Ben-Gurion University of the Negev
Faculty of Engineering Sciences
Department of Software and Information Systems Engineering

A machine-learning model for automatic detection of movement compensations in stroke patients

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE M.Sc DEGREE

By: Shir Kashi

August 2019

Ben-Gurion University of the Negev

Faculty of Engineering Sciences
Department of Software and Information Systems Engineering

A machine-learning model for automatic detection of movement compensations in stroke patients

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE M.Sc DEGREE

by: Shir Kashi

Supervised by: Prof. Lior Rokach, Dr. Shelly Levy-Tzedek and Prof. Boaz Lerner

Author:…….. Date:… 08/10/2019

Supervisor:.. …… Date:… 08/10/2019

Shelly Levy-Tzedek

Supervisor:…… Date:… 09/10/2019

Supervisor:…… …… Date:… 09/10/2019

Chairman of Graduate Studies Committee:…… Date:……

August 2019

Abstract

Motor rehabilitation is fastly becoming the last step in the chain of recovery from a neurological event. A leading cause of that is the increasing motor dysfunction impairment in more than 50% of the patients with stroke experience. Hence, when an individual suffers from an injury or illness such as a stroke, they usually undergo intense rehabilitation to restore motor function. During this process of rehabilitation after stroke, it is important that patients know how well they perform their exercise, so they can improve their performance in future repetitions. Standard clinical rating conducted by human observation is the prevailing way today to monitor the motor recovery of the patient. Therefore, a patient cannot know whether she is performing a movement properly while exercising by herself, e.g., at home. Adhering to the exercise regime makes the rehabilitation process more effective and efficient, and thus a system that can give the patient feedback on her performance is of great value.

Here, we report on two machine-learning-based automated models that we built. The first initial model was built to discern between stroke patients and healthy participants, whereas the second model, which represents our main purpose of this study, was to give patients accurate information on the compensatory (undesirable) movements that they make during a reach-to-grasp movement, in the absence of a therapist in the room. To construct those models, we used movement data recorded from 30 stroke patients and 16 healthy participants, who each performed 18 movements. Those movements were used to identify if the participant is a patient or healthy, and also to identify the presence of six types of compensatory movements in stroke patients' movement trajectories. In the second model, we used the RAkEL algorithm which concludes the random-forest algorithm for training this multilabel classification model. We achieved 85% macro-averaged precision across the six-movement compensations. This is the first study to automatically identify movement compensations based on stroke patients' data. We believe, this model can be adapted for use in in-clinic and at-home exercise programs for post-stroke patients.

Keywords— Compensations, machine learning, multi-label classification, RAkEL algorithm, random forest, stroke rehabilitation, time series

Acknowledgements

First, I would like to thank my advisor and mentor, Dr. Shelly Levy-Tzedek who has been continuously supportive since the days I began working in her lab as an undergraduate. Thanks for her patience, motivation, enthusiasm, and immense knowledge. And also, thanks for providing me extensive personal and professional guidance which helped me in all time of research. I would also like to extend my deepest gratitude to my advisor Prof. Lior Rokach, which his door was always open whenever I had a question about my research. He always steered me in the right direction whenever he thought I needed it. This work would not have been possible without his valuable comments and his tremendous academic support. Also, special thanks to my advisor, Prof. Boaz Lerner for his encouragement, insightful comments, and hard questions. Thanks for impressive attention to details and ideas, which has made this study more interesting, efficient, and effective. His contributions to the study were very helpful and meaningful.

Additionally, thanks to the Helmsley Charitable Trust through the Agricultural, Biological and Cognitive Robotics Initiative and the Marcus Endowment Fund, both at the Ben-Gurion University of the Negev for supporting the research. Also, special thanks to the Promobilia Foundation, the Borten Family Foundation, Israel Science Foundation, European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement for the financial supporting.

I want to thank Ronit Feingold-Polak for the significant professional guidance. Thanks for your patience and for always responding to any research questions I had about the field of physical therapy. Additionally, special mention goes to Anna Yelkin for collecting the patients' and controls' data and for her contribution to this study.

Finally, I must express my very profound gratitude to my parents and to my sister for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Thank you.

Contents

Acknowledgements 4
Introduction 9
1.1 The importance and limitations of self-exercise for stroke patients 9
1.2 From standard clinical rating to automated assessment 9
1.3 Our study 10
Background 12
2.1 Multi-label classification 12
2.2 Random Forest 13
2.3 XGBoost 15
2.4 The tsfresh package - Automated extraction of time-series features 17
2.5 RAkEL algorithm 19
2.6 Micro & Macro averaged precision 22
2.7 Hamming loss 23
2.8 AUC – Area under a ROC curve 23
2.9 LOOCV - Leave-one-out Cross Validation 25
Related work 27
Research Objectives and Expected Significance 30
4.1 Research Objectives 30
4.2 Limitations 30
4.3 Contributions 31
Method 32
5.1 Data Collection 32
5.2 Feature Generation 35
Experimental processes and algorithms 41
6.1 Experiment 1 - Distinguish between healthy and patient data 42
6.2 Experiment 2 - Movement-compensation detection 43
Evaluation Metrics and experimental Settings 45
7.1 Experiment 1 45
7.2 Experiment 2 45
Results 47
8.1 Experiment 1 47
8.2 Experiment 2 48

Conclusions and Future Work 55
References 59
תקציר 67

List of Figures

Figure 1. The evolution of tree-based algorithms over the years. 17
Figure 2. An example of the tsfresh feature extraction 19
Figure 3. The RAkEL algorithm 22
Figure 4. A ROC curve 24
Figure 5. The experimental setup for the data-collection phase 33
Figure 6. The location of the 11 body markers placed on each participant. 34
Figure 7. The process of movement segmentation 36
Figure 8. Example for the algorithm's output of the four time points that segment the movement. 37
Figure 9. The movement-compensation detection process 41
Figure 10. RAkEL algorithm using the 'random forest' classifier (k=4, M=16) 44
Figure 11. The ROC curve of the first experiment 47
Figure 12. Precision scores for each of the six compensations 48
Figure 13. Pairwise correlation between all compensation labels. 49
Figure 14. Pairwise correlation between all 156 features. 50
Figure 15. The micro-averaged precision of the 10 most informative features 52
Figure 16. The 'abs energy' feature. 53

List of Tables

Table 1. The handcrafted features.....39
Table 2. results of the first experiment 47
Table 3. Average predictive performance for different feature sets 48
Table 4. Micro averaged precision for the k parameter in the range [1,6] in the RAkEL algorithm.....49
Table 5. The 10 most-important features.....51
Table 6. The 5 most important features per table height (low/medium/high).....52
Table 7. The 5 most-important features per label.....53

Chapter 1

Introduction

1.1 The importance and limitations of self-exercise for stroke patients

The intensity and repetition of post-stroke training are key to the efficacy of the rehabilitation process [1]. Up to 77% of stroke survivors experience upper limb (UL) impairment, which affects their function and reduces health-related quality of life [2]. For effective and efficient rehabilitation of their upper limb functionality, self-exercising in between physical therapy sessions is vital, and yet, many patients do not follow their exercise regime, which can hamper their recovery [2]. One explanation for why compliance rates are low is that patients undergoing rehabilitation are not able to assess their own functional state and their performance without the therapist [3,4]. One of the major functional goals of rehabilitation after stroke is to retrain the coordination of reach-to-grasp (RTG) movements [5] (e.g., in order to pick up a cup to drink from). In individuals with stroke, goal-directed movements are characterized by slowness, spatial and temporal discontinuity and abnormal patterns of muscle activation and joint synergy [6-9]. Individuals with stroke were reported to have less smooth [7], less accurate and less efficient RTG movements compared to healthy individuals [10], as was measured by the index of curvature [11,12] and by the jerk [13,14] of their movements. Following a stroke, patients who are not able to coordinate their muscle-activation patterns to perform an RTG task as they did before they had a stroke, develop compensatory movement patterns – e.g., bending their trunk, rather than extending their elbow – to reach an object located at arm's length. Several such compensatory characteristics of movement have been described in RTG tasks, both in the trajectory and in the interjoint coordination of the movement [6,8].

1.2 From standard clinical rating to automated assessment

Cirstea and Levin (2007) demonstrated the importance of "knowledge of performance" in the upper limb rehabilitation process of stroke patients. That is, while the person is practicing RTG movements, she needs to know how well she performed the movement in order to improve her performance in future repetitions. The prevailing way to monitor motor recovery of the patient is carried out by direct human observation, using standard clinical ratings such as the Fugl

Meyer assessment (FMA), Functional Test for the Hemiplegic Upper Extremity (FTHUE), and the Brunnstrom stage [15-17]. The using of the FMA tool is found to be the most frequently and profitable compared to other scales such as the FIM and FTHUE. Padovani et al. (2016) found that 80% of the studies used this scale to evaluate the response for different types of therapies [10]. Moreover, according to Wang et al. (2014), the Brunnstrom approach is too coarse compared with the quantitative FMA since it just classifies stages of recovery into six stages. The Fugl-Meyer assessment is a representative evaluation tool that was developed to assess physical recovery following stroke survivors [16,24], particularly for the upper extremities [2,16,24]. It has been used to comprehensively evaluate upper extremity motor function, and it exhibited high reliability, validity, sensitivity and also can detect stroke recovery [3,16,24-32]. However, whereas those clinical scales are efficient tools, they also have some drawbacks [2,16-18,20]. The subjective judgement exercised by therapists using those tools may lack accurate quantifiable data [17], likely because it is difficult for the human eye to detect variations in a small-scale movement [20]. This makes it difficult to precisely evaluate gradual improvements in movement execution. In addition, the use of those tools is labor intensive and takes a considerable amount of time (at least 30 minutes) [2, 16, 18]. Furthermore, it is not suitable in the home settings as patients undergoing rehabilitation at home are not able to assess their own functional state with the FMA tool or other similar tools without a therapist [3,4]. In order to help stroke patients to continue the rehabilitation training after they leave the hospital, there is a need for automated assessment [4]. Automated assessment holds several benefits in comparison to assessment by human observation only in clinics [17]. For example, it can offer a more detailed tracking of the time course of recovery by identifying variations in their movements pattern [17,20] , it avoids the "test" situation which is often not representative of everyday function [17, 21], and also, it can make the assessment more objective by giving an automated score from a model instead of from a specific therapist [20]. In addition, the ability to make a quick and accurate evaluation could enable an efficient utilization of stroke-care resources, with clinician time being dedicated mainly to treatment, while the assessment is automated, even in the clinic [2, 18, 22, 23].

1.3 Our study

In this thesis, we used data from 30 stroke patients and 16 healthy participants, collected using a high-precision motion-capture system, to generate a multi-label classification model to detect movement compensations. To build such a model, it is necessary to choose the appropriate

method for this multi-label task, which is more complex than those having only a single outcome (i.e., a single compensation vs. multiple concurrent compensations). According to Tsoumakas and Katakis (2007), there are two main categories of multi-label classification methods: transformation methods and algorithm-adaptation methods. The first category transforms the multi-label classification problem into one or more single-label classification problem(s), while the second adjusts known single-label classifiers to handle multi-label data [33,34]. Since a main weakness of algorithm-adaptation methods is that they are mostly tailored to a specific classifier (e.g., SVM, or decision tree), they lack the ability to generalize, and thus the transformation methods perform better in this respect [33,34].

A paper on this work was submitted to a Special Section in IEEE Transactions on Emerging Topics in Computing (Q1, IF 4.9) by the authors: Shir Kashi, Ronit Feingold-Polak, Boaz Lerner, Lior Rokach, and Shelly Levy-Tzedek

Chapter 2

Background

2.1 Multi-label classification

Traditional single-label classification is concerned with learning from a set of examples that are associated with a single label λ from a set of disjoint labels L, |L| > 1. If |L| = 2, then the learning task is called binary classification while if |L| > 2, then it is called multi-class classification. In multilabel classification, the examples are associated with a set of labels $Y \subseteq L$ [33].

Multi-label classification is the supervised learning problem where an instance may be associated with multiple labels. This is opposed to the traditional task of single-label classification (i.e. multi-class, or binary) where each instance is only associated with a single class label [36]. There are two main categories of methods for multi-label classification: 1) problem transformation methods – methods that transform the multi-label classification problem either into one or more single-label classification or regression problems, and 2) algorithm adaptation methods – methods that extend specific learning algorithms in order to handle multi-label data directly [33].

Here, we will discuss the first method, which we used in this study, the problem-transformation method. There are five straightforward problem-transformation methods that force the learning problem into traditional single-label classification [33,37]. The first one subjectively selects one of the multiple labels of each multi-label instance and discards the rest. The second one simply discards every multi-label instance from the multi-label data set. These two problem transformation methods have a main a drawback, they discard a lot of the information content of the original multi-label data set [33].

The next three methods try to avoid discarding content that may be important in two different ways. The third problem-transformation method, called the Label Powerset (LP) approach, considers each different set of labels that exist in the multi-label data set as a single label. It so learns one single-label classifier $H: X \rightarrow P(L)$, where P(L) is the power set of L. One of the

negative aspects of this method is that it may lead to data sets with large number of classes and few examples per class [33]. The fourth method, called the Binary Relevance (BR) approach, dealing with this aspect by learning |L| binary classifiers Hl: X → {l, ¬l}, one for each different label l in L. It transforms the original data set into |L| data sets that contain all examples of the original data set, labelled as l if the labels of the original example contained l and as ¬l otherwise. It is the same solution used in order to deal with a single-label multiclass problem using a binary classifier [33]. For the classification of a new instance x this method outputs as a set of labels the union of the labels that are output by the |L| classifiers.
The last method is firstly, to decompose each example (x, Y) into |Y| examples (x, l) for all l ∈ Y. Then learn one single-label coverage-based classifier from the transformed data set. Distribution classifiers are those classifiers that can output a distribution of certainty degrees (or probabilities) for all labels in L. Finally, it post-processes this distribution to output a set of labels.

2.2 Random Forest

Random forest is considered an "ensemble learning" method, which is a method that generates many classifiers and aggregates their results. Two well-known methods of classification trees are boosting [38] and bagging [39]. In boosting, successive trees give extra weight to points incorrectly predicted by earlier predictors. In the end, a weighted vote is taken for prediction. In bagging, successive trees do not depend on earlier trees — each is independently constructed using a bootstrap sample of the data set. In the end, a simple majority vote is taken for prediction [40].

Random forests are grown using a collaboration of the bagging and the Iterative Dichotomiser 3 (ID3) principles (ID3 is a decision tree algorithm that is mainly used to produce classification trees). Each tree in the forest is grown in the following manner. Given a training set, a random subset is sampled (with replacement) and used to construct a tree which resembles the ID3 idea. However, every case in this bootstrap sample is not used to grow the tree. About one third of the bootstrap is left out and considered to be out-of-bag (OOB) data. Also, not every feature is used to construct the tree. A random selection of features is evaluated in each tree. The OOB data is used to get a classification error rate as trees are added to the forest and to measure the input variable (feature) importance. After the forest is completed, a sample can be classified by

taking a majority vote among all trees in the forest resembling the bootstrap aggregating idea [41].

Random forests, proposed by Breiman (2001), add an additional layer of randomness to bagging. In addition to constructing each tree using a different bootstrap sample of the data, random forests change how the classification or regression trees are constructed. In standard trees, each node is split using the best split among all variables. In a random forest, each node is split using the best among a subset of predictors randomly chosen at that node. Meaning, random forest only tests a limited number of the descriptors. Since the limited number is typically very small (the default in the software is the square root of the number of descriptors for classification), the search is very fast. Another difference between the standard decision tree and the random forest is the complexity. To get the right model complexity for optimal prediction strength, some pruning is usually needed for a single decision tree. This is typically done via cross-validation and can take up a significant portion of the computations. random forest, on the other hand, does not do any pruning at all. According to Svetnik (2003), in cases where there is an excessively large number of descriptors, random forest can be trained in less time than a single decision tree [41]. This somewhat counterintuitive strategy turns out to perform very well compared to many other classifiers, including discriminant analysis, support vector machines (SVM), and neural networks (NN) [40].

Another optional use of the random forest is to measure variable importance in a regression or classification problem in a natural way [40]. During the fitting process of the random forest to the data, the out-of-bag error for each data point is recorded and averaged over the forest [40]. In other words, if we change a single feature's input value and reclassify the record, we can determine that the feature's importance is based on the new classification. This is done using OOB data. Each variable m is randomly permuted and the permuted OOB cases are sent through the forest again. Subtracting the number of correctly classified cases using permuted data from the number of correctly classified cases using non-permuted data gives the importance value of variable m. These values are different for each tree, but the average of each value over all trees in the forest gives a raw importance score for each variable [42].

There are two more significant advantages of the random forest algorithm. First, they are parallelizable, meaning, that we can split the process to multiple machines to run. This results

in faster computation time. Boosted models are sequential in contrast and would take longer to compute. Also, each decision tree has a high variance, but low bias. But because we average all the trees in a random forest, we are averaging the variance as well so that we have low bias and moderate variance model [41].

2.3 XGBoost

Tree boosting was empirically proven to be a highly effective approach to predictive modeling. It has shown remarkable results for a vast array of problems [43,44]. More recently, a tree boosting method known as XGBoost has gained popularity by winning numerous machine learning competitions [43,44]. Xgboost is an improved algorithm based on the gradient boosting decision tree and can construct boosted trees efficiently and operate in parallel. The boosted trees in XGBoost are divided into regression and classification trees. The core of the algorithm is to optimize the value of the objective function. Unlike the use of feature vectors to calculate the similarity between the forecasting and history days, gradient boosting constructs the boosted trees to intelligently obtain the feature scores, thereby indicating the importance of each feature to the training model [43,44]. The more a feature is used to make key decisions with boosted trees, the higher its score becomes. The algorithm counts out the importance by “gain”, “frequency”, and “cover” [45]. “Gain” is the main reference factor of the importance of a feature in the tree branches. “Frequency”, which is a simple version of gain, is the number of a feature in all constructed trees and “Cover” is the relative value of a feature observation [45]. Tree boosting can be seen to adaptively determine the local neighborhoods of the model and can thus be seen to take the bias-variance tradeoff into consideration during model fitting. In addition, XGBoost introduces some subtle improvements which allow it to deal with the bias-variance tradeoff even more carefully [43].

One of the advantages of this algorithm is its handling of missing data [43]. The tree growing algorithm used by XGBoost treats missing values by learning default directions. At each node there are two possible directions, left or right. When data is missing, the default direction is taken. When there is missing data during training, the direction which minimizes the objective is learned from the data. [43].

To handle overfitting, there are three categories of regularization techniques applied in XGBoost [43]:

Boosting parameters

Boosting parameters are the number of trees M and the learning rate η [43]. Adding more trees (M), i.e. increasing the number of iterations, will increase the representational ability of the model. Selecting the number of trees M is thus crucial in order to achieve the right amount of representational ability. This can be viewed as the "main" tuning parameter of tree boosting. Moreover, the learning rate or shrinkage parameter η will generally shrink the added basis function at each iteration. Thus, by lowering the learning rate η, a larger number of trees can be added before the additive tree model will start to overfit the data [43].

Tree parameters

Tree parameters include constraints and penalties imposed on the complexities of the individual trees in the additive tree model [43]. These parameters can be seen to control the number of terminal nodes T in the tree, the size of the leaf weights w and the minimum sum of observation weights needed in a terminal node. To elaborate, XGBoost offers the possibility of constraining the complexity of the individual trees. Before XGBoost, complexity penalization was not commonly used for additive tree models. The penalization terms of the objective function can be written as follows [43]:

$$\theta(f) = \sum_{m=1}^{M} \gamma T_m + \frac{1}{2}\beta\|w_m\|_2^2 + \alpha\|w_m\|_1$$

The penalty is the sum of the complexity penalties of the individual trees in the additive tree model. There is a need to set those three parameters - γ, β and α in advanced, in a way it will lead to the best performance. The first term, T_m, is the number of terminal nodes of each individual tree m, the second is the L2 regularization on the term weights and the last one is the L1 regularization on the term weights [43].

Randomization parameters

randomization parameters control row and column subsampling [43]. By introducing randomization, the trees will indeed lose accuracy. However, they will also tend to be less

similar, i.e. more diverse. The fact that they are more diverse is beneficial when they are combined together. The positive effect of diversity often outweighs the negative effects of the lost accuracy of the individual trees. The tradeoff in selecting the right amount of diversity is known as the accuracy-diversity tradeoff in literature on ensemble methods [43].

Figure 1 demonstrates a chart of the evaluation of tree-based algorithms over the years. It stars from the Decision Trees which are a graphical representation of possible solutions to a decision based on certain conditions. The second algorithm is the Bagging, which is an ensemble meta-algorithm combining predictions from multiple-decision trees through a majority voting mechanism. Random forest is a Bagging-based algorithm where only a subset of features is selected at random to build a forest or collection of decision trees. In addition, it uses bootstrapping method for training/testing, which simply means generating random samples from the dataset with replacement. Boosting methods also produce multiple random samples, but it is done more thoughtfully. The subsequent samples depend on weights given to records in the previous sample which did not predict correctly - hence called weak learners. The final prediction is also not a simple average of all predictions, but a weighted average. Next, Gradient Boosting employs gradient descent algorithm to minimize errors in sequential models. At the end of this chart, there is the XGBoost algorithm which is an Optimized Gradient Boosting algorithm that works in parallel processing, perform tree pruning, handling missing values and use regularization to avoid overfitting/bias.

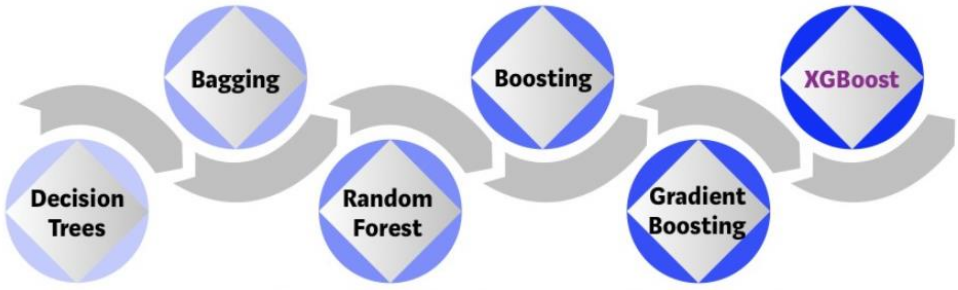


Figure 1. The evolution of tree-based algorithms over the years.

2.4 The tsfresh package - Automated extraction of time-series features

The tsfresh package generates about 3,000 time-series features automatically [46]. This package filters the features with respect to their significance for the classification task, while controlling the expected percentage of selected but irrelevant features. The features it extracts describe basic characteristics of the time series such as the number of peaks, average or

maximal value of a signal (Fig 2.), or more complex features such as the time reversal symmetry statistic. The features belong to one of three main categories [46].

1. Summary statistics- such as: maximum, minimum, mean, variance, standard deviation, skewness, kurtosis, length, median, quantile of empiric distribution.
2. Sample distribution - such as: absolute energy, augmented Dickey-Fuller test statistic, binned entropy, distribution characteristics, symmetry, mass quantile, number of data points above mean/median, and number of data points below median.
3. Observed dynamics – such as: autoregressive integrated moving average (ARIMA) model coefficients, continuous wavelet transformation coefficients, fast Fourier transformation coefficient, first index max, first index min, lagged autocorrelation, large number of peaks, last index max, last index min, longest strike above mean, longest strike above median , longest strike below mean , longest strike below median , longest strike negative , longest strike positive, longest strike zero, mean absolute change, mean absolute change quantiles, mean autocorrelation, mean second derivate central, number continous wavelet transformation peaks of size, number peaks of size, spektral welch density and time reversal asymmetry statistic [46].

For the sake of brevity, we elaborate here on seven of these features, which we found (chapter 6) to be most influential for the performance of our model:

1. Autocorrelation – Calculates the autocorrelation, where n is the length of the time series x_i, $\sigma 2$ its variance and μ its mean. l denotes the lag between observations:

$$\frac{1}{(\mathrm{n}\ -\ \mathrm{l})\sigma^2}\sum_{t=1}^{n-l}(x_t - \mu)(\ x_{t+l} - \mu)$$

2. Change quantiles mean – The mean absolute value of consecutive changes of the series x inside a corridor defined by the upper and lower quantiles of the distribution of x.
3. Change quantiles variance – The variance absolute value of consecutive changes of the time series, excluding extreme values (defined by a quantile corridor).
4. Energy ratio by chunks – The sum of squares of chunk i out of N chunks expressed as a ratio with the sum of squares over the whole series. N is the number of segments to divide the series into and i is the segment number (starting at zero) to return a feature on.

5. Abs energy – The absolute energy of the time series which is the sum over the squared values of the examined time series.
6. Larger standard deviation – Boolean variable denoting if the standard deviation of x is higher than r times its range, which is difference between maximum and minimum of x.
7. Ratio beyond $\boldsymbol{r}$ sigma – Ratio of values that are more than $r * std(x)$ (so they are r sigma) away from the mean of x divided by the length of the time series.

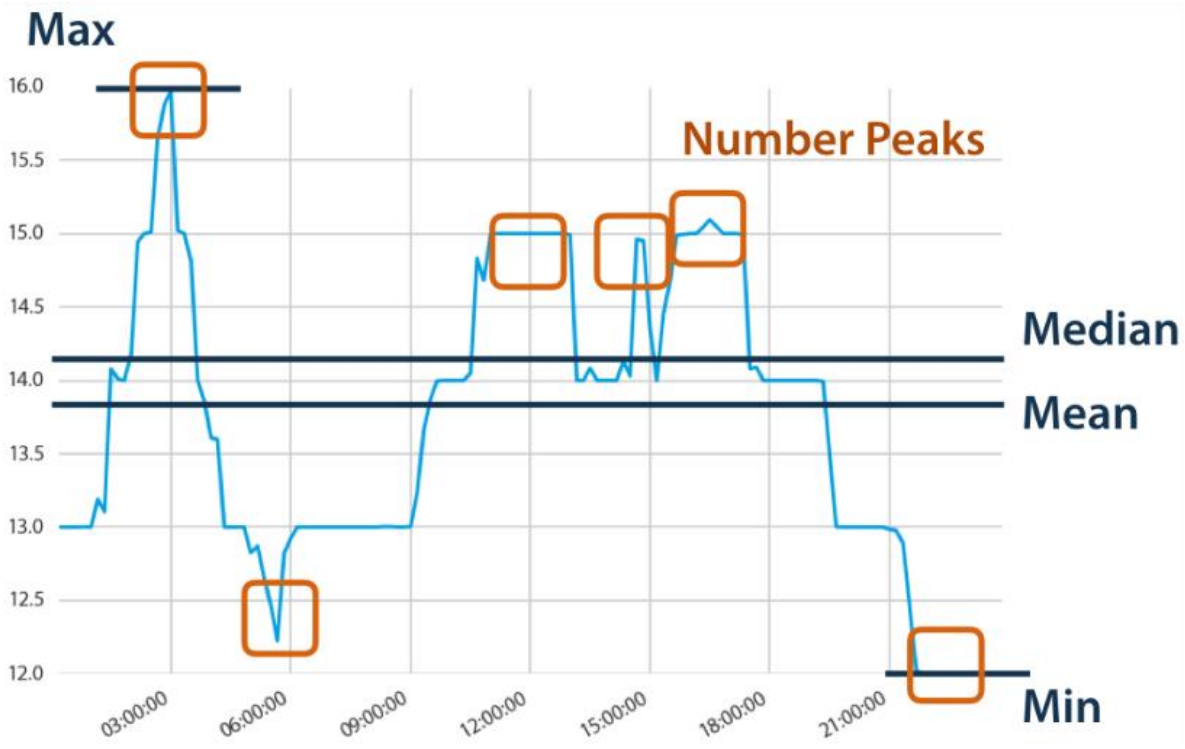


Figure 2. An example of the tsfresh feature extraction

2.5 RAkEL algorithm

We have shown in Section 2.1 that there exist two principal approaches for multi-labels classification over a set of labels L: problem transformation approach and problem adaptation approach. In this Section, we will review the RAndom k-labEL sets (RAkEL) algorithm (Fig.3), which is an ensemble method that follows the problem-transformation approach. The RAkEL algorithm seeks to classify multi-labeled instances while taking the relation between labels into account.

The goal is to output a multi-label classifier H that predicts a set of labels for each unseen instance. RAkEL handles multi-labeled data by generating LP-based multi-label classifiers for different small-size subsets of labels [46,47]. For a new instance, the LP classifiers output the most probable class which is a set of labels in the original multi-label representation. LP has the advantage of taking into account label correlations. In addition, the complexity of LP relies on the complexity of the single-label classifier with respect to the number of classes [46,47].

To reduce the computational complexity of the effective LP method for large numbers of labels and training instances, RAkEL involves "labels splitting" and "LP classification" and build a set of classifiers from different subsets of the finite set of labels [46,47]. In order to label unseen instances, the method combines the predictions of these multiple classifiers using a voting process to output the value of each label separately [46,47].

Let X denote an instance space and let Y = {l1, l2, . . . , lQ}, be the finite set of labels in a multi-label training task [46,47]. A training set of n instances is denoted by D = {(x_i, Y_i), i = 1, . . . , n}, where $x_i \in$ X is a feature vector describing instance i, and $Y_i \subseteq$ Y is the set of labels for that instance. The goal is to output a multi-label classifier H that predicts a set of labels for each unseen instance [46,47].

Label splitting

Given a size of label sets k, RAkEL constructs M random subsets of labels, { Zj | j = 1, . . . , M }, from Y. The different label sets may be overlapping and the overlap is certain when k × M > Q. For each label set Z_j , the associated training set, denoted as D_j , is obtained from the original training set D by replacing the label sets of training instances, Y_i , by their intersections with Z_j : D_j = {(x_i , $Y_i \cap Z_j$) | i = 1, . . . , n}. Note that this may lead to examples annotated by the empty set. These examples are not excluded from D_j but included in another class by considering the empty set as a new class [46,47].

LP classification

Since LP classifiers have some disadvantages, such as many different possible label sets, ($2^{|L|}$) and a small number of examples for each set of "new" single labels, each LP is trained using a different small random subset (k) of labels [46].

Each training set D_j is learnt by a single-label classifier H_j having as class values all the subsets of Z_j that are found in D_j. Given an new instance x, each single-label classifier H_j provides

binary predictions (+1 or −1) on each label in Z_j. The rest of labels (Y \ Z_j) are not learnt by H_j and their predictions by H_j are denoted by 0 [46,47].

Fusion of classifier decisions

To predict the set of labels for x, predictions of single-label classifiers are gathered and their mean is calculated separately for each label $l_q \in$ Y [46,47]. An adapted threshold t, usually equal to 0.5, is used in order to give the final decision. This intuitive threshold corresponds to the majority voting rule for the fusion of classifier decisions. Thus, the multi-label classifier H for the RAkEL method is determined as follows [46,47]:

$$H(x) = \{l_q \in Y \mid [\ (\sum_{j=1}^{M} (H_j(x, l_q) > 0)) / \sum_{j=1}^{M} |H_j(x, l_q)|] \geq t$$

This method can predict a label set that was not present in the initial training set because the final output of the multi-label classifier is assembled from different predictions of all single-label classifiers [46,47]. The number of single-label classifiers M and the number of labels in each model k are tunable parameters that need to be specified. For k = 1 and M = Q, RAkEL trains Q binary classifiers and we get the BR approach (mentioned in Section 2.1), while for k = Q and M = 1, we get the single-label classifier of the LP approach [46,47]. To improve the performance of the RAkEL method, one should increase the expected number of outputs per label, which means that k × M should take a large value. Since the complexity of RAkEL grows exponentially with the size of label sets k, but only linearly with the number of classifiers model, it is more usual to set k to a small value (e.g. k = 3), while giving to model a range of values going from Q to $2 * Q$ [46,47].

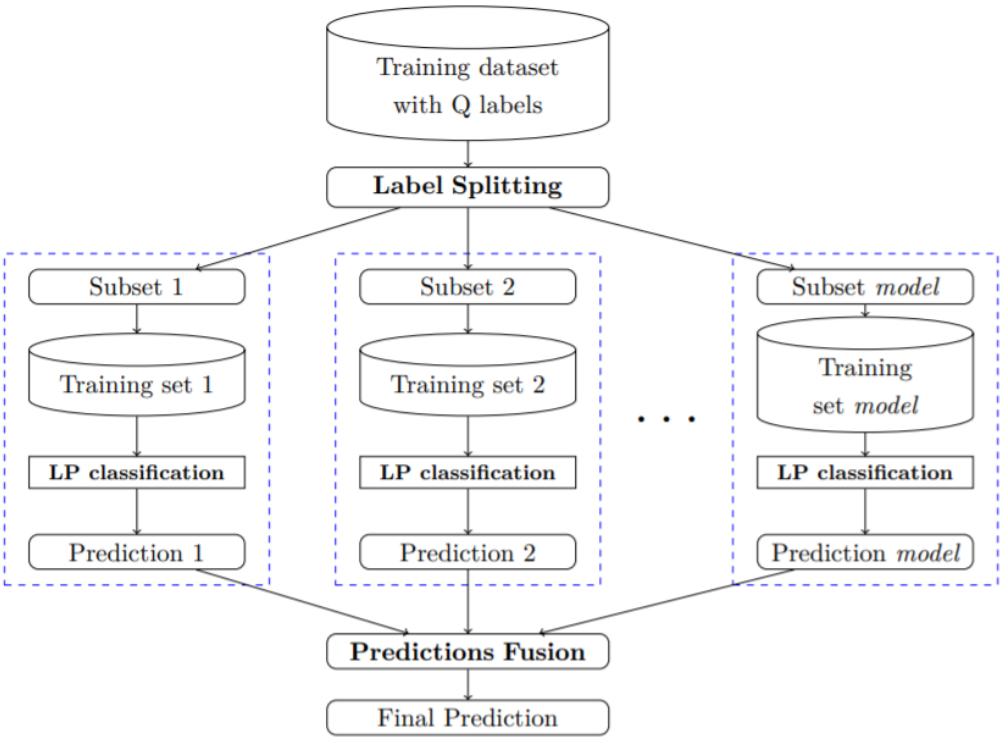


Figure 3. The RAkEL algorithm

2.6 Micro & Macro averaged precision

The precision metric defined as the probability that an object is relevant given that it is returned by the system [49]. In other words, as you can see in Equation 1, the precision for a class is the number of true positives (i.e., the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e., the sum of true positives and false positives, which are items incorrectly labeled as belonging to the class) [49]. In simple terms, high precision means that an algorithm returned substantially more relevant results than irrelevant ones.

$$(\mathbf{1})\ \mathbf{Precision} = \frac{TP}{TP + FP}$$

There are two label-based ways to calculate the precision over all the labels in a multi-label classification problem [50]. The calculation of this precision measure can be achieved using two averaging operations. It can be computed globally over all labels - which is the "micro-averaged precision" (Equation 2), or for each label separately and then be averaged over them, which is called "macro-averaged precision" (Equation 3) [50]. Macro-averaging gives an equal

weight to the performance on every category and is more influenced by the performance on rare categories. Micro-averaging, on the other hand, gives an equal weight to the performance on every instance, thus, favoring the performance on common categories [50].

$$(\mathbf{2})\ \mathbf{Micro-averaged\ precision} = \frac{\mathbf{1}}{|\boldsymbol{L}|} * \frac{\boldsymbol{TP_1} + \cdots + \boldsymbol{TP_{|L|}}}{\boldsymbol{TP_1} + \cdots + \boldsymbol{TP_{|L|}} + \boldsymbol{FP_1} + \cdots + \boldsymbol{FP_{|L|}}}$$

$$(\mathbf{3})\ \mathbf{Macro-averaged\ precision} = \frac{\mathbf{1}}{|\boldsymbol{L}|} * \left(\frac{\boldsymbol{TP_1}}{\boldsymbol{TP_1} + \boldsymbol{FP_1}} + \frac{\boldsymbol{TP_2}}{\boldsymbol{TP_2} + \boldsymbol{FP_2}} + \cdots + \frac{\boldsymbol{TP_{|L|}}}{\boldsymbol{TP_{|L|}} + \boldsymbol{FP_{|L|}}}\right)$$

Moreover, while micro-averaging can be used to know how the system performs overall across the data, macro-averaging is preferable if there is a class imbalance [50]. Providing both scores is more informative than providing either of them alone [50].

2.7 Hamming loss

The Hamming loss, also called example-based evaluation measure, is calculated based on the average differences of the actual and the predicted sets of labels over all test examples [50]. The Hamming loss evaluates how many times an instance-label pair is misclassified; in other words, a label not belonging to the instance is predicted or a label belonging to the instance is not predicted [51], i.e., the fraction of the wrong labels to the total number of labels. The Hamming loss was computed for each label and then was averaged over |L| and N, the numbers of labels and instances (movements), respectively.

$$\boldsymbol{Averaged-\ \ Hamming\ loss_{macro}} = \frac{\mathbf{1}}{\boldsymbol{N}} * \frac{\sum_j^N \sum_i^L [\boldsymbol{y_{pred}^i} \neq \boldsymbol{y_{true}^j}]}{|\boldsymbol{L}|}$$

Thus, unlike other evaluation measures described in this Section, for Hamming-Loss, the smaller the value, the better the multi-label classifier performance is [52].

2.8 AUC – Area under a ROC curve

Figure 4 shows the area under the receiver operating characteristic (ROC) curve, i.e., AUC, where a high AUC demonstrates a small top-left corner. A ROC curve is a two-dimensional

depiction of classifier performance [53]. Using of ROC graphs for diagnostic testing is very common, especially in the medical decision making field [53]. The ROC curve describes the tradeoff between true positives/hit rates (the x axes) and false positive/false alarm rates (the y axes) of classifiers [53]. To compare classifiers, we may want to reduce ROC performance to a single scalar value representing expected performance. A common method is to calculate the AUC [54, 55]. Since the AUC is a portion of the area of the unit square, its value will always be between 0 and 1. However, because random guessing produces the diagonal line between (0, 0) and (1, 1), which has an area of 0.5, no realistic classifier should have an AUC less than 0.5. Meaning, a perfect test has an AUC of 1, whereas random chance gives an AUC of 5 [56]. The AUC has an important statistical property: the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. This is equivalent to the Wilcoxon test of ranks [54, 55]. The AUC is also closely related to the Gini coefficient [56], which is twice the area between the diagonal and the ROC curve. Hand and Till (2001) point out that Gini + 1 = 2 · AUC [53,58]. Another property of the AUC is that it helps overcome imbalance since it is agnostic to the distribution of classes [59].

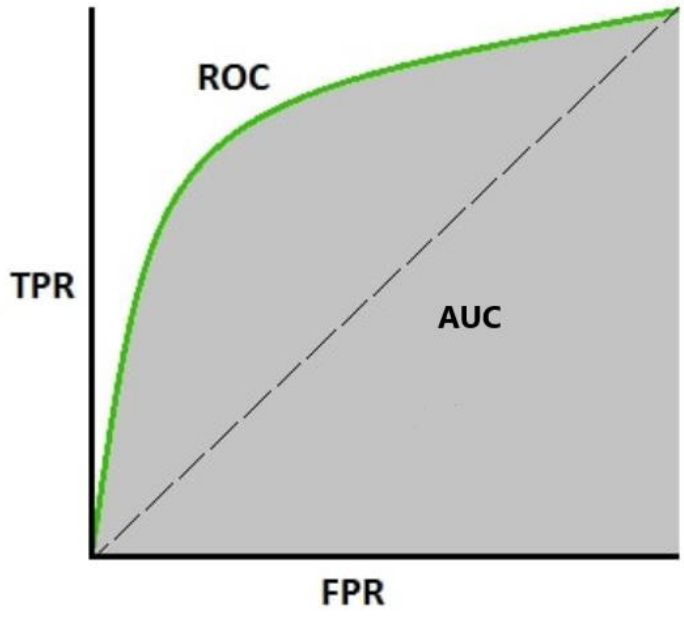


Figure 4. A ROC curve

It is possible for a high AUC classifier to perform in a specific region of the ROC space worse than a low AUC classifier. But in practice, the AUC performs very well and is often used when a general measure of predictiveness is desired [53].

Several researchers have investigated using AUC-ROC to inform the search heuristics of their algorithms. Ferri et al. (2002) alter decision trees to use the AUC-ROC as their splitting criterion, Cortes and Mohri (2003) show that the boosting algorithm RankBoost (Freund et al., 1998) is also well-suited to optimize the AUC-ROC. Joachims (2005) presents a generalization of SVM which can optimize AUC-ROC among other ranking metrics, Prati and Flach (2005) use a rule selection algorithm to directly create the convex hull in ROC space, and both Yan et al. (2003) and Herschtal and Raskutti (2004) explore ways to optimize the AUC-ROC within NN.

2.9 LOOCV - Leave-one-out Cross Validation

A k-fold cross validation procedure partitions the available data into k disjoint subsets. k models are then trained, each model being trained on a different combination of the k subsets and the test statistic evaluated over the remaining partition [68]. The mean of the test statistic for each of the k models is known as the cross-validation estimate of the test statistic.

The most extreme form of cross validation is known as Leave-One-Out Cross Validation (LOOCV) [68]. LOOCV provides a sensible model selection criterion as it has been shown to provide an almost unbiased estimate of the true generalization ability of the model [68]. The leave-one-out estimate (classification error) is computed by running the learning algorithm m times, each time removing one of the m examples, training on the remaining m-1 examples, and testing the resulting hypothesis on the removed example.

LOOCV is useful for avoiding the statistical problem of overfitting in models in which the same samples are trained and predicted, or when the number of examples is small and there are many variables [69, 70].

Efron (1983) conducted five sampling experiments and compared LOOCV, several variants of bootstrap, and several other methods. The purpose of the experiments was to investigate some

related estimators, which seem to offer considerably improved estimation in small samples. The results indicate that LOOCV gives nearly unbiased estimates of the accuracy, but often with unacceptably high variability, particularly for small samples.

Chapter 3

Related work

Several studies proposed a framework for automating upper limb assessments for stroke patients by using various sensors and classification schemes, most of them based on machine-learning algorithms [2,3,4,17,19,22,24,72,73]. The main purpose of most of these studies was to give an evaluation score per movement performed by the user.

Wang et al**.** (2014) presented an evaluation model that uses the SVM as a non-linear classifier. They used the algorithm to estimate the FMA score for Shoulder-Elbow movement. To build this model they recorded 20 movements from each of 24 stroke patients using two accelerometer sensor nodes, which were attached to the patients' forearms and upper arms. Statistical features were computed such as maximum magnitude, mean value and root mean square value of accelerometer data, root mean square value of the derivative of accelerometer data (jerk). In addition, maximum, mean value and standard deviation of the velocity of movement. The RMSE of the comprehensive model was 2.13 points, 7.1% of the corresponding highest score. A year later, Otten et al. (2015) introduced an evaluation model using an Artificial Neural Network (ANN) classifier, which outperforms the support-vector machine (SVM) classifier. They used various sensors, such as GPS sensors, direction sensors (i.e., magnetic compasses), and acceleration sensors (i.e., accelerometers) from an Android-based smartphone in order to record movements of eight healthy participants. They asked the participants to perform all movements in three ways: faultlessly, partially, and not at all (motionless). From these records, the authors calculated a set of movement features, such as elbow flexion, limb orientation, and joint angles. These features were used to determine a score for the participant's upper limb functionality, with a score of zero indicating the participant cannot perform any movement and a score of two indicating he or she can perform the movement faultlessly. Kwapisz et al. (2011) also used an ANN classifier, but it was used to evaluate functional movement of the lower limbs, such as walking, going up or down stairs, jogging, sitting and standing, and not for movements of the upper limbs. That study also included only healthy participants. The study describes a system that uses phone-based accelerometers to perform activity classification. Data was collected using a the accelerometer and a phone application that records users GPS and the features that were extracted from it were average acceleration, standard deviation, average absolute difference, average resultant

acceleration, time between peaks and binned distribution. Using those features the best model to predict the movement was the multilayer neural network with 91.7% accuracy while the accuracy of the logistic regression was 78.1% and of the J48 was 85.1%.

Later works argued that linear and nonlinear mapping methods such as neural network, SVM and linear regression often suffer from some drawbacks, such as too many parameters, time-consuming, etc. [73]. To reduce the number of parameters required in the ANN model, Yu et al. (2016) used extreme learning machine (ELM)-based ensemble regression model and compared its results to the results of an SVM algorithm. They proposed to monitor the functional movement of the upper limb and attempted to predict the user's FMA score using sensor data. Their proposed FMA framework contains two accelerometers and seven flex sensors were used to monitor the movement function of upper limb, wrist and fingers of 24 stroke patients. The model was aiming to map the sensor data to clinical FMA score, while there were three main types of features which were extracted from the sensor data: amplitude, mean value, and jerk. After applying the two models (SVM, ELM) the RMSE and the R-squared of the SVM were 11.87 and 0.84, respectively, and of the ELM were 12.19,0.83. After calculating T-test they found no obvious difference between the SVM and ELM algorithms in terms of accuracy ($p = 0.724 > 0.05$). Importantly, they found that feature selection - i.e., narrowing down the feature space to the most informative set of movement features - leads to significantly improved accuracy [4].

There is limited information on automatically identifying the specific compensations in stroke patients' movements. Tormene et al. (2009) used dynamic time warping (DTW) and OE-DTW (Open-Ended DTW) to provide real-time feedback to neurological patients undergoing motor rehabilitation. They generated a dataset of multivariate time series from a sensorized long-sleeve shirt. One of the experiments they conducted was to recognize incorrectly performed movements, and in those, identify the specific error that was performed. However, there was only one healthy participant in this experiment, whom they asked to perform very slow movements or mimic two options of compensatory actions: adduction of the upper limb on a frontal plane or on a sagittal plane [74]. Similarly, Kizony et al. (2014) proposed a system that was designed to provide a home-based tele-rehabilitation program based on one healthy participant performing compensatory and non-compensatory movements.

Previous works show that ANN, SVM, ELM, and DTW are beneficial for solving tasks such as multi-class classification and regression [2,3,4,17,19,22,24,72,73]. However, these

algorithms are not suitable for more complicated tasks, such as multi-label classification tasks, when attempting to identify more than a single component of the movement (e.g., both excessive bending of the trunk and elevation of the shoulder). Furthermore, in order to build a model that identifies movement components that are found primarily in patients' movements, it is vital to collect the movement data from the relevant patient population, rather than from healthy individuals. However, most of the models built by previous works were based on movements of healthy participants and not on movements of post-stroke patients (e.g., [72,73]). Moreover, the number of the participants in the study is also a major factor in generating a representative model. Yet, previous works had between 1-7 participants, which may have limited their applicability [3,72,74,75]. Finally, in order to build as accurate a model as possible, it is important to use high-precision sensors. Using a Kinect camera is rather common (e.g., [2,75,76]), being a readily available and relatively affordable tool, though it is limited in its capacity to correctly detect fine motions [2], which then limits the overall accuracy of the generated model.

Chapter 4

Research Objectives and Expected Significance

4.1 Research Objectives

To date, to the best of our knowledge, no algorithm has been developed to automatically identify the type of compensatory movements performed by patients who suffer from neurological conditions, such as a stroke. Here, we propose such an algorithm, with which compensatory movements, made when reaching a cup, grasping, and lifting it, can be automatically detected without the need for an on-site clinician to be present.

The machine-learning model we present here:

(1) Will allow the patients to practice the desired exercise movements, as instructed by the therapist, and avoid performing undesirable movement patterns known as "bad learned use" [32] during self-practice, by providing accurate information on what specific compensations he or she performed (e.g., elevation of the shoulder);

(2) Will enable the therapist to obtain information on the patient's at-home performance, in order to precisely adapt the overall training program to the patient's current ability; and

(3) Will serve as a personal ecological performance-assessment tool.

In the future, the algorithm could be used to give recommendations for updating the exercise program during a session, in accordance with the patient's performance.

4.2 Limitations

One possible limitation of our work is the sample size – 30 post-stroke individuals. While it is large compared to previous works with human participants, a larger sample size may allow for better generalization, and have a more balanced data set (a better representation of the different impairment levels) [56]. In addition, this group of participants was heterogeneous in terms of

their functional ability, with patients displaying low, moderate, and severe levels of impaired function. Despite this, the model we developed reached a macro-averaged precision of 0.85.

Another limitation is the use of markers for data collection. In this work, we aimed at generating an accurate algorithm for the identification of compensatory movements in stroke-patient movements. For that purpose, we used high-precision motion-capture device (V120: Trio OptiTrack, NaturalPoint, Inc., OR, USA, accuracy $\leq$ 1.0mm). It is conceivable that simpler implementations of the model may use a smaller number of sensors or potentially a different, low-cost, simple sensor system, which would add to the user's convenience of using the algorithm both in terms of procedure and price. Enabling a low-cost automated supervision of at-home practice would be important in that during home-based practice there are fewer constraints on time and space, so the patients can practice more frequently, for longer periods of time, and also according to their own schedule and terms [3, 4, 57]. Also, at home, the individuals can engage in more ecological exercises that are compatible with their everyday routines, which may be more indicative and useful from an evaluation session in the clinic [3, 4, 57].

4.3 Contributions

The algorithm we developed offers several benefits over existing models for movement evaluation:

(1) It uses everyday functional movements of the type individuals after stroke are often asked to practice; Thus, it avoids the artificial nature of a test situation which does not reflect real everyday movements [17, 20, 21];

(2) Unlike several of the models developed so far, it is based on data from stroke patients, rather than from healthy individuals; and

(3) It provides direct information on the compensatory movements that the individual performed – whether there were any, and which. Thus, it can address a potential concern that patients may have regarding at-home practice.

Chapter 5

Method

5.1 Data Collection

Participant

A total of 46 participants took part in this study. 30 post-stroke patients were recruited to the study from "Beit Hadar" Rehabilitation Center (14 females, 16 males, mean age 70.3±9.4 years). In addition, 16 age-matched healthy control participants were recruited from the community (11 females, 5 males, mean age 27.3±3.4 years). The study was approved by the Barzilai Medical Center's Helsinki Committee, and all participants signed an informed consent form to participate in this study.

Procedure

The participants were examined by a physical therapist between one to two weeks before their discharge date from the rehabilitation center. The examination was performed while the participants sat in front of a height-adjustable table. Participants were instructed to reach their impaired arm at a self-selected speed, forward, toward a cup located on the table, lift it, and place it on top of a 5 cm-high block, positioned on the table (Fig. 5). The participants were instructed to avoid bending their trunk as much as possible during the reach movement, but no restraint of the trunk was applied. RTG was performed at three different heights: (A) low - the height of the wrist when the hand is extended downwards, (B) medium, ~75 cm from the floor, the height of a standard table, and (C) high - the height of the participant's shoulder (Fig. 5). The cup was placed at an arm's distance, measured from the lateral acromion to the radial styloid process, to avoid excessive trunk movement during the reach movement. Reach and grasp movements were executed using an empty cup (273 gr) in half of the trials, and a cup filled with water (443 gr) in the other half (Fig. 5). Every combination of cup height and weight was repeated three times for a total of 18 RTG movements (3 heights x 2 weights x 3 repetitions). The order of the heights and weights was randomly set in order to prevent the influence of fatigue on particular combinations of height and weight.

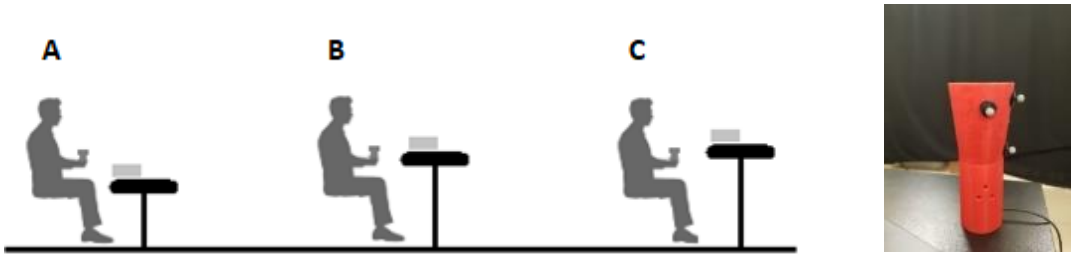


Figure 5. The experimental setup for the data-collection phase

Left: Participants were asked to reach to a cup placed on a table, pick it up, and place it back on a 5-cm block on top of the table. The table was set at three different heights: (A) low (~50 cm from the floor); (B) intermediate (~75 cm from the floor); and (C) high (~86-100 cm from the floor, depending on shoulder height). Right: The custom-built cup, embedded with a force sensor, with the three position markers (see text).

The reach movement was executed by the affected arm of the post-stroke patients. Since the affected arm could be either their dominant or their non-dominant arm, the control group were matched for dominance. That is, if half of the patient group reached with their non-dominant arm, then half of the control group were also asked to reach with their non-dominant arm. Starting position for the low height was with the arm held vertically at the side of the body. Starting position for the medium and high heights was with the arm placed on the ipsilateral thigh with palm facing down. Every reach combination of height and weight was evaluated three times, according to the participant's ability. That is, while the maximal total number of reaching trials was 18, some participants were not able to complete all trials, due to arm weakness, fatigue, pain, etc.

Motion-capture system

Position of the upper extremity joints during the RTG movement was recorded using a motion capture system V120: Trio (OptiTrack, NaturalPoint, Inc., OR, USA). The V120: Trio tracking system is a portable multiple-camera with a 6DoF optical object tracking technology. Eleven reflective markers were placed on the participants' upper body (Fig. 6). Markers were placed as follows: two markers were placed vertically aligned on the sternum to reflect the trunk motion (Fig. 6, Points 1-2), and one marker was placed on each of the following anatomical landmarks: lateral portion of the acromion [reflecting the scapular motion [69,77]] (Fig. 6,

Point 3), proximal humerus (Fig. 6, Point 4), lateral epicondyle of the elbow (Fig. 6, Point 5), the middle forearm (Fig. 6, Point 6), radial and ulnar styloid processes (Fig. 6, Points 7-8), the dorsal side of the palm at the axis along the middle part of the third metacarpal bone (reflecting the wrist motion) (Fig. 6, Point 9), thumb (Fig. 6, Point 10), and index finger (Fig. 6, Point 11). Two additional markers were placed vertically on the wall behind the participant to serve as stationary reference points, and three additional markers were placed on the cup and defined by the system as a rigid body, so that the cup location can be tracked during each recording. Data sampling speed of the Trio system is 120 Hz.

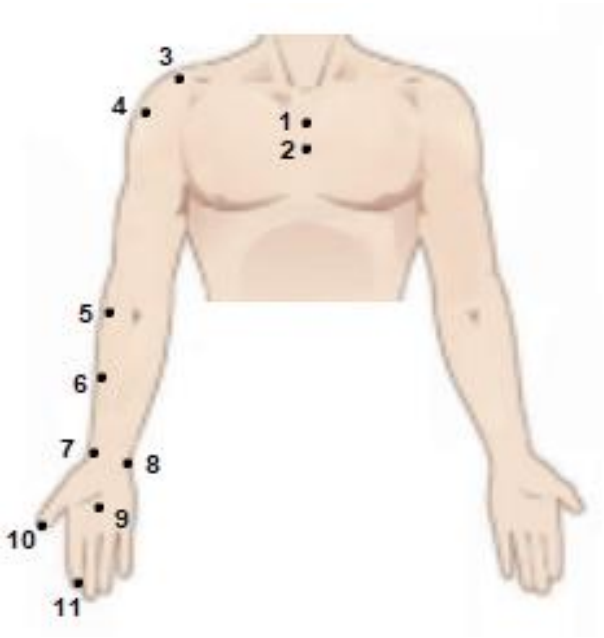


Figure 6. The location of the 11 body markers placed on each participant.

(1-2: sternum, 3: shoulder, 4: proximal humerus, 5: elbow, 6: the middle forearm, 7-8: radial and ulnar styloid processes, 9: wrist, 10: thumb, and 11: index finger)

Force sensor

Grip forces were measured with a 3D force sensor (Nano25-E Transducer, ATI Industrial Automation, INC) embedded in a custom-built 3D-printed cup (Fig. 5). The data sample

frequency of the force sensor was 100 Hz. The output from the force sensor was the summed grip forces applied on the cup.

Data collection and annotation

The data were collected by Ana Yelkin and Ronit Feingold-Polak, who also later annotated the data for existence of compensations. They are both physical therapists, with experience treating patients with stroke.

5.2 Feature Generation

To build an initial set of features from the collected data, we used two different approaches. The first is based on the analysis of the reach-grasp-lift movement by calculating (with Matlab R2017b) first the segments of each movement and then a wide variety of biomechanical hand-crafted metrics, including velocity, jerk, index of curvature, angels of the joints, etc., derived from the time series data generated using the markers and the force values measured during the movement. The second approach is based on time-series feature extraction using scalable hypothesis tests implemented by the tsfresh package (with python 3.6) [46].

Movement segmentation

RTG is divided into a transport component, which is the change in position of the hand over time, and a grasp component, which is the change of the distance between the index finger and thumb over [78]. In healthy individuals, certain elements in reaching and grasping display invariant behaviours suggesting key principles in motor control. For example, movement trajectories involving more than one joint tend to be straight, smooth, and have bell-shaped velocity profiles [7, 79]; Peak deceleration point usually occurs around the time of object contact [78]; The start time of the opening of the hand is correlated with the start time of hand movement toward the object, and the time of maximum hand opening is correlated with the time of peak deceleration of the hand [78]. Smoothness of the movement is widely regarded as a hallmark of coordinated movement. Jerk, the third derivative of position with respect to time, has been used as an empirical measure of this quality [80].

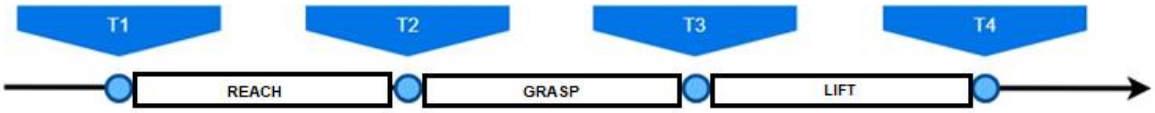


Figure 7. The process of movement segmentation

We automatically segmented each of the participants' movements into three segments, as shown in Figure 7, based on four time points (T1-T4), according to the phase of the movement: reaching to the cup (REACH), grasping the cup (GRASP), and raising the cup and placing it on the block (LIFT):

1. T1 – start of movement – the time at which 10% of the maximal velocity of the wrist is reached [6].
2. T2 – start of grasp – the time at which the participant applied force equal to 5% of the difference between the maximal and minimal forces applied on the cup in a given trial.
3. T3 – start of lift – the time at which the cup reached 10% of its maximal height during the trial.
4. T4 – end of movement – the time at which the movement ended, calculated as follows: the force trace was scanned from the end of the trial backward until the first time the value of the force was 20% of the maximal force during that trial. Then, the force trace was scanned from this point forward, to find the first time the value of the force was 5% of the maximal force during that trial. That point in the time series was set to be T4.

In each movement, T2 was first identified, then T1 was calculated in the interval between the start of the recording and T2, followed by T3, and then T4.

Figure 8 shows an example for the four time points (T1-T4) found by the algorithm:

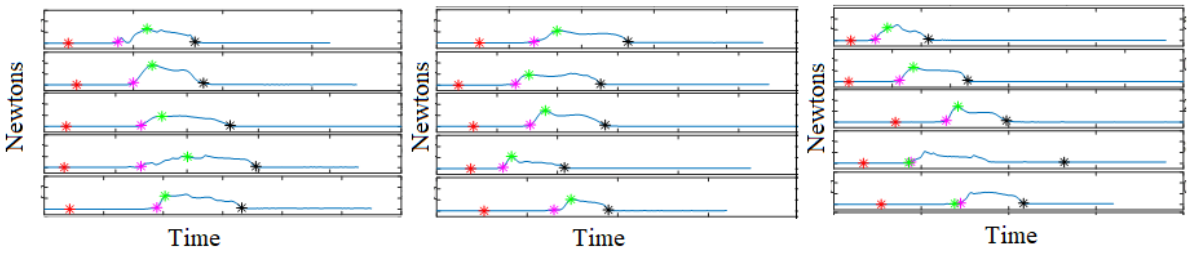


Figure 8. Example for the algorithm's output of the four time points that segment the movement. Force profiles through time from three different random movements (columns) from five different random participants (rows). The red, pink, green, and black asterisks represent T1, T2, T3, and T4, respectively.

Biomechanical handcrafted movement features

Kinematic features of the movement were calculated from six main categories: 'Jerk', 'Velocity', 'Angles', 'Aperture', 'Curvature', and 'Force', all of which are known to have different characteristics in movements of individuals after stroke [6, 7, 9, 10, 11, 13, 78, 81-84]. Table 1 shows the features we calculated from each category, the markers used to derive each feature, and for which segments of the movement the features were calculated: REACH (T1 to T2), GRASP (T2 to T3), LIFT (T3 to T4), and ALL (a unified segment that begins in T1 and ends in T4). The combination of two segments indicates a segment that is composed of both (e.g., GRASP + LIFT between T2 and T4).

In the 'Angles' category, six angles were calculated based on position data from the relevant markers:

1. Scapula elevation (ScapulaEle) – the angle between the two sternum markers and the shoulder marker (Fig. 3, Points 1-3).
2. Scapula rotation (ScapulaRot) – the angle between the bottom sternum marker and the shoulder marker (Fig. 3, Points 2-3).
3. Trunk– the angle between the two sternum markers (Fig. 3, Points 1-2) and the wall markers.
4. Elbow–the angle between the proximal humerus, elbow, and middle forearm markers (Fig. 3, Points 4-6).

5. Shoulder– the angle between the upper sternum, shoulder, and proximal humerus markers (Fig. 3, Points 1, 3-4).
6. Wrist– the angle between the middle forearm, the radial styloid process, and the wrist markers (Fig. 3, Points 6,7,9).

The correlations between pairs of angles were then calculated per movement, as well as between the angles and the aperture of the hand (see Table 1).

Category	Feature	Markers/Sensors	Segments
Jerk	**Mean Squared Jerk (MSJ)** – $$\frac{1}{\lvert T\rvert}\int \frac{1}{2}(\text{jerk}^2) / \max(\text{velocity})$$ *Velocity* - the first derivative of position with respect to time for the duration of the movement, T. *Jerk* -the third derivative of position with respect to time for the duration of the movement, T.	Wrist	REACH, GRASP, LIFT, ALL
Velocity	**Average, Maximum, Minimum, Time to Maximum**	Wrist, Elbow	REACH, GRASP, LIFT, ALL
Angles ScapulaEle ScapulaRot Trunk Elbow Shoulder Wrist	**Correlations** – Trunk ↔ Elbow Trunk ↔ Aperture Trunk ↔ Shoulder Trunk ↔ ScapulaEle Trunk ↔ ScapulaRot Trunk ↔ Wrist ScapulaEle ↔ Elbow ScapulaEle ↔ Aperture ScapulaRot ↔ Aperture Wrist ↔ ScapulaEle Wrist ↔ Elbow	Sternum 1-2, Shoulder, Humerus, Elbow, Forearm, Radial, Wrist	REACH, GRASP, LIFT, ALL
	Maximum, Minimum, Average	Trunk, Elbow, Shoulder, Wrist, Sternum 1-2	REACH, GRASP, LIFT, ALL
Aperture	**Maximum Distance –** between the positions of the thumb and index-finger markers.	Thumb, Index	REACH
	Time to Maximum Distance – The time elapsed from T1 to the time at which the participant reached the Maximum Distance.	Thumb, Index	REACH
	Time to T2 – The time elapsed from when the participant reached the Maximum Distance to T2.	Thumb, Index	REACH
	Std1– Standard deviation of the aperture amplitude between T1 and the time that 'Maximum Distance' was reached.	Thumb, Index	REACH
	Std2 – Standard deviation of the aperture amplitude between the time that 'Maximum Distance' was reached and T2.	Thumb, Index	REACH
Curvature	**Straight-line distance –** The length of a straight line connecting the locations of the wrist marker at T1 and T4.	Wrist	REACH, GRASP+LIFT
	Path length – The actual path length of the movement made between each segment.	Wrist	REACH, GRASP+LIFT
	IC – Index of curvature: the ratio between the Straight-line distance and Path length.	Wrist	REACH, GRASP+LIFT
Force	**Summed forces – Measured** across all durations of movement.	Force sensor	GRASP+LIFT
	Time to max force – Time duration between T2 and the time at which the maximum force was reached.	Force sensor	GRASP, LIFT
	Segment duration – Duration of a movement segment.	Force sensor	GRASP, LIFT, GRASP+LIFT
	Duration ratio – The ratio between the segment durations of GRASP and LIFT.	Force sensor	GRASP, LIFT
	Variance – Variance of the force measured during the segment.	Force sensor	GRASP, LIFT
	Time duration max aperture – The time duration between the time max aperture was reached and T2, divided by the variance in segment.	Force sensor	GRASP, LIFT
	Average force	Force sensor	GRASP, LIFT, GRASP+LIFT
	Std – standard deviation of the difference between any two consecutive force values.	Force sensor	GRASP+LIFT
	Variance X **Segment duration**	Force sensor	GRASP, LIFT
	Average X **Segment duration**	Force sensor	GRASP, LIFT

Table 1. The handcrafted features

The first column lists the main category they belong to, the second column describes the feature, the third column lists which markers were used for that feature's calculation and the forth column lists which segments the calculation was applied to.

Automated extraction of time-series feature

We used the Python 3.6 tsfresh package to generate 3,138 time-series features automatically [46]. This package filters the features with respect to their significance for the classification task, while controlling the expected percentage of selected but irrelevant features. The features it extracts describe basic characteristics of the time series such as the number of peaks, average or maximal value of a signal or more complex features such as the time reversal symmetry statistic (see details in Section 2.4). The time-series sequence we used as input to the tsfresh package is the traces of the three position axes (x, y, z) of the wrist marker (Fig. 6, Point 9) for each participant for each of the 18 RTG movements.

Chapter 6

Experimental processes and algorithms

Our main experiment was to construct an automatic algorithm that can identify whether and which compensations a post-stroke individual made when reaching to a cup, grasping, and lifting it. Before constructing this experiment, we first conducted an initial small experiment to investigate whether the features we extracted from the data are relevant for the study, by using them to distinguish between movements of post-stroke patients and those of healthy individuals.

Figure 9 shows the general process of generating both models, which we briefly overview here, and explain in detail below. The first two phases: Data Collection (01) and Feature Generation (02) which are described in Chapter 5, were identical in each experiment. In the Feature generation phase, movement features were generated using two methods in parallel and then combined: (1) biomechanics-inspired handcrafted features based on the motor-control literature, and (2) automatically extracted features by a dedicated software- the tsfresh package. Then, the last three phases were different in each experiment. The Feature Selection (03) was conducted in order to obtain an optimal set of features that will be the input to the relevant algorithm corresponding to the problem (04). Finally, in the Evaluation phase (0.5), we tested the performance of the model we built.

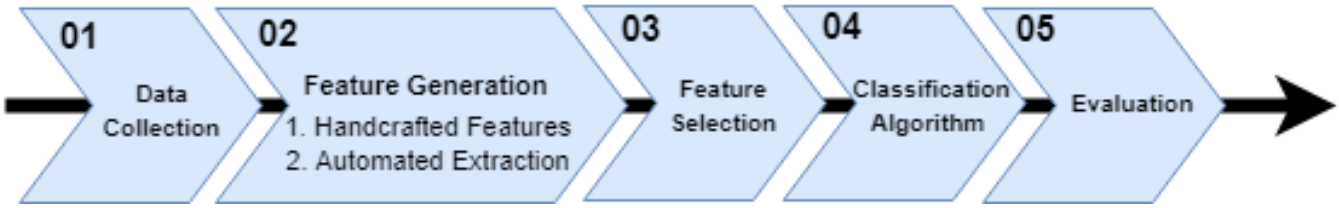


Figure 9. The movement-compensation detection process

6.1 Experiment 1 - Distinguish between healthy and patient data

The first algorithm in this study was used to differentiate between healthy participants' and stroke patients' data. A high discerning ability will imply that the features we used are relevant and important for our main purpose – detecting compensations in stroke patients' movement data (the second experiment).

Pre-Processing

Since the labels of this experiment are per participant (control/patient) instead of per movement, we made some manipulations to adjust the features to the problem. For each participant, for each feature calculated in Section 5.2, we calculated the sum, maximum and minimum overall the 18 movement of this participant.

Feature Selection

It was necessary to perform feature selection on the large set of features we generated using both methods (over 3,000) for two reasons: (1) to avoid over-fitting due to a large number of features compared to the number of training instances (movements); and (2) to identify the most meaningful features for a leaner and more efficient model. Feature selection allowed us to reduce the dimensionality of the feature space, and remove redundant, irrelevant, or noisy features, to enable the classification algorithm to be more accurate and rapid.

We used the WEKA 3.8 program [85] for the feature selection, and there we applied the supervised filter: 'Attribute selection' with 'BestFirst' as search method, which searches the space of attribute subsets by greedy hill-climbing augmented with a backtracking facility, on all features we got in phase 5.2, and then get a set of optimal features. The final set had 127 features.

The algorithm

For this initial study, we used the XGBoost algorithm, a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework (for more details – Section 2.3).

We took 70% from the participants to the train set and 30% to the test set, while keeping equal percentage of patients and healthy participants in each dataset.

We applied the XGBoost in R Studio version 3.4.3. The objective function we used is "binary:logistic", and the maximum depth of the tree we used is 3. The maximum number of boosting iterations is 500 with a 0.8 factor on sampling.

6.2 Experiment 2 - Movement-compensation detection

The second algorithm we present here was used to identify the exact set of compensations the patient performed in each movement. Thus, we are dealing with a multi-label classification task, since any given movement can have between zero and six compensations in parallel. Here we use the RAkEL algorithm, which is suitable for multi-label problems

Compensation labelling

Two expert physical therapists labelled each of the 18 movements per participant with the set of compensations a participant made, if any. The possible compensations were: trunk-flexion, scapula-elevation, scapula-rotation, shoulder-flexion, elbow-flexion, distal dys-synergy.

Feature Selection

For the reasons mentioned in ‘feature selection’ in Section 6.1, we applied here also feature selection. We used the WEKA 3.8 program [85] for this task and applied, for each single label (i.e., compensation), feature selection that searches the space of feature subsets by greedy hill climbing augmented with a backtracking facility. The features were evaluated by ‘CFS’ (correlation-based feature selection) [85], which evaluates the contribution of a subset of features by considering the individual predictive ability of each feature along with the degree of redundancy between them. Therefore, subsets of features that are highly correlated with the label (compensation) while having low intercorrelation among them are preferred [85].
After we applied this selection approach on each of the six compensation labels separately, we kept features that were selected for between two and five labels, since there were too many features that appeared only in a single label set. None of the features were selected for all the six labels. The final set had 156 features.

The algorithm

In this study, for our multi-label classification problem we used the RAkEL algorithm which is an ensemble approach - multiple learning algorithms, for multi-label classification over a set of labels L (the six compensation types; see details in Section 2.5). We trained the RAkEL model using a "random forest" as the base classifier, and the size of the feature subset was k=4. The number of LP models we used is 16 (Fig. 10).

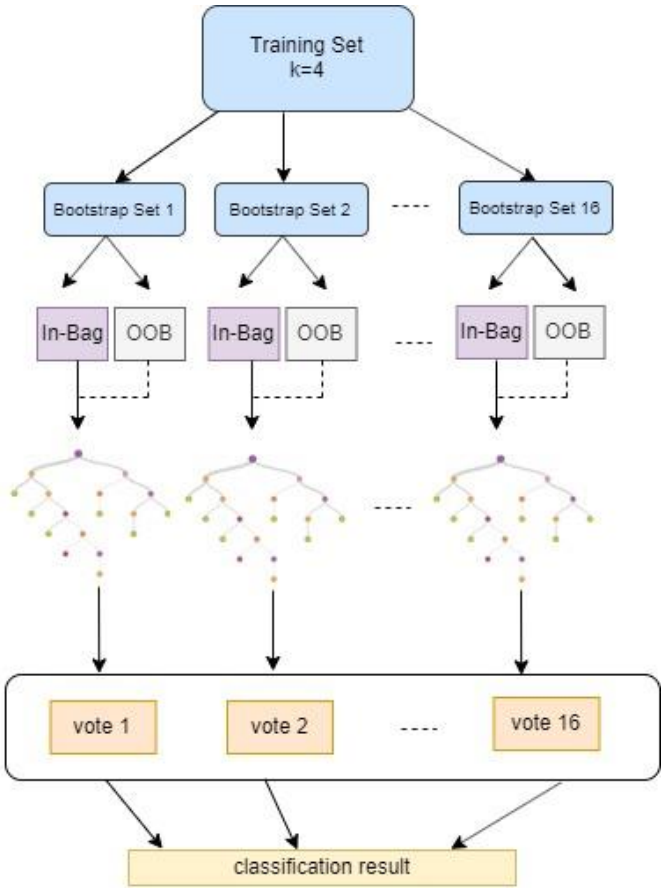


Figure 10. RAkEL algorithm using the 'random forest' classifier (k=4, M=16)

Chapter 7

Evaluation Metrics and experimental Settings

7.1 Experiment 1

AUC – Area under the curve

To evaluate the classifier's performance, we used the AUC measure. Using ROC graphs for diagnostic testing is very common, especially in the medical decision-making field [48,70] (see more details in Section 2.8).

Accuracy

The accuracy metric gives an average degree of similarity between the predicted and the ground truth label sets (N – number of participants):

$$\boldsymbol{Accuracy} = \frac{\mathbf{1}}{\boldsymbol{N}} * \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}} \frac{\boldsymbol{y}_{\boldsymbol{pred}}^{\boldsymbol{i}} \cap \boldsymbol{y}_{\boldsymbol{true}}^{\boldsymbol{i}}}{\boldsymbol{y}_{\boldsymbol{pred}}^{\boldsymbol{i}} \cup \boldsymbol{y}_{\boldsymbol{true}}^{\boldsymbol{i}}}$$

7.2 Experiment 2

Multi-label classification requires different evaluation measures than those used in traditional single-label classification. Several measures have been proposed in the past for the evaluation of multi-label classifiers. The example-based measures are calculated based on the average differences of the actual and the predicted sets of labels over all test examples. The label-based evaluation measures decompose the evaluation process into separate evaluations for each label, which they subsequently average over all labels [51,88].

We used both micro-averaging and macro-averaging (Section 2.6) and the Hamming loss (Section 2.7) to assess the multi-label classification performance under LOOCV, by which each of 30 models that was trained using data of 29 participants was then tested on data of the remaining participant (Section 2.9).

Chapter 8

Results

8.1 Experiment 1

As you can see in Table 2, the AUC, which represents degree or measure of separability between the participants, was 0.977 (you can see the ROC curve in Fig. 11), which means there is 98% chance the model will be able to distinguish between positive class (patients) and negative class (healthy). In addition, Table 2 shows that the model was able to predict in high accuracy of 0.931.

AUC	0.977
Accuracy	0.931

Table 2. results of the first experiment

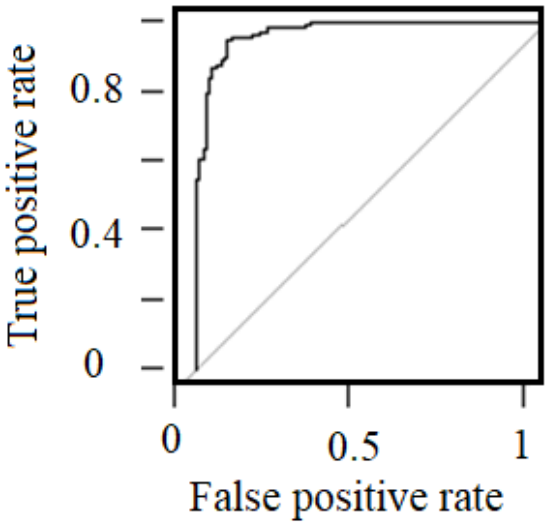


Figure 11. The ROC curve of the first experiment

8.2 Experiment 2

Formula / Features	Micro-averaged precision	Macro-averaged precision	Macro-averaged Hamming loss
Handcraft	0.78	0.81	0.21 **
tsfresh	0.76	0.81	0.22 *
Handcraft + tsfresh	0.81	0.85	0.19

Table 3. Average predictive performance for different feature sets
***/* indicate that the difference between Handcraft + tsfresh and the individual feature sets is statistically significant at p<0.05/ p<0.1, respectively.*

Table 3 presents the results obtained from the model in three feature settings: using handcrafted features (45 features), tsfresh features (111 features) and both (all 156 features). As we can see from the table, the macro and micro averaged precisions were higher when all 156 features were used, with a macro-averaged precision of 0.85, and a false-positive rate of 0.15. Not only that, but also the Hamming loss was the lowest in this condition. The calculation of the macro-averaged precision metric as we explain in Section 2.6 is done by computing the precision for each label (compensation) separately before averaging them. Figure 12 shows the precision for each label.

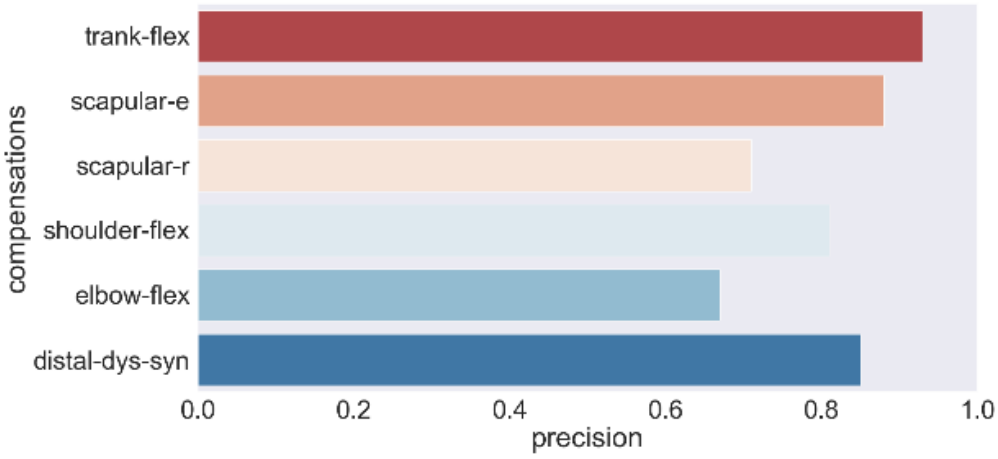
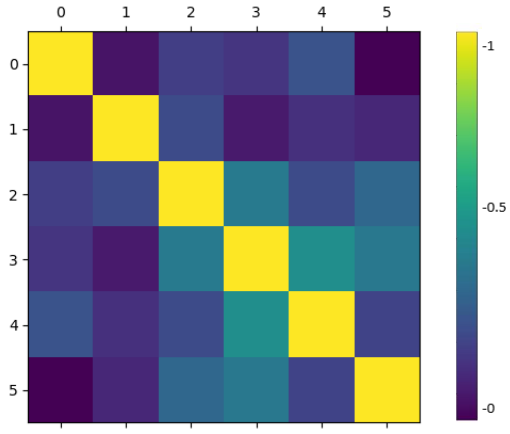


Figure 12. Precision scores for each of the six compensations**:** elbow flexion (elbow-flex), scapular rotation (scapular-r), shoulder flexion (shoulder-flex), scapular-elevation (scapular-e), distal-dys-synergy (distal-dys-syn) and trunk flexion (trunk-flex).

To check if there are corelative labels (i.e. compensations), Figure 13 shows the correlation between the labels.



	0	1	2	3	4	5
0	**1**	0.041	0.178	0.145	0.249	0.010
1	-	**1**	0.219	0.064	0.128	0.100
2	-	-	**1**	0.407	0.219	0.326
3	-	-	-	**1**	0.488	0.395
4	-	-	-	-	**1**	0.191
5	-	-	-	-	-	**1**

Figure 13. Pairwise correlation between the six compensation labels.
(0=trunk flexion, 1=scapular elevation, 2 = scapular rotation, 3=shoulder flexion, 4=elbow flexio n, 5=dys=synergy distal)

The best results obtained by the RAkEL algorithm were for k=4 and k=2. Since both give a micro-averaged precision of 0.81, but k=2 has a significantly longer running time, we opted for using k=4. Table 4 shows a grid search of parameter k and the corresponding micro-averaged precision. It is common to determine the number of classifiers (M) of the RAkEL algorithm as the number of labels or twice this number (i.e., in this case, between L=6 and 2*L=12). Since the larger the number of classifiers, the more reliable is the result, we tested, using trial and error, values of M>12 to find the optimal value for M, and concluded by choosing M=16

	k=1	k=2	k=3	k=4	k=5	k=6
Micro-averaged precision	0.80	0.81	0.77	0.81	0.75	0.75

Table 4. Micro averaged precision for the k parameter in the range [1,6] in the RAkEL algorithm

To test significance of the results in Table 3, we compared the macro-averaged Hamming loss, which is calculated for each movement of each patient, among the three features settings. First, we applied adjusted Friedman test [89] and rejected the null hypothesis that all settings perform the same (p-value<0.05). Once the null hypothesis was rejected, we used the post-hoc Nemenyi test in order to compare the settings with each other. The difference between the combined feature set and the handcrafted feature set was found to be statistically significant with p-value<0.05 (the combined is better). The difference between the combined feature set and the tsfresh feature set had a p-value<0.1. Finally, there was no statistically significant difference between the tsfresh feature set and the handcrafted feature set.

Figure 14 shows all 156 paired feature correlations. The matrix has the same number of rows and columns as the number of features. Cell (i,j) represents the correlation between features i and j. The small number of feature pairs with high correlation demonstrates that the features mostly contribute to the model individually, and not in combination with other features.

To find which of the 156 features were the most important, we applied the LOOCV protocol to calculate the macro-averaged precision of the model, each time with a different feature absent. Then, we extracted the 10 features that have the greatest impact on the model – i.e., those without which, the model's macro-averaged precision score was lowest. Table 5 lists the 10 most informative features, with three handcrafted features and seven which were generated by the tsfresh package. The tsfresh features were generated by the default hyper-parameter of the package without any optimization. Figure 8 shows values of the macro-averaged precision score of the model without each of the top 10 features.

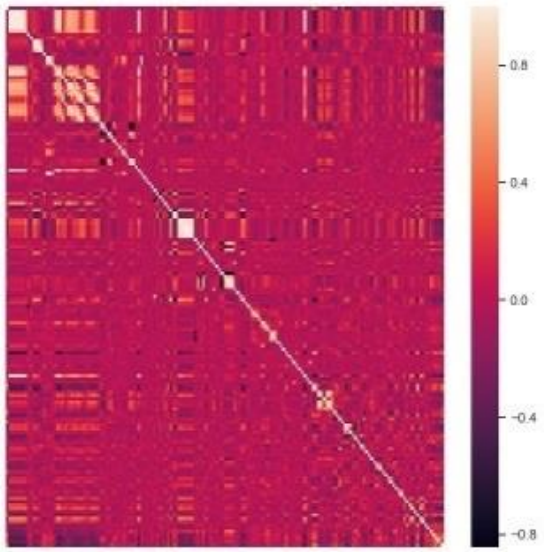


Figure 14. Pairwise correlation between all 156 features.

Figure 15 shows values of the macro-averaged precision score of the model without each of the top 10 features. Those top 10 features contain three handcrafted features and seven which were generated by the tsfresh package. The tsfresh features were generated by the default hyper-parameter of the package without any optimization.

Table 6 lists the top-5 features for each table height. Table 7 lists the top-5 features for each compensation.

	Feature	**Approach**	**Score**
1	Minimum elbow velocity	Handcrafted	0.8268
2	X wrist autocorrelation	tsfresh	0.8268
3	Correlation wrist ↔ trunk	Handcrafted	0.8263
4	X wrist variance - change quantiles	tsfresh	0.8261
5	X wrist energy ratio by segments (10,1)	tsfresh	0.8258
6	X wrist abs energy	tsfresh	0.8255
7	Mean force	Handcrafted	0.8252
8	Y wrist large std	tsfresh	0.8251
9	X wrist energy ratio by segments (10,6)	tsfresh	0.8249
10	Ratio beyond r sigma	tsfresh	0.8247

Table 5. The 10 most-important features

The description, source (handcrafted or based on tsfresh), and average precision score in the absence of each of these features

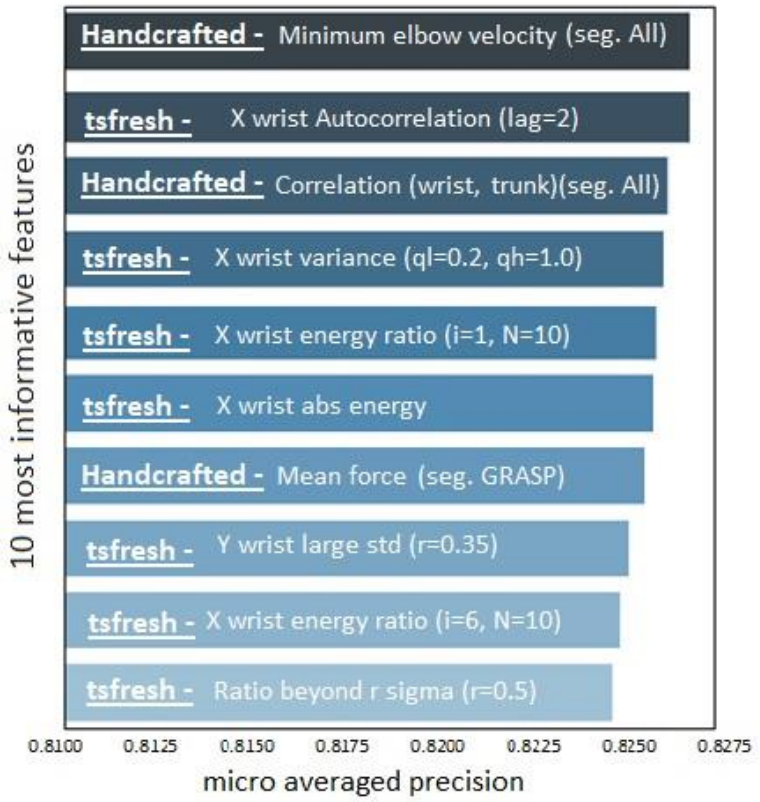


Figure 15. The micro-averaged precision of the 10 most informative features

	Low	Medium	High
1	Minimum elbow velocity (H)	X wrist variance – change quantiles (T)	Minimum elbow velocity (H)
2	X wrist autocorrelation (T)	Minimum elbow velocity (H)	Max angle of scapulae (H)
3	Z wrist mean – change quantiles (T)	Y wrist large std (T)	X wrist variance – change quantiles (T)
4	Correlation wrist, trunk (H)	X wrist mean – change quantiles (T)	Correlation trunk, scapulaRot (H)
5	X wrist energy ratio by chunks (T)	Ratio beyond r sigma (T)	Correlation wrist, trunk (H)

Table 6. The 5 most important features per table height (low/medium/high)

For each feature, it is noted (in parentheses) whether it was handcrafted (H) or generated by the tsfresh package (T).

	C1: trunk flexion	C2: scapular elevation	C3: scapular rotation	C4: shoulder flexion	C5: elbow flexion	C6: dys-synergy distal
1	Minimum velocity, GRASP (H)	Correlation (trunk, elbow) (H)	Minimum elbow velocity, LIFT (H)	X wrist energy ratio by segments (T)	Time to maximum velocity, REACH (H)	X_wrist__autocorrelation (T)
2	Minimum elbow velocity, ALL (H)	Correlation (trunk, scapularRot) (H)	Std diff (H)	X_wrist-autocorrelation (T)	Minimum velocity, LIFT (H)	Time to maximum velocity, LIFT (H)
3	Correlation (trunk, shoulder) (H)	Correlation (trunk, wrist) (H)	Time to T2 (H)	Correlation (trunk, wrist) (H)	Minimum elbow velocity, ALL (H)	Std_diff (H)
4	Y_wrist__percentage_of_reoccurring_datapoints (T)	Correlation (wrist, elbow) (H)	X_wrist__agg_"var" segment (T)	Correlation (trunk, shoulder) (H)	Mean force, LIFT (H)	wrist_abs_energy (T)
5	Correlation (trunk, wrist) (H)	Mean force, LIFT (H)	X_wrist__augmented_"pvalue" (T)	Mean force, LIFT, (H)	Time to T2 (H)	Mean force, LIFT phase (H)

Table 7. The 5 most-important features per label

The most informative five features (rows) for each of the six compensations (columns) are listed in order from the most informative (top) to least (bottom). Each feature name is followed by an indication of whether the feature was handcrafted (H) or generated by tsfresh (T).

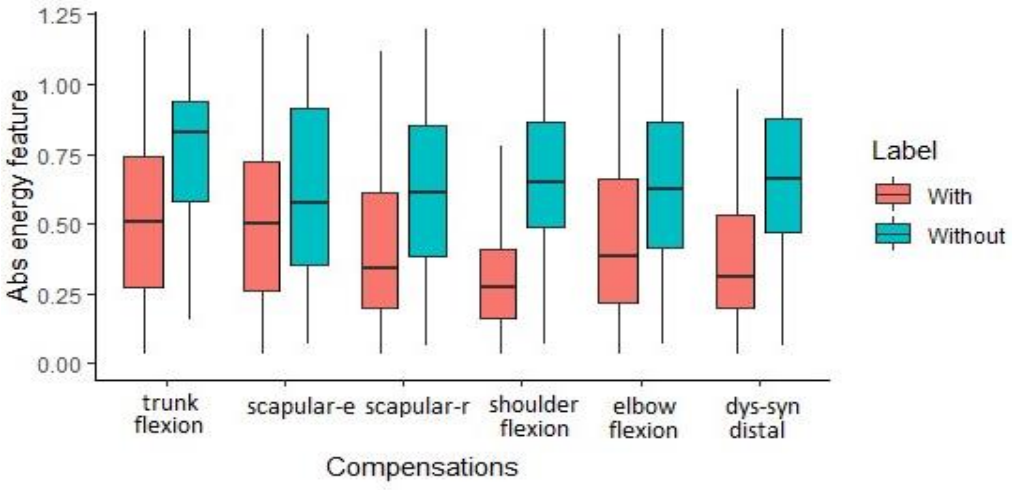


Figure 16. The 'abs energy' feature. This generated by tsfresh, one of the 10 most informative features, in movements with and without each of the six compensations. In red are the values of this feature for all the movements that include the compensation; In green are the values of this feature for all the movements without this compensation.

As an example of the information contribution of an individual feature, we show in figure 16 the value of the ‘X wrist abs energy’ feature for movements with and without each of the six compensations. As shown in the figure, movements with compensation tend to have lower values for this feature. One possible explanation for this is that, for those movements that included compensations, the compensation bypassed the need to extend the elbow, which led to less curvature in the motion of the wrist, resulting in a decrease in the energy of the wrist movement (for the energy calculation see the Section 5.2).

Chapter 9

Conclusions and Future Work

The main objective of this study was to construct an automatic algorithm that can identify whether and which compensations a post-stroke individual made when reaching to a cup, grasping, and lifting it. Our motivation in developing this algorithm was to enable individuals after stroke to practice everyday actions on their own, in addition to the practice they do during physical and occupational therapy sessions, by providing them with the information on whether they performed an undesirable compensation movement during their practice. In this experiment, we used data we recorded from position markers placed along the upper limb of 30 post-stroke individuals when they reached and grasped a cup placed at three different heights, with the cup being either empty or full. Feature generation and feature selection were made to find the optimal set of features which will result in the best performance of the algorithm. We produced two main sets of features: handcrafted and tsfresh-based features. Since the algorithm was trained to identify several compensations simultaneously made in an individual's movement, this is a multi-label task. Moreover, figure 13 shows that there are correlative labels, meaning, there are pairs of compensation that have high correlation. Therefore, we chose the RAkEL algorithm, which considering the correlations between the labels and achieved high classification rates, with 0.85 macro-averaged precision. That is, we identified the presence of the six main compensatory movements (described in detail in the Results section) in 85% of the cases, on average. The identification rate varies per compensation, and ranges between 67% and 93%. We found that the combination of the handcrafted features and the tsfresh features resulted in significantly more accurate identification rates, compared to using each set of features separately. The analysis we present in Tables 5 and 6 provides interesting insights into the most informative features in identifying the six compensations. Although there is an overlap in the most informative features across table heights (Table 6) and compensations (Table 7), they are not exactly the same features in the same order of importance for each of the table heights or compensations. That is, there are some main features that will be informative for all table heights and for all six compensations

(Table 5), as well as unique features that contribute specifically to the identification of a particular compensation, or to the idenfication of compensations performed at a certain table height.

The prevalent way to evaluate movements made by post-stroke individuals is by a physical therapist using standard clinical rating scales such as the FMA or the FTHUE [13]. Recently, an effort has been made to automate these clinical ratings using machine-learning algorithms [2, 3, 4, 16, 19, 20, 21, 22]. The motivation for automating the ratings is threefold: (1) to save time – evaluation with a model can take less than a few minutes, whereas performing the clinical test may take 30 minutes [2, 13, 15]; (2) to improve the precision of the evaluation, as the human eye may not detect small-scale movements that can be detected by accurate sensors [17]; and (3) to help post-stroke individuals to receive an evaluation of their movement quality when a clinician is not available to provide one, e.g., during home practice [4]. Using this system for home practice can also help patients who refrain from going to the clinic due to low availability and accessibility, lack of knowledge of opportunities, high costs of organized activity, inclement weather, or who feel uncomfortable exercising in public since they are concerned about how others might perceive them [45]. The models that have been developed thus far are useful in providing a clinical score. In contrast, our model is based on performing functional tasks. Rather than generating a score, it provides information on the exact compensation the individual performed. We anticipate this will be highly pertinent and informative output for the patients' rehabilitation process, as they work on recovering the ability to perform everyday tasks.We found that when using either the handcrafted features or the tsfresh-generated features, there was no significant difference in the macro-averaged Hamming loss, indicating that both sets of features are equivalent in their contribution to identifying the presence of compensations. However, their combination – handcrafted with tsfresh features – significantly improved the performance of the model, compared to using each set separately. A possible reason for this is that when we created the RGL segments, which were chosen based on the motor-control literature, we attempted to automatically segment the movements across all participants. Since each participant performs movements differently, there was a trade-off between the accuracy of the segmentation, which affects the model's output, and automaticity of the data segmentation. Since automation is a key feature in our system, we strove for automatic segmentation, which may have resulted in the loss of some participant-specific information. The tsfresh set of features, which were calculated for each movement as a whole without applying our segmentation rules, apparently added information that was lost in the calculation of the handcrafted set. Thus, the combination of both sets of features led to the best results.

The algorithm we developed offers several benefits over existing models for movement evaluation: (1) It uses everyday functional movements, of the type individuals after stroke are often asked to practice; Thus, it avoids the artificial nature of a test situation which does not reflect real everyday movements [14, 17, 18]; (2) It is based on data from stroke patients; (3) It provides direct information on the compensatory movements that the individual performed – whether there were any, and which. Thus, it can address a potential concern that patients may have regarding at-home practice. According to Moriss et al. (2017), translation of motivation into actual activity depends on capability for physical activity. For example, intrinsic influences, such as the fear of negative consequences of physical activity, may prevent individuals from being physically active [45]. Indeed, repeatedly performing compensatory movements is known as "bad learned use" [27], and should be avoided. Using a model as the one we present here, provides the users with accurate feedback on their movement performance. It can assist patients to perform movements more correctly and thus help them experience success, which, according to Moriss et al (2017), leads to increased motivation and is translated to confidence in the general capability to be active.

By tracking the individuals' performance over time (e.g., what compensations are present, and whether they diminish over time), both the clinical team and the patient can have an accurate evaluation of the process of recovery and identify particular recurring difficulties.

One possible limitation of our work is the sample size – 30 post-stroke individuals. While it is large compared to previous works with human participants, a larger sample size may allow for better generalization, and have a more balanced data set (a better representation of the different impairment levels) [46]. In addition, this group of participants was heterogeneous in terms of their functional ability, with patients displaying low, moderate and severe levels of impaired function. Despite this, the model we developed reached a macro-averaged precision of 0.85.

Another limitation is the use of markers for data collection. In this work, we aimed at generating an accurate algorithm for the identification of compensatory movements in stroke-patient movements. For that purpose, we used high-precision motion-capture device (V120: Trio OptiTrack, NaturalPoint, Inc., OR, USA, accuracy $\leq$ 1.0mm). It is conceivable that simpler implementations of the model may use a smaller number of sensors (in our model, we used only 6 of the 11 sensors we recorded from, namely: the sternum 1, sternum 2, shoulder, elbow, wrist and force sensors), or potentially a different, low-cost, simple sensor system, which would add to the user's convenience of using the algorithm both in terms of procedure and of price. Enabling a

low-cost automated supervision of at-home practice would be important in that during home-based practice there are fewer constraints on time and space, so the patients can practice more frequently, for longer periods of time, and also according to their own schedule and terms [3, 4, 47]. Also, at home, the individuals can engage in more ecological exercises that are compatible with their everyday routines, which may be more indicative and useful from an evaluation session in the clinic [3, 4, 47].

In this study, we constructed a model that identifies the presence of compensations in stroke patients' movements, to be used in the process of rehabilitation. We achieved 85% macro-averaged precision across the six movement compensations we studied. This is the first study to identify compensations based on stroke patients' data. Here, we used a high-precision movement-capture system. However, future work with a more affordable sensor system may open the possibility for stroke patients to use the model system for home-based training. The automated algorithm we present here may further be combined with socially assistive robotics (SAR), which can administer the exercise set, and provide feedback on the user's performance [48, 49, 50]. A potential line of future investigation would be how the compensation-specific information that the SAR may deliver to users affects their acceptance and level of trust in using such a device in the process of rehabilitation [51, 52]. Lastly, it would be instructive to collect and use data from the unimpaired arm when individuals with stroke perform functional RTG movements, as compensatory strategies may also involve the unimpaired side of the body.

References

[1] J. Brackenridge, L. Brandam, S. Lennon, J. Costi, D. Hobbs, A Review of Rehabilitation Devices to Promote Upper Limb Function Following Stroke., Neuroscience and Biomedical Engineering 4 (2016) 25-42.

[2] S. Lee, Y.S. Lee, J. Kim, Automated Evaluation of Upper-Limb Motor Function Impairment Using Fugl-Meyer Assessment., IEEE Transactions on Neural Systems and Rehabilitation Engineering 26(1) (2018) 125-134.

[3] J. Wang, L. Yu, J. Wang, L. Guo, X. Gu, Q. Fang, Automated Fugl-Meyer assessment using SVR model., IEEE International Symposium on Bioelectronics and Bioinformatics (ISBB), 2014, pp. 1-4.

[4] L. Yu, D. Xiong, L. Guo, J. Wang, A remote quantitative Fugl-Meyer assessment framework for stroke patients based on wearable sensor networks., Computer methods and programs in biomedicine 128 (2016) 100-110.

[5] V. Van Asch, Macro-and micro-averaged evaluation measures [[basic draft]]., CLiPS, Belgium, 2013, pp. 1-27.

[6] M. Cirstea, M. Levin, Compensatory strategies for reaching in stroke., Brain 123 (2000) 940-953.

[7] M.F. Levin, Interjoint coordination during pointing movements is disrupted in spastic hemiparesis., Brain 119(1) (1996) 281-293.

[8] T. Shaikh, V. Goussev, A. Feldman, M. Levin, Arm-trunk coordination for beyond-the-reach movements in adults with stroke., Neural Repair 28 (2014) 355-366.

[9] M.F. Levin, D.G. Liebermann, Y. Parmet, S. Berman, Compensatory versus noncompensatory shoulder movements used for reaching in stroke, Neurorehabilitation and neural repair 30(7) (2016) 635-646.

[10] C.E. Lang, J.M. Wagner, A.J. Bastian, Q. Hu, D.F. Edwards, S.A. Sahrmann, A.W. Dromerick, Deficits in grasp versus reach during acute hemiparesis, Experimental Brain Research 166(1) (2005) 126-136.

[11] M.C. Baniña, A.A. Mullick, B.J. McFadyen, M.F. Levin, Upper limb obstacle avoidance behavior in individuals with stroke., Neurorehabilitation and neural repair 31(2) (2017) 133-146.

[12] D.G. Liebermann, S. Berman, P.L. Weiss, M.F. Levin, Kinematics of reaching movements in a 2-D virtual environment in adults with and without stroke. , IEEE Transactions on Neural Systems and Rehabilitation Engineering 20(6) (2012) 778-787.

[13] R. Osu, K. Ota, T. Fujiwara, Y. Otaka, M. Kawato, M. Liu, Quantifying the quality of hand movement in stroke patients through three-dimensional curvature., Journal of neuroengineering and rehabilitation 8(1) (2011) 62.

[14] L. Dipietro, H.I. Krebs, B.T. Volpe, J. Stein, C. Bever, S.T. Mernoff, N. Hogan, Learning, not adaptation, characterizes stroke motor recovery: evidence from kinematic changes induced by robot-assisted therapy in trained and untrained task in the same workspace., IEEE Transactions on Neural Systems and Rehabilitation Engineering 20(1) (2011) 48-57.

[15] B. Hamilton, A uniform national data system for medical rehabilitation., Rehabilitation outcomes: analysis and measurement (1987) 137-147.

[16] J. Sanford, J. Moreland, L.R. Swanson, P.W. Stratford, C. Gowland, Reliability of the Fugl-Meyer assessment for testing motor performance in patients following stroke., Physical therapy 73(7) (1993) 447-454.

[17] G. Sprint, D.J. Cook, D.L. Weeks, V. Borisov, Predicting functional independence measure scores during rehabilitation with wearable inertial sensors, IEEE Access 3 (2015) 1350-1366.

[18] D.J. Gladstone, C.J. Danells, S.E. Black, The Fugl-Meyer assessment of motor recovery after stroke: a critical review of its measurement properties. , Neurorehabilitation and neural repair 16(3) (2002) 232-240.

[19] S.H. Lee, M. Song, J. Kim, Towards clinically relevant automatic assessment of upper-limb motor function impairment, IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), 2016, pp. 148-151.

[20] M.A. Villán-Villán, R. Pérez-Rodríguez, C. Gómez, E. Opisso, J.M. Tormos, J. Medina, E.J. Gómez, Automated Fugl-Meyer assessment for ABI subjects in upper limb physical (2015).

[21] B.T. McMahon, L.R. Shaw, advances in brain injury rehabilitation (1991).

[22] L.K. Kwah, R.D. Herbert, Prediction of walking and arm recovery after stroke: a critical review., Brain sciences 6 (2016).

[23] J.M. Veerbeek, G. Kwakkel, E. van Wegen, J.C. Ket, M.W. Heymans, Early prediction of outcome of activities of daily living after stroke: a systematic review, Stroke 42(5) (2011) 1482-1488.

[24] Mirbagheri, M. M., & Rymer, W. Z. (2008). Time-course of changes in arm impairment after stroke: variables predicting motor recovery over 12 months. Archives of physical medicine and rehabilitation, 89(8), 1507-1513

[25] Fugl-Meyer AR, Jaasko L, Leyman I, Olsson S, Steglind S (1975) The poststroke hemiplegic patient: a method for evaluation of physical performance. Scand J Rehabil Med 7: 13–31.

[26] Duncan, P. W., Propst, M., & Nelson, S. G. (1983). Reliability of the Fugl-Meyer assessment of sensorimotor recovery following cerebrovascular accident. Physical therapy, 63(10), 1606-1610.

[27] Padovani, C., Pires, C. V. G., Ferreira, F. P. C., Borin, G., Filippo, T. R. M., Imamura, M., & Battistella, L. R. (2016). Application of the Fugl-Meyer Assessment (FMA) and the Wolf Motor Function Test (WMFT) in the recovery of upper limb function in patients after chronic stroke: A literature review. Acta Fisiatr, 20, 42-49.

[28] Lin, J. H., Hsueh, I. P., Sheu, C. F., & Hsieh, C. L. (2004). Psychometric properties of the sensory scale of the Fugl-Meyer Assessment in stroke patients. Clinical rehabilitation, 18(4), 391-397.

[29] Jørgensen, H. S., Nakayama, H., Raaschou, H. O., Vive-Larsen, J., Støier, M., & Olsen, T. S. (1995). Outcome and time course of recovery in stroke. Part I: Outcome. The Copenhagen Stroke Study. Archives of physical medicine and rehabilitation, 76(5), 399-405.

[30] Duncan, P. W., Goldstein, L. B., Matchar, D., Divine, G. W., & Feussner, J. (1992). Measurement of motor recovery after stroke. Outcome assessment and sample size requirements. Stroke, 23(8), 1084-1089.

[31] Kwakkel, G., Kollen, B., & Twisk, J. (2006). Impact of time on improvement of outcome after stroke. stroke, 37(9), 2348-2353.

[32] Sullivan, K. J., Tilson, J. K., Cen, S. Y., Rose, D. K., Hershberg, J., Correa, A., ... & Duncan, P. W. (2011). Fugl-Meyer assessment of sensorimotor function after stroke: standardized training procedure for clinical practice and clinical trials. Stroke, 42(2), 427-432

[33] G. Tsoumakas, I. Katakis, Multi-label classification: An overview., International Journal of Data Warehousing and Mining (3) (2007) 1-13.

[34] L. Rokach, A. Schclar, E. Itach, Ensemble methods for multi-label classification, Expert Systems with Applications 41(16) (2014) 7507-7523.

[35] M. Alaverdashvili, I. Whishaw, A behavioral method for identifying recovery and compensation: Hand use in a preclinical stroke model using the single pellet reaching task. , Neuroscience and Biobehavioral Reviews 37 (2013) 950-967.

[36] Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. Machine learning, 85(3), 333.

[37] Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. Pattern recognition, 37(9), 1757-1771.

[38] Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. The annals of statistics, 26(5), 1651-1686.

[39] Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

[40] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[41] Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., & Feuston, B. P. (2003). Random forest: a classification and regression tool for compound classification and QSAR modeling. Journal of chemical information and computer sciences, 43(6), 1947-1958.

[42] Breiman, L., & Cutler, A. (2007). Random forests-classification description. Department of Statistics, Berkeley, 2.

[43] Nielsen, D. (2016). Tree Boosting With XGBoost-Why Does XGBoost Win" Every" Machine Learning Competition? (Master's thesis, NTNU).

[44] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). ACM.

[45] Zheng, H., Yuan, J., & Chen, L. (2017). Short-term load forecasting using EMD-LSTM neural networks with a Xgboost algorithm for feature importance evaluation. Energies, 10(8), 1168.

[46] Christ, M., Kempa-Liehr, A. W., & Feindt, M. (2016). Distributed and parallel time series feature extraction for industrial big data applications. arXiv preprint arXiv:1610.07717.

[47] Tsoumakas, G., Katakis, I., and Vlahavas, I. Random k-Labelsets for Multilabel Classification. IEEE Transactions on Knowledge and Data Engineering 23, 7 (July 2011), 1079–1089. 2, 6, 23, 48, 49, 68

[48] Tsoumakas, G., and Vlahavas, I. Random k-Labelsets : An Ensemble Method for Multilabel Classification. In the 18th European Conference on Machine Learning (Warsaw, Poland, 2007), Springer Berlin Heidelberg, pp. 406–417. 48, 49

[49] Goutte, C., & Gaussier, E. (2005, March). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In European Conference on Information Retrieval (pp. 345-359). Springer, Berlin, Heidelberg.

[50] Y. Yang, X. Liu, A re-examination of text categorization methods., In Sigir 99(8) (1999) 99.

[51] M.L. Zhang, Z.H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, Pattern recognition 40(7) (2007) 2038-2048.

[52] Cherman, E. A., Spolaôr, N., Valverde-Rebaza, J., & Monard, M. C. (2015). Lazy multi-label learning algorithms based on mutuality strategies. Journal of Intelligent & Robotic Systems, 80(1), 261-276.

[53] Fawcett, T. (2006). An introduction to ROC analysis. Pattern recognition letters, 27(8), 861-874.

[54] Bradley, A.P., 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recogn. 30 (7), 1145–1159.

[55] Hanley, J.A., McNeil, B.J., 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143, 29– 36.

[56] Zou, K. H. (2002). Receiver operating characteristic (ROC) literature research. On-line bibliography available from:< http://splweb. bwh. harvard. edu, 8000.

[57] Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. Wadsworth International Group, Belmont, CA.

[58] Hand, D.J., Till, R.J., 2001. A simple generalization of the area under the ROC curve to multiple class classification problems. Mach. Learning 45 (2), 171–186.

[59] Menon, A. K., & Elkan, C. (2011, September). Link prediction via matrix factorization. In Joint european conference on machine learning and knowledge discovery in databases (pp. 437-452). Springer, Berlin, Heidelberg.

[60] Ferri, C., Flach, P., & Hernández-Orallo, J. (2002, July). Learning decision trees using the area under the ROC curve. In ICML (Vol. 2, pp. 139-146).

[61] Cortes, C., & Mohri, M. (2003). AUC optimization vs. error rate minimization advances. Neural Information Processing Systems. MIT Press, Cambridge.

[62] Joachims, T. (2005). A support vector method for multivariate performance measures. Proceedings of the 22nd International Conference on Machine Learning. ACM Press.

[63] Prati, R., & Flach, P. (2005). ROCCER: an algorithm for rule learning based on ROC analysis. Proceeding of the 19th International Joint Conference on Artificial Intelligence. Edinburgh, Scotland.

[64] Yan, L., Dodier, R., Mozer, M., & Wolniewicz, R. (2003). Optimizing classifier performance via the WilcoxonMann-Whitney statistics. Proceedings of the 20th International Conference on Machine Learning.

[65] Herschtal, A., & Raskutti, B. (2004). Optimising area under the ROC curve using gradient descent. Proceedings of the 21st International Conference on Machine Learning (p. 49). New York, NY, USA: ACM Press.

[66] Srinivasan, A. (2003). The Aleph Manual Version 4. http://web.comlab.ox.ac.uk/ oucl/ research/ areas/ machlearn/ Aleph/.

[67] Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In Proceedings of the 23rd international conference on Machine learning (pp. 233-240). ACM.

[68] Cawley, G. C., & Talbot, N. L. (2004). Fast exact leave-one-out cross-validation of sparse least-squares support vector machines. Neural networks, 17(10), 1467-1475.

[69] Kim, I. J., Lim, S. B., Kang, H. C., Chang, H. J., Ahn, S. A., Park, H. W., ... & Choi, H. S. (2007). Microarray gene expression profiling for predicting complete response to

preoperative chemoradiotherapy in patients with advanced rectal cancer. Diseases of the Colon & Rectum, 50(9), 1342-1353.

[70] Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences, 95(25), 14863-14868.

[71] Efron, B., & Tibshirani, R. J. (1993). An introduction to the bootstrap. Chapman and Hall, New York, NY. Farrell, J., Johnston, M. and Twynam, D.(1998),‘‘Volunteer motivation, satisfaction, and management at an elite sporting competition’’, Journal of Sport Management, 12, 288-300.

[72] P. Otten, J. Kim, S.H. Son, A framework to automate assessment of upper-limb motor function impairment: A feasibility study., Sensors 15(8) (2015) 20097-20114.

[73] J.R. Kwapisz, G.M. Weiss, S.A. Moore, Activity recognition using cell phone accelerometers, ACM SigKDD Explorations Newsletter 12(2) (2011) 74-82.

[74] P. Tormene, T. Giorgino, S. Quaglini, M. Stefanelli, Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation., Artificial intelligence in medicine 45(1) (2009) 11-34.

[75] R. Kizony, P.L. Weiss, O. Elion, S. Harel, I. Baum-Cohen, T. Krasovsky, M. Shani, Development and validation of tele-health system for stroke rehabilitation., International Journal on Disability and Human Development 13(3) (2014) 361-368.

[76] M. Capecci, M.G. Ceravolo, F. Ferracuti, S. Iarlori, V. Kyrki, S. Longhi, F. Verdini, Physical rehabilitation exercises assessment based on hidden semi-markov model by kinect v2, IEEE-EMBS International Conference In Biomedical and Health Informatics (BHI), 2016, pp. 256-259.

[77] P.W. McClure, L.A. Michener, B.J. Sennett, A.R. Karduna, Direct 3-dimensional measurement of scapular kinematics during dynamic movements in vivo. , Journal of shoulder and elbow surgery 10(3) (2001) 269-277.

[78] A.J. Turton, P. Cunningham, E. Heron, F. van Wijck, C. Sackley, C. Rogers, P. van Vliet, Home-based reach-to-grasp training for people after stroke: study protocol for a feasibility randomized controlled trial., Trials 14(1) (2013) 109.

[79] T. Flash, N. Hogan, The coordination of arm movements: an experimentally confirmed mathematical model., Journal of neuroscience 5(7) (1985) 1688-1703.

[80] N. Hogan, D. Sternad, Sensitivity of smoothness measures to movement duration, amplitude, and arrests, Journal of motor behavior 41(6) (2009) 529-534.

[81] M.A. Murphy, C. Willén, K.S. Sunnerhagen, Kinematic variables quantifying upper-extremity performance after stroke during reaching and drinking from a glass., Neurorehabilitation and neural repair 25(1) (2011) 71-80.

[82] P.M. van Vliet, M.R. Sheridan, Coordination between reaching and grasping in patients with hemiparesis and healthy subjects., Archives of physical medicine and rehabilitation 88(10) (2007) 1325-1331.

[83] S.M. Michaelsen, R. Dannenbaum, M.F. Levin, Task-specific training with trunk restraint on arm recovery in stroke: randomized control trial, Stroke 37(1) (2006) 186-192.

[84] S.M. Michaelsen, S. Jacobs, A. Roby-Brami, M.F. Levin, Compensation for distal impairments of grasping in adults with hemiparesis, Experimental Brain Research 157(2) (2004) 162-173.

[85] G. Holmes, A. Donkin, I.H. Witten, Weka: A machine learning workbench, (1994)

[86] G. Doquire, M. Verleysen, Feature selection for multi-label classification problems, In International work-conference on artificial neural networks, 2011, pp. 9-16.

[87] Cawley, G. C., & Talbot, N. L. (2003). Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers. Pattern Recognition, 36(11), 2585-2592.

[88] R.E. Schapire, Y. Singer, BoosTexter: A boosting-based system for text categorization., Machine learning 39(2-3) (2000) 135-168.

[89] W.W. Daniel, Friedman two-way analysis of variance by ranks., Applied nonparametric statistics (1990) 262-274.

[90] J.H. Morris, T. Oliver, T. Kroll, S. Joice, B. Williams, Physical activity participation in community dwelling stroke survivors: synergy and dissonance between motivation and capability. A qualitative study., Physiotherapy 103(3) (2017) 311-321.

תקציר

יותר מ50% מהאנשים אשר לקו באירוע מוחי, תפקודם המוטורי נפגע. לכן, השיקום המוטורי הפך במהרה לשלב הכרחי בתהליך ההחלמה מאירוע נוירולוגי. מכאן, כאשר אדם סובל מפציעה, מחלה או אירוע נוירולוגי, הוא בדרך כלל עובר שיקום אינטנסיבי על מנת לשחזר את התפקוד המוטורי. במהלך תהליך השיקום, חשוב שהמטופל ידע האם הוא מבצע את התרגילים בצורה תיקנית ונכונה, למען שיקום מירבי. דירוג קליני סטנדרטי שנערך על ידי התבוננות אישית של מטפל מוסמך\פיזיותרפיסט היא הדרך הרווחת כיום לפקח על איכות תרגול התנועות של המטופל. לכן, מטופל המבצע תנועות לבד, ללא פיקוח, אינו יכול לדעת אם הוא מבצע תנועה כראוי. ההקפדה על משטר תרגול התנועות הופכת את תהליך השיקום ליעיל יותר ויותר ולכן מערכת שיכולה לתת למטופל משוב על ביצועיו היא הכרחית ובעלת ערך רב.

כאן, אנו מדווחים על שני מודלים אוטומטים מבוססי למידת מכונה שבנינו. המודל הראשון נבנה כדי להבחין בין חולי שבץ מוחי למשתתפים בריאים, ואילו המודל השני, המייצג את מטרתו העיקרית של מחקר זה, היה לתת למטופלים מידע מדויק על הקומפנסציות (תנועות המפצות\הלא רצויות) שהם מבצעים במהלך ביצוע התנועה, בהיעדר מטפל/פיזותרפיסט בחדר. על מנת לבנות את שני המודלים, השתמשנו במידע שהקלטנו מראש של 30 חולי שבץ ו16 אנשים בריאים, אשר ביצעו כל אחד 18 תנועות שונות. תנועות אלו עזרו להבחין אם המטופל חולה או בריא, ובנוסף עזרו להבחין בקיומם של שישה סוגי קומפנסציות בתנועות השייכות לחולי שבץ. במודל השני, המרכזי, השתמשנו באלגוריתם הנקרא RAkEL אשר מכיל בתוכו גם את האלגוריתם Random-Forest , בכדי להתמודד עם בעיית הmulti-label, החזרת כל סוגי הקומפנסציות שהופיעו בתנועה מסוימת.

הגענו לתוצאה של macro averaged precision 85% על פני ששת סוגי הקומפסנציות בתנועות שבחנו. זהו המחקר הראשון החוקר זיהוי אוטמטי של קומפנסציות בתנועה המבוסס על תנועות של אנשים חולי שבץ. אנו מאמינים שמודל זה יכול להתאים לשימוש הן בתוך הקליניקנה והן בביתו של המטופל ולהוות חלק מתוכנית האימון.

אוניברסיטת בן-גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת מערכות מידע ותוכנה

מודל מבוסס למידת המכונה המזהה קומפנסציות בתנועות אצל מטופלים חולי שבץ

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: שיר קשי

מנחה/ים: פרופ' ליאור רוקח, ד"ר שלי לוי-צדק, פרופ' בועז לרנר

חתימת המחבר..... תאריך..08/10/2019

אישור המנחה/ים..... ..... תאריך..08/10/2019

..... תאריך.....9/10/2019

..... תאריך.....9.10.2019

אישור יו"ר ועדת תואר שני מחלקתית..... תאריך.....

אוניברסיטת בן-גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת מערכות מידע ותוכנה

מודל מבוסס למידת המכונה המזהה קומפנסציות בתנועות אצל מטופלים חולי שבץ

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: שיר קשי

תשרי תש"פ

אוגוסט 2019