products for motorcyclists and equestrian riders [5] [6] [7] [8] [9]. The air bags deploy when a fall is detected, protecting critical areas such as the hips and in the case of the vests, the upper body and head. Note hip air bags also do not prevent fractures in other parts of the body besides the hips and offer no head protection.

Among traditional hip protectors, there is some evidence that they slightly decrease the chance of hip fractures, but could also potentially increase the chance of a pelvic fracture, and more critically, previous studies showed low rates of long term adoption by users [10]. Note also that hip protectors do not offer any benefits in terms of aid in balancing, physical support, nor fall prevention, but only serve to protect from injury in the event of a fall.

Another limitation to those wearable airbags is that they do not provide mobility aids. Balance-impaired older adults must use some form of mobility aids, like canes and walkers. The current work aims to integrate an injury prevention mechanism into a mobility aid to achieve both mobility aid and injury prevention.

C. PRIOR WORK IN FALL PREDICTION & INJURY PREVENTION
While many fall prediction algorithms exist, and there are multiple open source datasets available, there are few data trials in which a mobility aid was used [11] [12] [13] [14] [15] [16]. For perspective, almost a quarter of the U.S. population 65 years and older use a mobility aid and over 11% use walkers [17]. It is expected that the use of a mobility aid would alter signals used to predict a fall since posture and gait are affected, and additional reaction forces can act on the body via the hands holding the walker.

Furthermore, many of these predictors are used purely for observation and sending alerts, or there is no validation with a real-time injury prevention system. This is significant because systems that generate a real-time physical response will have more stringent requirements for their fall predictors. Real-time fall prevention systems will need a sufficiently low false positive rate, since unnecessary physical responses to a false fall prediction will disturb daily life. Additionally, adequate lead-time will be necessary to allow the physical response to occur in time to prevent injury. Note that lead-time requirements will be determined by the deployment time of the physical response.

D. APPROACH TO CHALLENGES
The current work aims to fill the gaps among those existing technologies. Specifically this work focuses on the segment of the elderly population that is balanced impaired, and therefore at a higher fall risk, and requires a walking aid. To address the lack of fall predictors that are designed to be used for mobility aid users, this work constructs the machine learning based prediction model using data of people falling with a walker, one of the most common mobility aids among older adults [17].

Falls can be very diverse and occur over different time periods, varying from almost instant instability in the case of a trip, to a more prolonged instability period for falls caused by improper weight shift [18]. This paper aims to identify these diverse fall types and improve the identification rate of all types of falls by training separate models for faster and slower falls.

Additionally, to address the strict requirements of real-time injury prevention systems, this paper introduces a technique aimed at reducing the false positive rate. This is done by delaying the prediction when in the early stages of instability, to reduce the likelihood of confusing a safe motion with that of a slow fall.

Lastly, data often have limited predictive power for certain phenomenon, independent of the modeling methods. No matter the techniques employed in modeling the data, features selected may not be able to adequately capture the distinguishing characteristics of different phenomena, e.g. fall and nonfall motions in this case. The data's predictive power can be evaluated with the use of the Lipschitz quotient.

E. GOALS & PAPER OUTLINE
This paper presents three techniques for improving the performance of a fall prediction classifier to make it suitable for use with a new fall injury prevention device that also functions as a mobility aid for the balance impaired elderly population.

Here is a summary of the novelty of this work:
1) Collect fall data with use of mobility aid.
2) Introduce technique to identify diverse fall types - by separately learning characteristics of fast and slow falls.
3) Present a technique for predicting the time remaining until a fall, so that the false positive rate can be reduced by observing more data and making more cautious predictions until the deadline of the fall/no-fall decision.
4) Analyze a given dataset using the Lipschitz quotient, and augment the data by incorporating additional signals to remedy the error-prone, confounding fall cases.
5) Design and build a new fall prevention system by integrating an injury prevention mechanism into a mobility aid.

The outline of the rest of this paper is as follows. Section II describes human subject tests of falling while using a mobility aid and performing motions seen in activities of daily living (ADL). The three methods for achieving high fidelity fall prediction will be addressed in Section III-A, and their
prediction performance and discussion will be presented in Section IV. The design, construction, and testing of a novel untippable walker that can prevent fall injuries is in Section V. Lastly, the limitations of this work are acknowledged in Section VI and concluding remarks are in Section VII.

II. DATA COLLECTION & PREPARATION

Over a hundred videos of real-life elderly falls in two long term care facilities were reviewed from an online database [19]. The falls were often unique in the exact falling mode and posture the body took while falling. This indicates it would be very difficult or impossible to develop an analytical model of the dynamics of a fall that could generalize well enough to cover all possible falling modes. So, as most works in this area have concluded, it is necessary to take a more data driven approach to capture the complex and underlying signal trends that characterize a fall. To generate fall data taken while using the walker frame, falls were realistically reproduced and simulated with a dozen human subjects.

Various data of reenacted falls have been created [20] [21] [22] [23]. While these data are valuable, they are fall cases without the use of supporting devices. Additionally, while there are some data bases of real-life falls, many are not public. Falling patterns and features of measured signals are highly dependent on the walking aids they use. This necessitates the collection of new fall data and a prediction model for a specific type of walking aid.

In order to know when the injury prevention device needs to respond, the user’s real-time stability state must be observed. Therefore, key signals associated with the user’s stability must be captured by sensors mounted on the device or wearable support interface. Since two-thirds of a person’s mass is concentrated in the torso, the pose of this segment of the body largely influences the center of mass (COM), making this a key area to monitor. Furthermore, since falling is a dynamic event, it is expected that instantaneous kinematic parameters, such as acceleration and angular velocity of the user’s torso, will contain important clues on their stability level and likelihood of falling.

This signal information must be reliably available to provide constant monitoring of the user’s stability and predict falls in their early stages. Sensing modalities such as cameras, that are mounted on the robot, are at high risk of occlusion and cannot obtain a suitable viewing angle due to the close proximity of the support frame to the user. Therefore, sensors that directly measure changing dynamics and positions of the user are needed. A torso harness’s close conformity to the human body provides an ideal mounting location for this type of sensor, such as inertial measurement units (IMU’s).

A. EXPERIMENTAL DATA COLLECTION

Data were collected while human subjects were caused to fall from an external disturbance and when they reenacted common fall types. Data was collected according to the IRB protocol #2104000353 approved by the Massachusetts Institute of Technology Committee on the Use of Humans as Experimental Subjects. Twelve people participated in this study, consisting of 5 females and 7 males. Ten were in their twenties and two were above age 65, one female and one male. MIT’s Clinical Center for Translational Research Center was used to collect data. Participants pushed a walker frame while wearing a medical grade harness attached to an overhead rail system, as shown in Figure 1. The harness did not impede normal walking movements and the rail was moved along with the participant so it did not impose any resistance on the participant.

![Figure 1. Human subject data recorded at MIT’s Center for Clinical and Translational Research. Participants wore harness attached to overhead rail system.](image1)

![Figure 2. Location of IMU](image2)

An Mbientlab MetaMotionR 3-axis IMU was used to record acceleration and angular velocity data. The IMU was attached to the center of the harness at navel height as shown in Figure 2. Videos of all trials were also recorded.

Falls were simulated by reenacting the most common trends and fall initiations observed in the database of elderly falls [19]. Elderly falls usually occur while walking, and the primary modes of fall initiation are improper weight shifting, followed by tripping [24].

The participants performed 5 different types of test cases.
1) Walking
2) Performing motions associated with activities of daily living (ADL)
3) Being pushed to induce a loss of balance
4) Simulating tripping over one’s own foot
5) Simulating losing one’s balance by improper weight transfer
At least 5 instances in the three loss-of-balance cases described above were tried for each subject.

The participants were shown a video of an elderly person walking with a walker, and instructed to walk in a similar manner. This mostly entailed reducing their walking speed from their normal speed. To obtain data of normal walking motions and common actions seen in daily life, data was recorded while participants walked with the walker and pretended to reach for and pick up objects.

To reproduce the sudden tripping fall case, the participants were unexpectedly pushed from behind in the center of the lower back forwards or to the side while walking in a straight line with the walker. The simulated falls caused by tripping were performed by first walking several steps before tripping over their own feet. While performing falls, the test subjects attempted to take several recovery steps before completely falling to match preventative measures similar to those observed in real falls. Since the participants were wearing a harness, the height they could fall was limited and they never contacted the ground.

B. DATA PREPROCESSING

The data was sampled at 100 Hz and noise was removed using a low pass filter. From the IMU, there are 6 features total in a sample, including the 3-axis accelerometer and 3-axis gyroscope readings. Each feature was mean centered and scaled to have unit variance. The data was filtered with a 2nd order Bessel filter with a 5 Hz cutoff frequency. Examples of typical signals from each fall type are shown in Figure 3.

C. DATA SEGMENTATION AND ANNOTATION

The data was labeled by certain stages of the motion. The unstable region consists of normal movements. The imbalance onset marks the start of the unstable region. For the pushed fall, the imbalance onset is when the person’s hand makes contact with the participant’s back. For the simulated tripping fall, the imbalance onset is when one foot first crosses the path of the opposite foot. Lastly, for the simulated improper weight transfer, the imbalance onset is when the top half of the body started to tilt, or when the center of the hips were no longer positioned over the feet. After some point in the unstable region, the individual cannot be prevented from falling. In this work the start of the descent phase of the fall is when the harness straps become taught.

The fall data was extracted by watching the videos with falls and noting the timestamps of the imbalance onset, and the start of the descent phase, as shown in 4. The data was reformulated as a supervised learning problem by labeling sliding time windows of the data as shown in Figure 5. Therefore, the model has between the imbalance onset and some time before the start of the descent phase point, depending on the speed of response of the physical injury prevention device, to predict the fall.

The time period classified as a fall, the period directly before the fall, and the ADL motions were divided into many overlapping windows of length \( n_w \). The length of the time window was a hyperparameter that is tuned. Each window was shifted by an interval equal to the sampling period. A window is equivalent to one sample and determines how much previous data is included in classifying a slice of time as a fall or not a fall. Therefore, many more samples were generated for training the Long-Short Term Memory (LSTM) model than if just the original recorded sequences were used. All the windows in the stable region are labeled as “not a fall”, or \( y = 0 \), and all the windows in the unstable region are labeled a “fall”, \( y = 1 \). The number of fall training samples \( h \) is a function of the window length and the length of the fall cases.

\[
h = \sum_{i=1}^{m} n_{ei} = \sum_{i=1}^{m} \frac{(\delta_{ti} - n_w)}{\tau} (1)
\]

where \( m \) is the number of recorded falls, \( n_{ei} \) is the number of samples in the \( i^{th} \) fall case, \( \delta_{ti} \) is the length of the \( i^{th} \) fall, \( \tau \) is the sampling period, and \( n_w \) is the window size.

The dataset is naturally heavily imbalanced for this problem - there are far fewer fall samples than non-fall samples because falling is a brief event. As such, the dataset was augmented by mirroring the fall data about the sagittal plane. For example, the mirrored signals of a fall to the left side would look like a fall to the right side. Data was only mirrored about the sagittal plane as it is assumed that falls to the left and right would have analogous signals, where this would not be the case for forward and backward falls.

The distribution of the data is shown in Figure 6. The dataset distribution is 220,633 negative sample windows and 14,618 positive sample windows.

III. BASELINE MODEL & METHODS FOR HIGH-FIDELITY PREDICTION

Simultaneously achieving a high rate of fall identification and a low false positive rate is a challenge for real-time prediction of a fall. Here we introduce the baseline model, and propose three methods which are effective specifically for real-time fall prediction.

A. BASELINE MODEL

The baseline model consisted of a standard Long Short Term Memory (LSTM) network that classifies a stream of data as a fall or not a fall. Windows/samples are input in batches of size \( n_b \). Each time slice of data \( z_i \) in a window \( x_i \), is input into the LSTM cell sequentially. A time slice has \( n_f = 6 \) features. The softmax activation function outputs a final prediction probability of the class value which is then rounded to give a value of 0 or 1 for the fall prediction. Binary cross entropy (BCE) is used as the loss function.

B. SEPARATE LEARNING OF FAST & SLOW FALLS

There can be a lot of variation in falls. Some falls occur more slowly, such as that caused by an improper weight shift [18],
FIGURE 3. Representative examples of filtered acceleration and angular velocity data for each fall type. The vertical red dotted line corresponds to the imbalance onset. Data prior to the line is in the stable region and labeled as not a fall, and data after the red line is in the unstable region and labeled as a fall. The data is cut off at the start of the descent phase. The simulated weight transfer is a slower fall and has differing signal patterns.

FIGURE 4. Key frames in a fall sequence used to label data. Data is not used once the straps become taught at the start of the descent phase. The data prior to the imbalance onset is labeled as not a fall and is in the stable region. Between the imbalance onset and start of the descent phase is the unstable region and labeled as a fall.

and other falls occur more rapidly, such as that caused by pushing and tripping disturbances. As can be seen in Figure 3, the signal patterns are significantly different between the faster and slower falls, although both must be judged as a fall. With the limited dataset, it can be challenging to train one predictor to accurately detect very distinct fall types. In this work we look at slow and fast falls separately, and train separate LSTM models for identifying each. To this end the training data are split into two datasets. One is for detecting a fast fall, where the target output is 1 only for fast falls and 0 for elsewhere, including slow falls. The other LSTM is for detecting only slow falls where the target output is 1 for slow falls and 0 for the other samples. This labeling scheme ensures each LSTM network is still presented with all possible data types during training. The predicted outputs from the two LSTMs are combined as disjunction (logical OR).

C. PREDICTION OF TIME REMAINING UNTIL START OF THE DESCENT PHASE

Fall prediction is a time-critical decision. Before the start of the descent phase, a decision must be made so that the fall prevention mechanism can be deployed in time. The decision must not be made in haste, however. Both false negative and false positive rates must be low. In the literature obtaining low
false positive rates below 1%, or sensitivity scores above 99%, has been hard to achieve [15] [11]. Impractically high false positive rates are a major barrier when used in conjunction with a physical means to prevent a fall or reduce risk of injury. This section presents a new method for reducing the false positive rate in time-critical decision making.

Although fall prediction is time-critical, a decision must be made only by the start of the descent phase. If a fall is predicted in an early stage of unstable behavior, there will be time to confirm the predicted fall. One more cycle of data can be acquired and processed to confirm the predicted fall. This is especially needed for dealing with slow falls where an elderly person loses balance for a short time but recovers from it. It may become a false positive if the predictor makes a decision based on an early stage unstable behavior alone. The early stages of the slow falls can resemble the non-fall ADL motions. Therefore, it is logical to wait until the signal pattern is more decidedly a fall before making a decision.

To accomplish this we introduce another LSTM predictor to estimate the time remaining \( \hat{y}_{TR} \) until the start of the descent phase, as defined in section II. Thus if the time remaining predicted is sufficiently long, \( \hat{y}_{TR} > \text{thresh}_{TR1} \), the fall risk decision can be delayed until after another cycle of data collection and prediction is completed. If the predicted time remaining is below the threshold \( \text{thresh}_{TR1} \), then the final decision must be made without taking new data. It can examine a series of previous predictions over a "look-back period" \( lb \) until the current time step. Specifically, these past predictions are averaged over the look-back period, and if this average is greater than \( \text{thresh}_{TR2} \), then a fall is predicted. These more cautious decision strategies can be applied as long as the system knows how much time remains. The additional LSTM that estimates the time remaining enables these strategies.

While the concept of predicting the time until an event has been reported in [25] [26] [27] [28] [29] [30], predicting the time remaining until a fall appears to be unique. Most previous applications are for a much longer time horizon, predicting the remaining usable life of machinery or in survival analysis, the time until death of a living organism [27] [26] [25]. In those prior works, the objective is to estimate some type of remaining time, while our case uses the time estimate for conducting confirmation of a decision or making more cautious decisions in order to reduce the false positive rate.

Also in our work, the time remaining until a fall could be on the order of 100’s of milliseconds, while the prior works are on the order of days and months.

D. ASSESSING SENSOR DATA DEFICIENCY VIA LIPSCHITZ QUOTIENT

Even after the two methods described above are applied, there may be some confounding situations in the data which are difficult to distinguish between a fall and not a fall. Input signals from a wearable sensor may be unable to capture a critical feature of the body movement. This can result in a situation where two sets of input signals are very close, but they have different target outputs, one fall and the other non-fall. In other words, the distance between the two sets of input signals is very small, while the target outputs are far apart. Such a situation can be detected with the use of Lipschitz analysis [31] [32] [33].

As illustrated in Figure 7, a predictor can be represented as
a nonlinear map, \( f : X \mapsto Y \), from a vector in an input space, \( x_i \in X \), to an output space, \( y_j \in Y \).

Consider a distance between any arbitrary two points in the input space, say points \( x_i \) and \( x_j \), and their correspondent outputs, \( y_i \) and \( y_j \). Compute the ratio of the output distance to the input distance, called the Lipschitz quotient:

\[
q(i,j) = \frac{|y_i - y_j|}{|x_i - x_j|}
\]

where \( x_i, x_j \in X, y_i, y_j \in Y, y_i = f(x_i) \), and \( y_j = f(x_j) \). If this quotient is especially high for a pair of inputs \( x_i, x_j \in X \), it is very difficult to predict the correct outputs from the given sensor signals.

Suppose that data from a suite of sensors do not span the entire input space. Instead, the sensor space is a subspace of the entire input space, \( X' \subset X \). In other words, the sensors cannot capture a certain signal, which is necessary for making the correct prediction. If this missing signal determines the output, a one-to-multiple-point correspondence occurs between the dimension-deficit sensor space and the output space. See Figure 7-(B). There exists at least a pair of points in the dimension-deficit sensor space that are mapped to different points in the output space. This implies that the Lipschitz quotient is very large.

This represents a confusing situation; there is no way the output can be predicted from the deficit sensor information. This confounding case always occurs somewhere in the dimension-deficit input space if some variables pertinent to prediction are missing in the input space.

When a high Lipschitz quotient is found at a certain region in the input space, additional information must be sought to augment the input sensor space. As illustrated in Figure 7-(C), a third axis must be added to the input space so that two confounding points, say points \( i \) and \( j \) in the figure (C), can be separated along the new axis. With this new variable along the added axis, the Lipschitz quotient can be lowered. In the following we will find such additional variables in the context of fall prediction with a wearable sensor.

E. MODEL TRAINING & COMBINING PREDICTIONS

The first of our new methods implemented was the separate LSTM models for predicting fast and slow falls. During training the baseline LSTM network is presented with all the data. For the second architecture with separate fast and slow fall predictors, each LSTM network must still be presented with all possible data that could see in real-time. Thus this data is labeled as follows. For the fast fall predictor, the data is labeled as \( X_{train, fast} = \{ADL/Walk \mapsto 0; X_{slow} \mapsto 0; X_{fast} \mapsto 1\} \) and likewise for the slow fall predictor, \( X_{train, slow} = \{ADL/Walk \mapsto 0; X_{slow} \mapsto 1; X_{fast} \mapsto 0\} \). Therefore, the fast fall predictor only learns to identify the fast falls, and the slow fall predictor only learns to identify the slow falls.

The second method is that represented in Algorithm 1 and Figure 8, which adds in the time remaining prediction after the slow fall predictor. The output of the slow fall predictor trained on the dataset \( X_{train, slow} \) for the current data sample \( x_i \) is \( \text{Confid}(x_i; X_{train, slow}) \), the probability of a slow fall. The time remaining predictor is only used in the case when there is a positive prediction from the slow fall predictor (line 3) and a negative prediction from the fast fall predictor (line 6). When the fast fall predictor makes a positive prediction, the final prediction is taken as a fall since time is limited in that case due to the faster fall dynamics (line 12).

**Algorithm 1 Combined Prediction**

**Input:** \( x_i \)

**Output:** \( \hat{y}_{final, t} \)

1. Make predictions for current timestep:
2. \( \hat{y}_{fast, t} = \text{Confid}(x_i; X_{train, fast}) \geq \text{thresh}_{fast} \)
3. if \( \hat{y}_{fast, t} = 0 \)
4. \( \hat{y}_{slow, t} = \text{Confid}(x_i; X_{train, slow}) \geq \text{thresh}_{slow} \)
5. \( \hat{y}_{TR, t} = \text{Pred}(x_i) \)
6. if \( \hat{y}_{slow, t} = 1 \) and \( \hat{y}_{TR, t} \leq \text{thresh}_{TR1} \)
7. \( \hat{y}_{final, t} = \frac{1}{lb} \sum_{k=t}^{t+lb} \text{Confid}(x_k; X_{train, slow}) \geq \text{thresh}_{TR2} \)
8. else
9. \( \hat{y}_{final, t} = 0 \)
10. end if
11. else
12. \( \hat{y}_{final, t} = 1 \)
13. end if

Training the model for predicting the time remaining until the start of the descent phase is a regression problem. For each fall trial, the time remaining until the start of the descent phase is calculated for each time step (which is at most 1.5 s), and the data from the ADL and walking trials are labeled as 5. This data is then scaled to be between 0 and 1. Weighted mean squared error (W. MSE) is used as the loss function for the time remaining prediction. Because of the imbalanced dataset, errors for the non-fall target values were weighted by 0.3. This reduces the penalty on missed predictions for nonfalls and places more emphasis on predicting the time remaining after the imbalance onset, or in the unstable region.

The parameters for each architecture are shown in Table 3 in the appendix.

IV. PREDICTION RESULTS

A. PERFORMANCE METRICS

1) Identified falls rate

The identified falls rate is the rate of falls correctly identified out of all fall cases. Only a single window in the unstable region shown in Figure 5 must be identified for that particular fall to be correctly predicted.

2) Sensitivity & Specificity

The sensitivity is equivalent to the true positive rate. The specificity is the rate of correctly identified non-fall windows, or test samples, which is equal to (1 - False Positive Rate). All of the windows in the stable region shown in Figure 5

\(^1\text{Algorithm 1, line 7: The previous confidence values up to } t - lb \text{ are stored}\)
would have to be identified for each trial to achieve a perfect specificity score.

These metrics are also appropriate for naturally unbalanced datasets, since falling is a rare event, and the percentage of positive, fall cases, is about 5%.

3) $F_{\beta}$ Score

Additionally, the maximum $F$-$\beta$ scores are calculated for each architecture.

$$F_{\beta} = \frac{1 + \beta^2 \cdot \text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \quad (3)$$

The precision is the number of true positives over the total predicted number of predicted positive values. The recall in this case is the same as the sensitivity.

When $\beta = 1$, $F_{\beta}$ is equal to the F1 score. The F1 score is the weighted harmonic mean of precision and recall. The F2 score gives more weight to the recall score whereas the F0.5 score gives more weight to the precision score. The selection of the score depends on the user’s preference. On one hand all the falls should be identified, but the false positive rate must be sufficiently low otherwise the system will be impractical.

4) Precision-Recall Curve

The Receiver Operating Characteristic (ROC) curve is a standard way to show the performance of one model at different discrimination thresholds, or hyperparameter values. Because this dataset is imbalanced, a precision-recall curve is used rather than a traditional sensitivity vs. specificity ROC curve to compare the different architectures.

For the baseline model, the threshold for a positive fall prediction was varied. For the separated falls architecture, this threshold is independently varied for the slow and fast fall predictors. For the separated falls with time remaining architecture, this fall threshold is varied as for the separated falls architecture, and the time remaining threshold $thresh_1$, $thresh_2$, and look back $lb$ parameter values are also swept through different values. All hyperparameter values searched are specified in Table 4 in the Appendix. Note that a look back value of $lb = 1$ indicates that only the prediction at the current time step is used as the final prediction.

5) Lead Time

Other performance metrics are the prediction time $t_p$, and more importantly, the lead time $t_l$ before the start of the descent phase. The prediction time is measured from the onset of imbalance to when a window is first correctly classified as a fall. This time metric can be quantified by noting the first true positive prediction for a given fall case. Similarly, the lead time is the time from when the fall was first predicted to the start of the descent phase. The objective is to maximize this time, or minimize the prediction time to allow maximal time for a physical injury prevention response.

B. PREDICTION RESULTS - ORIGINAL DATA FEATURES

Figure 9 shows the precision-recall curve for the baseline architecture and the two new proposed methods at different parameter values when the prediction is repeated every 50 ms. For clarity, for each unique recall score, only the models with the highest precision scores are shown in the plots. The separated falls and separated falls with time remaining prediction achieve better combined precision-recall values than the baseline architecture.

The max $F$-$\beta$ scores were selected for each architecture type. The $F$-$\beta$ scores shown in the top part of the original dataset section of Table 1 indicate that the separated fast and slow fall architecture and the architecture with the time remaining prediction outperformed the baseline model over the range of hyper parameters tested.

Table 2 shows the highest identified falls rate achieved at less than 1% false positive rate for each architecture. The separated falls architecture on the original data performed significantly better and adding the time remaining further reduced the false positive rate.

C. PREDICTION RESULTS - WITH EXTRA DATA FEATURES

As the models were trained extensively with different hyperparameters, it is possible that the 6 IMU data features selected are insufficient for identifying the fall. Therefore, the
Lipschitz quotient is used to assess alternative data features that could improve performance over the original features selected. The additional features selected consisted of the original 6 IMU features (acceleration and angular velocity in 3 axes each) integrated over the past 500 ms. This is equivalent to the linear velocity and change in angular position over the past time window, respectively. This increases the total number of features to 12. A histogram of Lipschitz quotients show in Figure 10 indicates that the addition of these new data features does indeed lower the mean of the Lipschitz quotient by a significant percentage, indicating higher data predictability.

Figure 11 compares the precision-recall curves for the separated falls architecture and the architecture with time remaining prediction with and without the extra data features validated with Lipschitz theory. Adding the extra features clearly pushes the result into a more favorable region of the precision-recall curve. Additionally, the lower parts of Table 1 show the results with the extra features. Each architecture outperforms its counterpart with the original dataset, which is consistent with the Lipschitz analysis.

On the dataset with extra features, the F0.5 and F1 scores increase, but the F2 score decreases slightly. However, this did not impact the total number of identified falls in the test set, so it is not a concern.

Using the dataset with additional features improved the identified falls rate and false positive rate of the baseline model over the previous best model by 6% and 0.07%, respectively. Adding the separated falls increased the identified falls rate by over 3% from the new baseline. The time remaining prediction further decreased the false positive rate.

For the best architecture, the separated falls with extra features, the only missed fall of the 31 trials in the test set was a simulated tripping over the foot trial. Therefore, it had a 97% identified falls rate at a 0.17% false positive rate.

<p>| TABLE 1. Max F-beta scores obtained from parameter sweep of each architecture. For the original features the scores improve with the separated falls and time remaining techniques. Adding the extra features verified with Lipschitz analysis improved the baseline scores above any previous score and the F1 and F0.5 scores are only marginally improved by using the separated falls and separated falls with time remaining on the dataset including the extra features. |</p>
<table>
<thead>
<tr>
<th>Architecture</th>
<th>F2 Score</th>
<th>F1 Score</th>
<th>F0.5 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.669</td>
<td>0.643</td>
<td>0.645</td>
</tr>
<tr>
<td>Sep. Falls</td>
<td>0.776</td>
<td>0.725</td>
<td>0.738</td>
</tr>
<tr>
<td>Sep. Falls with TR</td>
<td>0.767</td>
<td>0.730</td>
<td>0.749</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iden. Falls Rate</th>
<th>FP Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>39.4%</td>
<td>0.15%</td>
</tr>
<tr>
<td>Sep. Falls</td>
<td>87.9%</td>
<td>0.82%</td>
</tr>
<tr>
<td>Sep. Falls with TR</td>
<td>87.9%</td>
<td>0.75%</td>
</tr>
</tbody>
</table>

<p>| TABLE 2. Identified falls rate at less than 1% false positive rate for each architecture. The separated falls architecture increases the identified falls rate (especially for the original features) and the time remaining further lowers the false positive rate. The baseline model improves significantly by using the integrated features. |</p>
<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iden. Falls Rate</th>
<th>FP Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>93.9%</td>
<td>0.53%</td>
</tr>
<tr>
<td>Sep. Falls</td>
<td>97.0%</td>
<td>0.29%</td>
</tr>
<tr>
<td>Sep. Falls with TR</td>
<td>97.0%</td>
<td>0.17%</td>
</tr>
</tbody>
</table>

D. PERFORMANCE ON DIFFERENT TRIAL TYPES

Figure 12 shows the lead time for each fall type for the best model, the separated falls with time remaining architecture trained on the data with extra features. For the best model with a 97% identified falls rate and 99.83% specificity the average lead time for all falls was 250 ms or more. The simulated weight transfer, the slow fall type, had the most variability in the resultant lead time. Seven of the false positives were in the ADL trials and 4 were during walking trials. Most of the false positives are during the ADL trials. The ADL trials are most likely to resemble a fall since they involve bending over and reaching motions.

E. DISCUSSION

Overall, the separated falls and separated falls with time remaining techniques successfully increased the number of identified falls and decreased the false positive rate, respectively.
FIGURE 10. Lipschitz histogram comparing the original slow and fast fall datasets with 6 IMU features to the datasets with the additional integrated features. The mean of the Lipschitz quotient for the dataset with extra features is shifted lower, indicating higher data predictability.

Of the three techniques discussed, using additional data features, validated with the Lipschitz quotient technique, had the largest impact on performance. Note that more features are not always better as at some point increasing the number of features being used will hurt performance.

There are several hyperparameters specific to this problem. This includes the length and overlap of the time windows. Shorter time windows make faster prediction times/longer lead times possible. However, short window sizes may not be long enough to characterize the trends that signify a fall, which mean fewer fall windows will be correctly predicted during a given fall and there will be more incorrectly classified “stable” windows. In previous work a window size of 500 ms was found to produce the best identified falls rate and false positive rate [34]. Therefore a window size of 500 ms was selected for this work. This window size also produced the best results with the aforementioned performance metrics. Additionally, how frequently a prediction is made is also a hyperparameter that will affect the identified falls rate and false positive rate. In this work the prediction rate was 50 ms for feasibility on an embedded device.

V. DESIGN, IMPLEMENTATION, AND TESTING OF A FALL INJURY PREVENTION SYSTEM

A. INJURY PREVENTION STRATEGY - RAPID & STABLE EXPANSION OF BOS

We aim to develop a fall injury prevention system specifically designed for the balanced-impaired elderly population. This population is at a higher fall risk as determined by traditional clinical tests and require the use of a walking aid. As addressed previously, existing fall injury prevention devices, such as airbags, do not provide those older adults with support while walking. This device aims to provide mobility aid and prevent injury from falls.

Our strategy is to use a standard walker and upgrade it to a fall injury prevention system. Specifically, we aim to make a walker untippable and, in case of a fall, suspend the human body with a cable connecting the human to the frame of the untippable walker. To prevent a walker from tipping, the Base of Support (BOS) of the walker frame is enlarged as shown in the right of Figure 13. This enlarged BOS, however, does not fit a home environment; it does not go through a narrow doorway, does not fit in a confined space, such as a bathroom and a closet. The standard-size walker on the left of Figure 13 can meet the doorway width constraint and is usable in a bathroom. Nonetheless, the walker BOS is too small to prevent tipping from occurring. To resolve this conflicting requirement between stability and maneuverability of standard walkers, we propose an expandable BOS mechanism. As detailed later, the walker is equipped with auxiliary legs that can be deployed in a short time to expand the BOS.

FIGURE 11. Precision-recall curve comparing the separated falls with time remaining architecture with the original data features and extra data features informed by Lipschitz.
The user wears harness shorts consisting of a pair of spandex shorts with a built-in load bearing harness that attaches to the walker frame via cables, and can support the body in the event of a fall. An IMU sensor unit is also attached to the harness shorts. A battery is used to power the electronics and servo motors so that the robotic walker is completely wireless.

B. MAJOR COMPONENTS

1) Auxiliary Legs
   The rapid expansion of the BOS can be achieved with use of spring-loaded auxiliary legs that can be unfolded in a short time. As shown in Figure 14, each auxiliary leg is stored on a leg of the walker, where the spring is fully compresses. As soon as the leg is unlocked by a servo latch, the auxiliary leg swings down and contacts the floor. With ratchet gears and pawls at the joint of the auxiliary leg, the leg is prevented from bouncing back and bears the load. We have found that this vertical deployment of auxiliary legs is advantageous in the following aspects:
   - By exploiting gravity the leg deployment is accelerated;
   - In case an obstacle is placed on the floor, the descending leg is stopped at the height of the obstacle, adapting to the uncertain environments; and
   - The auxiliary leg can be retracted and stored along the leg of the walker, fitting the limited space.

   The length of the legs were determined as to prevent tipping of the walking in the event of a fall by a 90 kg person. The force diagram is shown in Figure 14. The deployment of the auxiliary legs triggers brakes on the two rear walker wheels to prevent the walker from sliding during a fall. This last feature was implemented after getting feedback from clinicians at the Spaulding Rehabilitation Hospital in Charlestown, MA.

2) Harness Shorts
   The modified walker frame is intended to support the entire weight of a fallen person. Therefore the user needs to attach to the frame. This is accomplished by embedding a harness in an otherwise normal pair of pants [35]. The custom harness is sewn into a pair of spandex shorts such that it rests comfortably against the skin. The harness shorts are worn under ordinary pants, making them invisible to the outside. There are loops above each hook where the user can attach to the walker frame via a short tether. The discreet support provided by these pants help reduce potential stigmas associated with wearing medical harnesses, enables the user to wear a comfortable and protective harness all day, and will support the user’s entire body weight during a fall.

3) Control System
   An Nvidia Jetson Nano mounted on the walker is to continuously read the IMU data and run it through the fall prediction model. Once a fall is predicted, all four servo motors are immediately activated in order to release the auxiliary legs. It was found that the minimum inference time on the Jetson Nano was 20 ms.

C. EXPERIMENTAL TESTS

1) Deployment Experiment
   The auxiliary legs can deploy as fast as 400 ms. This is expected to be sufficiently quick for establishing a new BOS upon detection of a loss of balance in the user based on a prior study of elderly fall times [18]. Additionally, due to the ratcheting mechanism, the walker auxiliary legs are adaptable to various environments as shown in Figure 16. This enables the walker to establish a stable structure despite obstacles such as uneven floors and narrow hallways.

2) Model Integration with Device & Timing of Events
   The average lead time (from imbalance onset to the start of the descent phase) for the best model on all three fall types was 250 ms or higher. The smallest lead time lower quartile of all three fall types was 100 ms. Figure 17 shows the stages of the fall on the bottom timeline, and the stages of the fall prediction and injury prevention system on the top timeline.

   In one study of elderly falls, it was found that the average duration from the descent initiation to pelvis impact was 583 ms [18]. Since the average descent time is 583 ms, the deadline for prediction must be sometime before the start of the
descent phase to allow enough time for device reconfiguration prior to ground contact.

The baseline model was tested in real-time on the device to verify that simulated falls could be predicted quickly enough in real-time. Results for one person found that when performing simulated falls the device could deploy fast enough before the weight was completely born by the harness. Formal human subject tests were not conducted for this phase. It is expected that using the more complex architectures will increase the inference time, and the duration between predictions may need to be increased slightly from 50 ms.

The performance of this prototype demonstrated the feasibility of applying real-time fall prediction to a fall prevention device other than air bags. Future device development would incorporate the more complex model architectures, and could also focus on an improved BOS deployment system, such as that shown in Figure 18. This alternative deployment configuration for the auxiliary leg would be more practical in a populated environment.

VI. LIMITATIONS & FUTURE WORK

1) Limited Dataset

Some of the limitations to this work include the limited data of 12 subjects and only 2 older people. None of the participants normally use a walker in daily life and were trained to do so prior to data collection.

2) Cross-Validation

Whole participants were not left out for validation since the results differed depending on which participants were left out and 12-fold cross validation would require training 3 models per left out subject, resulting in 36 models for 12 subjects.
Torsion spring

Ratchet gear & pawl

Compression spring

Leg extension

Leg ext. stop

Brace

FIGURE 18. Alternative auxiliary leg deployment design in the normal/stowed configuration (left) and the deployed configuration (right). This design has the rotation point near the top of the walker. It is not as adaptable to obstacles in the environment but is safer in populated environments.

3) Public Dataset

We are working to make the dataset we collected public for other researchers to use by pursuing an amendment to the original IRB protocol.

4) Validation on Walker

Only one person was used to test the baseline fall predictor in real-time. The latest and best architecture using the extra features should be evaluated in real-time. In the future a formal study would be necessary with multiple participants to verify the walker performance.

VII. CONCLUSION

This paper detailed new techniques for fall prediction and the resulting model was implemented on a new fall injury prevention device that can also provide mobility support. Three new methods were introduced to improve performance over a baseline LSTM network that predicts if the user would fall based on signals from an IMU worn by the user. The first method was to separately train models on each fall type with a distinct pattern, which consisted of fast and slow falls. The second method involved adding the time remaining prediction to the separated falls prediction, to delay the prediction at the start of slower falls, where instability is initially pretty mild and whose signals could be similar to ADL motions. The third method involved adding extra data features to make the fall trials with similar inputs but belonging to different target classes more distinct in the input space. The predictability of the dataset with these new features was checked compared to the original features using the Lipschitz quotient. The fast and slow fall datasets with the new features were found to have lower mean Lipschitz quotients, indicating overall higher data predictability, than the equivalent original datasets.

For the original dataset, the separated falls successfully increased the number of identified falls. Likewise, for the original dataset, the separated falls with time remaining decreased the false positive rate as intended.

Different features were added to the original dataset to further improve performance. The potential of these data features was validated with the Lipschitz quotient technique. The trained models for all three architectures performed better than any of the models with the original 6 data features. However, the separated falls and time remaining techniques had little impact on performance with this new dataset. The best model was the separated falls with time remaining with the extra features, and it was able to achieve an identified falls rate of 97% with a false positive rate of 0.17%. Only 1 of the 31 falls tested were missed, which was a trial of simulating tripping on one’s foot.

Therefore, the results can be interpreted as two approaches to improving fall prediction model performance. If no additional data may be collected, separating out distinct fall types and additionally predicting the time remaining in the fall can help to improve performance. Alternatively, if additional sensor signals may be used, or existing signals can be transformed to different features, then these new candidate data features may first be checked with the Lipschitz quotient method. Using more informative sensor signals that are distinct in the input space for each target class will create an easier modeling problem. The Lipschitz quotient enables a comparison of candidate data features that is quicker and easier than training and tuning models on each candidate set of data features.

This paper also detailed the design of a reconfigurable walker that is activated by the fall prediction model. The walker adopts a new stabler configuration if a fall is predicted. The harness short allows the user to move freely during normal operation, provides a comfortable interface to the machine during normal use, and can support the entire body weight for brief periods during emergencies. It will prevent the user from contacting the ground during a fall as long as the support structure remains in a stable configuration. The robotic walker is able to deploy its legs as fast as 400 ms, allowing sufficient time for the walker to expand its BOS and prevent the fall.

REFERENCES


### APPENDIX.

#### TABLE 3. LSTM parameter values

<table>
<thead>
<tr>
<th>Param.</th>
<th>Baseline</th>
<th>Fast</th>
<th>Slow</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Original Features</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># LSTM layers</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># dense layers</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># LSTM hidden units</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td># dense hidden units</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Learning rate</td>
<td>5e-5</td>
<td>5e-5</td>
<td>5e-5</td>
<td>1e-3</td>
</tr>
<tr>
<td>Batch size</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>600</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>Epochs</td>
<td>120</td>
<td>120</td>
<td>120</td>
<td>250</td>
</tr>
<tr>
<td>Loss func.</td>
<td>BCE</td>
<td>BCE</td>
<td>BCE</td>
<td>W. MSE</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
<td>Adam</td>
<td>Adam</td>
<td>Adam</td>
</tr>
<tr>
<td><strong>Extra Features</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># LSTM layers</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># dense layers</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># LSTM hidden units</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td># dense hidden units</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Learning rate</td>
<td>5e-5</td>
<td>1e-3</td>
<td>1e-3</td>
<td>5e-5</td>
</tr>
<tr>
<td>Batch size</td>
<td>30</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>Epochs</td>
<td>120</td>
<td>250</td>
<td>250</td>
<td>120</td>
</tr>
<tr>
<td>Loss func.</td>
<td>BCE</td>
<td>BCE</td>
<td>BCE</td>
<td>W. MSE</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
<td>Adam</td>
<td>Adam</td>
<td>Adam</td>
</tr>
</tbody>
</table>

#### TABLE 4. Hyperparameter values searched for each architecture

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Values Tested</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline</strong></td>
<td></td>
</tr>
<tr>
<td>threshold all falls</td>
<td>( {x \in \mathbb{Z}</td>
</tr>
<tr>
<td><strong>Separated Falls</strong></td>
<td></td>
</tr>
<tr>
<td>threshold fast falls, ( \text{thresh}_{\text{fast}} )</td>
<td>( {x \in \mathbb{Z}</td>
</tr>
<tr>
<td>threshold slow falls, ( \text{thresh}_{\text{slow}} )</td>
<td>( {x \in \mathbb{Z}</td>
</tr>
<tr>
<td><strong>Time Remaining</strong></td>
<td></td>
</tr>
<tr>
<td>threshold TR 1, ( \text{thresh}_{\text{TR1}} )</td>
<td>{0.1, 0.25, 0.5, 0.75, 0.9, 0.95}</td>
</tr>
<tr>
<td>threshold TR 2, ( \text{thresh}_{\text{TR2}} )</td>
<td>{0.1, 0.2, 0.3, 0.4, 0.5}</td>
</tr>
<tr>
<td>look back, ( \text{lb} )</td>
<td>{1, 3, 5, 10}</td>
</tr>
</tbody>
</table>

***