doi:10.20944/preprints202107.0519.v1

Article

The Stumblemeter: design and validation of a system that detects and classifies stumbles during gait

Dylan den Hartog [1], Jaap Harlaar [1, 2,*] and Gerwin Smit [1]

[1] Delft University of Technology, Dept. of Biomechanical Engineering, 2628 CD Delft, The Netherlands; D.D.G.denHartog@student.tudelft.nl (D.d.H); J.Harlaar@tudelft.nl (J.H.); g.smit@tudelft.nl (G.S.)
[2] Erasmus Medical Center, Dept. Orthopaedics, 3015 GD, Rotterdam The Netherlands; j.harlaar@eramusmc.nl
* Correspondence: j.harlaar@tudelft.nl

Abstract: Stumbling during gait is commonly encountered in patients who suffer from mild to serious walking problems, e.g. after stroke, in osteoarthritis, or amputees using a lower leg prosthesis. Instead of self-reporting, an objective assessment of the amount of stumbles in daily life would inform clinicians more accurately and enable the evaluation of treatments that aim to achive a safer walking pattern. An easy to use wearable might fullfill this need. The goal of the present study was to investigate whether a single inertial measurement unit (IMU) placed at the shank and machine learning algorithms could be used to detect and classify stumbling events in a dataset comprising of a wide variety of daily movements. Ten healthy test subjects were deliberately tripped by an unexpected and unseen obstacle while walking on a treadmill. The subjects stumbled a total of 276 times, both using an elevating recovery strategy and a lowering recovery strategy. Subjects also performed multiple Activities of Daily Living. During data processing, an event-defined window segmentation technique was used to trace high peaks in acceleration which could potentially be stumbles. In the reduced dataset, time windows were labelled with the aid of video annotation. Subsequently, discriminative features were extracted and fed to train seven different types of machine learning algorithms. Trained machine learning algorithms were validated using leave-one-subject-out cross-validation. Support Vector Machine (SVM) algorithms were most succesful, and could detect and classify stumbles with 100% sensitivity, 100% specificity and, 96.7% accuracy, in the independent testing dataset. The SVM algorithms were implemented in a user-friendly, freely available, stumble detection app named *Stumblemeter*. This work shows that stumble detection and classification based on SVMs is accurate and ready to apply in clinical practise.

Keywords: Stumbling; detection; machine learning; inertial measurement unit; accelerometer; gyroscope; amputee; osseointegration

1. Introduction

1.1 Stumbling in individuals with impaired gait

Among non-disabled older adults, tripping over an obstacle has consistently been reported as the leading cause of falls [1-3] accounting for 33 [3] to 53 percent of all falls [2]. Fall risk is even increased in chronic disorders like osteoarthritis [4], stroke [5], and leg amputees [6]. During gait, an individual may be particularly susceptible to tripping or stumbling at the instant when the swing foot reaches its peak forward velocity and, simultaneously, the vertical distance between the swing foot and the ground reaches a local minimum [7]. This point in the gait cycle has been referred to as the instant of minimum toe clearance (MTC). Theory predicts that small MTC and larger toe clearance variability increase the probability that the swing foot will contact an unseen obstacle, initiating a stumble [8]. In the absence of compensatory strategies, the lack of ankle dorsiflexion muscles for individuals with a prosthesis is expected to affect MTC, possibly increasing the likelihood of stumbling over an obstacle [9]. Measuring the number of stumbles during daily life could be an effective way to identify older adults and impaired individuals who are prone to fall.. Fall risk is directly associated with stumbles [10, 11]. Sgyrley *et al.* [12] found that elderly who reported multiple near falls were more likely to fall prospectively. For amputees osseointegration is an innovative way to anchor the prosthesis to the bone of the stump. Such a direct skeletal connection of the prosthesis is claimed to provide superior walking stability [13, 14]. However, scientific evidence is required to support these claims. Screening for individual fall risk is advocated for groups at risk (e.g. in osteoarthrosis [4]), to indicate and tailor falls prevention interventions.

1.2 Automatic stumble detection for an objective evaluation of fall risk

Assessing the number of stumbles is often based on subjective self-reports [15]. Even though these self-reports are a low-cost solution, they are not very accurate and reliable, but seriously biased, due to denial and under- or overestimation of the true occurrence of the stumbling events [14, 16]. Therefore, accurate and reliable methods for objective detection of stumbles are required. Automatic stumble detection would enable clinicians to objectively assess patients who are at fall risk, or monitor how an older individual's fall risk changes over time. Moreover, such a system could be used and evaluate the efficacy of interventions that aim to promote walking safety. Furthermore, it can be used to monitor patient progress during falls prevention training programs. In addition to the number of stumbles, also identification of the type of stumble recovery strategy used could be important information to inform therapists. The body has two primary approaches to recovering from stumbles [17, 18]. In the elevating strategy, the obstructed foot is lifted over the impeding object and swung quickly forward to take the weight. In the lowering strategy, the obstructed foot in put to the ground to take body weight while the other leg performs a quick recovery step.

1.3 Wearable sensors and machine learning

In near fall detection research, the rapid development in sensor technology and improvement of data processing capabilities of devices has led to a shift from self-reports to remote monitoring using wearable sensors and advanced detection algorithms, as it gives the opportunity to potentially collect data outside the laboratory setting [19]. Especially the combination of an accelerometer and a gyroscope, also known as an inertial measurement unit (IMU), has become more popular as the development of micro-electro-mechanical systems (MEMS) technology has led to a low cost, low mass, and low energy consumption of sensors.

However, if a stumble detection system is to be used in a real world environment it is hard to distinguish peaks in acceleration and angular velocity that are caused by stumbles from peaks that are caused by other movements, such as walking down the stairs. Threshold-based algorithms no longer suffice and more advanced algorithms are required to separate stumbles from other movements. Machine learning involves the development of algorithms that would enable computers to learn complex patterns and make intelligent decisions based on these algorithms, without explicitly being programmed to do so [20]. The development of advanced machine learning algorithms offers the possibility to classify complex data. However, machine learning is still a relatively new field in stumble and fall detection research. High false positive and false negative rates are one of the main reasons remote stumble and fall detection devices are not widely used yet in daily life [21]. Moreover, a lower-leg based stumble detection with a single sensor system has not been developed yet.

The goal of the present study is to investigate whether a single IMU sensor placed on the lower leg together with machine learning algorithms can be used to detect and classify stumbling events, with high sensitivity and specificity, in a context of activities of daily life.

2. Materials and Methods

2.1 Participants, experimental set-up, and protocol

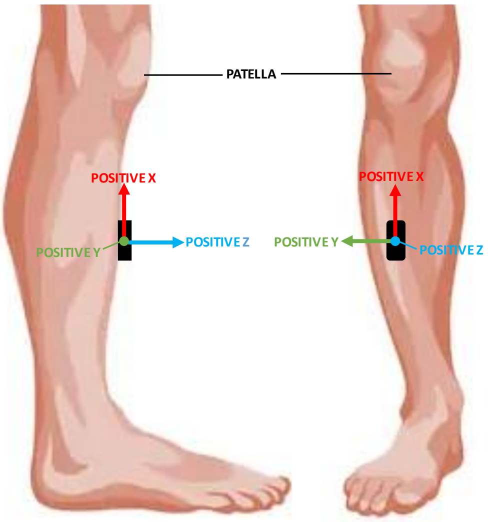
Ten healthy volunteers (9 young (25.4 ± 1.5 years) and 1 older (60 years)) participated in the study. The study was approved by the TU Delft Human Research Ethical Committee (HREC-1304). All risks and precautions of the experiment were explained to the participants, after which they read and signed the informed consent form.

To make the participants stumble unexpectedly, a stumbling device based on the design by King *et al.* [22] was built. The device consists of a ramp-based obstacle delivery apparatus that releases an obstacle onto a treadmill. The obstacle was made out of aluminum and weighted approximately 6 kilograms. The horizontal velocity at treadmill touchdown could be modified by changing the point along the ramp where the obstacle is held by an electromagnet. When the obstacle was released it rolled down the ramped track, on a set of flanged roller bearings mounted on shoulder bolts threaded into each corner of the obstacle, and then slid onto the treadmill belt. Firm foam padding was attached to the front and bottom of the obstacle to protect the subjects' toes and the treadmill belt, respectively.

Participants were asked to walk steadily on a treadmill and manage the unexpected tripping perturbations. To prevent subjects from hearing or seeing the obstacle being deployed, each subject listened to music via earbuds, and a shield was placed directly above the stumbling device, to occlude visual perception of the obstacle sliding on the treadmill. Participants wore a safety harness that was attached to the ceiling by a cord and a stiff spring, to prevent them from falling. Participants were given several minutes to walk on the treadmill before testing, to acclimate to the setup. During the stumbling trials, the treadmill speed was changed after every three consecutive stumbles, ranging from 1 - 5 m/s to elicit different gradations of stumbles and to prevent habituation. Changes in treadmill speed were chosen randomly. Release of the obstacle on the treadmill happened about once every minute. The legs of the participants were videotaped, to classify a trial as either a successful stumbling trial or a mistrial. A trial was labelled as successful, if there was a clear impact of the swinging foot with the obstacle during the swing phase. Trials were labelled as unsuccessful if the subject stepped on or over the obstacle. For each test subject, at least 20 successful stumbles were recorded. Stumbling trials were divided into two classes based on the recovery strategy used: the elevating strategy or the lowering strategy [23, 24]. The experimenter ensured that about the same number of elevating stumbles and lowering stumbles were evoked by manually timing the release of the obstacle. The recovery strategy was determined by the trajectory of the perturbed foot after impact:

- *Elevating strategy:* After impact with the obstacle, the perturbed foot lifts up and over the obstacle, landing past the obstacle. This strategy is used when the foot is perturbed in the early swing phase (5-50% of the entire swing phase).
- *Lowering strategy*: After impact with the obstacle, the perturbed foot lowers in front of the obstacle, while the other foot performs a recovery step and lands past the obstacle. This strategy is used when the foot is perturbed in the late swing phase (40-75% of the entire swing).

One Ax6 inertial measurement unit (IMU) from Axivity, Newcastle upon Tyne, UK, was used during this study. The IMU was placed on the tibia, 20 cm below the patella, using sports tape. The sensor was set to record at 100 Hz, with an accelerometer range of ± 8 g and a gyroscope range of ± 500 dps. The placement and directions of the axes of the IMU are shown in Figure 1.



`

Figure 1. Placement and direction of the axes of the Axivity sensor.

After the stumbling trials, the participants performed several Activities of Daily Living (ADLs) that resemble common movements that are present in the daily life of individuals with a prosthesis. The inclusion of ADLs in the training dataset is necessary to properly train the machine learning classification models and reduce the amount of false positives and false negatives when the system is used in the real world. Each ADL was explained and demonstrated to the participants by the experimenter. The participants then performed the ADLs themselves. See Table 1.

Table 1. Activities of Daily Living

ADL	Amount / time	Instructions
Walking straight	5 min	1, 2, 3, 4, and 5 m/s on a treadmill (1 min each)
Walking corner	10x	Walk 90 degree and 180 degree corners
Come to a halt	10x	Stand still after walking. Repeat 10 times.
Sitting and rising	10x	Sit down on a chair and rise from a chair in different ways and speeds. Repeat 10 times
Pick up object from ground	10x	Throw a small ball on the ground and then pick it up from the ground in different ways and speeds. Repeat 10 times.
Walking upstairs and downstairs	5x	Walk up and downstairs in different ways and using speeds. Repeat 5 times

2.2 Dataset and software

Accelerometer and gyroscope data collected during the experiments were uploaded to a computer via Omgui, an open-source lightweight application. Omgui is used to set up and configure the Axivity sensors, as well as to visualize the data.

The video recordings of the legs of the test subjects were synchronized with the IMU sensor data via ELAN 5.9. In ELAN 5.9 video recordings and IMU sensor data can be synchronized and played back. To obtain the ground truth, the experimenter used this application to manually label the different activities in MATLAB. Motions were labelled as 'Stumble (Elevating)', 'Stumble (Lowering)', and 'Other'. Mistrials were labeled as 'Other', and kept in the dataset. The labelled activities (classes) from the video footage were treated as the ground truth, to train the machine learning models.

MATLAB (by MathWorks Inc) was used for processing the data and developing the machine learning models.

2.3 Data pre-processing

The dataset required minimal pre-processing. Data from the IMU contains seven columns. The first column contains the time as a serial date number. Columns 2 to 4 and 5 to 7 contain the accelerometer and gyroscope data in X, Y, and Z direction, respectively. The logging frequency was set to 100 Hz, resulting in 100 data points per second. The serial date numbers were converted to a datetime array using the *datetime* function. The resultant acceleration without gravity and resultant angular velocity at each time was calculated using equations (1) and (2):

$$a_r = \sqrt{a_x^2 + a_y^2 + a_z^2} - 1 \tag{1}$$

$$\omega_r = \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2} \tag{2}$$

In total 8 signals were used for machine learning; $a_x, a_y, a_z, a_r, \omega_x, \omega_y, \omega_z$ and ω_r.

2.4 Window segmentation and labelling

In machine learning problems where time series come into play, the data should be adequately partitioned and labelled with the corresponding activity, to distinguish between the different classes. In human activity classification, several windowing techniques are used to divide the sensor data into smaller time segments (or windows), also known as window segmentation. Subsequently, feature extraction is applied to each window separately.

In our approach, an event-defined window segmentation method is used. The first step in the event-defined window segmentation method is to find potential stumbles in the dataset. As each stumble is characterized by peaks in acceleration, the *findpeaks* option in MATLAB is used to locate peaks in the dataset. As both stumbles using an elevating strategy and lowering strategy are characterized by high acceleration peaks in the z-direction, the *findpeaks* function in MATLAB was used to find peaks in this signal. To reduce the amount of peaks considered, a threshold value of 1.75g was empirically chosen, as this value was just low enough to capture all stumbling peaks, as well as some peaks caused by other movements. Also, the time interval in between peaks was set to 4 seconds, ignoring lower peaks within this range. This ensures that stumbles are not detected multiple times, as the acceleration signal may cross the 1.75g threshold multiple times during a stumble. The *findpeaks* option returns the locations (indices) of the peaks. After the locations of the peaks were found, these locations were used as the centers of the windows. Time windows of 256 milliseconds were created as this length is enough to fully capture a stumble. For each location, time extending 127 milliseconds to the left and 128 milliseconds to the right, respectively, was considered to form one window. See figure 2. As the sampling frequency of the IMU was 100 Hz, each time window includes 256 data points for one signal. As there are 8 signals, each window contains 8x256 = 2048 data points. The event-defined window segmentation method reduces the computational time as only parts of the data that are potential peaks are fed to the machine learning algorithms and the rest of the data, the vast majority, is ignored for the rest of the process.

Next, the time windows were labelled. During this study we evaluated two different approaches to classify the data into three classes: *Stumble (elevating)*, *Stumble (lowering)* and *Other*. In the first approach, the three-class classification approach, we tested different machine learning algorithms their capability to directly classify the data into the three classes. We use dataset D1 to evaluate this approach. In our second approach, the double binary classification approach, we tested different algorithms their capability to first classify the data into two classes: *Stumble* and *Other*. Subsequently, all windows predicted as stumbles were classified as either *Elevating* or *Lowering* using a second machine learning algorithm. We used datasets D2 and D3 to test this approach. See Table 2 for an overview of the three datasets.

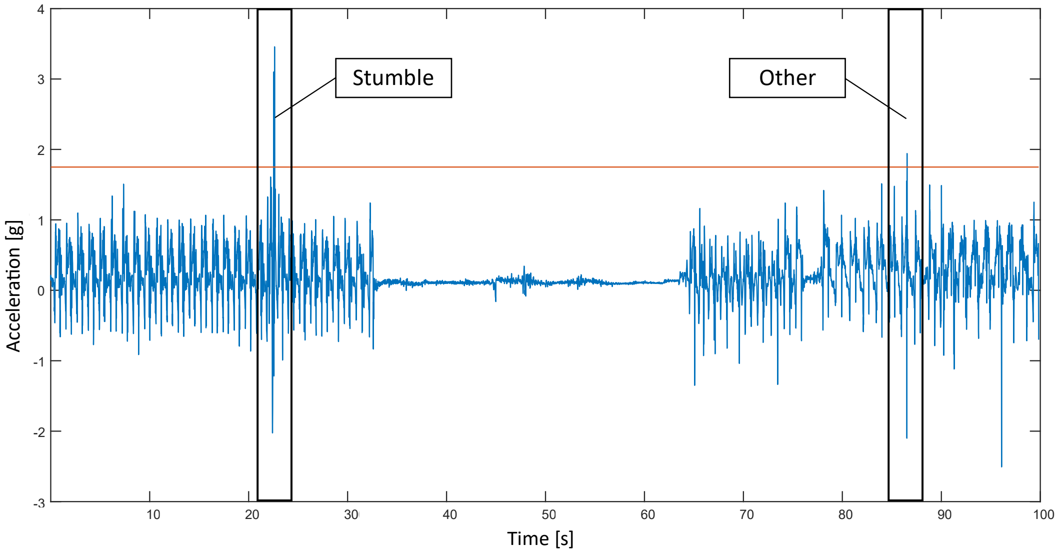


Figure 2. Event-defined window segmentation technique. Time windows were created around the intersection points with the threshold line.

Table 2. Labelled datasets

Dataset	Classes	Amount of windows (validation)	Amount of windows (test)
D1) Three-class classification	Stumble (elevating)	132	12
	Stumble (lowering)	114	18
	Other	329	77
D2) Stumble detection	Stumble	246	30
	Other	329	77
D3) Stumble type classification	Elevating	132	12
	Lowering	114	18

2.5 Feature selection and extraction

Feature selection is an important area in machine learning. It is the process of selecting relevant features to construct a model. The main idea behind feature selection is that some features are redundant or irrelevant and can therefore be removed without much information loss. Research has shown that it is an effective way to improve the learning process and recognition accuracy, and decreases the complexity and computational cost. Some models are negatively affected by irrelevant features [25]. The main objective of feature selection in supervised machine learning is to improve the classification accuracy and reduce complexity [26, 27].

In this study, both time domain and frequency domain features were tested and selected. A Fast Fourier Transform was used to extract frequency-domain features. Initially, 42 different feature classes were tested for usability. For each time window, a single feature class was extracted per IMU signal, creating 8-dimensional feature vectors (1 feature class × 8 signals). These feature vectors were then fed to different machine learning algorithms, to test the feature classes' predictive power. Feature classes were only selected if they were capable of achieving at least 70% sensitivity and specificity with a machine learning algorithm, indicating there is a strong correlation with the output. A total of 17 features classes passed the first selection round. See table 3.

Table 3. Feature classes

Nr	Feature class
1	Interquartile range

2	Kurtosis
3	Mean
4	Median
5	Mean absolute deviation
6	Maximum
7	Minimum
8	Peak-magnitude-to-RMS-ratio
9	Spectral entropy
10	Prominence
11	Root-mean-square level
12	Root-sum-of-squares level
13	Range
14	Skewness
15	Standard deviation
16	Sum of local maxima and minima
17	Variance

Next, for each time window all 17 features classes were extracted for each of the 8 IMU signals, creating 136-dimensional feature vectors (17 classes × 8 signals). This means that for each time window there are 136 features that could describe the characteristics of that window. An effective way to identify redundant and irrelevant features is correlation analysis. Correlation analysis is a statistical method used to evaluate the strength of relationships between quantitative variables. The Pearson correlation coefficient was used to evaluate the degree of linear correlation between two features or between feature and class. All features were correlated with each other, as well as the output class. Features that had a high correlation coefficient with each other (above 0.9) or a low correlation coefficient (between -0.2 and 0.2) with the output were excluded. These thresholds were chosen by determining the classification accuracy of the models with different thresholds. Eliminating features with a feature-to-feature correlation coefficient lower than 0.9 or a feature-to-output correlation coefficient outside the -0.2 to 0.2 range resulted in lower classification accuracy. In total 16 features were excluded as they were either redundant or irrelevant. As 16 features were eliminated, the total number of features fed to the machine learning algorithms was 136 – 16 = 120.

Finally, we normalized the extracted features to rescale the data to a common scale. Supervised machine learning algorithms learn the relationship between input and output and the unit, scale, and distribution of the input data may vary from feature to feature. This will impact the classification accuracy of the models. In this work, the data was normalized by converting the original scaling to a range of 0 to 1.

2.6 Machine Learning algorithms

After the sensor data was properly processed and the features were extracted, the next step is to feed these feature vectors to a machine learning algorithm. In this study, 7 types of machine learning algorithms were tested: Decision Tree, Discriminant Analysis, Logistic Regression, Naïve Bayes, Support Vector Machine (SVM), k-nearest neighbors (KNN), and Ensemble Learning. Each type of machine learning algorithm has hyperparameters to select. For each type of machine learning algorithm, the optimal set of hyperparameters was found for the three different machine learning classification datasets, by using a Bayesian Optimization Algorithm with 30 iterations. See Appendix A for an overview of the Machine Learning algorithms that were trained and evaluated, with their optimal hyperparameters.

2.7 Training, validating and testing

To evaluate the different machine learning algorithms, the dataset was divided into a training dataset, validation dataset and testing dataset. See figure 3. To determine the optimal hyperparameters leave-one-subject-out cross-validation (LOOCV) was used together with Bayesian Optimization, on the data from subjects 1 to 9, the younger test subjects. Like k-fold cross-validation, the data was partitioned into training data and validation data. The validation dataset provides an evaluation of a model fit on the training dataset while tuning the model's hyperparameters. With 9 subjects, the cross-validation process iterated 9 times. For each iteration, the data of the left-out subject was used as validation data and the data of the remaining subjects as training data. After the 9 iterations, the predicted labels of the validation data were compared with the true labels. Trained models with the optimal hyperparameters, found using Bayesian Optimization, were exported.

The trained models were then evaluated with the testing data from the remaining subject 10, the older test subject. The testing dataset is a dataset used to provide an unbiased evaluation of a final model fit on the training dataset. This testing dataset was not used for training. The predicted labels of the testing data were compared with the true labels.

Next, the total performance for each model was calculated with different metrics. For this study, the most important performance metrics to compute were sensitivity (also called true positive rate, hit rate, or recall), specificity (also called true negative rate), and accuracy. See equations (3), (4) and (5). These metrics were calculated after training and validating with LOOCV (validation scores), and after testing the exported models on the holdout data from subject 10 (test scores). It should be noted that the specificities were calculated over the reduced dataset, containing just time windows with peaks that cross the threshold.

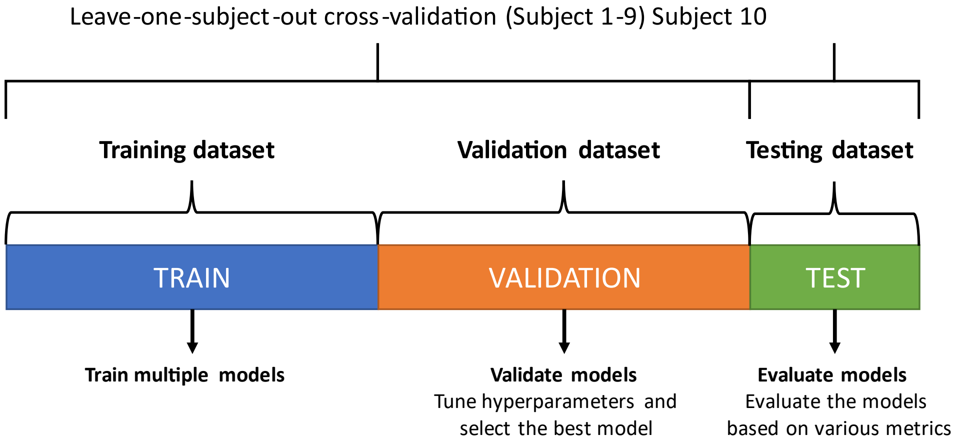

Leave-one-subject-out cross-validation (Subject 1-9) Subject 10
Training dataset
Validation dataset
Testing dataset
TRAIN
VALIDATION
TEST
Train multiple models
Validate models
Tune hyperparameters and select the best model
Evaluate models
Evaluate the models based on various metrics

Figure 3. Division of the dataset in a training dataset, validation dataset and testing dataset.

$$Sensitivity = \frac{TP}{TP + FN} \tag{3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{4}$$

$$Accuracy = \frac{TE + TL}{Total\ stumbles} \tag{5}$$

TP represent the true positives (true stumbles), TN represents the true negatives (true ADLs), FP represents the false positives (ADLs misidentified as stumbles), and FN represents the false negatives (stumbles misidentified as ADLs). TE represent true elevating stumbles and TL the true lowering stumbles.

3. Results

In total 276 successful stumbles were captured by the IMU, of which 134 were stumbles that were recovered using the elevating strategy and 132 that were recovered using the lowering strategy. Subject 10 stumbles 30 times and recovered from 5 perturbations by jumping over the obstacle with both legs at the same time. These 'hopping' recoveries were labelled as elevating, as the obstructed foot was lifted over the object directly after the collision. No separate class was created for these 'hopping' stumbles as there was simply not enough data to do so. The dataset of all subjects combined, including both the stumbling data and ADLs, is approximately 11.5 hours long.

In this chapter, all the different machine learning algorithms are validated using leave-one-subject-out cross-validation and tested by using the exported models on the holdout data from subject 10. In paragraph 3.1, single machine learning algorithms were used to separate three classes directly*: Stumble (elevating), Stumble (lowering)* and *Other.* In paragraph 3.2 two machine learning algorithms were used in series, to first separate all stumbles from all other peaks, and subsequently differentiate between the type of stumble recovery strategy. Sensitivity and specificity were computed to validate and evaluate the model's ability to separate the stumbles from the other data. Accuracy was computed to evaluate the model's ability to distinguish between stumbles where an elevating strategy was used and stumbles where a lowering strategy was used.

3.1. Three-class classification approach

Since we do not only want to separate stumbles from the other data, but also want to distinguish between the type of stumble (elevating strategy / lowering strategy), there are three classes in this machine learning classification problem. Intuitively, a single machine learning model can be used to classify the data into the three classes. See figure 4.

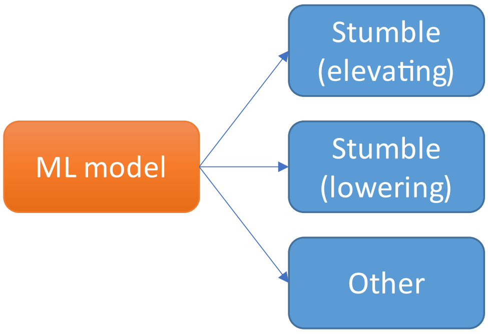


Figure 4. Single machine learning model to classify data into three classes.

Table 3 and 4 show the results for this classification problem. We used dataset D1 to validate and evaluate this approach. The model predictions were compared with the true labels. Sensitivities and specificities were calculated by taking both types of stumbles together as the positive class and the 'Other' windows as the negative class. Accuracy was calculated by the number of correctly classified stumbles (elevating as elevating and lowering as lowering) divided by the number of detected stumbles. Highest sensitivities, specificities and accuracies where achieved with the SVM model during both validation and testing.

Table 3. Results three-class classification approach (validation)

ML model	Sensitivity [%]	Specificity [%]	Accuracy [%]
SVM	98.4	99.4	98.5
Ensemble Learning	98.0	98.5	93.4
Discriminant Analysis	97.2	97.0	90.0

KNN	97.2	95.1	74.1
Naïve Bayes	91.1	95.4	75.4
Decision Tree	87.8	93.3	77.8

Table 4. Results three-class classification approach (testing)

ML model	Sensitivity [%]	Specificity [%]	Accuracy [%]
SVM	96.7	100	96.6
Ensemble Learning	93.3	98.7	92.9
Discriminant Analysis	93.3	97.4	89.3
KNN	90	93.5	88.9
Naïve Bayes	90	93.5	77.8
Decision Tree	83.3	94.8	72.0

3.2. Double binary classification approach

In our second approach, the classification problem was split into two parts. First, a machine learning model was used to detect stumbles in the data. We call this the stumble detection problem. Subsequently, a second machine learning model was used to classify the stumbles as either a stumble where an elevating strategy was used or a stumble where a lowering strategy was used. We will call this the stumble type classification problem. See figure 5.

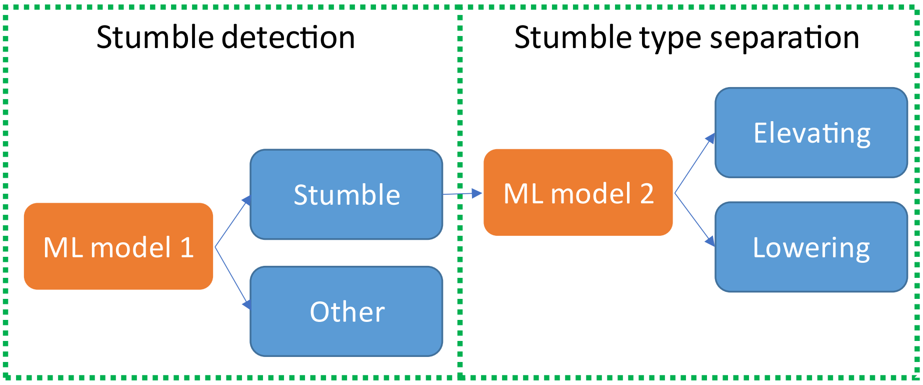


Figure 5. Two binary machine learning models in series.

Tables 5 and 6 show the results for the stumble detection problem, validated and evaluated with dataset D2. The model predictions were compared with the true labels. A confusion matrix was created for each model and the sensitivities and specificities were calculated for both validation and testing. Best results were achieved with the SVM model, with a 100% sensitivity and 100% specificity in the testing dataset. Tables 7 and 8 show the results for the stumble type classification problem, validated and evaluated with dataset D3. The accuracy was defined as the amount of correct predictions over the total amount of predictions. Again a SVM model outperformed other models, with an accuracy of 96.7% in the testing dataset.

Table 5. Results stumble type classification problem (validation)

ML model 1	Sensitivity [%] (validation)	Specificity [%] (validation)
SVM	98.8	100
Discriminant Analysis	98.0	96.3
Ensemble Learner	97.6	98.2

Logistic Regression	97.6	94.8
KNN	96.3	92.4
Naïve Bayes	88.2	90.4
Decision Tree	88.0	93.6

Table 6. Results stumble type classification problem (testing)

ML model 1	Sensitivity (testing)	Specificity [%] (testing)
SVM	100	100
Discriminant Analysis	96.7	97.4
Ensemble Learner	96.7	97.4
Logistic Regression	90.0	93.5
KNN	90.0	92.2
Naïve Bayes	86.7	89.6
Decision Tree	83.3	88.3

Table 7. Results stumble type classification problem (validation)

ML model 2	Accuracy [%] (validation)
SVM	95.1
Ensemble Learner	91.9
Discriminant Analysis	87.0
KNN	86.6
Logistic Regression	85.4
Naïve Bayes	84.2
Decision Tree	81.3

Table 8. Results stumble type classification problem (testing)

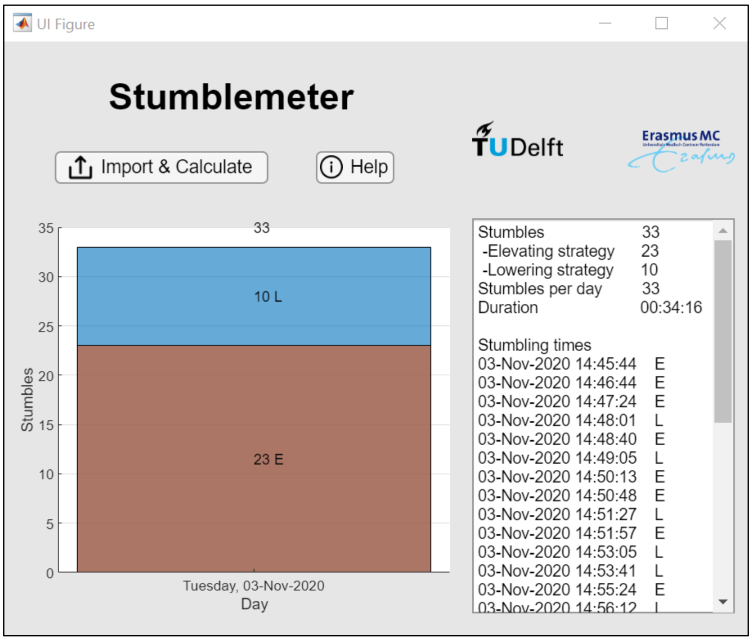
ML model 2	Accuracy [%] (testing)
SVM	96.7
Ensemble Learner	93.3
Discriminant Analysis	83.3
KNN	80.0
Logistic Regression	80.0
Naïve Bayes	76.7
Decision Tree	73.3

3.3. Final model

For our final model, we look at the results of the previous two paragraphs. As precise detection of stumbles is prioritized over accurate stumble type classification, the main demand for the final model is its ability to detect as many stumbles as possible with keeping the amount of false positives as low as possible. For both validation and testing, highest sensitivity and specificity were achieved with the double binary classification approach. For both the stumble detection problem and the stumble type classification problem the best results were achieved with SVM's. For stumble detection, the SVM achieved 100% sensitivity and 100% specificity, in the testing dataset. For the stumble type classification, the SVM was able to classify the stumble recovery type with 96.7% accuracy, in the testing dataset. Therefore, for our final model we use these two SVM models with optimized hyperparameters in series.

3.4. *Stumblemeter app*

To make the programming work of this study accessible for clinicians, an application named *Stumblemeter* was created (see Supplementry Materials). After uploading the .cwa file containing the IMU data, the application automatically performs all the steps required for machine learning classification. See figure 6 for the interface of the stumblemeter app. The application displays the amount of stumbles in the form of a histogram. In the text area, the total amount of stumbles during a measurement is displayed, as well as the times when a stumble occurred. Depending on the physical activity of an individual with a prosthesis, the computation time for a 7-day measurement is about 3 minutes.



```
Stumbles              33
 -Elevating strategy  23
 -Lowering strategy   10
Stumbles per day      33
Duration              00:34:16

Stumbling times
03-Nov-2020 14:45:44   E
03-Nov-2020 14:46:44   E
03-Nov-2020 14:47:24   E
03-Nov-2020 14:48:01   L
03-Nov-2020 14:48:40   E
03-Nov-2020 14:49:05   L
03-Nov-2020 14:50:13   E
03-Nov-2020 14:50:48   E
03-Nov-2020 14:51:27   L
03-Nov-2020 14:51:57   E
03-Nov-2020 14:53:05   L
03-Nov-2020 14:53:41   L
03-Nov-2020 14:55:24   E
03-Nov-2020 14:56:12   L
```

Figure 6. Interface of the *Stumblemeter* app. The number of stumbles per day is displayed on the left in the form of a histogram. The total number of stumbles, the duration of the measurement and the individual stumbling times are displayed on the right.

4. Discussion

4.1. *Recap*

During this study, seven types of machine learning algorithms were trained, validated and tested. For each type, optimized of hyperparameters were found, using Bayesian Optimization. All the models were first validated using leave-one-subject out cross-validation, and then exported. The exported models were then tested on the testing dataset, which included the data from the older test subject. We found that using two binary SVM models in series produced better results that using a single SVM to directly classify the data into three classes. Therefore, these two SVM's are used in the final model. Even though the subjects performed a multitude of different ADLs, no other movements were recognized as a stumble in both the validation dataset and the testing dataset.

4.2. *Internal validity*

The data was split into a training dataset, validation dataset and testing dataset. The validation dataset is a sample of data held back from training the model that is used to give an estimate of model skill while tuning model's hyperparameters. The validation dataset is different from the test dataset, that is also held back from the training of the model, but is instead used to give an unbiased estimate of the skill of the final tuned model when comparing or selecting between final models.

For each type of machine learning algorithm, Bayesian Optimization was used during leave-one-subject-out cross-validation with data from 9 subjects to find the optimal set of hyperparameters. Models were trained with Bayesian Optimization to minimize the error function defined with respect to the training dataset. The performance of the models was compared by evaluating the error function using an independent validation dataset, which provided an evaluation of a model fit on the training dataset while tuning the model's hyperparameters, and the models having the smallest error with respect to the validation dataset were selected. Since this procedure can itself lead to some overfitting to the validation dataset, the performance of the selected models was confirmed by measuring its performance on a third independent test dataset, containing the data from subject 10. Testing on the unseen data provided an unbiased evaluation of a final model fit on the training dataset. The predicted labels were compared with the true labels and the sensitivity, specificity, and accuracy of the models were calculated to evaluate the models. By using this method it is certain that the testing data could not have influenced the training of the models.

4.3. Comparison with previous studies

In previous near fall detection research, only two studies achieved 100% specificity. Aziz *et al.* [28] did include multiple ADLs in their dataset. However, the way they recreated stumbles is questionable, as they had their participants act out a stumble on a mattress after watching a video. It remains unclear whether their system would be able to accurately detect a real-world stumble. Moreover, their setup is too impractical for clinical use: it consists of five sensors of which one was placed on the head. The other study that achieved 100% specificity, by Choi *et al.* [29], added just three ADLs in their dataset: standing, walking, and lying down, and did not include activities with high acceleration peaks. Such a limited dataset lacks realistic representation of real-life activity, which could result in an overestimation of the practical performance. Also, two sensors were used, which is less attractive for practical use than our single sensor system.

All in all, we expect that the stumblemeter presented in this study will outperform previously reported systems. Importantly, the machine learning algorithm was trained and tested with naturally occurring stumbles in a dataset that contains a representative amount of ADLs. For clinical feasibility, it is important that the single sensor can be attached to the shank in an unobtrusive way and can be worn for a longer period of time, e.g. a week.

This study also aimed to create an algorithm that is able to determine the type of stumble; whether an elevating recovery strategy was used or a lowering recovery strategy. A second model was used to classify detected stumbles into the two classes. This model was trained, validated and tested, separately. We found that an optimized SVM was able to distinguish between the two types of strategies with 96.7% accuracy in the testing dataset.

In terms of computational cost, we cannot compare our system with other systems as they did not give any specifications on that matter. However, it is evident that the event-defined window segmentation technique that was introduced ensures that the computational cost is considerably lower, than when the full dataset has to be processed. We made use of the fact that all stumbles are paired with high peaks in acceleration. By using a threshold, the vast majority of irrelevant data (95.5% in our dataset) is eliminated in an early stage. As a result, for a limited amount of time windows features have to be extracted and fed to the machine learning models. In this study, 682 high peak time windows that were created, of which 276 (40.5%) were stumbles and 406 (59.5%) were non stumbles. The estimated computational time for a week-long measurement is 3 minutes at most, depending on the user activity.

4.4. Practical application in clinical research

The *stumblemeter* has been validated and tested in healthy people. Strickly sproken, this will not guarantee that it will work as well in the target population: individuals with walking diffulties, e.g. those walking with a prosthesis. The system would have to be tested separately for the different pathologies. Nonetheless, it is expected that *stumblemeter* will work as desired during clinical research in e.g. the amputee population. Shirota *et al.* [30] showed that transfemoral amputees generally exhibited typical able-bodied recovery strategies (elevating and lowering) when recovering from stumbles on both the sound and prosthesis sides. Throughout the swing phase, amputees used similar recovery strategies to able-bodied subjects for perturbations that occurred at similar time points in the gait cycle. However, two out of eight amputees in their study used a novel hopping strategy when tripped using a tether on the prosthesis side in early to mid-swing. This strategy was also found in the older test subject in our study. Such recoveries were labelled as elevating, as the obstructed foot was lifted over the object directly after the collision. These 'hopping' stumbles were classified as elevating stumbles and all detected correctly. Therefore it is expected that our system is able to detect such stumbles, even though it is not specifically trained to classify this particular recovery type. Follow-up research on individuals with prosthetic legs should be conducted to validate this expectation.

5. Conclusion

This work shows that stumble detection and classification based on an IMU and SVMs is extremely accurate and ready to apply in clinical practice. Our proposed system consists of just one small IMU sensor, which can easily be integrated into the pylon of a prosthesis or attached to the shank, leaving no burden for the users. Out of the 30 evoked stumbles from an independent experiment, the optimized SVM model was able to detect all of them (100% sensitivity). Moreover, our models did not give any false positive predictions (100% specificity), even though the dataset comprised of a wide variety of daily movements. This is the first study aiming to classify the type of stumble recovery strategy and did this with 96.7% accuracy. The user-friendly *Stumblemeter* app makes it quite straightforward for clinicians to analyze the data. The introduction of the *Stumblemeter* enables clinicians to objectively assess fall risk in older adults, amputees, and other individuals with gait impairments, outside a laboratory or clinical setting.

Supplementary Materials: The computer code (in MATLAB) and software (ready to use Windows app) are openly available from https://doi.org/10.4121/14473269.

Author Contributions: Methodology, software, experiments, design, writing—original draft preparation, D.d.H.; conceptualize, supervision, writing—review and editing, J.H. and G.S. All authors have read and agreed to the published version of the manuscript

Funding: This research was funded by Delft University of Technology.

Institutional Review Board Statement: The study was conducted according to the guidelines of the Declaration of Helsinki, and approved by the Human Research Ethics Committee of Delft University of Technology (ID 1304 approved on 09-10-2020).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The data presented in this study are openly available in 4TU.ResearchData at https://doi.org/10.4121/14473320.

Acknowledgments: The authors wish to express their gratitude to Erin Macri, Ajay Seth, Jan van Frankenhuyzen for their help and insightful comments during this work and like to thank the enrolled subjects who voluntarily participated in this study.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Decision Tree: optimized hyperparameters for each dataset

ML model	Maximum number of splits	Split criterion	Surrogate decision splits
Decision Tree (D1)	59	Maximum deviance reduction	Off
Decision Tree (D2)	38	Gini's diversity index	Off
Decision Tree (D3)	19	Gini's diversity index	Off

Table A2. Discriminant Analysis: optimal hyperparameters for each dataset

ML model	Discriminant type
Discriminant Analysis (D1)	Linear
Discriminant Analysis (D2)	Linear
Discriminant Analysis (D3)	Linear

Table A3. Logistic Regression

ML model
Logistic Regression (D2)
Logistic Regression (D2)

Note 1: Logistic Regression can only be used for binary classification problems.

Note 2: Hyperparameter optimization does not apply to Logistic Regression as there are no hyperparameters to alter.

Table A4. Naïve Bayes: optimized hyperparameters for each dataset

ML model	Distribution type	Kernel type	Support
Naïve Bayes (D1)	Kernel	Gaussian	Unbounded
Naïve Bayes (D2)	Kernel	Triangle	Unbounded
Naïve Bayes (D3)	Kernel	Gaussian	Unbounded

Table A5. Support Vector Machine: optimized hyperparameters for each dataset

ML model	Kernel function	Box constraint level	Kernel scale
SVM (D1)	Linear	1	8.5
SVM (D2)	Gaussian	12.5	26
SVM (D3)	Linear	1	25

Table A6. K-Nearest Neighbors: optimized hyperparameters for each dataset

ML model	Number of Neighbors	Distance metric	Distance weight
KNN (D1)	1	Cosine	Squared inverse
KNN (D2)	5	Spearman	Squared inverse
KNN (D3)	1	Correlation	Inverse

Table A7. Ensemble Learner: optimized hyperparameters for each dataset

ML model	Ensemble method	Learner type	Max number of splits	Number of learners	Learning rate	Subspace dimension
Ensemble Learner (D1)	Bag	Decision Tree	320	12	-	53
Ensemble Learner (D2)	GentleBoost	Decision Tree	5	485	0.001864	38
Ensemble Learner (D3)	Bag	Decision Tree	182	27	-	35

References

1. Li, W., Keegan, T. H. M., Sternfeld, B., Sidney, S., Quesenberry, C. P., & Kelsey, J. L. Outdoor Falls Among Middle-Aged and Older Adults: A Neglected Public Health Problem. *American Journal of Public Health* **2006**, *96*, 1192–1200.
2. Blake, A. J., Morgan, K., Bendall, M. J., Dallosso, H., Ebrahim, S. B. J., Arie, T. H. D., Fentem, P. H., & Bassey, E. J. Falls by elderly people at home: Prevalence and associated factors. *Age and Ageing* **1988**, *17*, 365–372.
3. Berg, W. P., Alessio, H. M., Mills, E. M., & Tong, C. Circumstances and consequences of falls in independent community-dwelling older adults. *Age and Ageing* **1997**, *26*, 261–268.
4. Ofori-Asenso, R., Ackerman, I. N., & Soh, S. Prevalence and correlates of falls in a middle-aged population with osteoarthritis: Data from the Osteoarthritis Initiative. *Health & Social Care in the Community* **2020**, *29*, 436–444.
5. Weerdesteyn, V., De Niet, M., Van Duijnhoven, H. J., & Geurts, A. C. Falls in individuals with stroke. *J Rehabil Res Dev* **2008**, *45*, 1195-213.
6. Hunter, S. W., Batchelor, F., Hill, K. D., Hill, A. M., Mackintosh, S., & Payne, M. Risk Factors for Falls in People With a Lower Limb Amputation: A Systematic Review. *PM&R* **2016**, *9*, 170–180.
7. Winter, D. A. (1992). Foot Trajectory in Human Gait: A Precise and Multifactorial Motor Control Task. *Physical Therapy* **1992**, *72*, 45–53.
8. Begg, R., Best, R., Dell'Oro, L., & Taylor, S. Minimum foot clearance during walking: Strategies for the minimisation of trip-related falls. *Gait & Posture* **2007,** *25*, 191–198.
9. Rosenblatt, N. J., Bauer, A., Rotter, D., & Grabiner, M. D. Active dorsiflexing prostheses may reduce trip-related fall risk in people with transtibial amputation. *Journal of Rehabilitation Research and Development* **2014**, *51*, 1229–1242.
10. Byju, A. G., Nussbaum, M. A., & Madigan, M. L. Alternative measures of toe trajectory more accurately predict the probability of tripping than minimum toe clearance. *Journal of Biomechanics* **2016**, *49*, 4016–4021.
11. Santhiranayagam, B. K., Sparrow, W. A., Lai, D. T. H., & Begg, R. K. Non-MTC gait cycles: An adaptive toe trajectory control strategy in older adults. *Gait & Posture* **2017**, *53*, 73–79.
12. Srygley, J. M., Herman, T., Giladi, N., & Hausdorff, J. M. Self-Report of Missteps in Older Adults: A Valid Proxy of Fall Risk? *Archives of Physical Medicine and Rehabilitation* **2009**, *90*, 786–792.
13. Sehatzadeh, S., Tiggelaar, S., & Shafique, A. Osseointegrated Prosthetic Implants for People With Lower-Limb Amputation: A Health Technology Assessment. *Ontario health technology assessment series* **2019**, *19*, 1–126.
14. Advantages and disadvantages Osseointegration. Available online: https://www.radboudumc.nl/en/patientenzorg/behandelingen/osseointegration/advantages-and-disadvantages (accessed on 17 March 2021).
15. Hajj Chehade, N., Ozisik, P., Gomez, J., Ramos, F., & Pottie, G. Detecting stumbles with a single accelerometer. Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society; EMBS; 2012; 6681–6686
16. Mackenzie, L., Byles, J., & D'Este, C. Validation of self-reported fall events in intervention studies. *Clinical Rehabilitation* **2006**, *20*, 331–339.
17. Eng, J. J., Winter, D. A., & Patla, A. E. Strategies for recovery from a trip in early and late swing during human walking. *Experimental Brain Research* **1994**, *102*, 339–349
18. Pavol, M. J., Owings, T. M., Foley, K. T., & Grabiner, M. D. Mechanisms Leading to a Fall From an Induced Trip in Healthy Older Adults. *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences* **2001**, *56*, 428–437.
19. Patel, S., Park, H., Bonato, P., Chan, L., & Rodgers, M. A review of wearable sensors and systems with application in rehabilitation. *Journal of NeuroEngineering and Rehabilitation* **2012**, *9*, 21.
20. What is Machine Learning? A definition - Expert System. Available online: https://www.expert.ai/blog/machine-learning-definition/ (accessed on 17 March 2021).
21. Noury, N., Fleury, A., Rumeau, P., Bourke, A. K., Laighin, G. Ó., Rialle, V., & Lundy, J. E. Fall detection - Principles and methods. Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology; 2007; 1663–1666.
22. King, S. T., Eveld, M. E., Martínez, A., Zelik, K. E., & Goldfarb, M. A novel system for introducing precisely-controlled, unanticipated gait perturbations for the study of stumble recovery. *Journal of NeuroEngineering and Rehabilitation* **2019**, *16*, 69
23. Schillings, A. M., van Wezel, B. M. H., Mulder, T., & Duysens, J. Muscular Responses and Movement Strategies During Stumbling Over Obstacles. *Journal of Neurophysiology* **2000**, *83*, 2093–2102.
24. Shirota, C., Simon, A. M., & Kuiken, T. A. Trip recovery strategies following perturbations of variable duration. *Journal of Biomechanics* **2014**, *47*, 2679–2684.
25. Kuhn, M., & Johnson, K. *Applied Predictive Modeling*, 1st ed.; Springer Medizin Verlag: 2013.
26. Acharya, A., & Sinha, D. Application of Feature Selection Methods in Educational Data Mining. *International Journal of Computer Applications* **2014**, *103*, 34–38.
27. García, S., Luengo, J., & Herrera, F. *Data Preprocessing in Data Mining*, 1st ed.; Springer Publishing; 2014.
28. Aziz, O., Park, E. J., Mori, G., & Robinovitch, S. N. Distinguishing near-falls from daily activities with wearable accelerometers and gyroscopes using Support Vector Machines. Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society; EMBS: 2012; 5837–5840.
29. Choi, Y., Ralhan, A. S., & Ko, S. A study on machine learning algorithms for fall detection and movement classification. Proceedings of the International Conference on Information Science and Applications; ICISA: 2011.

30. Shirota, C., Simon, A. M., & Kuiken, T. A. Transfemoral amputee recovery strategies following trips to their sound and prosthesis sides throughout swing phase. *Journal of NeuroEngineering and Rehabilitation* **2015**, *12*.