UNIVERSITY OF EXETER

Computer Science Department

B.Sc. Computer Science
Computer Science Department

Inference on Physical Therapy Exercises using Inertial Sensors

Candidate

Haris Aftab

Student ID 700074345

Supervisor

Dr. Johan Wahlström

University of Exeter

Academic Year
2022/2023

Abstract

The utilization of physical therapy exercises constitutes a potent modality for rehabilitating the mobility and functional capacity of individuals suffering from a diverse array of impairments stemming from disability, illness, or injury. However, their success is contingent upon the patient's ability to adhere to the prescribed guidelines of the physical therapy routines. Inertial Measurement Units (IMUs) are a device that employs accelerometers, gyroscopes, and magnetometers to accurately measure and report the motion of a body. A patient's ability to accurately execute physical therapy exercises can be facilitated through the use of IMUs. Our objective is to leverage machine learning techniques such as Support Vector Machines to classify and evaluate the physical therapy exercises performed using data gathered from these devices. To accomplish this, we have conducted an in-depth analysis using data acquired from the PHYTMO database, which comprises inertial data from 30 volunteers performing six exercises with varying degrees of form correctness. In this report, we also investigate the efficacy of different window lengths and overlaps, and evaluate their impact on model performance. Through experimentation, we determined that a window length of 4 seconds and 25% overlap resulted in the highest accuracy rate of 95.5%. The success of the project will be determined by the implementation of rigorous evaluation techniques, which will allow for a comprehensive comparison of results with previous, comparable studies.

Contents

1 Introduction

Physical therapy is a vital element in providing targeted rehabilitation services to individuals seeking a healthier and more active lifestyle that is tailored to their specific needs. As an integral part of preventive health care, physical therapy exercises promote overall well-being while mitigating the risk of chronic ailments. Due to its conservative nature, it has earned its recommendation at the earliest indication of any problem by primary care physicians. The benefits of physical therapy exercises are manifold, ranging from improving mobility to managing pain and aiding in injury recovery. One of the most significant advantages of physical therapy exercises is their ability to enhance mobility, regardless of age. By employing stretching and strengthening exercises, physical therapy can restore an individual's ability to move, even in the case of difficulty standing, walking, or moving. Another noteworthy benefit is the potential to avoid surgery, as physical therapy may help manage pain or facilitate injury recovery. Even in cases where surgery is necessary, pre-surgical physical therapy can be beneficial as it can aid in a faster recovery by building strength and improving physical fitness. This can also help in reducing healthcare expenses by avoiding surgery. By strengthening weakened parts of the body and improving gait and balance, physical therapy exercises can restore mobility, allowing individuals to move with greater ease and independence. This can lead to an improved quality of life by reducing dependence on others for everyday tasks such as dressing, bathing, and toileting [1].

As the world's population continues to age, with an estimated increase in the proportion of those over 60 from 12% to 22% between 2015 and 2050 [2], there will be an ensuing surge in demand for healthcare and social services, leading to a significant rise in associated expenses. Traditional physical therapy, which involves patients receiving treatment in clinics or hospitals under the supervision of therapists, can be inconvenient for patients, interrupting their daily activities. Moreover, this approach may be cost-prohibitive, particularly for elderly individuals requiring long-term therapy to maintain their health.

Physical therapists always conduct a comprehensive assessment of a patient's strength, range of motion, and pain points to devise a tailored treatment plan aimed at restoring optimal functionality or helping the patient adjust to any new limitations. Once their inpatient therapy session concludes, patients are usually provided with an exercise regimen to continue their progress at home. Unfortunately, transitioning from supervised therapy to self-directed exercises at home can interrupt the continuity of care and hamper the efficacy of the treatment. To maintain the benefits gained from physical therapy, patients must remain active and perform the exercises correctly to target specific muscles without placing undue stress on them. Failure to execute exercises properly can engage unintended muscles, potentially hindering progress and leading to further complications [3].

These factors are driving the importance of home-based supervision systems. Evaluating the effectiveness of these exercises can be challenging, as it often requires subjective assessments by therapists. Machine learning (ML) offers a promising solution to this problem by providing an objective and automated method for classifying and evaluating physical therapy exercises. Through the use of Inertial Measurement Units (IMUs) and ML algorithms such as Support Vector Machines (SVMs), it is possible to gather and analyse data on a patient's movement patterns, allowing for more accurate and personalized assessments of their physical progress. IMUs provide several advantages over other sensing solutions in exercise recognition. They are wearable and easily integrated into devices like smartphones and fitness trackers, do not require a line of sight like cameras, and are low-cost with long battery life meaning them ideal for large-scale deployment. This approach also has the potential to improve the continuity of care by providing patients with the ability to continue their exercises at home, while still receiving personalized feedback on their performance. In this paper, we explore the benefits of using machine learning techniques to classify and evaluate physical therapy exercises and investigate the effectiveness of various window lengths and overlaps in improving the accuracy of the models. By doing so, we aim to contribute to the growing body of research on the application of ML in healthcare and its potential to improve patient outcomes in physical therapy.

2 Project Specification

2.1 Aims and Objectives

Aims:

- To investigate the feasibility and effectiveness of using machine learning algorithms for the classification and evaluation of physical therapy exercises.
- To develop a system that can evaluate and classify the correctness of physical therapy exercises using IMU sensors.
- To investigate the efficacy of different window lengths and overlaps and evaluate their impact on model performance.

Objectives:

- Acquire the raw IMU signals from the PHYTMO database, comprising four IMUs attached to the upper and lower limbs in each exercise recording, capturing tri-axial accelerometer, gyroscope, and magnetometer data.
- Utilize sliding windows to segment the raw IMU data, with window sizes ranging from 100 to 500 samples, corresponding to windows between 1 and 5 seconds, and varying the window overlap size using 0%, 25%, 50%, and 75% overlaps.
- Conduct meaningful feature extraction on the signal windows over each axis to capture pertinent information.
- Label the samples as required by supervised learning algorithms, classifying exercises as correct or incorrect in performance.
- Pre-process the features and labels for more effective classifier training.
- Train the ML algorithm for the classification and evaluation of physical therapy exercises.
- Utilize leave-one-out cross-validation (LOOCV) on the model to calculate performance metrics such as accuracy, precision, sensitivity, F1-score, and specificity.
- Compare the results obtained with related studies to evaluate the efficacy of the developed model.

The success of this project will be judged by its performance metrics which will be evaluated utilizing the LOOCV technique. Success will be determined by achieving a minimum threshold of 90% for all performance metrics, including accuracy, precision, sensitivity, F1-score, and specificity. These metrics are pivotal in assessing the model's ability to precisely classify and evaluate exercises. More detail of success criteria in section 3.5.

3 Design

3.1 Dataset

The PHYTMO dataset [4] is a highly valuable resource for researchers and practitioners interested in physical therapy and movement analysis. It consists of data collected from 30 healthy and robust volunteers, who performed six physical therapy exercises and three gait variations that are commonly prescribed to elderly people as part of a physical therapy exercise routine. The dataset is notable for its high quality and range of exercises, making it a highly relevant resource for researchers interested in studying the biomechanics of human movement and developing new algorithms and techniques for analysing movement data. The exercises and gait variations performed by the volunteers have been carefully selected based on their ability to help elderly people retain their functional capacity [5]. Studies have shown that consistently performing these exercises over a relatively short period of time can lead to improved functional abilities in older individuals [6]. The wearable sensors used in the study were carefully calibrated and synchronized to ensure accurate measurement of the variability in execution. The sensors included 3-axis gyroscopes, accelerometers, and magnetometers, with a range of 2000°/s, 16g, and 1,300μT, respectively. The data were sampled at a rate of 100Hz for the gyroscopes and accelerometers and 20Hz for the magnetometers, providing a high level of detail and precision.

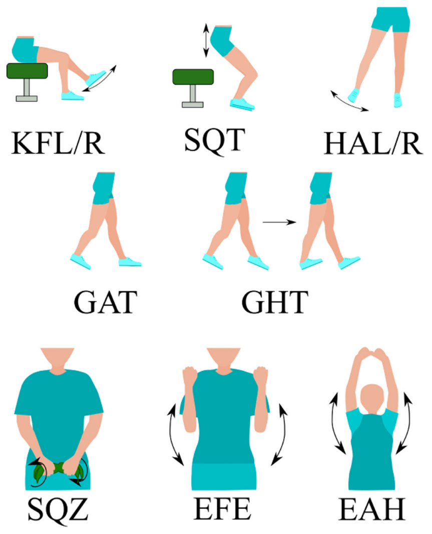


Figure 1: Visual representation of the exercises examined in this research [7].

The PHYTMO database is organized in a meticulous manner, with each dataset consisting of a series of text files stored in the widely-used Comma-Separated Values (CSV) format. The database is further divided into two categories: upper and lower limbs, with each group being segregated based on the age range of subjects, which spans from 20 to 70 years of age. Each group is then subdivided into distinct folders based on the specific body part where the IMUs are placed, namely the left or right side, and the corresponding limb segment. For the lower limb group, these segments are classified as Lshin, Lthigh, Rshin, and Rthigh, while for the upper limb group, the segments are classified as Larm, Lforearm, Rarm, and Rforearm.

In this project, we will employ the data from the most active sensor positions during exercises on both the left and right sides. Specifically, the sensors on the left and right forearm for upper body exercises and the sensors on the left and right shin for lower body exercises will be utilized, as it will capture the majority of the movement and motion associated with the exercises. Notably, exercises such as KFE and HAA are performed unilaterally, and as such, an additional identifying character is included at the end of the filename to indicate the leg on which the exercise is being performed (e.g. KFEL and KFER). To ensure an active signal is always present, we will use only the sensor position on the same side of the body as these exercises are being performed. Otherwise, if, for instance, we employ the sensor on the left shin whilst KFE is being executed on the right leg, the signal will be flat and fail to accurately represent the exercise being performed.

Exercise	Correct execution	Wrong execution
Knee flex-extension (KFE)	With the left leg stationary, the right leg is extended in the sagittal plane from an initial position of 90° knee flexion until maximum extension. This is repeated on opposite legs.	• Deviations from the sagittal plane • Concentrating the movement on the thigh • Non-unilateral leg movement
Squats (SQT)	Perform the exercise by sitting on a chair without bending the knees or hips laterally and standing up again upon touching the chair.	• Sitting on the chair • Not touching the chair • Using hands to support their weight
Hip abduction (HAA)	Stood up, the right leg is lifted straight out to the side in their frontal plane, while keeping the left leg still - both legs remaining straight.	• Deviations from the frontal plane • Knees bent • Insufficient motion control
Elbow flex-extension (EFE)	Both arms move from straight to maximum elbow flexion in sagittal plane, shoulders stationary.	• Unilateral arm movement • Directing the force towards the back • Deviations from the sagittal plane
Extension of arms over head (EAH)	Both arms start together with straight elbows and make an arch until reaching the maximum elevation of the hands.	• Unilateral arm movement • Arms raised below head level • Moving both hands apart
Squeezing (SQZ)	Perform an anti-symmetric squeezing motion of the fabric while holding it in front of them with straight arms.	• Unilateral wrist movement • Rotating the wrists in different directions.

Table 1: Exercises included in the database, that are also used in this study, together with descriptions of their proper and improper execution [4].

3.2 Signal Processing

In order to achieve a more accurate and comprehensive representation of an object's movement and orientation in space, we will harness the full capabilities of the accelerometer, gyroscope, and magnetometer sensors. The accelerometer measures linear acceleration, which denotes the rate of change of velocity in a straight line, while the gyroscope measures angular velocity, which refers to the rate of change of angle or orientation [8]. The magnetometer, on the other hand, gauges the strength and direction of magnetic fields. Employing all three types of sensors allows us to obtain a more comprehensive and accurate representation of an object's movement and orientation in space, as each sensor provides distinct information that can be combined to produce a more complete picture of the object's motion. For example, accelerometers solely provide information about linear acceleration, but when paired with gyroscope data, we can determine the object's angular velocity and, by integrating over time, its orientation. Meanwhile, magnetometer data can be leveraged to provide a reference point for the object's orientation relative to the Earth's magnetic field. By tapping into the unique insights provided by each sensor, we can ensure a more nuanced and sophisticated understanding of the object's movement, leading to more precise and reliable recognition of different exercises.

3.2.1 Signal Segmentation

Once the CSV file is loaded into a 2-dimensional array where each row is a single signal and contains 10 elements, which represent the sensor readings for that signal, we segment the data into sliding windows. Segmenting signal data into sliding windows is a common practice in signal processing and analysis, especially for time-series data such as sensor measurements. The primary reason for this is to deal with the problem of non-stationarity [9], which refers to the fact that the statistical properties of the signal may change over time. By segmenting the signal data into small time intervals, or windows, we can assume stationarity within each window, which simplifies the analysis and allows for more accurate feature extraction and classification. Sliding window segmentation allows for a more fine-grained analysis of the signal data. By breaking the data into smaller segments, we can capture the temporal variations and patterns within the signal at a higher resolution. This helps to ensure that important information in the signal is not lost or smoothed over by the analysis. By sliding the window over the signal, we can extract features at different time points, which can be used to capture the evolution of the signal over time and improve the accuracy of activity recognition.

We will consider five distinct window sizes, with sample counts spanning from 100 to 500, corresponding to the application of windows within the range of 1 to 5 seconds. There are both advantages and disadvantages to using smaller and larger window sizes when segmenting time series data [10]. Smaller window sizes, such as 1-2 seconds, allow for finer-grained analysis of the data, capturing more nuanced changes and fluctuations within the signal. This level of detail can be particularly useful when analysing complex movements or activities that involve rapid changes in motion, such as dancing or martial arts. Additionally, smaller window sizes can help reduce the risk of data loss or distortion due to signal noise or drift, as they limit the amount of time that any one segment of the signal is analysed. On the other hand, larger window sizes, such as 3-5seconds, offer a broader perspective on the data, capturing more general trends and patterns over a longer period of time. This can be particularly useful when analysing activities that involve slower, more sustained movements, such as physical therapy exercises. Larger window sizes can also help reduce the computational complexity of the analysis by reducing the number of individual segments that need to be analysed, potentially improving the speed and efficiency of the classification process [11]. The optimal window size will be chosen empirically by testing the range of window sizes and evaluating the performance of the model based on the highest accuracy and other performance metrics.

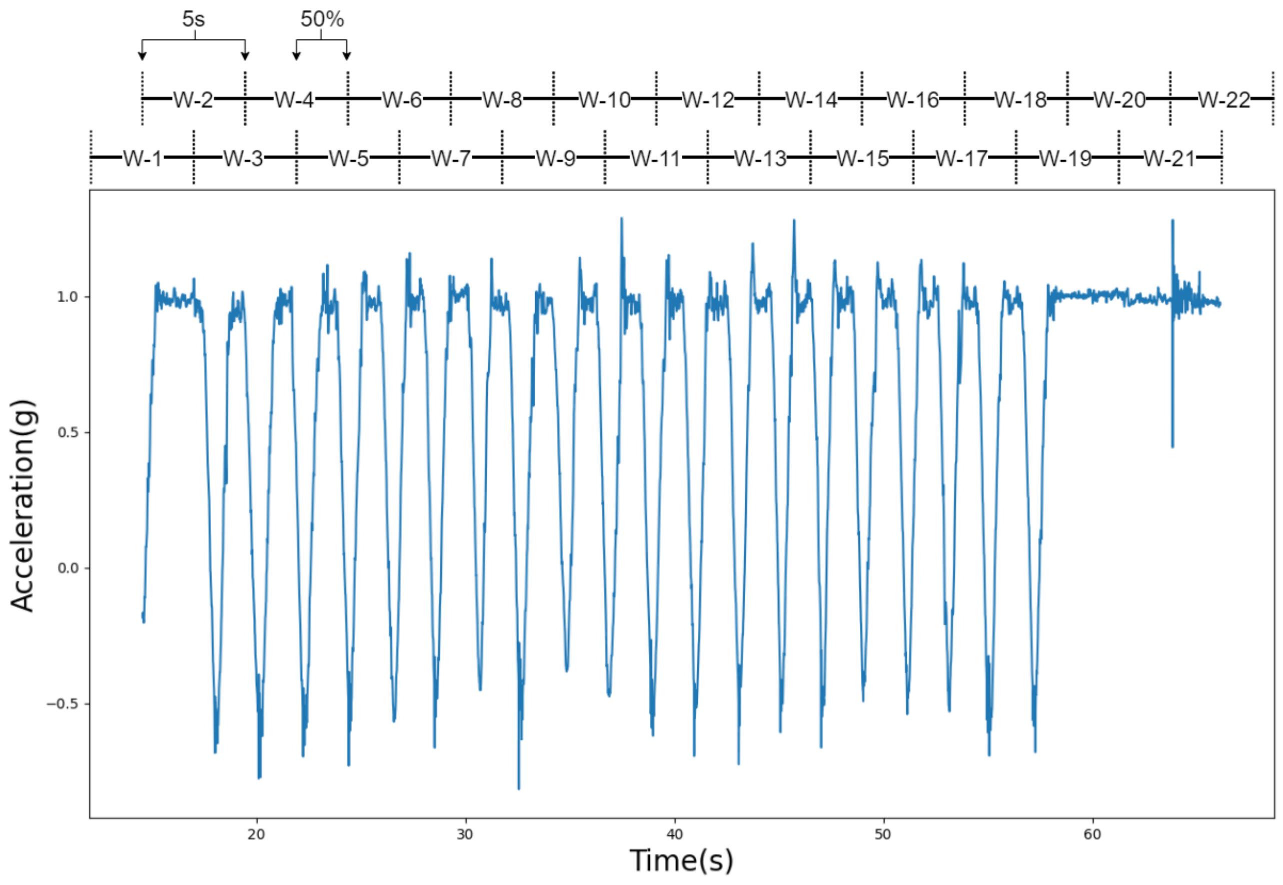


Figure 2: This plot depicts the acceleration-time graph of the elbow flex-extension physical therapy exercise captured from the x-axis of the accelerometer. The data is segmented using 5-second windows with 50% overlap.

We will also consider four different window overlap sizes: 0%, 25%, 50%, and 75%. By utilizing different amounts of overlap, we can observe the effects of varying amounts of correlation between adjacent window segments. Utilizing a smaller window shift can be advantageous in detecting minuscule changes in the signal, that might otherwise be overlooked with a larger shift. This can facilitate precise feature extraction and classification, potentially resulting in higher accuracy. However, this benefit is counterbalanced by the increased computational complexity and lengthier processing times associated with smaller window shifts, which can pose a challenge when dealing with a large volume of data. Additionally, smaller window shifts may increase the risk of overfitting and reducing the generalization of the model due to the potential for introducing noise into the data. On the other hand, a larger window shift offers the advantage of requiring less data to be processed, thereby resulting in faster processing times, and decreased computational complexity. Moreover, it can help reduce the amount of redundant information captured in consecutive windows. However, a larger shift can miss crucial signal changes that occur between windows, which may be significant for accurate feature extraction and classification. Consequently, the larger window shift could result in lower classification accuracy due to incomplete signal representation. Finally, when using non-overlapping windows, each window starts exactly where the previous window ended. One benefit of non-overlapping windows is that they simplify the data processing pipeline by avoiding the need for overlap computation - reducing the computational complexity of the system. However, non-overlapping windows have some limitations. One major drawback is that they can result in loss of information due to the abrupt change between segments, which can affect the accuracy of feature extraction and classification [12].

3.2.2 Feature Extraction

In activity recognition, a critical step is to extract relevant features from the signal that accurately represent the underlying activity patterns. These features serve as inputs for the machine learning model, enabling it to differentiate between different activities. The chosen features must strike a balance between discriminative power and computational efficiency. To this end, two main types of features are extracted: time-domain features and frequency-domain features [13]. The former provide insight into the statistical characteristics of the data over time, offering crucial information about the temporal behaviour of the signal. These features can capture important properties such as amplitude, variability, and patterns of the signal. On the other hand, frequency-domain features capture the distribution of signal power across different frequency bands. These features can provide insight into the oscillatory behaviour of the signal, offering a different perspective on the signal characteristics.

Choosing the appropriate set of features is critical to achieving high accuracy in activity recognition, as it enables the machine learning model to effectively capture the underlying activity patterns. However, we must also be cautious about overfitting the features [14]. If the features are too specific to the training data, they may not generalize well to new, unseen data, resulting in high accuracy during training but poor performance on new data. To avoid overfitting, it is essential to carefully select the set of features and ensure that they capture the most relevant information about the activity patterns without incorporating noise or idiosyncrasies unique to the training data. This requires striking a balance between discriminative power and generalizability in feature selection. By selecting a set of features that achieves this balance, the model can achieve high accuracy while avoiding overfitting.

Features to be extracted from each window:

- **Mean:** provides a measure of the central tendency of the data and is useful for characterizing the average behaviour of a signal. In activity recognition, mean values can help to distinguish between different activities by capturing the differences in movement patterns.
- **Standard deviation:** a statistical measure that provides information on the level of variability or noise in the data. This feature can capture the changes in the pattern of sensor data over time, which is particularly useful in activity recognition.
- **Maximum and minimum:** represents the upper and lower bounds of the data range, respectively. These features can provide valuable insights into the nature and intensity of the signal. For instance, a high maximum value of an accelerometer signal during a window may suggest

a sudden or vigorous movement, while a low maximum value may indicate a more gradual or subdued movement.

- **Median:** provides a robust measure of the central tendency that is less influenced by outliers compared to the mean. This makes it a useful feature for capturing the typical behaviour of a sensor signal. Its resistance to extreme values makes it a valuable feature for capturing the typical behaviour of a sensor signal. Furthermore, the median's ability to reveal information about the symmetry and shape of the distribution of a signal makes it relevant for specific activity recognition tasks.
- **Skewness:** a measure that describes the symmetry of the data distribution and identifies the presence of outliers. It can be particularly relevant for certain activities where a non-normal distribution is expected, such as those involving sudden and jerky movements. By analysing the skewness of the accelerometer readings, it is possible to identify the extent to which the distribution deviates from normal and infer the underlying characteristics of the activity.
- **Kurtosis:** a measure that evaluates the shape of a distribution of a specific feature within a data window, indicating the degree of peakedness or flatness compared to a normal distribution. A higher kurtosis value suggests a more peaked distribution, while a lower value indicates a flatter distribution.
- **Mode:** a measure that represents the value that occurs most frequently in a distribution, providing valuable insight into the most common or typical value for a specific feature within a given window of data.
- **Correlation coefficient:** a measure of the degree of linear relationship between two signals captured within the same time window. This measure reflects the similarity of the two signals within the given time window, revealing whether they are positively or negatively correlated. The correlation coefficient is a valuable feature that can provide insight into how strongly two variables are related to each other.

3.3 Data Labelling

Data labelling is an essential step in machine learning that involves assigning meaningful and descriptive labels to a given dataset, thereby enabling the machine learning model to learn from the data and make accurate predictions on new, unseen data. The labelling process involves annotating data with labels that correspond to specific characteristics of the data. The data labels we create include information on the exercise being performed and whether the exercise is correctly or wrongly performed. As we are using six physical therapy exercise for our classification, the output of this process includes twelve labels, comprising six labels for each exercise, representing both the correct and incorrect forms.

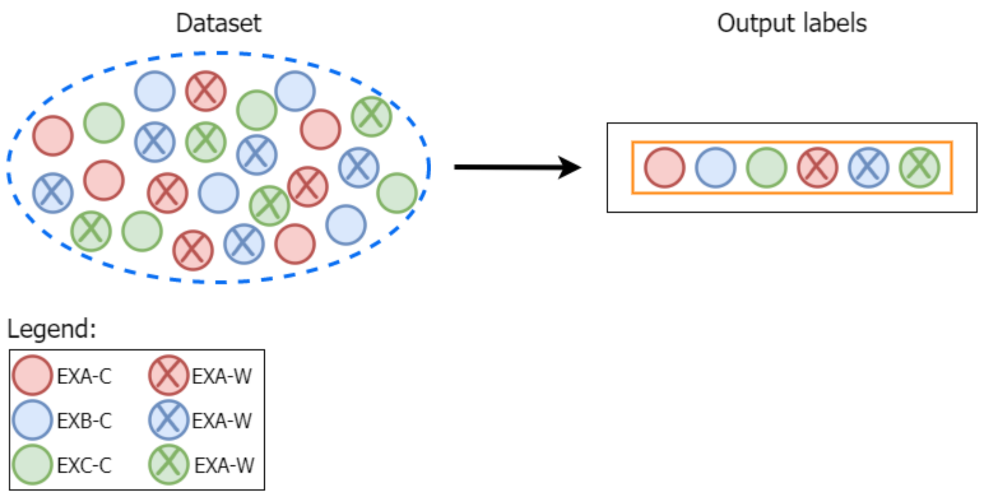


Figure 3: Illustration of the classification approach for the recognition and evaluation of exercises used. Each hollow circle represents a correctly performed exercise. Each circle with a cross represents an incorrectly performed exercise.

3.4 Machine Learning Algorithm

To recognize exercises, we will be utilizing a SVM - a supervised machine learning algorithm extensively employed in classification tasks [15]. SVMs are particularly adept at handling high-dimensional data and are resilient to noise and outliers. Given that the data will be acquired from various sensors placed on different parts of the body, a high-dimensional feature space will be generated. SVMs detect a hyperplane that maximizes the separation of classes in this feature space, leading to the identification of the decision boundary that best distinguishes between the classes. As a result, SVMs are well-suited for multi-class classification challenges. Furthermore, in cases where classes are not linearly separable, which is often the scenario with exercise recognition, SVM has been demonstrated to deliver outstanding results. Essentially, the decision boundary that distinguishes between various exercises may not be a linear function, but a complex nonlinear boundary. By training SVM to discover the optimum nonlinear decision boundary, we can attain high classification accuracy.

In contrast, other classifiers such as k-Nearest Neighbours (KNN) and Decision Trees (DT) have limitations in handling high-dimensional data and dealing with noisy data. KNN is sensitive to irrelevant features and noisy data, and decision trees can be prone to overfitting, which reduces their generalization ability. Several studies have shown SVMs to outperform other classifiers in exercise classification tasks. For example, in a study by Villa et al. [7] SVMs outperformed KNN, DT, and Random Forest (RT) classifiers in recognizing eight different exercise classes using data from wearable sensors. Similarly, in a study by Preatoni et al. [16] SVMs achieved higher classification accuracy compared to KNN in recognizing four different functional training drills using data from wearable sensors.

3.5 Evaluation Method

To evaluate the performance of our machine learning model, we have opted to use the leave-one-out cross-validation (LOOCV) technique. This approach involves training the model on all but one sample in the dataset and testing it on the left-out sample. This iterative process is repeated for each sample in the dataset, and the performance metrics are then averaged over all iterations. LOOCV's advantage over other validation techniques is that it utilizes all available data to train the model, which helps improve the accuracy of the model by minimizing overfitting. Furthermore, LOOCV provides an unbiased estimation of the model's performance on new, unseen data, which is essential in assessing the model's generalization ability [17].

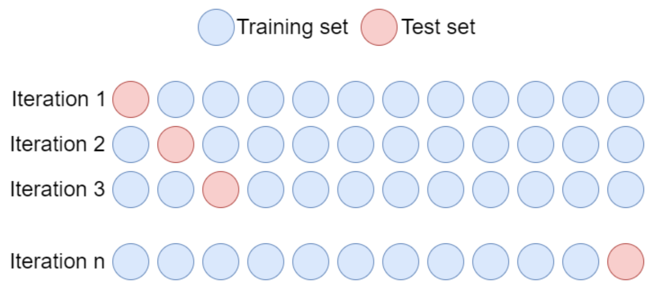


Figure 4: Leave-one out cross validation.

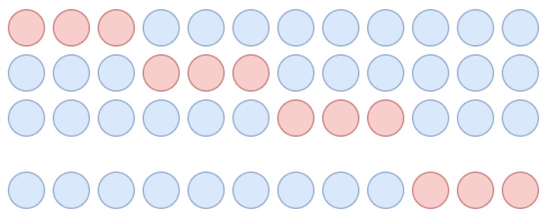


Figure 5: K-fold cross validation.

Compared to k-fold cross-validation, LOOCV offers more significant advantages. For example, k-fold cross-validation randomly splits the dataset into k-folds, using k-1 folds to train the model and one fold to test it. This process is repeated for each fold, and the performance metrics are averaged over all iterations [18]. However, since only a fraction of the data is used for testing at a time, this can introduce randomness into the results, depending on the specific way in which the data is divided. Furthermore, k-fold cross-validation requires the selection of the number of folds, which can be challenging in practice, and may lead to different results depending on the choice of k.

In contrast, LOOCV provides a more robust estimation of the model's performance, as it uses all available data for testing, and avoids the need to choose the number of folds. As such, LOOCV is often considered a gold standard for evaluating machine learning models, as it provides an accurate and unbiased estimate of the model's performance. In summary, LOOCV is a more accurate and robust validation technique compared to k-fold cross-validation, making it an excellent choice for evaluating machine learning models as shown by Meijer et al. [19].

The project's success criteria will be demonstrated through the evaluation stage, where performance metrics will be utilized to determine the model's success. The evaluation process will employ LOOCV to calculate various performance metrics that reflect the model's effectiveness. The model will be evaluated in terms of the following performance metrics:

- **Accuracy:** measures the proportion of correctly classified samples (correctly labelled as correct or incorrect out of all samples. It gives an overall idea of the model's performance in correctly classifying samples as correct or incorrect.
- **Precision:** measures the proportion of true positives (samples correctly labelled as correct) out of all samples classified as correct. Precision indicates the model's ability to avoid false positives.
- **Sensitivity:** measures the proportion of true positives out of all actual positives (samples that are actually correct). Sensitivity indicates the model's ability to correctly detect correct samples.
- **F1-score:** a harmonic mean of precision and sensitivity, which balances both measures. It is a useful metric when the classes are imbalanced and provides a more comprehensive assessment of the model's performance.
- **Specificity:** measures the proportion of true negatives (samples correctly labelled as incorrect) out of all actual negatives (samples that are actually incorrect).

Drawing on comparable studies, we anticipate that the success of this project will hinge on attaining a minimum threshold of 90% for all the metrics outlined above. To contextualize this target, it is worth noting that Attal et al. [20] reported a performance of over 92% for all metrics when employing an SVM, while Villa et al. [7] achieved a success rate exceeding 93% for all metrics using a similar approach.

4 Development

4.1 Initial Data Development

To efficiently collect relevant data for our model from the extensive PHYTMO database, we utilize list comprehension to generate a list of CSV file paths. To achieve this, we employ the versatile *product* function from the *itertools* module [21] to generate all possible combinations of relevant factors including age groups, sensor positions, exercise names, individuals, sets, and form correctness. After generating these combinations, we validate the existence of the CSV file for each combination and append the path to the list of files to be processed.

4.2 Signal Processing Development

4.2.1 Segmentation

The input data files are segmented into smaller windows of a specified window size and window overlap between adjacent windows. To segment the data, the function creates an array called *segmented_data* with the dimensions of *(n_windows, window_size, data.shape[1])*. Here, *n_windows* represent the number of windows that can be created based on the specified window size and window

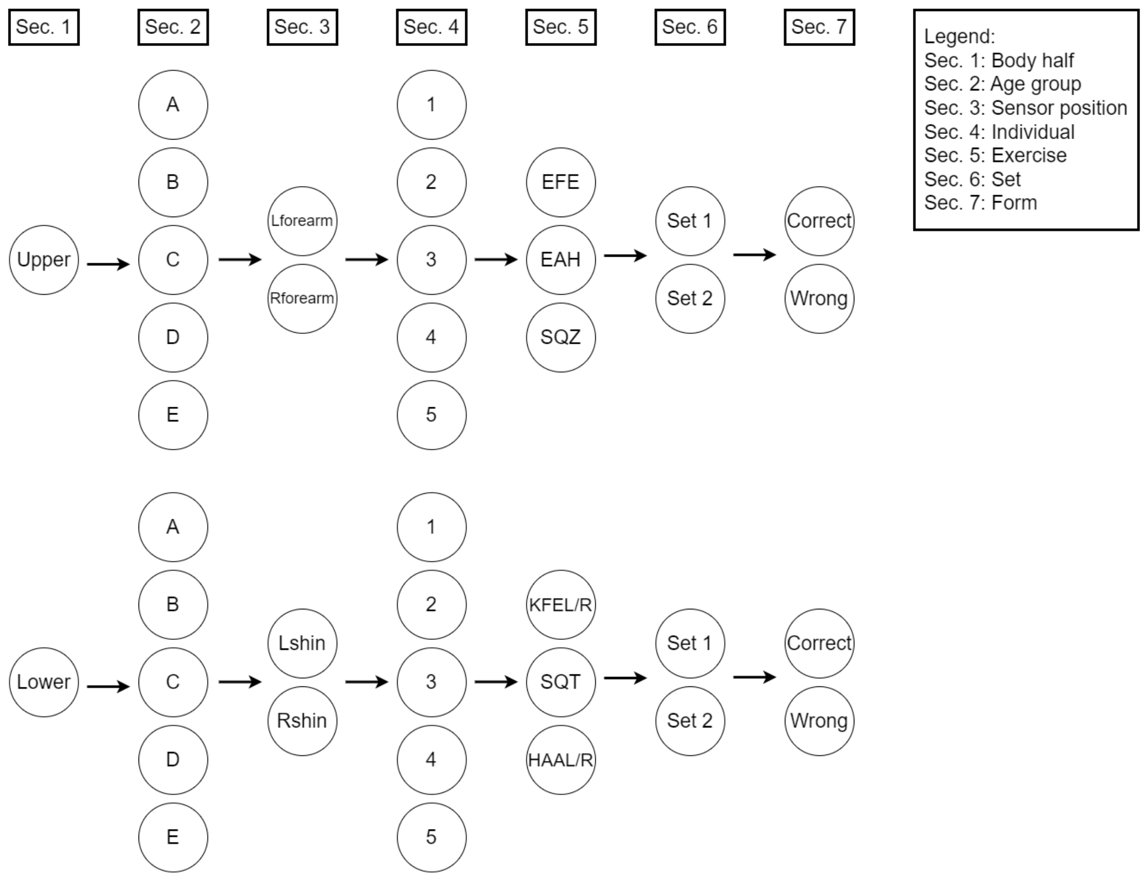


Figure 6: Illustration of the structure of the files

overlap. On the other hand, *data.shape[1]* corresponds to the number of sensor signals being used in the analysis. In this case, the value of *data.shape[1]* is 9, which accounts for the 3-axis readings of accelerometer, gyroscope and magnetometer. A loop is then used to iterate through each window of the input data. For each window, the corresponding segment of data is extracted from the input data and added to *segmented_data* using array slicing.

4.2.2 Feature Extraction

In order to obtain the features required for our machine learning model, we first obtain an array of windows from the segmented data of an exercise, and then extract a list of features from each window using time and frequency features. The time feature's function calculates the time-domain features of the windowed signal in each axis. This includes the mean, standard deviation, maximum, minimum, median, skewness, kurtosis, and mode. Furthermore, the function computes pairwise correlations between each axis using the *np.corrcoef* function.

The frequency feature's function utilizes the Welch method to compute the power spectral density (PSD) of the windowed signal in each axis [22]. The *welch* function from the *scipy.signal* module [23] is used to compute the PSD. The Welch method is a widely used signal processing technique that involves dividing a longer signal into overlapping segments and then computing a modified periodogram for each segment. These individual periodograms are then averaged to obtain an estimate of the overall power spectrum. This method significantly reduces the variance of the estimate of the power spectrum by averaging over multiple segments [24]. The function computes the same type of statistical features as the time domain features, except for the mode and the correlation coefficient. The mode is not suitable for use in frequency features because it is a measure that is specific to discrete data. The frequency features, on the other hand, are calculated based on the PSD of the signal, which is a continuous function of frequency. Similarly, the correlation coefficient is not applicable to frequency-domain analysis because it does not consider the spectral properties of the signals.

Regarding results, the Welch method is particularly useful for analysing non-stationary signals where the frequency content of the signal changes over time. By dividing the signal into shorter segments and computing the PSD for each segment, the Welch method is capable of capturing changes in the frequency content of the signal over time. In contrast, the time-domain features computed by the time feature's function provide information about the statistical properties of the signal over time but do not capture changes in frequency content [25].

The final features array exhibits a 2D structure, namely *n_samples* and *n_extracted_features*. The former refers to the number of files included in the dataset, each uniquely identified by a path, as illustrated in Figure 6. The value of *n_samples* is determined by summing the number of all possible paths in the figure, which amounts to 1200 prior to preprocessing. The latter dimension, *n_extracted_features*, represents the total number of features extracted from each file. It is computed by multiplying the number of features (195) by the number of windows. The number of windows is determined by several factors, including signal length, window size, and window shift.

4.2.3 Labelling

In order to match a label with its corresponding feature, we had to take into account the two factors that define an exercise: the type of exercise and its form correctness. To achieve this, we created a label that consists of two parts. The first part is a number from 1 to 6 that corresponds to the type of exercise, while the second part is either 0 or 1, representing correct and incorrect form respectively. To concatenate these two parts into a single label, we multiply the first part by 10 and add the second part. We start the first part from 1 rather than 0, as multiplying by 0 would result in a label of 0. We extracted this information from the file names in the same way as described in section 4.1, ensuring that the order of the features corresponds to the order of their corresponding labels.

		Physical Therapy Exercises					
		EAH	EFE	SQZ	HAA	KFE	SQT
Form	Correct	10	20	30	40	50	60
	Wrong	11	21	31	41	51	61

Table 2: Physical therapy exercises and their respective labels.

4.3 Machine Learning Development

4.3.1 Pre-processing

After extracting the features and labels for each exercise and storing them in lists, the data is pre-processed before feeding it into the SVM. First, the maximum length of the arrays in the features list is determined using the *max()* function [26] with a generator expression that iterates over the features list and returns the length of each array. Next, each array in the features list is padded with zeros using *np.pad()* [27] so that it has the same length as the maximum length. The padded arrays are stacked into a single 2D array using the *np.vstack()* function [28], which ensures that each instance has the same number of features and a consistent shape that can be fed into the machine learning algorithm.

Next, any rows from the features array that contain *NaN* values are removed using the *np.isnan()* function [29] with the .any() method [29] to find rows that contain at least one *NaN* value. The resulting Boolean mask is used to filter both the features array and the labels so that only rows without

NaN values are retained.

After removing *NaN* values, the maximum length of the arrays in the updated features array is determined, and the arrays are padded to that length using *np.pad()* again. The reason for using padding twice is to first ensure that all arrays within the features of an exercise have the same length by padding them with zeros. Then, the second padding ensures that all the arrays in the features array for all exercises have the same length by padding the shorter ones with zeros. Although it is possible to combine the two padding steps, this could result in more padding than necessary for some arrays, potentially increasing memory requirements and computational time. The current approach only pads the necessary elements to make all the arrays in the features array have the same length.

Finally, a *StandardScaler* [30] object is initialized and fitted to the padded features using the *fit_transform()* method. Standardization is an important step in machine learning because it ensures that all features are on the same scale, which is crucial for certain algorithms such as SVM. Standardization also transforms the data such that each feature has a mean of 0 and a standard deviation of 1, making it easier to compare and combine features and potentially leading to better performance [31]. In addition, machine learning algorithms often assume that the data is normally distributed, and standardization can help to ensure that this assumption is met.

4.3.2 Support Vector Machine

The SVM process is split into two functions that are used for training and evaluating the model using LOOCV. The first function is designed to take in two arguments: the feature array and the label array containing the features and labels of all instances, respectively. LOOCV process is implemented in this function by parallelizing the process using a multiprocessing pool. Specifically, *pool.starmap()* [32] is used to apply the second function to each instance in the feature array in parallel, and the resulting predictions are combined to produce the complete set of predictions. The utilization of a multiprocessing pool significantly reduces the training time of the SVM model, as LOOCV is computationally intensive, especially for large datasets. The pool creates a set of worker processes that can execute tasks in parallel, with each process handling a different subset of the data. We then evaluated the predictions using well-established metrics such as accuracy, precision, sensitivity, F1-score, and specificity, all of which are available in the *sklearn.metrics* module [30].

In the second function, the index i of the instance to predict, along with the feature array and the label array, are passed as arguments. A linear kernel SVM object is then instantiated, and the training features and labels are obtained by removing the ith instance from the features and labels arrays. The SVM model is subsequently trained on the remaining instances using *svm.fit()* [30], and the label of the ith instance is predicted using the trained SVM model and *svm.predict()* [30]. Finally, the predicted label is returned as a 1D array to ensure that it has the same shape as the original label array.

We experimented with different kernel functions to optimize the SVM hyperparameters. Kernel functions are used to transform the input data into higher-dimensional spaces where it may be easier to separate the classes [33]. The three types of kernels we considered are:

- **Linear kernel:** the simplest kernel function and works best when the data is linearly separable. It projects the data onto a higher-dimensional space where the classes are separable by a linear boundary. The linear kernel function is defined as:

$$K(x_i, y_j) = x_j \cdot x_i^T \tag{4.1}$$

- **Polynomial kernel:** is a non-linear kernel that works well when the data has a curved decision boundary. It adds polynomial terms of degree d to the linear kernel function to map the data into a higher-dimensional space. The polynomial kernel function is defined as:

$$K(x_i, x_j) = (1 + x_j \cdot x_i^T)^d \tag{4.2}$$

  where d is the degree of the polynomial.

- **Gaussian (RBF) kernel:** is also a non-linear kernel that maps the data into an infinite-dimensional feature space by considering all possible polynomials of all degrees. It is defined as:

$$K(x_i, x_j) = \exp(-\gamma * ||x_i - x_j||^2) \tag{4.3}$$

  where γ is a hyperparameter that controls the width of the Gaussian function.

The decision boundaries and the classification regions of different SVM classifiers with distinct kernel functions are depicted in Figure 7. These plots showcase the classification regions' background colours for each class, with the training points represented by points. These visualizations aid in comprehending the classification patterns of diverse SVM classifiers with various kernel functions, providing insights into the strengths and limitations of different kernel functions in SVM classifiers. Figure 8 demonstrates that the linear kernel exhibits the highest accuracy, and is hence used for the main model.

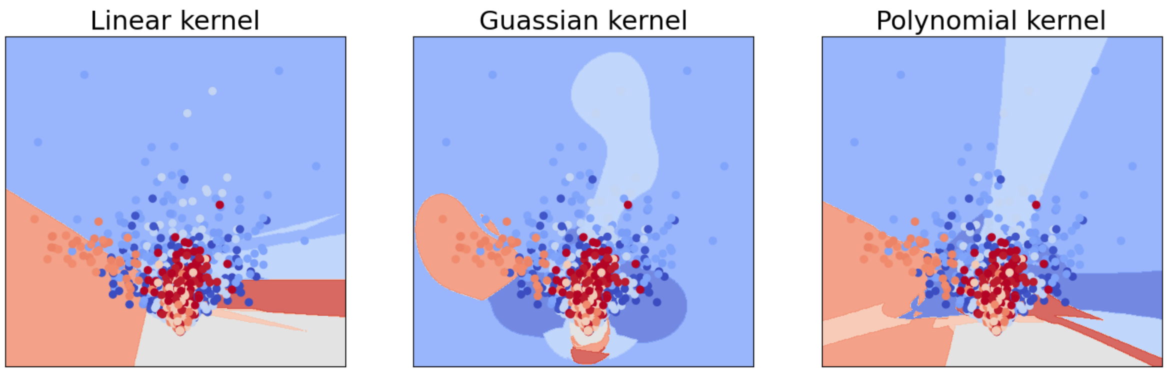


Figure 7: Comparison of different SVM kernels.

5 Results & Evaluation

5.1 SVM Kernel

After testing degrees 1 through 10 for the polynomial kernel, it was found that a degree of 1 provided the optimal results for the exercise classification model. This may be due to the fact that a degree of 1 results in a linear decision boundary, similar to that of the linear kernel. However, unlike the linear kernel, the polynomial kernel with a degree of 1 allows for the inclusion of non-linear terms in the decision boundary, providing a greater degree of flexibility in the model's classification capabilities. Furthermore, increasing the degree beyond 1 may result in overfitting, where the model becomes too complex and begins to fit to the noise in the training data rather than the underlying pattern. This can lead to a decrease in the model's generalization performance on new, unseen data. Therefore, the degree of 1 for the polynomial kernel strikes a balance between model complexity and flexibility, resulting in the optimal classification performance for the exercise classification task.

The linear kernel, as opposed to the polynomial and Gaussian kernels, is a simpler and more computationally efficient kernel. It operates by calculating the dot product between two vectors, which makes it well-suited for linearly separable datasets. In the case of exercise recognition, the linear kernel outperformed the polynomial and Gaussian kernels, as seen in Figure 8, because the exercise data is more linearly separable, meaning that the features of each exercise were more easily distinguishable from one another in a linear fashion. This would allow the linear kernel to accurately classify the exercises without the need for more complex and computationally expensive kernels. Furthermore, the polynomial and Gaussian kernels may have been more prone to overfitting the data, which can lead to poor generalization performance on new data. This can be especially problematic for exercise recognition, as the goal is to accurately classify exercises based on sensor data. Overall, the choice of kernel function for SVMs depends on the complexity of the dataset and the degree of non-linearity present in the data. In the case of exercise recognition, the linear kernel provided the best results due to the relatively simple and linearly separable nature of the exercise data.

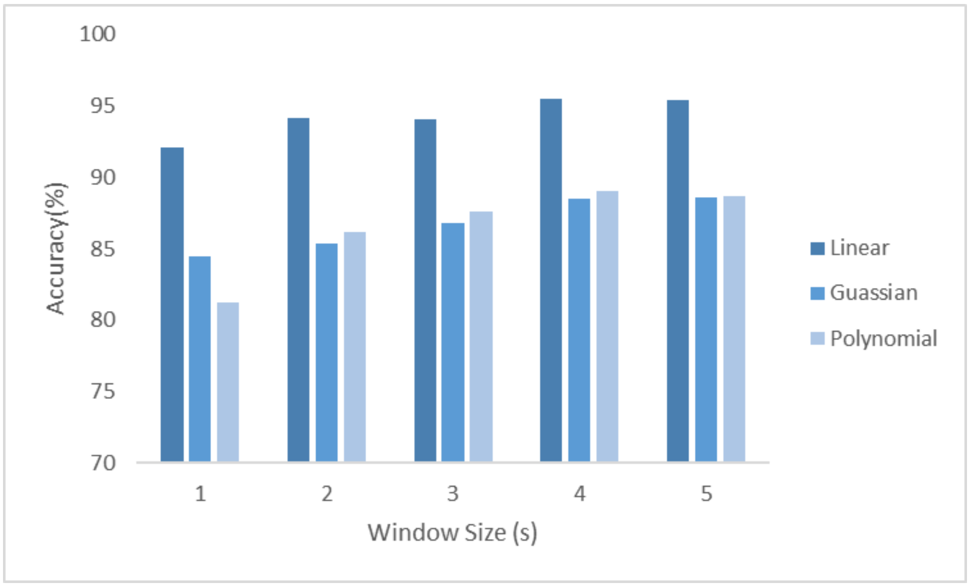


Figure 8: Accuracy of three different SVM kernel functions, namely linear, Gaussian, and polynomial, with varying window sizes. The window shift is set to 25%.

5.2 Window Size & Window Shift

To evaluate the performance of our model in classifying physical therapy exercises, we conducted a rigorous analysis to determine the optimal window length and window shift for achieving the highest accuracy. This involved testing each window size (ranging from 1-5 seconds) and window shift (ranging from 25-100%) for their accuracy, precision, sensitivity, f1-score, and specificity.

A higher overlap in windows means that the same data is included in multiple windows, resulting in redundant information. However, it has been observed that a higher overlap in windows leads to improved classification accuracy in exercise recognition tasks. In Table 3, it is observed that a window shift of 25% results in the highest metrics across all windows sizes. This can be attributed to two key reasons. Firstly, a higher overlap increases the amount of data that is used for training and testing the classifier. This can help to improve the generalization ability of the model, making it more robust to variations in the input data. By including more data points in each window, the model can capture more nuanced patterns in the sensor data, which can be crucial for accurately classifying different exercise movements. Secondly, a higher overlap can also help to reduce the effects of variability in the data. In exercise recognition tasks, there can be significant intra-class variation in the sensor data, which can make it challenging to distinguish between different exercise movements. By including more data points in each window and increasing the overlap, the model can average out some of this variability and better capture the underlying patterns in the data.

The choice of window size is an essential parameter in exercise recognition using machine learning from IMUs. There is an inherent trade-off between the window size and the amount of information captured. Smaller window sizes capture more detailed information but can result in noisy and incomplete data. Larger window sizes, on the other hand, provide a more stable and complete representation of the data but may lose important temporal information. The results in Table 3 and Table 4 indicate that larger window sizes yield better classification accuracy than smaller ones. Notably, the 4s window size achieved the highest evaluation metrics, with an impressive 95.5% accuracy. The 5s window size also performed well, with a close second at 95.4% accuracy. In contrast, the 1s window size had the lowest accuracy rate of 92.1%, suggesting that the smaller window size did not provide sufficient data to capture the complex exercise patterns. The trend observed in the results supports the notion that a larger window size provides a more comprehensive representation of the physical therapy exercise movements, enabling better identification and classification of exercise types.

Using larger window sizes, such as 4s and 5s, offers a notable advantage in exercise recognition as it enables the capture of more comprehensive exercise information, while simultaneously minimizing the effects of intra-class variability. By providing more data, it becomes easier to differentiate between distinct exercises, while accounting for small variations in executing a single exercise. This advantage is due to the fact that larger window sizes allow for a greater volume of data to be captured, thus providing a more detailed and accurate representation of the complex and nuanced nature of the exercise. This increased volume of data can help to account for the subtle variations that can occur in the execution of an exercise, while simultaneously enabling the recognition of key features of the exercise. Moreover, larger window sizes can be useful in reducing the impact of noise and variability in the data, resulting in a higher level of accuracy in exercise recognition.

In contrast to larger window sizes, smaller window sizes may prove inadequate in capturing the nuances of an exercise, leading to a reduction in classification accuracy. This is due to the fact that smaller windows are more susceptible to noise and incomplete data. Noise can stem from natural variations in movement patterns or sensor data errors, while incomplete data may result from the truncation of the start or end of a repetition within a window, resulting in the loss of crucial information. Although smaller window sizes may be advantageous for detecting fast and dynamic exercises, such as jumping jacks or burpees, that necessitate a higher sampling rate to capture all the required information, physical therapy exercises typically involve deliberate and controlled movements performed at a slower pace. Thus, a larger window size is imperative for capturing a more thorough understanding of the movements being performed.

Window Size(s)	Window Shift(%)	Accuracy(%)	Precision(%)	Sensitivity(%)	F1-Score(%)	Specificity(%)
1	**25**	**92.1**	**92.3**	**92.1**	**92.1**	**97.9**
1	50	91.9	92.2	91.9	91.8	96.9
1	75	90.9	91.1	90.9	90.9	96.2
1	100	90.7	91.0	90.7	90.7	94.5
2	**25**	**94.1**	**94.2**	**94.1**	**94.1**	**96.9**
2	50	93.7	93.9	93.7	93.7	96.9
2	75	93.2	93.3	93.2	93.2	95.8
2	100	92.5	92.7	92.5	92.5	95.8
3	**25**	**94.2**	**94.3**	**94.2**	**94.2**	**97.9**
3	50	94.1	94.1	94.1	94.1	96.9
3	75	93.4	93.5	93.4	93.4	97.9
3	100	92.6	92.8	92.6	92.5	97.6
4	**25**	**95.5**	**95.6**	**95.5**	**95.5**	**97.9**
4	50	95.2	95.3	95.2	95.2	97.9
4	75	93.9	94.0	93.9	93.9	96.9
4	100	92.7	92.9	92.7	92.7	96.9
5	**25**	**95.4**	**95.5**	**95.4**	**95.4**	**96.9**
5	50	94.3	94.3	94.3	94.2	96.9
5	75	94.3	94.4	94.3	94.3	95.8
5	100	91.9	91.9	91.9	91.9	93.7

Table 3: Results for the SVM classification on physical therapy exercises using a linear kernel. The most optimal row for each window size is highlighted in green.

The Decision Tree (DT) algorithm is commonly used as a baseline in machine learning tasks due to its simplicity and interpretability, which provides a reasonable initial performance benchmark. In contrast, the Random Forest (RF) algorithm is an ensemble method that leverages multiple decision trees to arrive at a final prediction. It does so by generating a set of decision trees, each using a random subset of the features and data. These trees are then trained independently, and each one makes a prediction based on its own rules. The final prediction is arrived at by taking the majority vote of all the individual predictions made by the trees.

As evidenced in Table 4, the RF algorithm is the second-best algorithm, with an accuracy of 94.2% achieved at the most optimal window size of 5 seconds. Similarly to the SVM models, the RF algorithm produced better metrics as the window size increased. RF has several advantages over DT, including the ability to handle missing values and noisy data more effectively, as well as reducing the issue of overfitting. This is because individual decision trees in the RF are trained on different subsets of the data, increasing the probability that each tree captures a different aspect of the data. This reduces the variance of the model and results in better generalization performance. Therefore, RF is a more robust and accurate model compared to DT in complex and noisy datasets.

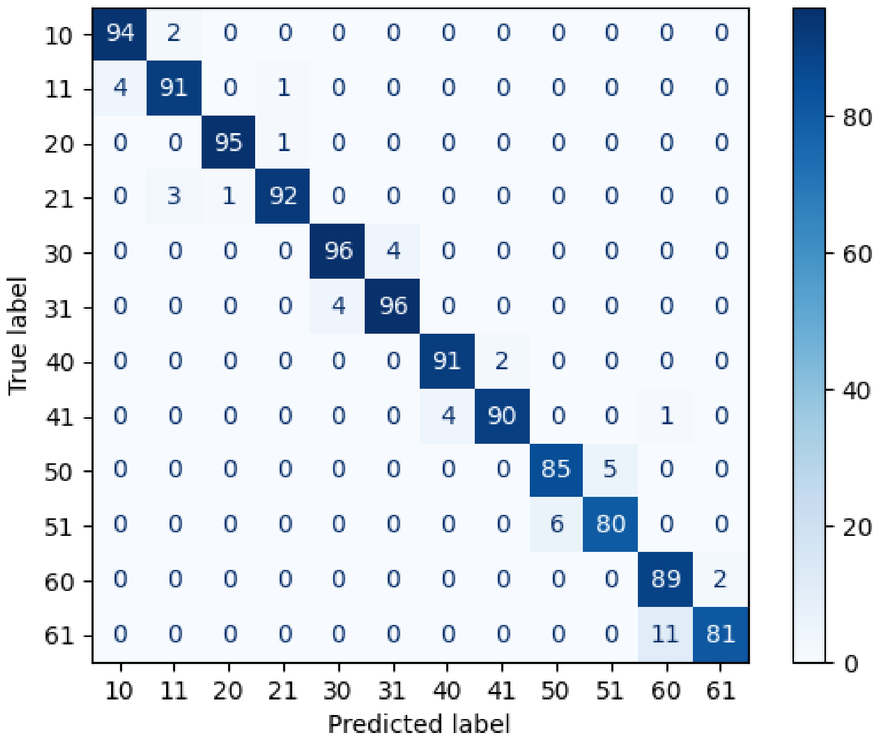
	Window Size (s)									
	1		2		3		4		5	
	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)
SVM_l	92.1	92.1	94.1	94.1	94.1	94.1	95.5	95.5	95.4	95.4
SVM_g	84.5	84.6	85.4	85.5	85.4	85.5	88.5	88.7	88.6	88.7
SVM_p	81.2	81.2	86.2	86.1	86.2	86.1	89.0	89.0	88.7	88.6
RF	88.9	88.8	92.5	92.5	92.5	92.5	93.9	93.9	94.2	94.2
DT	72.4	72.4	75.6	75.6	75.6	75.6	79.2	79.2	79.0	78.9

Table 4: Accuracy and F1-Score Results for SVM with Linear, Gaussian, and Polynomial Kernels, Random Forest, and Decision Tree, using Various Window Sizes with 25% Window Shift.

5.3 Confusion Matrix

The confusion matrix presented in Figure 9 depicts the performance of the SVM model with a window size of 4s and a window shift of 25%. The diagonal elements of the matrix represent the accurate predictions, while the off-diagonal elements represent incorrect predictions. This model demonstrates a remarkable accuracy rate of 95.5%, indicating that it has effectively distinguished between different exercises in the dataset and accurately identified the majority of the samples. Furthermore, a precision of 95.6% shows that the model has a high level of confidence in its positive predictions and only a small percentage of these predictions are false positives. The high sensitivity score of 95.5% indicates that the model is correctly identifying a significant portion of the positive samples in the dataset, resulting in a low false-negative rate. The F1-score of 95.5% reflects a balance between precision and sensitivity, indicating that the model correctly identified the positive class while minimizing the number of false positives and false negatives. A high specificity of 97.9% suggests that the model has excellent performance in correctly identifying negative samples. Overall, the results demonstrate that the model performs exceptionally well in distinguishing between different exercises and has a low rate of misclassification.

Figure 9 demonstrates the model's exceptional accuracy, as only two exercises were misclassified as another exercise. Specifically, the model classified an incorrect form of EAH as an incorrect form of EFE, and an incorrect form of HAA as a correct form of SQT. The misclassification of the incorrect form exercises can be attributed to the EAH being performed with such poor form that it appears similar to EFE, as both exercises involve similar motions, as shown in Figure 1. Additionally, the misclassification between HAA and SQT can be explained by the sensor position for all lower body exercises being on the shins. As minimal movement occurs in the shin area during a squat, the model may struggle to differentiate between the two exercises, causing confusion. The high misclassification rate between the correct and incorrect forms of the squat can also be attributed to this issue, as the limited details from the data recorded from the shin sensors make it challenging for the model to make accurate predictions.



	10	11	20	21	30	31	40	41	50	51	60	61
10	94	2	0	0	0	0	0	0	0	0	0	0
11	4	91	0	1	0	0	0	0	0	0	0	0
20	0	0	95	1	0	0	0	0	0	0	0	0
21	0	3	1	92	0	0	0	0	0	0	0	0
30	0	0	0	0	96	4	0	0	0	0	0	0
31	0	0	0	0	4	96	0	0	0	0	0	0
40	0	0	0	0	0	0	91	2	0	0	0	0
41	0	0	0	0	0	0	4	90	0	0	1	0
50	0	0	0	0	0	0	0	0	85	5	0	0
51	0	0	0	0	0	0	0	0	6	80	0	0
60	0	0	0	0	0	0	0	0	0	0	89	2
61	0	0	0	0	0	0	0	0	0	0	11	81

Figure 9: Confusion matrix showing the performance of the classification model tuned with the optimal window size, 4s, and window shift, 25%. Rows represent the true class labels, and columns represent the predicted class labels. Refer to Table 2 for the corresponding exercise names based on the provided labels.

6 Critical Assessment of Final Product

The ultimate criterion for the success of this project, as outlined in section 3.5, was to achieve a minimum threshold of 90% for all performance metrics. Through an extensive optimization process, we arrived at the most effective approach, which entailed using an SVM model with a linear kernel, a window size of 4s, and a window shift of 25%. Remarkably, this combination yielded an impressive result, with all performance metrics reaching or exceeding 95.5%, as indicated in Table 3. With a 25% window shift, we found that the utilization of a 4-second window size resulted in an impressive 3.6% improvement in model accuracy compared to the 1-second window size. Furthermore, we observed that the employment of a 4-second window size in conjunction with a 25% window shift produced a 3.0% boost in accuracy compared to the utilization of a 100% window shift (no overlap).

In comparing our optimal window length of 4s with other studies, it is noteworthy that similar findings have been reported. Tapia et al. [34] conducted a study using 30 physical gymnasium activities collected from 21 individuals and varied the window length from 0.5 to 17 seconds with 50% overlap. They found that a window length of 4.2s resulted in the highest recognition accuracy performance of 94.6%. Villa et al. [7] also tested window sizes from 1 to 3 seconds using 30 volunteers over a set of 8 exercises, achieving exercise evaluation between 93.6% and 100%. Again, their results align with ours, indicating that larger window sizes have a greater impact on exercise classification. Moreover, Wang et al. [35] explored the impact of sliding window length on indoor human motion modes and concluded that longer window sizes generally result in better recognition performance. However, blindly increasing window size may not always yield significant improvement after a certain cut-off window length. For an F1-Score of 99%, an average window size of 3.7s is required, while a window size of 2.5-3.5s with an F1-Score of 95% is recommended to reduce recognition latency. Therefore, our findings corroborate those of previous studies, indicating that the choice of larger window length is crucial in achieving high performance metrics for exercise recognition.

Regarding the significance of window shift in our model, we have demonstrated that a greater window overlap leads to superior accuracy results, particularly when compared to the absence of window overlap. In a study by Dehghani et al. [12], the efficacy of overlapping windows was investigated. Their research employed a window shift of 200ms, meaning that a 5s window shared 4.8s of data with the previous window, constituting a window shift of 4%. As a result of window overlaps, the model achieved an average performance increase of 10% compared to the same model that did not utilize window overlaps. This result is consistent with our findings, which also emphasize the significance of window overlap in our exercise classification model. Putra et al. [36] also explored the benefits of overlapping and non-overlapping windows in machine learning-based fall detection. Four distinct window overlaps ranging between 25% and 90%, as well as a non-overlapping sliding window, were tested on fall data using an SVM, and F1-Scores were calculated. Their findings revealed that the higher the overlap, the greater the F1-Score, with the highest overlap generating an F1-Score of 88%, and the lowest overlap having an F1-Score of 87.2%. However, the non-overlapping window had a lower F1-Score of 84% compared to the overlapping windows. Hence, our research substantiates the conclusions drawn by previous studies, highlighting the criticality of selecting a larger window overlap for obtaining superior performance metrics in exercise classification.

One of the key contributions of this study is its unique exploration of the joint impact of varying window size and shift on exercise recognition, which sets it apart from previous investigations in this domain. Specifically, while prior research has focused on examining either window size or shift, few if any studies have sought to investigate the joint effects of these parameters on the accuracy of physical therapy exercise recognition. By adopting this innovative approach, our study provides novel insights into the optimal configuration of window size and shift for achieving high-performance metrics in physical therapy exercise recognition using IMUs.

The objectives outlined in section 2.1 have been successfully achieved in this project. The efficacy of machine learning in classifying and assessing physical therapy exercises has been demonstrated by our impressive performance metrics, as displayed in Table 3. While Figure 9 indicates near-perfect exercise classification, the evaluation of exercise form remains a limitation of this study. The model's ability to accurately evaluate exercise form is crucial to ensure reliable predictions, thus its improvement is paramount to enhance the overall performance of the model. One potential approach to improving the model's evaluation of exercise form is to augment the quantity and quality of data utilized in the training process. This could include integrating additional sensor measurements or implementing more advanced machine learning algorithms such as deep learning. Another strategy involves refining the current feature extraction techniques employed by the model, thereby ensuring the most pertinent and informative features are being utilized. Ultimately, the model's ability to accurately assess exercise form will depend on various factors, and a multi-pronged approach may be necessary to achieve optimal performance.

The findings obtained through this project have the potential to significantly benefit the scientific community in multiple ways. Firstly, this study highlights the efficacy of employing IMUs for physical therapy exercise classification, which holds great promise for aiding patient rehabilitation and monitoring. This could serve as a catalyst for further research in this area, potentially resulting in the development of more sophisticated and precise exercise classification models that can be utilized in clinical settings. Secondly, this project offers valuable insights into the impact of varying window size and window shift on exercise recognition performance. By optimizing these parameters, this study was able to achieve remarkable performance metrics, setting a high standard for future research endeavors in this domain. Researchers can leverage this information to design more effective models for exercise recognition and classification. Thirdly, this study contributes to the rapidly expanding literature on machine learning in healthcare. As machine learning algorithms continue to evolve, they hold the potential to revolutionize the healthcare industry by enabling more precise and efficient diagnosis, treatment, and monitoring of patients. These findings offer valuable information on the use of machine learning in physical therapy, which can inspire further research in this area and lead to the development of more sophisticated and effective healthcare technologies. Overall, the insights gained through this project could have a profound impact on the scientific community and pave the way for significant advancements in physical therapy, machine learning, and healthcare as a whole.

6.1 Future Work

One potential avenue for future work involves the integration of the developed classification model into a wearable device, such as a wrist-worn sensor. This could enable real-time tracking of a patient's performance during physical therapy exercises, with immediate feedback provided to the patient via their smartphone. Such a system could greatly enhance the ability of patients to perform exercises correctly, reducing the risk of injury and potentially improving rehabilitation outcomes. To achieve this goal, several technical challenges would need to be addressed. For example, the developed model would need to be optimized for deployment on a low-power wearable device, with careful consideration given to the computational resources required. Additionally, the sensor measurements would need to be carefully calibrated and validated to ensure accuracy and consistency of the captured data. Furthermore, the implementation of the proposed system would require considerations beyond technical challenges. Ethical and privacy concerns surrounding the collection and storage of patient data would need to be addressed, and user-centered design principles would need to be employed to ensure the system is intuitive and user-friendly. Overall, the integration of the developed classification model into a wearable device has the potential to revolutionize the field of physical therapy by enabling personalized, real-time feedback to patients during exercise.

To further enhance the performance of the classification model and provide more personalized exercise programs, incorporating patient-specific data could be a promising future direction. This could involve enabling patients to input their personal information, such as age, sex, and medical history, which could be utilized to adapt the model to each patient's unique characteristics. Training the model with data collected from patients with different conditions could also prove advantageous, as different conditions may require different exercise regimes. As such, allowing the user to enter their condition could enable a more personalized evaluation of their exercise movement. Furthermore, leveraging the classification model to provide tailored feedback and guidance to each patient could lead to improved patient outcomes and greater engagement with physical therapy exercises. These potential future directions have the potential to further enhance the capabilities of the classification model and ultimately improve patient outcomes in physical therapy.

7 Conclusion

In conclusion, this project aimed to investigate the feasibility and effectiveness of using machine learning algorithms for the classification and evaluation of physical therapy exercises, and to develop a system that can evaluate and classify the correctness of physical therapy exercises using IMU sensors. The project also aimed to investigate the efficacy of different window lengths and overlaps and evaluate their impact on model performance.

To achieve these aims, raw IMU signals were acquired from the PHYTMO database, and sliding windows were used to segment the data, with window sizes ranging from 100 to 500 samples and varying window overlap sizes. Meaningful feature extraction was performed on the signal windows over each axis, and the samples were labelled as required by supervised learning algorithms. The features and labels were pre-processed for more effective classifier training, and the ML algorithm was trained for the classification and evaluation of physical therapy exercises. LOOCV was utilized on the model to calculate performance metrics such as accuracy, precision, sensitivity, F1-score, and specificity. The results obtained were then compared with related studies to evaluate the efficacy of the developed model.

The findings of this project demonstrate that using machine learning algorithms for the classification and evaluation of physical therapy exercises is both feasible and effective, and the developed system using IMU sensors can accurately evaluate and classify the correctness of physical therapy exercises. Additionally, the investigation into the efficacy of different window lengths and overlaps provided insights into the impact of these parameters on model performance. The use of LOOCV for model evaluation allowed for more reliable and unbiased assessment of the model's performance.

Overall, this project contributes to the growing body of literature on the use of machine learning algorithms for the classification and evaluation of physical therapy exercises and provides a promising approach for future research in this area. The developed model has the potential to assist clinicians and patients in accurately assessing exercise performance and improving rehabilitation outcomes.

References

[1] Burke Rehabilitation Hospital. *Physical Therapy*. `https://www.burke.org/medical-services/physical-therapy/`. Accessed on April 11, 2023.

[2] World Health Organization. *Ageing and Health*. `https://www.who.int/news-room/fact-sheets/detail/ageing-and-health`. Accessed on April 11, 2023. 2022.

[3] One Step. *3 Reasons Why Your Physical Therapy Isn't Working*. `https://www.onestep.co/resources-blog/3-reasons-why-your-physical-therapy-isnt-working`. Accessed on April 11, 2023. 2022.

[4] Sara García-de-Villa, Ana Jiménez-Martín, and Juan Jesús García-Domínguez. "A database of physical therapy exercises with variability of execution collected by wearable sensors". In: *Scientific Data* 9.1 (2022), p. 266.

[5] Alvaro Casas-Herrero et al. "Effect of a multicomponent exercise programme (VIVIFRAIL) on functional capacity in frail community elders with cognitive decline: study protocol for a randomized multicentre control trial". In: *Trials* 20.1 (2019), p. 362.

[6] María Romero-García et al. "Effect of a Multicomponent Exercise Program (VIVIFRAIL) on Functional Capacity in Elderly Ambulatory: A Non-Randomized Clinical Trial in Mexican Women with Dynapenia". In: *The journal of nutrition, health & aging* 25.2 (2021), pp. 148–154.

[7] Sara García-de-Villa et al. "Simultaneous exercise recognition and evaluation in prescribed routines: Approach to virtual coaches". In: *Expert Systems with Applications* 199 (2022), p. 116990.

[8] Naser El-Sheimy and Ahmed Youssef. "Inertial sensors technologies for navigation applications: state of the art and future trends". In: *Satellite Navigation* 1.1 (2020), p. 2.

[9] Eduardo Ogasawara et al. "Adaptive Normalization: A novel data normalization approach for non-stationary time series". In: 2010, pp. 1–8.

[10] Andreas Bulling, Ulf Blanke, and Bernt Schiele. "A Tutorial on Human Activity Recognition Using Body-Worn Inertial Sensors". In: *ACM Comput. Surv.* 46.3 (2014), p. 33.

[11] Miguel Ja'en-Vargas et al. "Effects of sliding window variation in the performance of acceleration-based human activity recognition using deep learning models". In: *PeerJ Computer Science* 8 (2022), e1052.

[12] Amir Dehghani et al. "A Quantitative Comparison of Overlapping and Non-Overlapping Sliding Windows for Human Activity Recognition Using Inertial Sensors". In: *Sensors* 19.22 (2019), p. 5026.

[13] Waltenegus Dargie. "Analysis of Time and Frequency Domain Features of Accelerometer Measurements". In: *2009 Proceedings of 18th International Conference on Computer Communications and Networks*. 2009, pp. 1–6.

[14] Xue Ying. "An Overview of Overfitting and its Solutions". In: *Journal of Physics: Conference Series* 1168.2 (2019), p. 022022.

[15] M.A. Hearst et al. "Support vector machines". In: *IEEE Intelligent Systems and their Applications* 13.4 (1998), pp. 18–28.

[16] Ezio Preatoni, Stefano Nodari, and Nicola Francesco Lopomo. "Supervised Machine Learning Applied to Wearable Sensor Data Can Accurately Classify Functional Fitness Exercises Within a Continuous Workout". In: *Frontiers in Bioengineering and Biotechnology* 8 (2020).

[17] Sylvain Arlot and Alain Celisse. "A survey of cross-validation procedures for model selection". In: *Statistics Surveys* 4.none (2010), pp. 40–79.

[18] Tzu-Tsung Wong and Nai-Yu Yang. "Dependency Analysis of Accuracy Estimates in k-Fold Cross Validation". In: *IEEE Transactions on Knowledge and Data Engineering* 29.11 (2017), pp. 2417–2427.

[19] Rosa J. Meijer and Jelle J. Goeman. "Efficient approximate k-fold and leave-one-out cross-validation for ridge regression". In: *Biometrical Journal* 55.2 (2013), pp. 141–155.

[20] Ferhat Attal et al. "Physical Human Activity Recognition Using Wearable Sensors". In: *Sensors* 15.12 (2015), pp. 31314–31338.

[21] Python Software Foundation. *Python itertools Documentation*. `https://docs.python.org/3/library/itertools.html`. Accessed on April 24, 2023. 2021.

[22] Pauli Virtanen et al. "scipy.signal.welch". In: *SciPy 1.7.0 Reference Guide* (2021).

[23] SciPy developers. *SciPy Reference Guide*. Accessed on April 24, 2023. 2021.

[24] P. Kumar Rahi and Rajesh Mehra. "Analysis of power spectrum estimation using Welch method for various window techniques". In: *International Journal of Emerging Technologies and Engineering* 2.6 (2014), pp. 106–109.

[25] Sana Ullah Jan et al. "Sensor fault classification based on support vector machine and statistical time-domain features". In: *IEEE Access* 5 (2017), pp. 8682–8690.

[26] Python Software Foundation. *Built-in Functions: max*. `https://docs.python.org/3/library/functions.html#max`. Accessed on April 24, 2023. 2021.

[27] NumPy Developers. *numpy.pad*. `https://numpy.org/doc/stable/reference/generated/numpy.pad.html`. Accessed on April 24, 2023. Aug. 2021.

[28] NumPy developers. *numpy.stack*. `https://numpy.org/doc/stable/reference/generated/numpy.stack.html`. Accessed on April 24, 2023. 2021.

[29] Python Software Foundation. *Built-in Functions: any*. `https://docs.python.org/3/library/functions.html#any`. Accessed on April 24, 2023.

[30] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[31] Caitlin A. Owen, Grant Dick, and Peter A. Whigham. "Feature Standardisation in Symbolic Regression". In: *AI 2018: Advances in Artificial Intelligence*. Cham, 2018, pp. 565–576.

[32] Python Software Foundation. *multiprocessing - Process-based parallelism*. `https://docs.python.org/3/library/multiprocessing.html`. Accessed on April 24, 2023. 2021.

[33] Arti Patle and Deepak Singh Chouhan. "SVM kernel functions for classification". In: *2013 International Conference on Advances in Technology and Engineering (ICATE)*. 2013, pp. 1–9.

[34] Emmanuel Munguia Tapia et al. "Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor". In: *2007 11th IEEE international symposium on wearable computers*. 2007, pp. 37–40.

[35] Gaojing Wang et al. "Impact of Sliding Window Length in Indoor Human Motion Modes and Pose Pattern Recognition Based on Smartphone Sensors". In: *Sensors* 18.6 (2018).

[36] I Putu Edy Suardiyana Putra et al. "An Event-Triggered Machine Learning Approach for Accelerometer-Based Fall Detection". In: *Sensors* 18.1 (2018).