MILITARY UNIVERSITY OF TECHNOLOGY

Ph.D. THESIS

„Optimization of Medicine Dosing in Parkinson's Disease, Based on Signals from Sensor Measurements”

Tomasz GUTOWSKI, M.Sc.

Supervisor: **Ryszard ANTKIEWICZ, Ph.D., D.Sc.**

Assistant supervisor**: Mariusz CHMIELEWSKI, Ph.D.**

Warsaw 2024

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Ryszard Antkiewicz and Mariusz Chmielewski, for their invaluable time, assistance, and constructive feedback, which have been instrumental in the preparation of this dissertation.

I am also deeply thankful to Stanisław Szlufik and the members of the NEKON scientific circle, without whose support it would have been impossible to obtain the necessary data for conducting this research.

For their unwavering support and assistance, I extend my heartfelt thanks to my parents and sisters, especially Kasia, who has been a pillar of strength and support throughout my journey. I would also like to express my deepest appreciation to my close friends, Daniel, Karolina, and Kasia, whose motivation and encouragement have been vital to the completion of this work.

Table of Contents

1. Introduction

1.1. Parkinson’s disease

Parkinson's disease (PD) is a progressive neurodegenerative disorder that significantly impacts the lives of over six million individuals globally [1]. As a complex condition, it presents a spectrum of symptoms that progressively impair motor and non-motor functions.

The disease is characterized by the degeneration of dopaminergic neurons in the substantia nigra (presented in Figure 1), the region of the brain, that has an important role in movement and reward system [2]. These neurons are responsible for dopamine production and their impaired functioning causes the manifestation of symptoms of the disease. The most commonly known symptoms of the disease include the impairment of motor functions of the patient such as tremors, bradykinesia (slowness of movement), muscle stiffness, and posture instability [3]. The onset of the symptoms is usually gradual, and they usually manifest on one side of the body first. As the disease progresses, the symptoms spread to the entire body of the patient [2].

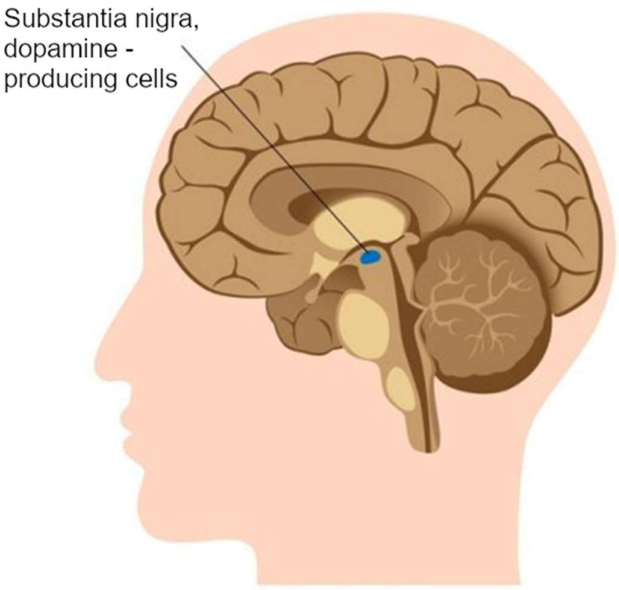

Figure 1 The location of substantia nigra in the human brain, where dopaminergic neurons reside [4]

PD patients do not suffer only from motor symptoms affecting their physical health. They experience a wide range of non-motor symptoms too. These usually include apathy, sleep problems, cognitive impairment, depression, anxiety, and sensory

abnormalities [3]. Their presence and severity can precede the motor symptoms and the diagnosis of the disease by even 10 years. They are increasingly recognized for their impact on patient quality of life and disease burden [3].

Diagnosing Parkinson's disease (PD) is not straightforward tasks. Clinicians usually look for common PD symptoms like tremors and slowness of movement and check how these symptoms improve with specific PD medications, particularly those that increase dopamine, a key brain chemical. The mainly used medicine in this approach is levodopa – a dopamine precursor able to cross the blood-brain barrier [5]. While brain scans, like dopamine transporter scans, can help in the diagnosis, they do not provide a definite answer. The differential diagnosis of PD distinguishing it from other diseases that have a similar profile, but are caused by different factors, e.g., essential tremor and atypical parkinsonian disorders. These, while having similar symptoms require another approach to therapy, making the correct diagnosis crucial [6].

Due to the fact there is no cure for the disease [7], PD is characterized by the progression. As the number of functioning dopaminergic neurons in substantia nigra decreases, the disease progresses and becomes more difficult to manage. The progression greatly varies among patients, making it a disease requiring highly individualized treatment. Clinicians often use rating scales, like the Hoehn and Yahr [8] scale and the Unified Parkinson's Disease Rating Scale (UPDRS) [9], to assess disease progression and the impact of symptoms on daily living.

The exact cause of Parkinson's Disease is unknown. However, it is attributed to a combination of genetic, environmental, and age-related factors [2]. Certain genetic mutations have been identified in familial PD cases, offering insights into potential disease mechanisms [10]. Environmental factors, such as pesticide exposure and rural living, have been associated with an increased risk of developing PD, though the direct causality is still being researched [11].

Up to this day, there is no cure for PD, and treatment focuses on managing symptoms and improving patient quality of life. Pharmacological treatments, most notably levodopa, aim to maintain the expected level of dopamine in the brain, to make up for degenerated neurons [12]. Other medications used include dopamine agonists, MAO-B inhibitors, and COMT inhibitors. Advanced stages of PD may warrant surgical interventions, such as deep brain stimulation. All of these treatments, while effective in

managing symptoms, do not stop or slow down the disease's progression and their impact on patient's condition may decrease over time [13].

In conclusion, PD is a complicated disease with a significant impact on individuals and healthcare systems. Its differential diagnosis can be sometimes difficult, and the main medication used to decrease the symptom severity is levodopa – a dopamine precursor.

1.2. Detection and evaluation of PD symptoms

As previously stated, PD is characterized by both motor and non-motor symptoms, both affecting the patient's daily life. The main symptom associated with the disease is the tremor, which can manifest in three main forms: rest tremor – when the muscles are resting, postural tremor – apparent when the muscles are keeping a posture and kinetic tremor – present when patients are in movement [14]. In the case of PD, rest tremor is the most common, typically occurring at a frequency between 4 and 6 Hz [3]. It often manifests in one limb and is often characterized by a circular movement of the thumb and index finger.

Another common and key symptom of PD is bradykinesia, which refers to slowness of movement [15]. It affects a range of activities and can manifest in different ways, including reduction of automatic movements (e.g., blinking), difficulties with starting a movement (e.g., standing up), overall slowness during activities (e.g., drawing, writing) and limited facial expressions. Another symptom, important in PD diagnosis is rigidity, observed in the form of muscle stiffness of arms or legs [3]. It can result in a decreased range of motion, as well as pain in the muscles or joints. It also affects daily activities e.g., reducing arm swinging while walking and facial expressions.

As the disease progresses the symptoms become more visible and the movement troubles start to affect the whole body leading to other symptoms such as balance problems, falling, freezing of gait (difficulty of moving feet forward while walking, being "stuck" in place) [2,3]. Apart from that patients often experience problems with writing and drawing, their handwriting becomes less readable and micrographia (small, cramped handwriting) might be present as well. The disease might also affect the speech of the patients, causing them to mumble and slur words. The overall speaking difficulty can make them difficult to understand.

In addition to described motor symptoms, many patients experience also non-motor symptoms, their onset can sometimes precede the motor symptoms. These include apathy, lack of emotional involvement, daytime sleepiness, and other sleep disorders. Among PD patients constipations are quite common [3].

1.3. Therapy characteristic

Parkinson's disease is a result of degeneration of dopaminergic neurons, which leads to decreased dopamine levels causing most of PD symptoms. The main goal of therapy is to return to the desired dopamine level, therefore mitigating the symptoms. There are multiple approaches to achieve that.

The most common medication used in PD is levodopa [12], which stands as the most effective treatment. Levodopa is a dopamine precursor, capable of crossing the blood-brain barrier and is subsequently metabolized into dopamine in the central nervous system. Therefore, it counteracts the dopamine deficiency that is central to PD pathology. However, as the disease progresses the patients begin to experience fluctuations in their state, known as the ON/OFF phenomenon [16]. These two states refer to the condition of the patient. The ON state indicates that the medication is effective, and the patient does not experience the symptoms of the disease. When patients are in the OFF state, it signifies that they are not under the influence of medication and their condition is worsening – the medication's effects are diminishing, and they experience the symptoms. Figure 2 presents how the effect of the treatment changes, as the disease progresses. In the early stages of the disease, patients' symptoms are fully controlled by levodopa medication. Nevertheless, with every year the therapeutic window gets thinner, and the patients are more likely to experience symptoms during the day [17].

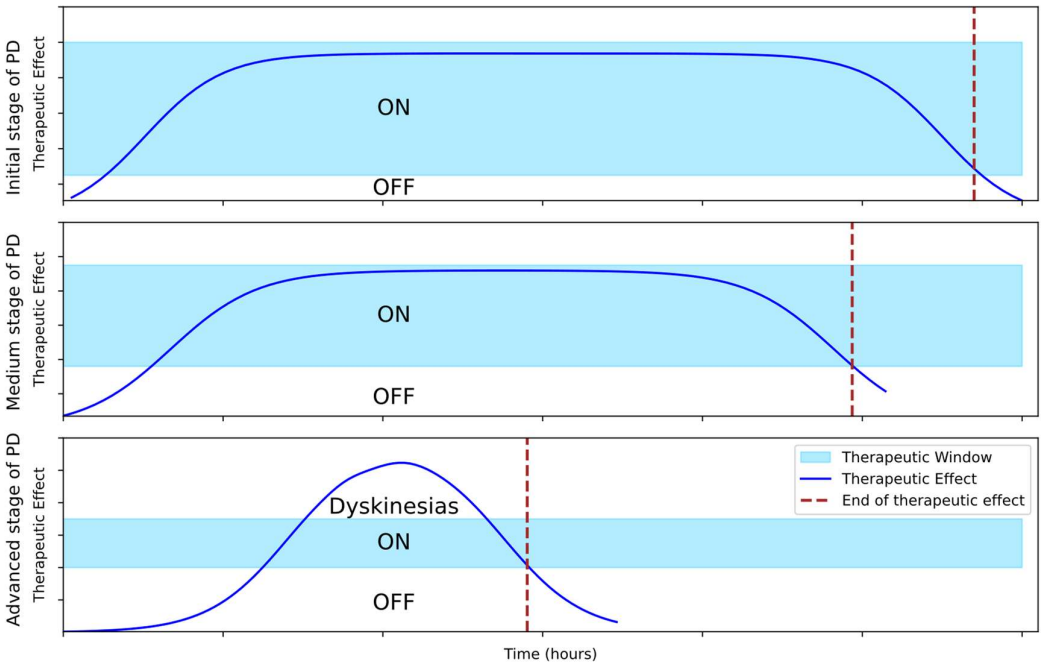

Figure 2 Therapeutic effect of levodopa medication after intake in different stages of PD

Managing fluctuations in PD symptoms often requires careful adjustment of the levodopa regimen, sometimes supplemented with other drugs to extend the ON periods, and reduce the OFF periods. This may include using extended-release formulas of Levodopa or adding medications that prolong levodopa's effect.

Experiencing the ON/OFF fluctuations is not the only challenge for PD patients, as presented in Figure 2. PD patients might experience levodopa-induced dyskinesias, a significant side effect associated with long-term use of levodopa. These dyskinesias are represented by involuntary, erratic movements usually affecting the limbs. They are not a symptom of the disease, but a side effect of medication, typically experienced in the ON state, when levodopa concentration in the blood is the highest. This condition is thought to arise due to the pulsatile stimulation of dopamine receptors in the brain.

Due to the short half-life of levodopa, which leads to a short wearing off, it is often taken along with other substances that are meant to improve its therapeutic effectiveness. For instance, carbidopa prevents the premature conversion of levodopa to dopamine, by inhibiting the enzyme responsible for the conversion [18], resulting in the prolonging the ON time after intake. A similar effect has benserazide, another aromatic L-amino acid decarboxylase inhibitor (AADC), which allows more levodopa to pass the blood-brain barrier and improve the effect of levodopa. These substances additionally reduce nausea,

vasoconstriction, and arrhythmia, caused often by peripheral dopamine (present outside the nervous system), and are often combined in the same pills with levodopa for convenient administration.

Improving levodopa effectiveness can also be achieved by catechol-O-methyltransferase (COMT) inhibitors such as entacapone and tolcapone which can be applied alongside carbidopa or bensarazide. These inhibitors; however, reduce methylation of levodopa leading to more levodopa passing to the brain, increasing its bioavailability, the effects of these inhibitors are illustrated in Figure 3.

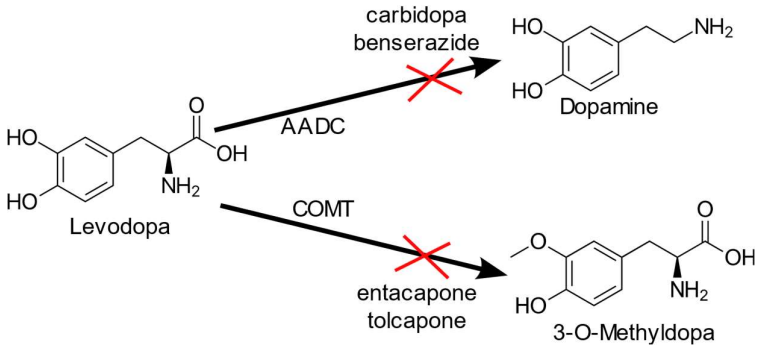

Figure 3 The metabolism of levodopa before crossing the blood-brain barrier.

Another medication group used in PD treatment is dopamine agonists (e.g., Ropinirole and Pramipexole), which activate dopamine receptors similarly to dopamine [7]. These can be used alone or together with levodopa medication. Adamantane derivatives are also commonly prescribed in PD, providing a complex effect on the patient, being agonists and antagonists of multiple receptors. However, they are often used alongside levodopa, to decrease the intensity of levodopa induced dyskinesias. To reduce the breakdown of dopamine, another group can be applied – Monoamine oxidase B (MAO-B) inhibitors, by slowing down this process, they increase the level of dopamine and can be used alone in early stages of PD or together with other medication in later stages. The last group of commonly used medication is called anticholinergics. They block the actions of acetylcholine, the imbalance between levels of acetylcholine and dopamine in the brain is the reason for many PD symptoms. These are the oldest PD medications, which might also cause significant side effects, especially in older patients [7].

Beyond oral medication, advanced therapeutic approaches are available for PD patients [19]. They typically involve a more invasive procedure or surgical interventions.

Such options are generally considered when conventional approaches are insufficient or result in side effects. To ensure that the benefits outweigh the risks, every patient must go through a complex qualification process. Most widely used advanced therapies include Deep Brain Stimulation (DBS), Duodopa and apomorphine [19].

DBS involves placing electrodes in the brain to deliver electrical impulses. These can treat motor fluctuation, tremor and dyskinesias. During therapy, the DBS device is programmed by the clinician, and is configured individually for every patient. Apomorphine is a dopamine agonist, it is usually not administered orally in PD, but can be used as ad-hoc treatment for unpredictable OFF states or as continuous infusion treatment. It is reported to be effective against both motor and non-motor advanced symptoms [19,20]. Duodopa uses a levodopa carbidopa intestinal gel administered directly into the small intestine through a surgically placed tube. It is advised for patients that experience significant fluctuations when using levodopa orally. Using the infusion pump makes it possible to provide a continuous administration of medication at a steady or modifiable rate.

Recent years brought new approaches and medication for PD patients, these are aimed at improving their quality of life and decreasing the severity of symptoms. Even though levodopa is still the most commonly used medicine, many patients take not one, but many different medicines to improve their condition. The treatment process needs to be highly individualized, acknowledging the distinct symptom patterns and disease progression in each patient. Continuous monitoring and adjustments in therapy are essential, demanding close collaboration among patients, caregivers, and healthcare professionals.

While current treatments for PD provide a variety of pharmacological and surgical treatments, they are not free from side effects and sometimes are difficult to implement. With the expansion of computer science and artificial intelligence, these new technologies have the potential to improve the treatment by providing more personalized approaches, improving diagnostics and state evaluation precision and optimizing medication regiments. The integration of advanced data analysis and predictive modeling could lead to significant improvements in the management of PD, promising a future where the impact of this condition is greatly minimized, and patient quality of life is substantially improved.

1.4. Problem definition

The management of PD presents a significant challenge that requires precision, adaptability, and good understanding of each patient's experience with the disease. The traditional approach to PD treatment is usually reactive rather than proactive [21]. Adjustments to medicine intake schedules follow observed changes in the patient condition and their insufficient response to current treatment. This sometimes requires multiple visits. The neurologists follow the trial-and-error approach to find the most suitable medicine intake schedule for the patient based on the reports regarding their condition and therapy efficiency as well as results of state evaluation scales. This is ineffective, time-consuming and takes into account subjective measures. Therefore, the idea of improving this process using computer science technologies is explored.

This dissertation addresses this problem and introduces a method designed to create optimal medicine intake schedules, currently focused on levodopa – main medication in PD. However, it can be easily extended to account for more sophisticated treatment regimes. The main goal of the method is to suggest medicine doses and their intake times that keep the patient in an optimal state throughout the day. To calibrate the method to individual patient needs and their disease profile, examinations performed by the patient using mobile phones and sensor devices as well as demographic and clinical data are utilized. These examinations allow to capture the responses to medication, and this can be used to find the doses that put the patient in the optimal state.

Solving this research problem requires discovering the answers to three research questions. These questions also form the steps required to implement the method.

The first question is “Can the data collected during examinations with using mobile phones and wearable sensors be used to assess the current state (represented by the severity of symptoms) of a PD patient?”. This is the first step of solving the problem. To explore it, a method is built which, based on registered sensor signals evaluates the current symptom severity for the patient. The dissertation presents a method that uses machine learning algorithms to solve it.

When a model for present state evaluation is available, research can be conducted to answer the second question: “Can the data collected regarding medication and patient profile be used to build a model capable of predicting the individual patient response to medicine doses?”. To find the answer to this question, prediction models are explored that

based on the recorded medicine intakes and previous responses to medication, should be capable of inferring individual responses to other doses. To achieve it, machine learning models, tasked with predicting the patient states after medicine intakes are used.

Once the model for predicting future patient states is available the last question can be investigated “Can the medicine response model be used to find optimal medicine intake schedules for PD patients?”. Solving this problem has been explored using optimization methods. These test potential medicine schedules, in order to find the one that provides the best results for specific patients.

In conclusion, the method proposed in this dissertation represents a step towards the future of PD treatment. It extends the traditional focus on symptomatic management to include an emphasis on improving patient quality of life through more personalized, data-informed treatment strategies. This method makes a transition to proactive healthcare, integrating insights from computer science with clinical expertise.

1.5. Structure of dissertation

The thesis consists of eight chapters, that together form the description and testing of the method to create individualized medicine intake schedules for PD patients.

The first chapter, the introduction, has an informative nature. Its goal is to provide the medical background of PD, its symptoms, and the characteristics of the therapy. The focus is on presenting the challenges in the treatment and is necessary for understanding the need for the presented method and forming the research problem.

The second chapter focuses on the analysis of currently available research on the topic. It includes descriptions of research regarding patient state assessment using sensors and machine learning. The research addresses both evaluating specific symptoms and providing an overall representation of the patient state. The chapter concentrates also on application of computer science in the therapy of PD, discussing models for predicting response to medication and automated mechanisms for medication dosing.

The third chapter describes three datasets, which are later used in the following chapters to test and validate proposed methods. The datasets contain sensor, clinical and demographic data for the patients. Each of the datasets covers a different scope making them useful in different parts of the thesis.

The fourth chapter describes the first part of the method - the construction and testing of machine learning (ML) models capable of predicting the patient's current state. The models are built to predict individual symptom severities and the overall state related to PD. Both, simple ML algorithms and deep learning methods are used.

The fifth chapter focuses on building models capable of predicting future states under the influence of medication. These ML models are tasked to predict individual patient responses to medication. This is performed initially on simulated patients and is followed by experiments on real patients.

The sixth chapter presents the use of previously designed models by conventional optimization methods and reinforcement learning (RL) to create medicine intake schedules for PD patients. These results are compared with current methods and the prescriptions by a neurologist.

The seventh chapter contains a description of the tool created to implement the proposed method. It is a system consisting of two software applications. A mobile application created to collect data from patients, capable of leading them through the therapy, and a web application designed for clinicians supervising the treatment.

The remaining parts of the thesis focus on the discussion regarding the complete proposed solution and final conclusions regarding the applied approach.

2. The review of the current state of research regarding the application of computer science methods in the management of Parkinson's disease

Computer science has had an important role in medicine for many years, with applications ranging from the management of patient records to complex diagnostic algorithms [22,23]. With the development of artificial intelligence, mainly machine learning, the possibility to aid medicine has expanded even more allowing to capture the rules that clinicians follow in their practice [24].

The applicability of machine learning and other computer science-related fields is currently being explored for Parkinson's disease. Data analysts and other specialists are working on exploring the possibilities of using collected data to improve the diagnosis, prognosis, and treatment of the disease. However, most research has been done regarding the diagnosis and evaluation of disease symptoms.

This chapter concentrates on the review of the current state of research regarding the application of computer science methods, most importantly machine learning, in the management of Parkinson's disease. The focus is on exploring how it is used to enhance diagnosis, monitor disease progression and symptoms, and optimize therapy. By examining these different approaches, from rule-based to complex methods using deep learning and differential equations, the review aims to present the diverse and innovative ways these technologies are applied. Understanding these applications does not only clarify the technological capabilities and challenges faced but also identify the gaps in current research that this dissertation aims to address.

2.1. Machine learning for patient state assessment

Aiding the diagnosis of Parkinson's disease using machine learning typically involves analyzing measurements or signal data collected from patients. This data, often comprising motion signals from accelerometers, voice recordings or other sensor-based information, captures single or multiple symptoms of the disease. Later, the data is processed with ML algorithms to provide the diagnosis. In the processing stage, important features such as tremor frequency, gait patterns, or vocal changes are extracted and fed into machine learning models. Alternatively, deep learning approaches can be used, which directly analyze the raw signal, automatically identifying relevant features for prediction.

This methodology is similarly employed in predicting symptom severity, disease progression, or response to medication evaluation, based on observed symptoms. While the fundamental approach remains consistent across these applications, the specific modeling techniques and outcome predictions may vary. This could range from diagnosing the disease to evaluating symptom severity and predicting disease progression. These approaches will be discussed in following paragraphs, in more detail, encompassing different sensors and outcomes for diagnosis, symptom severity assessment, and disease progression monitoring.

The first approach uses inertial sensors, such as accelerometers and gyroscopes placed on patient limbs to infer the evaluation based on motor symptoms of the limbs like tremor or bradykinesia, muscle stiffness or more complex symptoms like freezing of gait, falling down. These studies usually register the signals while the patient is performing specific tasks e.g., walking, extending their arms. One study [25], considering data from 15 patients and 15 healthy controls (healthy individuals participating in the study to provide a control group), focuses on building two ML models. These, based on accelerometer, gyroscope, and magnetometer signals, collected while the patients were performing 8 tasks, are able to successfully distinguish between patients and healthy controls. The approach uses signal processing methods to extract features and feed them into the models. Patel et al. used a Support Vector Machine (SVM) to build a model that estimates the severity of PD symptoms based on data from wearable accelerometers – 8 placed on different body parts. This approach required the patients to complete six specific tasks [26]. Similar data was collected in the MJFF Levodopa Response Study [27] which was used in a DREAM Challenge [28] to build prediction models of bradykinesia, dyskinesia and tremor severities based on accelerometer readings from Shimmer3 sensors. Both of these studies resulted in well-fitted models for individual symptom predictions. Related research has been repeated in a few other studies [29,30], with different scopes of tasks completed by patients, other ML methods, and different signal processing algorithms.

Another task the researchers have been focusing on is predicting not individual symptoms severities, but rather their overall condition. One of the studies concentrated on detecting ON/OFF states based on inertial sensor data using ML [31]. Their approach, unlike previously mentioned, did not require the patient to complete specific tasks, data was collected in the background. In Thomas et al. the Treatment-Response Index (TRIS)

was predicted based on the sensor data captured during the pronation-supination task. This index represents the patient response to medication, with values ranging from -4 (severe symptoms) to +4 (sever dyskinesia) [32]. Sotirakis et al. (2023) focused on assessing the progression of the disease by predicting the scores for part III of the MDS-UPDRS scale [33]. However, this still required the patients to perform predefined tasks.

Some of the latest research explored taking a different approach than instrumental tasks performed by the patient. Instead, the researchers suggested collecting the data in the background by wearables, while the patient is performing daily activities. This is less troublesome for the patient. Unfortunately, in most cases, the quality of the data decreases, because of the noise of daily tasks. Some approaches included algorithms that allowed the calculation of symptom severity scores [34], while others used machine and deep learning methods to detect tremor [35,36], falling [37] and diagnose the disease.

The patient's drawing and writing capabilities have also been recognized as valuable sources for the evaluation and detection of PD. Capturing the handwriting gives the ability to detect, not only micrographia, but main motor symptoms such as tremor and bradykinesia too. Pereira et al. focused on using spiral images drawn on paper to perform the evaluation [38]. The diagnosis was performed using simple ML methods applied to features extracted from the drawings. Using this approach, while convenient for patients, made it difficult to capture the dynamics of hand movements.

Further research was set up to capture more data, Rios-Urrego et al. used a tablet with a stylus that allowed capturing six signals, not only the horizontal and vertical position, but also the azimuth angle, distance from tablet and altitude angle [39]. Similarly, in this study features from the signals were extracted to capture significant characteristics for PD diagnosis, these were then fed into ML models to train and provide a diagnosis in the future. The researchers examined signals for drawing the spiral and writing a sentence.

While the main task across publications is spiral drawing, due to its versatility, research regarding writing and signatures has also been conducted. A study conducted by Drotár et al. [40] led to the publication of a public dataset - Parkinson's Disease Handwriting Dataset (PaHaW). It contains examinations from patients and healthy controls consisting of spiral drawings, and writing of single letters, syllables, words, and sentences. They collected the data using a tablet and a stylus with a 200 Hz frequency for

the following: x and y coordinates, timestamp, button status (on or off-screen), pressure, tilt, and elevation. This has been used in multiple studies that extracted different features for classification. Most focused on using calculated features regarding the dynamics of movement in different directions and pressure [40] and fractional derivatives [41]. Some approaches also used deep learning methods to train the model [42]. Compared to sensor data from wearables, handwriting data coming from various datasets exhibits less variability. This is because the sensors in different studies may be placed on different body parts, in varying orientations and use different sampling frequencies, complicating the use of the data from multiple studies to train a single model.

Recent years brought research regarding PD diagnosis through vocal analysis. Using machine learning and deep learning methods gives the chance for early detection and monitoring of the disease progression. To perform the diagnosis different types of voice recordings are used including sustained vowel phonations, repeating syllables, reading text, and free speech. Little et al. [43] used a dataset of 195 recordings of vowel phonations. They extracted features from the recording, including a newly introduced feature – pitch period entropy. These features were then fed into ML models, achieving classification accuracy above 90% for distinguishing PD patients from healthy controls.

The study by Bayestehtashk et al. [44] calculates voice features based on three tasks: sustained phonation, repeating “pa-ta-ka” syllables, and reading text. In this case, the researchers decided to build a model to predict patients’ UPDRS scale scores, they used different feature extraction methods and various ML models to complete this task. The model using the text reading recording resulted with the lowest error value. Research conducted by Rueda and Krishnan [45] focused on analysis of voice recordings of sustained “a” vowel, singing the “a” vowel up and down the scale and a one-minute monologue. They used mel-frequency cepstrum coefficients and intrinistic mode functions, common approaches to extract features in voice analysis. The research did not include any ML methods but presented the analysis of these features’ values. The UPDRS score was used as a predicted value in a study conducted by Frid. et al. [46]. They used features extracted from recordings of phonetically and phonemically balanced text readings to train a Support Vector Machine to predict UPDRS scores (0-5 with 0.5 resolution) resulting in an average accuracy of 81.8%.

For processing the speech of PD patients deep learning approaches have been tested too. Wodzinski et al. [47] utilized a modified version of the ResNet network consisting of four convolutional layers to process the spectrograms of filtered recordings of sustained vowel phonations. The dataset consisted of 50 patients, and 50 healthy controls, with three recording of each patient. The training process led to an accuracy of 91.7% in the 10-fold cross-validation approach. Rehman et al. for the detection of PD used recurrent neural networks [48]. Long short-term memory (LSTM), and Gated recurrent unit (GRU) architectures were used to build four different models. Based on 195 recordings registered from 31 patients the models were trained. The authors used both the initial dataset, and a new balanced one created using sampling techniques. Their best model – a combination of the LSTM and GRU resulted in a perfect classification (100% accuracy). In 2018, Orozco-Arroyave et al. published an open-source software – NeuroSpeech, designed to help researchers with the analysis of PD patient speech [49]. This software allows users to input voice recordings, process them and extract features based on predefined sets in order to perform the analysis of phonation, articulation, prosody, diadochokinetic and intelligibility. This software is capable of generating reports, which might be helpful for neurologists and speech therapists.

Aside from the detection of PD based on IMU sensor data, writing, and speech, other approaches have been investigated to perform a non-invasive diagnosis or state evaluation. One innovative method, proposed by Szymański et al. involves using eye-tracking devices, they have used two systems for recording eye movements [50]. The dataset consisted of recording from 8 PD patients in different sessions (different patient ON/OFF states a different treatment applied). Using the eye-tracking and patient clinical data they built ML models to predict the session type of the recording and the total UPDRS score of the patient. The acquired accuracy of UPDRS score prediction reached 85.7% showing promise in this approach to assessing patient condition.

Voice-related EEG (electroencephalogram) signal analysis is another approach for PD diagnosis. Recent studies have developed models that analyze EEG signals related to voice activities to distinguish between PD patients and healthy controls. One such approach [51] uses graph learning models on voice-related EEG signals, showing superior performance in accurately classifying PD. These models, including graph signal processing-graph convolutional networks (GSP-GCNs), have demonstrated high

accuracy in diagnosing PD, highlighting the potential of EEG signal analysis as a powerful tool for early PD detection.

Another notable method used for assessing PD patient condition is finger tapping, a method that leverages the rhythmic movement of fingers as a diagnostic tool. Studies have explored how finger tapping rates, regularity, and force differ between individuals with Parkinson's Disease (PD) and healthy controls. One study focuses on results collected with a smartphone application, where users are asked to alternately tap two rectangles on a screen with an index finger [52]. Based on the recorded results, a set of features was selected allowing the discrimination between patients and healthy controls with the area under the receiver operating characteristic curve of 0.92. Researchers have also built models capable of analyzing video recordings of the finger-tapping activity. Khan et al. [53] focused on distinguishing PD patients from healthy controls and predicting the severity of symptoms on a scale from 0 (normal) to 3 (severe). This was achieved by employing a newly constructed computer vision algorithm, which used face detection to track the movement of index fingers, thereby enabling the extraction of important features. These were fed into an SVM model and resulted in accuracies of 88% for severity classification and 95% for distinguishing between patients and healthy controls.

Typing patterns of PD patients have also been an area of investigation to provide a non-invasive approach to disease detection or symptom severity evaluation. Researchers have analyzed data regarding typing on a traditional keyboard on a computer and other devices such as mobile phones. This procedure can be also applied in passive monitoring since it does not necessarily require the patients to complete any additional tasks. Warwick used the data collected by a PC application that captured keystrokes resulting in measurements collected from 103 participants (32 PD patients). Using ML models, it was possible to get a perfect discrimination on the test set and an accuracy of 94% on another, publicly available dataset. These results open new possibilities for diagnosis, which could be easily used as low-cost screening tests for PD [54]. Typing habits have also found an application in assessing the response to PD medication. In a study by Matarazzo et al. [55] data was being collected for 6 months from 31 PD patients and 30 healthy controls. The response to medication was observed as a change in part III of the UPDRS scale. Using a recurrent neural network, they built models to predict the change in the UPDRS score and to classify if patients have improved or not. While

the first task resulted in a correlation coefficient of only 0.33 between real and predicted values, the balanced accuracy for the classification task was equal to 76.5%.

The presented ideas for the assessment of PD patients' conditions usually utilized only one source of data such as sensor readings or voice recordings. Recent years brought the development of complex solutions that are capable of handling data from multiple sources, which usually leads to precise and more meaningful results, which give a more objective evaluation. These usually require additional attention from the patient as more examinations are performed. Some of these approaches are based on specific software, a computer, or a mobile phone application providing feasibility of use and accessibility, making them perfect tools for monitoring of changing conditions.

In a study by Aghanavesi et al., a Treatment Response Index from Multiple Sensors (TRIMS) has been defined [56]. It represented a response to medication on a scale from -3 (severe symptoms) to 3 (severe dyskinesia) based on signals registered from multiple sensors and completed tasks. IMU sensors were placed on limbs while the patient was performing the leg agility, pronation-supination, and walking tests. After extracting features from all three tests, 178 features were available for 204 observations. Employing ML techniques allowed them to build a model that provided outputs that had a 0.93 correlation with neurologist’s evaluations.

In March 2015, a mPower study was launched [57], focusing on collecting data from volunteers via an iOS mobile application. The participation was open to PD patients and other volunteers (control group). During the study, participants completed standard surveys used for assessment of the disease and completed a set of exercises using the application (maximum three times a day). These included: a short memory exercise, alternate tapping of two points with two fingers, sustained phonation of the “a” vowel, and a walking task. These activities have also been investigated in previously mentioned studies in a controlled environment. Schwab and Karlen have used the data collected in the mPower dataset to build ML models to distinguish PD patients from healthy controls [58]. They used two approaches, extracting features and applying an ML model and a deep learning model containing convolutional layers. The models were built for each task separately. However, the best results were achieved when all of the tasks were considered together.

Zhan et al. utilized data collected using an Android mobile application in order to predict the severity of the disease [59]. Their study included a definition of the mobile Parkinson disease score (mPDS) which is expected to represent the severity of the disease captured by predefined activities on a scale from 0 to 100. Their approach used weak supervision – with an assumption, that symptom severity was higher before taking the next medication dose, than an hour after it. Correctly ranking these pairs allowed for the definition of the score. The state evaluation was performed based on five aspects of the disease: gait, balance, finger tapping, speech, and reaction time, each task has a different weight assigned in the final mPDS score. Their attempts at defining the scale led to a high correlation (>0.8) of mPDS with both the III part of MDS-UPDRS scale and the total MDS-UPDRS score.

Another interesting study – CIS-PD, sponsored by Michael J. Fox Foundation used iPhone devices along with smartwatches to collect data [60]. During the study, accelerometer data was collected continuously using a smartwatch, 12 hours per day, for 25 days per month. Every day the participants were also asked to provide at least three evaluations of their symptoms – constipation, balance, bradykinesia, speech impairment, dyskinesia, tremor, and gait with five severity scores. Their medication was also being tracked using the application. This dataset has been used in the BEAT-PD challenge, where the participants were tasked with building ML models for the prediction of symptom severities experienced by the patients, based on their evaluations.

Every year, innovative approaches are explored to improve the assessment and monitoring of PD, leveraging advancements in technology and data analysis. While this discussion has highlighted a range of methods—from IMU sensor data, writing, and speech analysis to novel techniques combining data from many sources, like finger tapping and keyboard typing—it represents only a fraction of the ongoing research. The field continues to evolve, with each new study offering the potential for more accurate diagnostics, better patient care, and insights into PD's complex nature. This underscores the importance of continuous exploration and adaptation in the quest to fully understand and treat PD.

The conducted review will be utilized to develop custom methods for patient state identification using measurements performed with mobile and wearable devices. These methods should consider the precision provided by specific sensors and devices, as well

as the possibility of data fusion of individual measurements of various symptoms ensuring the individualization of current state assessment for every patient.

2.2. Computer science for improved therapy

Shifting from diagnosis and evaluation of PD symptoms, this part of the review highlights the role of computer science in enhancing therapeutic interventions for PD. It focuses on innovative methods such as personalized exercise programs facilitated by software applications, which are instrumental in improving the management and quality of life for individuals affected by PD. These advancements demonstrate the significant potential of computer science to offer novel solutions for therapy in neurodegenerative diseases.

In 2005, Chan et al. published their results of a 4-year cohort study on PD [61]. The purpose of the study was to describe the pharmacokinetics of levodopa in the analyzed population. They investigated 20 patients and checked their reaction to levodopa infusion administered with oral carbidopa dose. As a result, a pharmacokinetic model was built, which describes how the body affects medication after administration. The model consists of two compartments and is personalized through patient-specific parameters. For the population, estimates of parameters are provided along with variabilities values between subjects, occasions, and within a trial. This model has been further developed by Westin et al. [62]. It has been applied for levodopa duodenal infusion and a pharmacodynamics part was added to the model. This new pharmacokinetic-pharmacodynamic (PK/PD) model consists of five equations. They allow the calculation of concentration of levodopa in different compartments using patient-specific parameters. The pharmacodynamic part of the model brings the possibility of predicting the effect that levodopa medication would have on a patient at specific times using the previously mentioned TRS scale. In their study, they compared real levodopa concentrations and TRS scores with the results suggested by patient-specific PK/PD models.

The definition of the model made it possible to predict patient responses to medication, helping neurologists in treatment adjusting. Thomas et al. [63] used them to create an algorithm for suggesting the infusion rate for Duodopa. To evaluate the model, they generated simulated patients (through PK/PD model parameters) resulting in 23 patients. For each of them, optimization was performed to find out which infusion rate keeps the patients' TRS scores at 0 – optimal state. It resulted in a 0.88 correlation

between the optimal infusion rate and the suggested infusion rate by neurologist. The promising results of this study were further investigated, which led to the formulation of an algorithm for supporting neurologists in oral levodopa dosing [64]. During the study, the patients were administered a levodopa/carbidopa dose, and with defined intervals, their state was evaluated by clinicians as they completed a pronation-supination task with Shimmer3 sensors on their wrists. Based on patient's states captured using the TRIS index and the medicine dose, the PK/PD model parameters were estimated to fit the levodopa response curve. Fitted, modified PK/PD models could then be used to predict the response to oral doses of levodopa and carbidopa pills. Based on the range of experienced states by the patients, the objective function has been defined for the morning and maintenance dose effects. Employing the exhaustive search, an optimal size for the morning and maintenance dose were found, which were highly correlated (at least 0.8) with the neurologist's suggestions.

An interesting approach has been presented by Watts et al. [65] for creating individualized treatment. Based on the literature review, they constructed possible patient profiles and generated their responses to medication as bradykinesia and dyskinesia severities that could be read from sensors. These were generated using sigmoid functions with uniformly generated noise. Constructed models were used as an environment for the reinforcement learning agent. Using asynchronous advantage actor-critic (A3C) it was possible to train the agent to suggest doses of medication that minimized the negative effects of PD and overdosing. The reward was based on predicted bradykinesia and dyskinesia scores.

Kinesia 360, a wearable biosensor for continuous monitoring was tested in a 12-week study by Isaacson et al. [66]. The goal was to check if it could help improve motor symptoms management for patients using rotigotine – PD medicine, a dopamine agonist. The patients were split into an experimental group (EG) and a control group. For EG patients, the clinicians used the collected data to adjust medicine dosing. After modifying the schedule, patients were again evaluated using the UPDRS scale and it has significantly improved their overall condition.

As highlighted in this literature review, advancements in technology have led to diverse approaches for assessing Parkinson's Disease (PD) patients, ranging from wearable sensors to machine learning models. These methods offer valuable insights into

disease progression, medication management, and therapy optimization. Despite these advancements, there is still a gap in the field: the need for a solution that can improve the therapy of many medications while considering patient-specific features. Current approaches often rely on generalized dosing schedules or subjective assessments of symptom severity, which may not adequately account for individual variations in disease presentation and treatment response. Therefore, there is a pressing need for personalized therapeutic interventions that can adapt in real-time based on patient-specific data, ensuring optimal outcomes and minimizing adverse effects. This highlights the demand for advanced models that take into account multiple aspects of the disease, which differ between patients to support clinical decision-making. Creating a solution addressing these issues has the potential to significantly improve the quality of life of PD patients and disease management.

2.3. Hypothesis and method overview

Despite significant advancements in the treatment of PD, current treatment strategies largely rely on generalized dosing schedules that often fail to accommodate the individual needs of patients. This general approach can lead to inadequate symptom control and a bigger chance of medication-related side effects, highlighting a critical gap in personalized PD management. Current research focuses on the potential of machine learning and sensor technologies in enhancing disease monitoring, yet there remains a substantial unexplored territory in their application to dynamically optimize medication dosing tailored to individual patients.

The dissertation presents a method to create individual medicine intake schedules, which was initially described by Gutowski and Chmielewski [67]. Since the publication, the method has been further developed based on experiments and achieved results [68,69]. Figure 4 presents the outline of the method, the steps needed to prepare and apply the method, as well as its main components.

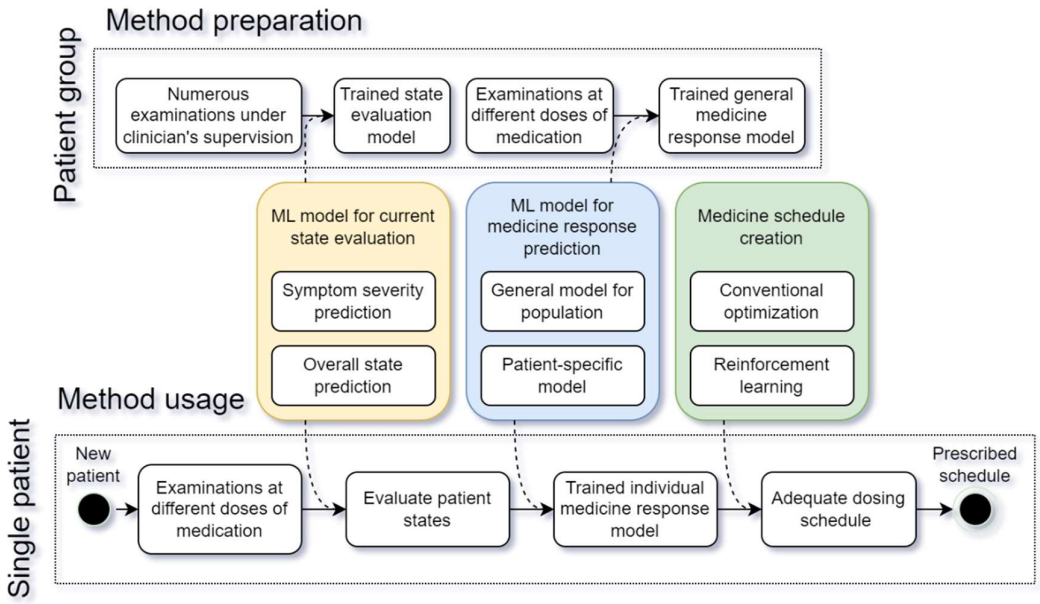

Figure 4 Chart presenting the outline of the method, steps needed to prepare and use it

The system consists of three main components. The first component is the ML model, which is used for state evaluation based on sensor signals. This model is trained based on data from numerous patients and preferably state evaluations provided by different clinicians, in order to provide maximum objectivity. The second component is also an ML model, trained on patient state data along with medication information. It is capable of predicting future patient states after a medicine dose is taken. The third component uses the second model to test different medicine dosing schedules, which is performed by optimization algorithms and reinforcement learning agents, that try to find optimal dosing regimens for specific patients.

To apply the method for a new patient a series of examinations should be performed with different doses of medication. These examinations are then evaluated using the state evaluation ML model. Resulting states are then fed together with dosing data into the medicine response prediction model. Retraining the model makes it possible to capture the individual character of the patient's reaction to the medication. This model can be then employed in optimization algorithms and the reinforcement learning environment to find medicine doses that keep the patient in the desired state throughout the whole day.

The described and developed method is integrated into a system that includes both a mobile app and a web-based platform, enhancing real-time data collection and patient

monitoring. The mobile application allows patients to record symptoms and medication effects daily using sensors built into smartphones and dedicated wearables. Given the limited precision of these measurement devices, careful selection of data processing methods is crucial for training the ML models and accurately identifying patient states. The web platform offers clinicians a dashboard to view comprehensive patient data, facilitating better treatment oversight and adjustments. This integration supports continuous patient care and enables dynamic treatment personalization.

The main hypothesis is that the application of the described method, which utilizes machine learning algorithms and sensor data within an integrated system comprising both mobile and web platforms, can create personalized medicine intake schedules, which can improve the management of Parkinson's disease symptoms. This approach is expected to enhance the precision of dosing adjustments, reduce side effects, and improve overall patient quality of life by maintaining an optimal therapeutic state throughout the day.

3. Data collection process

This study showcases research performed using mainly 3 datasets, two of which were previously collected by other institutions and were reused in this study for further research, while the third dataset was built specifically for this study. These datasets include:

- Michael J. Fox Foundation (MJFF) Levodopa Response study dataset [27], referred to as *MJFF dataset,*
- dataset from a clinical study that was performed at Sahlgrenska University Hospital in Gothenburg, Sweden, between August 2016 and February 2017 [64], referred to as *Swedish dataset,*
- dataset collected at the Medical University in Warsaw specifically for this study, referred to as *MUW dataset*.

These datasets were used for:

- building ML models to evaluate patient's current state,
- building ML models for predicting the response to medication.

3.1. MJFF dataset

The dataset was created as a part of the Levodopa Response study, which was funded by MJFF. The main purpose was to study the feasibility of monitoring PD symptoms and motor fluctuations at home. It was performed at two locations in New York and Boston and depending on the location the patients were fitted with 3 (New York) or 8 (Boston) sensors. All of them wore a GeneActive watch on their more affected wrist and the Pebble watch on the other side, an additional phone – Samsung Galaxy Mini was attached to the front of the waist to track lower limbs symptoms. The additional five sensors for Boston patients were Shimmer3 sensors placed on each limb and the last one on lower back. The placement of sensors is presented in Figure 5.

Figure 5 Placement of sensors in the MJFF Levodopa Response Study

Participants underwent a 4-day study comprising of two visits, scheduled on the first and the fourth day. On the first day, they were expected to arrive in the ON state. Initial data regarding their state was collected:

- demographic data,
- medical history,
- MDS-UPDRS scale results,
- metadata of used sensors.

The demographic data included their gender, birth date, height, weight, length measurements of limbs, and dominant hand. Medical history consisted of diagnosis date, the most affected side by the disease, presence of specific symptoms, time of levodopa dose and regular medication information. The MDS-UPDRS score was given separately for every part of the scale, along with the Hoehn & Yahr scale result. For every patient, the placement of sensors was recorded along with their identification numbers. The initial evaluation was followed by a completion of a set of exercises called “activities of daily living”, presented in Table 1. This set of tasks was completed by the patient 6 to 8 times during the visit, approximately every 30 minutes, under the supervision of the clinician. Every time the patient was completing a task, the clinician evaluated three main symptoms: tremor, bradykinesia, and dyskinesia. These were evaluated for upper limbs separately and together for both lower limbs. In the case of tremor, the severity was evaluated on a scale of 0 to 4. Bradykinesia and dyskinesia were assessed for presence/absence only. For some exercise only a subset of evaluations was performed, their scope is presented in Table 1. Additionally, for patients fitted with Shimmer3

sensors, the scores for lower limbs were provided separately, and for the dyskinesia and bradykinesia severity scores from 0 to 4 were provided as well. Afterwards, the patients were discharged home, with the sensors. During the following 2 days they were performing daily activities and recording their medication intakes. On the fourth day, they returned to the laboratory in the OFF state, and their state was evaluated the same way as during the first day. However, after the first set of tasks were performed, they were given levodopa medication and repeated the tasks 5 to 7 times. At the end, they returned the sensors and completed a feedback survey about the study.

Table 1 The list of tasks performed in the MJFF Levodopa Response study with the scope of clinician's evaluations

Task	Abbr.	Tremor	Bradykinesia	Dyskinesia	Samples
Standing	Stndg	X	-	X	1497
Walking straight	Wlkgs	X	X	X	1490
Walking while counting	Wlkgc	X	X	X	1495
Going up the stairs	Strsu	X	X	X	238
Going down the stairs	Strsd	X	X	X	239
Walking through a narrow passage	Wlkgp	X	X	X	1493
Finger to nose – right arm	Ftnr	X	RH	X	2667
Finger to nose – left arm	Ftnl	X	LH	X	2666
Repeated arm movement – right arm	Ramr	X	RH	X	2664
Repeated arm movement – left arm	Raml	X	LH	X	2662
Sit to stand	Ststd	X	X	X	1495
Drawing and writing on a paper	Drawg	X	H	X	1487
Typing on a computer keyboard	Typng	X	H	X	1495
Assembling nuts and bolts	Ntblt	X	H	X	1520
Take a glass of water and drink	Drnkg	X	H	X	1488

Organizing sheets in a folder	Orgpa	X	H	X	1489
Folding a towel	Fldng	X	H	X	1491
Sitting	Sittg	X	-	X	1494

X – evaluated for both upper limbs and lower limbs, H – evaluated for both upper limbs, LH – evaluated for upper left limb, RH – evaluated for upper right limb.

The study included 28 patients of whom 11 were examined in New York and 17 in Boston, 19 males and 9 females.

In 2017, a part of the dataset was used in the Parkinson's Disease Digital Biomarker DREAM Challenge [28]. The participants were tasked with feature extraction and selection. These feature sets were later evaluated by training and testing basic machine learning models (random forests, SVM, k-NN, elastic net, neural nets) which were used to predict the presence/severity of each of the symptoms. In the challenge, the participants could use raw accelerometer signal recorded during tasks execution along with metadata (patient identifier e.g., 3_BOS, site – Boston/NYC, device – Pebble/GENEActiv, device side – Left/Right, visit – 1/2, session – 1-8, task – presented in Table 1). The solutions were evaluated using area under the precision-recall curve (AUC PR) [70]. They were compared with the baseline model which was created using only meta data features.

The models created during the challenge used information about the session number, visit, patient identifier and site. Using this data, makes it nearly impossible to apply the trained model in other conditions, on newly collected measurements, which might be carried out in a different location, performed on a patient that was not within the selected group or carried out outside the study's visit and session regime. Furthermore, the trained models in the challenge could have learned the fact that certain symptom severities were more likely to occur only with certain patients, during certain sessions and visits. Consequently, this aspect makes the results and the model less adaptable to alternative scenarios.

3.2. Swedish dataset

This dataset was collected at the Sahlgrenska University Hospital in Gothenburg, Sweden for the study of creation dosing schedules for oral levodopa in advanced stages of PD [64]. The data was collected between August 2016 and February 2017 and 31

patients were initially recruited, but only 25 were qualified to participate in the study. Characteristics of patients are presented in Table 2.

Table 2 Characteristics of the patient population, represented by medians and interquartile range values (in brackets)

	Sex	**Age**	**BMI**	**Years since disease onset**	**Years since diagnosis**	**Years with motor fluctuations**	**MDS-UPRDRS score**
Patients	15 males, 10 females	68.0 (9.00)	25.2 (4.58)	11.0 (10.0)	9.50 (8.25)	4.00 (4.50)	61.5 (36.5)

The patients that were recruited for the study needed frequent doses of levodopa medicine, with a dosing frequency lower than 4 hours and the study schedule expected the patients to show up for three visits with a two-week break between them. During the first visit the patients had their traditional dosing schedule converted to an equivalent using levodopa-carbidopa microtablets [71]. These allow for the dosing of levodopa with the precision of 5 mg – the capacity of a single pill. The newly prescribed schedules did not use additional PD medication, but only these levodopa-carbidopa tablets were applied. After the visit, the patients were equipped with a Parkinson's KinetiGraph device (PKG) [34]. This device using inertial sensors worn by the patient analyzes the signal and provides the clinician with scores for bradykinesia and dyskinesia changing throughout the day, aiding them in adjusting the treatment. These scores are calculated algorithmically based on accelerometer and gyroscope signals. The device is approved to be used with PD patients and is sometimes useful for clinicians in the treatment phase of PD. The patients were tasked to wear the PKG device for 6 days prior to the second visit while following the updated treatment using microtablets.

During the second visit, a neurologist evaluated the results provided by PKG in a form of reports with charts, generated in PDF format. Based on that, the schedules were updated to respond to patient needs. After the visit, patients were once again asked to wear the device. On the third visit, the adjustment results were evaluated using PKG generated reports.

During the second visit patients also participated in an additional test, to access their response to levodopa medication. All of them were requested to wear the Shimmer3 sensors, equipped with accelerometers and gyroscopes, on both wrists. At specific times they were asked to perform the hand pronation-supination activity for 20 seconds [32]. The tests were conducted 8 to 13 times per patient, with the initial examination starting at time zero and subsequent tests occurring at 20, 40, 60, 80, 110, 140, 170, 200, 230, 260, 290, and 320 minutes thereafter. During the second examination, which occurred 20 minutes after the initial assessment, patients received a dose of levodopa-carbidopa that was 120% of their predetermined morning dosage, prescribed by their neurologist. This approach enabled the capture of how the patient's state changes from the baseline value to the maximum effect of the levodopa dose and back to the baseline, allowing for the recording of patient-specific responses to levodopa. To make it possible, each examination was video recorded, and two neurologists used the TRS scale [32], with values from -3 (severe symptoms) to 3 (severe dyskinesias), to evaluate the patient's condition using consensus. This led to single values (from -3 to 3) regarding the patient's states.

At the time of the first visit patient demographic and clinical data was collected and during each of the visits the patient's state was also evaluate using clinical scales. The most significant scales for this research are presented in Table 3, along with medians and interquartile ranges (IQR) calculated for the total scores across the patient population.

Table 3 Most important scales used in the Swedish study

Scale results		**Median (IQR)**
MDS_UPDRS	Total score and score for each part of MDS-UPDRS scale	61.5 (36.5)
EQ-5D-5L	Scores for answers to the EQ-5D-5L questionnaire	0.681 (0.284)
MADRS	Scores for answers to the MADRS scale	5.5 (5.5)
PDQ-8	Scores for answers of the PDQ-8 questionnaire	5.5 (9.5)
H&Y	Hoehn and Yahr scale result	2.0 (1.0)

As a result of this study, a substantial quantity and scope of data has been collected. The data that was provided by the researchers included mainly:

- patient demographic and clinical data,
- accelerometer and gyroscope signals from the Shimmer3 sensors,
- TRS state evaluations provided by the neurologists,
- scales results for each patient visit,
- dosing schedules suggested by the clinician consisting of morning and maintenance dose sizes and time intervals,
- dose times and sizes administered during second visits.

3.3. MUW dataset

The main dataset used in this research was created as a result of cooperation of two researchers, author of this dissertation, representing the Military University of Technology and Dr. Szlufik representing the Medical University of Warsaw. The dataset consists of data collected by Dr. Szlufik from patients not only with PD, but also other similar neurological disorders, such as essential tremor (ET) [72], multiple system atrophy (MSA), progressive supranuclear palsy (PSP), corticobasal syndrome and dementia with Lewy bodies [73]. The dataset is an outcome of a study to use a mobile application in the differential diagnosis and treatment of tremor in patients with essential tremor, Parkinson's disease, and atypical parkinsonism, which has been approved by the Bioethics Committee of the Medical University of Warsaw. During the study, the data is collected in two modes: under clinicians supervision, and individually by the patient. The process is supported and organized by an information system further described in the chapter "System for tracking PD patients' therapy". The system consists of a mobile application and a web portal. The data is mostly collected using the mobile application.

The mobile application is designed to collect patient demographic and clinical data upon registration of the patient. However, its main goal is to allow to evaluate the patient state and keep track of state changes and medicine schedules and intakes. The state evaluation can be performed in two ways: using the conventional approach, which involves clinical scales completed by clinicians or patients, and a set of exercises completed with a mobile phone and wearables – specifically, Myo armband [74] and Biopoint/BioArmband designed by SiFiLabs [75]. In the clinical supervision mode, at the hospital, the clinician is responsible for completing and video recording the state

evaluation using scales, in case of PD patients the third part of the MDS-UPDRS [76] scale was used to evaluate the patients' state. However, the range of scales used depended on the patient, their disease, condition, and clinician's decision. The scope of the scales used for PD patients is presented in Table 4.

Table 4 Scales implemented in the application to evaluate PD patient's state

Scale	Description
PDQ-8	Assess difficulties across 8 dimensions of daily living.
PDQ-39	
NMSQ	Assess non-motor symptoms of Parkinson's disease.
NMSS	
MDS-UPDRS	Assess various aspects of Parkinson's disease including non-motor and motor experiences of daily living.
UDyS-RS	Assess dyskinesia severity.
Hauser diary	Keep track of dyskinesias and on/off states.
VAS	Visual analogue scale, patients evaluate their pain on a scale.
DDAS-21	Patients complete this scale to evaluate their depression, anxiety, and stress.
SWLS	Used to assess the satisfaction with life.
MoCA	Assess cognitive impairment of the patient.
EQ-5D	Assess patient quality of life.
BDI	Used in depression diagnosis.
BAI	Used to evaluate the level of anxiety.
SRMI	Assess manic symptoms in individuals.
DAS	Assess the activity of rheumatoid arthritis.
TRS	Assess tremor severity.
ESS	Assess the sleepiness of patients.
HADS	Assess anxiety and depression.
QUIPS-RS	Assess Impulsive-Compulsive Disorders.
AES	Assess apathy.
TAS-20	Assess difficulty in identifying and describing emotions.
FAS	Scale used to access patient's fatigue.

Apart from the scales, to evaluate patient's condition, examinations performed with the mobile device are used, after each examination the patients are asked to evaluate their state on a scale from -4 (severe symptoms) to +4 (severe dyskinesia) and if the clinicians are conducting the examination, they are asked to provide additional information including: state evaluation according to doctor (scale from -4 to 4 and -10 to 10), current phase: ON/OFF and the evaluation of individual symptoms – tremor, bradykinesia, rigidity and dyskinesia.

The application allows performing four types of examinations::

- sensor examinations,
- screen exercises,
- writing exercises,
- voice exercises.

The main goal of sensor examinations is to detect motor symptoms of Parkinson's disease, particularly, tremors, bradykinesia, and dyskinesia. At that time, the application allows collecting data with built-in sensors in the mobile device – the accelerometer and gyroscope at a frequency of 50 Hz and sensors from wearable devices – accelerometer, gyroscope, and EMG data from MYO armband, accelerometer, gyroscope, EMG, ECG, PPG, and EDA data from Biopoint. Before starting the examination, the patients are asked to put on the wearable sensors (if applicable) on one or two arms and hold the mobile phone in the primarily examined hand.

The first sensor task is focused on detecting rest tremor. The patient is asked to keep his hands on knees or a horizontal platform, for 30 seconds, while the sensor data is collected. The second task is focused on detecting postural tremor – for 30 seconds the patient extends his arms in front of them, for the sensors to register the data. This is followed by the pronation-supination task performed also for 30 seconds – this task is primarily for detecting bradykinesia. The sensor data can be also collected for another 30 seconds, while the patient is completing further tasks – this is done to assess the kinetic tremor experienced by the patient.

The patients then continue to screen exercises, which require them to touch specific areas of the screen in the shortest time. The idea for this task set was previously described by Gutowski and Chmielewski [77]. During the first task, a square (5 x 5 cm) is displayed, and the user is asked to click it as many times as possible in 20 seconds. This

is followed by a 4x4 grid of squares, with one of them highlighted. After the user clicks it, a new square is highlighted for the user to click, this is performed for 20 seconds. During tasks 3 and 4 the user is displayed the 4x4 grid of squares with randomly assigned numbers from 1 to 16. Their goal is to click them in ascending order. The difference between these two tasks is that during task 3 all the clicked squares remain highlighted, allowing the user to clearly see which are remaining, while in task 4 only the most recently, correctly clicked square is highlighted, these tasks have a maximum time of 90 seconds, but finish earlier if the user completed clicking. During the last task, the user is presented with two squares placed horizontally next to each other. Their goal is to click them alternately with two fingers – index and middle, this task is performed for 30 seconds. These tasks are aimed at detecting bradykinesia and a decline in mental functions. When the examination is performed data is saved upon every touch action – press and release. The following parameters are saved:

- time of action,
- duration of action,
- X and Y offset from the middle of the clicked area (square),
- boolean value representing if the click was correct (correct square was clicked),
- size – representing the pressure applied on the screen during the action.

This data is then saved and can be used for analysis of the patient's current state. The view of these tasks has been presented in Figure 6.

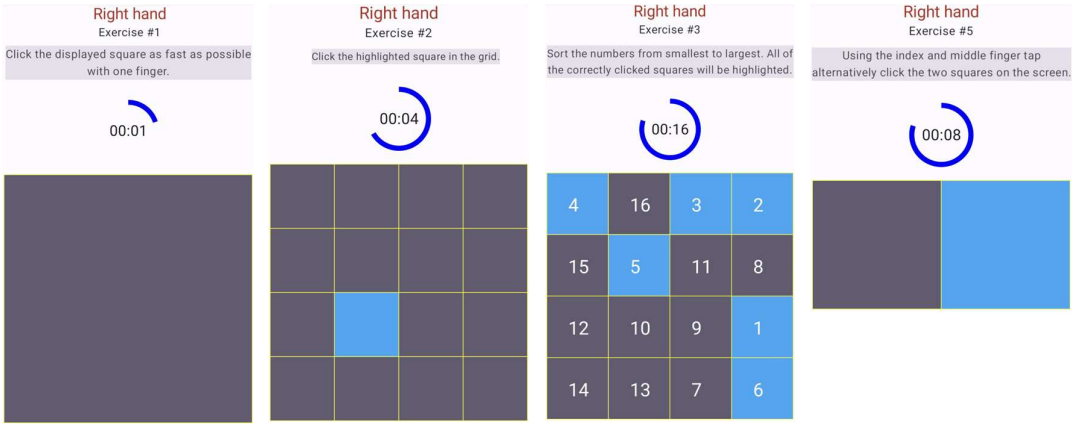

Figure 6 Screen exercises performed by patients for each hand

Writing exercises are aimed at patients who have writing disorders e.g., micrographia. However, they can be also used to detect slowness of movement and kinetic tremor. This task is to be performed with a stylus to closely reflect the conditions experienced while writing on paper. Currently, four tasks have been defined in the application to access the patient's writing and drawing skills, their definition was based on previous research [41,78–80]. The first two require drawing a shape along the line, first a spiral (5x5 cm), then a triangle. The patient is instructed where to start the drawing, and that it should be performed with one movement, without removing the stylus from the screen. The following two tasks focus on writing skills and are performed only for the dominant hand of the patient. First, the patient writes the Polish word "koparka" (excavator) three times. This word was chosen because it does not cause any spelling difficulties, which might influence the speed of writing. Then the patient writes a whole sentence "Jutro będzie ładna pogoda" twice. The meaning of this sentence is "The weather tomorrow will be good". The sentence is simple and is often used in accessing PD patients' handwriting in Poland. When these tasks are performed, the following data is registered at every touch action:

- type of action (press, release, move),
- time of action,
- duration of action,
- orientation of the device,
- X and Y coordinates on the screen,
- size – representing the pressure applied on the screen during the action.

This data is saved for every drawing/writing task, Figure 7 shows the application views of these tasks.

Figure 7 Handwriting exercises performed by patients

To perform the voice examination the built-in microphone is used of the mobile device. These tasks are aimed at capturing the voice disorders the patients are experiencing. Based on the current literature four task have been designed [81–84]. First, the patient sustains the "a" vowel for 10 seconds, then repeats the syllables "pa-ta-ka" for 10 seconds. Task 3 requires the patient to read a following text in Polish "Wyrosły w lesie grzyby duże, ogromne talerze. Grzybobrania nadszedł czas. Hej, pędźmy w las!". This little poem is used by speech therapists to assess speech disorders and is balanced phonetically. It has the following meaning: "There grew big mushrooms in the forest, huge caps. The time for mushroom picking has come. Hey, let's rush into the forest!". The last task includes free speech for at least 10 seconds, the patient is asked to describe his recent meals and physical activity. This may not only provide data for detecting speech disorders but also provide information regarding their medicine metabolism and condition since it has been proven that diet and physical exercise influence the condition of PD patients [85]. Sound recordings in wave format are the result of this task.

The complete list of exercises that were performed by patients is presented in Table 5.

Table 5 The summary of exercises performed using the mobile application

Type	**Number**	**Definition**
Sensor	1	Keeping hands on a horizontal surface
	2	Keeping hands extended in front
	3	Pronation-supination task
Reaction	1	Square clicking

	2	Highlighted square clicking
	3	Sorting numbers (all clicked squares highlighted)
	4	Sorting numbers (only last clicked square highlighted)
	5	Alternating finger tapping
Handwriting	1	Drawing a spiral
	2	Drawing a triangle
	3	Writing a single word three times
	4	Writing a sentence two times
Voice	1	Sustaining the “a” vowel
	2	Repeating “pa-ta-ka”
	3	Reading a text
	4	Free speech

After completion of every examination, the patient or clinician is asked to answer a few questions regarding the condition during the examination. The form for the patient includes an individual subjective state evaluation on a scale from -4 (severe symptoms) to +4 (severe dyskinesias) with 0 being the optimal state and a place for additional comments regarding the condition. The form for the clinician additionally requires providing the following data based on the clinician’s knowledge:

- state evaluation on a scale from -4 to +4,
- state evaluation on a scale from -10 (severe symptoms) to +10 (severe dyskinesia),
- individual symptom evaluation on a scale from 0 to 4 for bradykinesia, tremor, dyskinesia, and stiffness,
- current phase of the patient: ON/OFF,
- clinician initials.

In the study, for some patients, apart from the examination and scale data, information regarding taken medicine doses was collected. In that case the name, dose size and intake times were collected. Using the web application clinicians provided data regarding other tests that were performed on the patients such as blood tests, posturography etc. However, this was not performed for all patients.

This study had no funding and therefore it was difficult to organize the examinations and achieve the completeness and fillability of data at a commercial level. This is the main reason for missing data in the dataset. A neurologist Stanisław Szlufik from the Mazovian Bródno Hospital in Warsaw was responsible for providing state evaluations, which were treated as the ground truth for experiments described in the thesis. At the time of data analysis, this dataset contained accounts of 352 patients with PD, resulting in 739 examinations. However, not all the examinations included all of the exercises because the scope of the scales and examinations was expanding as the data was collected, to capture the scope and magnitude of PD symptoms and for each examination the clinician had the ability to restrict the scope of exercises. The characteristics of the dataset are presented in Table 6.

Table 6 Characteristics of the MUW dataset

Characteristic	**Value**
Total number of patients	241
Age*	62.0 (11.1)
Years since diagnosis*	10.5 (6.10)
Patient sex	98 female, 143 male
Examination count	739
Examinations per patient*	3.07 (2.77)
States according to clinician*	-1.64 (1.38)
States according to patient*	-1.66 (1.42)
Examinations with state assessment according to doctor	700
Examinations with symptom assessment	356

* - represented by mean and standard deviation

4. Patient state evaluation

The evaluation of PD patient's condition is usually performed by a clinician using scales that result in a score giving an overview of the patient's state. The evaluation of the patient state with the scales can be time-consuming, as it requires the patient to perform a series of exercises. Currently, the most common scales in PD are UPDRS [9] and its revised version MDS-UPDRS [76] developed by the Movement Disorder Society in 2008. The MDS-UPDRS consists of 4 parts, each of them aimed at capturing other aspects of the disease. The first part focuses on non-motor aspects of experiences of daily living, which is completed by both the clinician and the patient. Part two, which is also completed by the patient, focuses on motor aspects of experiences of daily living. In the evaluation of patient's current state, the most significant is the third part, because it consists of questions regarding the currently experienced motor symptoms. The last part focuses on dyskinesias and their presence, severity and caused discomfort. Most of the questions require the evaluation of specific symptoms on a scale from 0 to 4. For each option, a description is provided.

Other commonly used scales, while being usually faster to complete, such as PDQ-8, PDQ-39, NMSS etc., focus mostly on the progression of the disease and do not capture the changing state during the day. Another solution used to track patient state is Patient Hauser Diaries [86]. For every hour during the day, the patient is expected to choose one of the following options: sleep, OFF, ON, or dyskinesias. Keeping track of patient's state this way is a good method to evaluate the current therapy and a base for applying some changes in it, based on patient responses.

These described approaches for state assessment of PD patients are widely used. However, they usually require the presence of an experienced clinician, capable of evaluating specific symptoms and the overall state of the patient. In these scales, the assessment is performed visually by a clinician. This makes them subjective and less precise, due to differences between different clinicians and their capabilities. Even though, the instructions and the descriptions of all the questions are usually very precise.

This chapter focuses on building a method for objective evaluations, of individual symptoms and overall state with regards to the disease. Using data collected from wearable sensors and other devices, ML models are trained to provide more objective values, representing patient condition. This step is performed using both the MJFF and

WUM datasets, which contain sensor signals and clinician-provided symptom evaluations. Utilizing these datasets allows for examining the effectiveness of both classical and deep ML methods in patient state identification, given access to different scopes of sensor data. Based on the results of this research, guidelines will be defined for selecting the appropriate methods for patient state identification within the proposed system.

For the MJFF dataset, both deep learning and classical ML methods are examined. The considered problems include binary detection of symptoms and symptom intensity evaluation, which is approached using classification methods to assign severities into five levels and using regression with a continuous set of values.

For the MUW dataset, the focus is on classical ML methods. The problems considered include the evaluation of symptom intensity using regression with a continuous set of values and the overall patient state assessment. The overall state is evaluated according to both the clinician and the patient, using regression with a continuous set of values for both perspectives.

4.1. MJFF dataset

4.1.1. Severity classification from smartwatch data

Before having collected enough data in the MUW study, to be able to create a valuable model for prediction of symptoms severities, a number of experiments have been conducted using publicly available datasets regarding PD. The purpose of this was to get to know the characteristics of this type of data and the build models that might be later used in transfer learning [87] approaches. Selected parts of this section have been previously presented at the European Symposium on Artificial Neural Networks [68].

In this section the use of MJFF dataset is described. It was partially used in the DREAM challenge previously discussed in Data collection process

chapter (p. 27). In the challenge, the participants received only data from smartwatches worn on upper limbs, which was only a fraction of the whole dataset. They were tasked with extracting features that could be used by ML algorithms to detect bradykinesia or dyskinesia in the limb or assess the severity of tremor on a scale of 0 to 4. In the challenge the participants were allowed to use information about the session number, visit, patient identifier and site. Utilizing them in the model limits its applicability

to new measurements from different locations or patients not in the original group, or outside the study's sessions. Additionally, the models trained in the challenge might have recognized that specific symptom severities tended to occur with certain patients, during particular sessions and visits. To be able to predict the severity of tremor and the presence of bradykinesia and dyskinesia it is necessary to build a model that does not use that meta data as input.

This research goes further than the tasks defined in the challenge. Firstly, a broader range of sensor data is utilized to build the models. This includes not only signals from smartwatches placed on the upper limbs but also data from Shimmer3 sensors placed on all limbs, both upper and lower. These additional sensors provide a more comprehensive dataset by capturing movement and activity from all four limbs. Furthermore, for the Shimmer3 sensors, the evaluations of bradykinesia and dyskinesia were not binary but were conducted on a scale from 0 to 4, allowing for more nuanced assessment.

Additionally, in this study, less metadata is used (no information regarding the patient identifier, session, or visit number), making trained models applicable in processing other datasets and not learning to do the prediction based on the patient IDs. Not only feature extraction is performed, but the models are finalized for 2 tasks – classification and regression. Classification, based on available values, can be performed to detect the presence of the symptom (binary classification) or to predict the severity of the symptom as a member of a class representing a severity score (from 0 to 4), which was assigned by the clinician. Symptom severities are in fact not discrete variables, but continuous. Discretization is in fact performed only to make the data more manageable for humans and enable the neurologists to evaluate patient condition with a predefined accuracy. However, assessing the severity of disease symptoms is a regression task. Therefore, regression models are built to better reflect patient's condition, wherever it is possible.

Using the dataset, it was possible to extract a different number of samples for each of the symptoms. For GENEActive and Pebble smartwatches (SW) it was: dyskinesia – 12 883, bradykinesia - 8 347, tremor – 12 883, for Shimmer3 sensors (SH) it was: dyskinesia – 16166, bradykinesia – 7466, tremor – 16166. Not all symptoms were evaluated during every exercise and unfortunately some of the data was missing. The

number of samples available for every performed task is presented in Table 7. For the purpose of the experiment, the datasets have been split into training and testing sets, for each of the symptom separately. The datasets and the split were different for every symptom due to the different scope and number of measurements. However, each of the training sets contained about 75% of the samples leaving 25% in the testing set. The split process was random, but the class contribution in the sets was kept similar. Performing the split has highlighted an existing imbalance in the contribution of each performed task within the training and testing sets, which is inherent to the MJFF dataset.. The training process is to be performed for three different dataset combinations: smartwatches dataset, Shimmer3 dataset and the two of them combined. This gives a good overview over the ability for the ML model to learn specific patterns in accelerometer signals.

Table 7 The number of samples available for the training process split between symptoms (T-tremor, B – bradykinesia, D- dyskinesia) and performed tasks (abbreviations explained in Table 1 – p. 29)

Sensors	**Smartwatches**			**Shimmer3**		
Task	T	B	D	T	B	D
drawg	692	413	692	795	402	795
drnkg	693	693	693	795	402	795
fldng	696	696	696	795	402	795
ftnl	1076	514	1076	1590	408	1590
ftnr	1077	562	1077	1590	396	1590
ntblt	704	371	704	795	402	795
orgpa	694	694	694	795	402	795
raml	1072	476	1072	1590	408	1590
ramr	1074	516	1074	1590	396	1590
sittg	699	331	699	795	0	795
stndg	702	0	702	795	0	795
strsd	106	106	106	133	133	133
strsu	105	105	105	133	133	133
ststd	700	409	700	795	795	795
typng	700	368	700	795	402	795
wlkgc	700	700	700	795	795	795
wlkgp	698	698	698	795	795	795

wlkgs	695	695	695	795	795	795

For smartwatches, the evaluation of bradykinesia and dyskinesia was binary, categorized simply as “Yes” or “No”, whereas tremor severity was quantified using a discrete scale ranging from 0 to 4, with increments of 1. In contrast, when utilizing Shimmer3 sensors, both tremor and dyskinesia severities were measured on a comprehensive 0 to 4 scale. However, the assessment of bradykinesia with Shimmer3 sensors showed variability. While some measurements adhered to the 0 to 4 scale, others used the binary Yes/No evaluation. This inconsistency in the evaluation of bradykinesia and dyskinesia, particularly the mix of binary and scale-based assessments, makes data preprocessing necessary for machine learning applications. While converting scale-based assessments to binary is straightforward (assigning 'No' to zeros and 'Yes' to any value above), developing a dataset with severity evaluations presents a more difficult challenge. To address this, two methodologies are proposed. The first method interprets 'No' as 0 and 'Yes' as 1. The second strategy is applicable only to dyskinesia and Boston patients, who were equipped with both smartwatches and Shimmer3 sensors. The consistent 0 to 4 scale assessments provided for Shimmer3 sensors are used to replace the binary evaluations assigned to smartwatch sensors measurements. These adjustments are implemented exclusively for the purposes of regression analysis and multiclass classification tasks, ensuring a uniform dataset capable of supporting algorithmic predictions.

Although the clinicians thoroughly assessed symptoms' severities on a scale from 0 to 4, the dataset is not diverse in terms of severity class representation. Due to low numbers of samples where symptoms were present and an even lower number for high severities the dataset is severely imbalanced. Some classes have over 483 times more samples than others e.g., tremor severities, the exact class distributions are presented in Figure 8. The high imbalance in the dataset makes it difficult to build a model capable of predicting less represented classes. To handle this, appropriate steps need to be taken, such as using unconventional loss functions or resampling techniques.

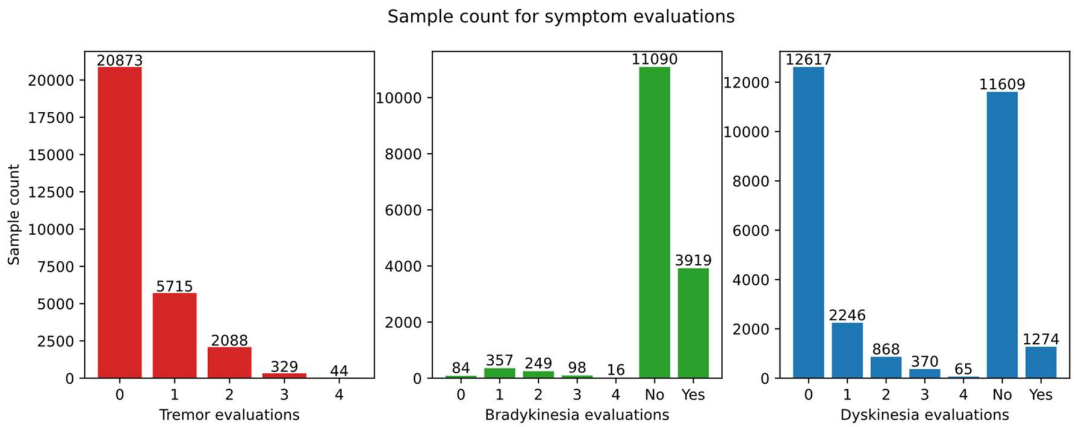

Figure 8 Histogram presenting the distribution of symptom severities for MJFF dataset

4.1.2. Deep learning model

In order to make the created model applicable to sensor measurements outside the Levodopa Response Study only the raw accelerometer signal was used across 3 axes: X, Y, and Z, along with limited metadata: the task code, the limb the sensor was worn on upper/lower and left/right, and the device type: GENEActive/Pebble/Shimmer3. This approach ensures that the model is independent of the study's specific regime. Importantly, patient identifiers, location, visit number, and session number were not used to train the model predicting the intensity and presence of PD symptoms, ensuring broader applicability, and reducing potential biases.

Before the data is provided to the network's inputs, basic transformation is performed. To keep the network's dimensions constant, it is ensured that all the samples are of equal length – 4000 values per each axis (signal frequency – 50 Hz). When the sample is longer, only the last 4000 samples are considered - the beginning of the signal is often noisy and might contain data before the patient starts executing the task. In case of shorter recordings, the signal is padded with zeros until it matches the desired length. The next step is normalization of the signals. It is performed based on the mean and standard deviation for each of the axes. For the task type, device type, and limb features one-hot encoding [88] is performed and followed by normalization to ensure correct distinction between values by the model.

To predict the symptoms an artificial neural network (ANN) [89] was selected. ANNs are machine learning models inspired by the structure of neurons in the human brain, designed to recognize patterns and solve problems by processing input data through layers of connected neurons. Basically, it involves matrix multiplications of input and

weights followed by the application of activation functions, which enable the network to learn and model complex non-linear relationships. The architecture includes an input layer – receiving data, hidden layers, which compute the transformations, and the output layer providing the result, these are presented in Figure 9. During the training process of a neural network, gradients of the loss function with respect to the weights and biases are calculated. This is used to find a way the weights should be updated to minimize the value of the loss function. This process is called back-propagation.

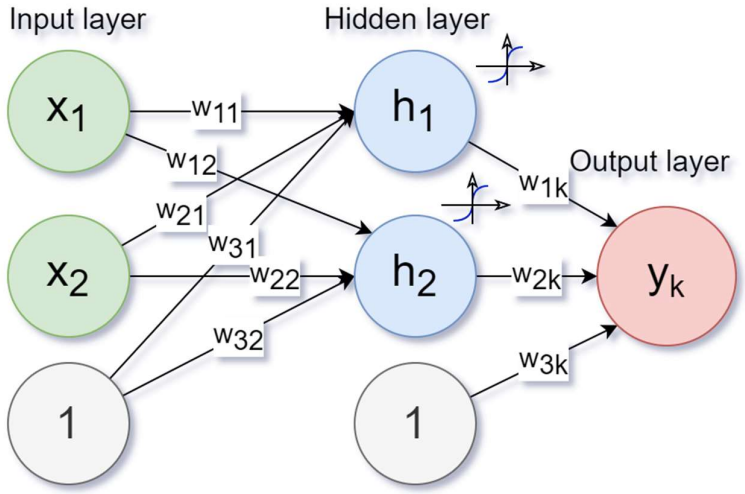

Figure 9 Structure of an ANN – multilayer perceptron

In the simplest neural network - the multilayer perceptron, the neurons in all of the layers are fully connected, meaning that every neuron in the previous layer is connected to a neuron in the following layer. However, deep neural networks have been initially chosen to solve the task. These do not require the calculation of features but can process raw data and calculate the features by themselves using specific layers e.g., convolutional layers [90]. Deep learning allows using different types of layers that can be connected in various ways to provide easy data processing and good prediction results. Gutowski [68] presented an approach that used only convolutional, fully connected, and dropout layers, from the PyTorch library [91], but after further experimentation with different architectures, other layer types and their combinations have been explored and these experiments were implemented using TensorFlow [92].

The main layer type used is a convolutional layer [90]. It is widely used for image and video recognition and natural language processing, where it is able to adaptively and automatically learn spatial hierarchies of features from input. Convolutional layers can

process multidimensional data e.g., three-dimensional networks to process videos, two-dimensional to process images and one-dimensional to process signals like audio recordings or in this case, sensor signals, collected with accelerometers. The greatest advantage of these layers is that they can perform feature extractions directly from raw data without the need for manual feature extraction. They operate by applying filters (or kernels) over the input data to extract features by performing convolution operations, which involve element-wise multiplication of the filter with the input followed by a summation, as presented in Figure 10. In most cases, there are multiple convolutional layers stacked one after another. The earlier layers are tasked with detecting fewer, more simple patterns – they contain smaller and fewer filters, while later layers focus on detecting complex features based on features already detected by previous layers. The hierarchical approach to feature detection allows them to build a comprehensive understanding of the data.

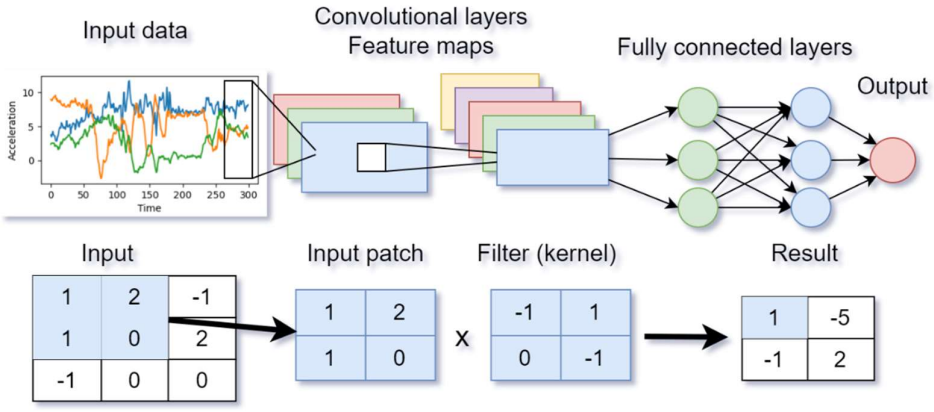

Figure 10 Structure of convolutional neural networks and convolution example

Beside defining the filter size and the number of filters, it is possible to define additional parameters such as stride and padding. Stride defines how many units the filter moves across the data in each step. The bigger the stride the smaller the result of the application of the layer, by default it is equal to 1. As a result of padding, additional values (usually zeros) are added around the border of the network. This allows for the filters to be applied at the edges of the data, therefore making it possible to capture features that might be present there. Depending on the shape of the input, the developer should also define the number of channels. For black and white images there would be one channel, for colored images three, representing the red, green, and blue components. For processing inertial sensor data usually three channels are used, one for each axis: X, Y

and Z. Every convolutional layer is typically followed by an activation function, just as with fully connected layers. Usually, it is the rectified linear unit (ReLU) activation function, which introduces non-linearity to the model and is fast to compute. This efficiency makes it preferable to other activation functions such as the hyperbolic tangent (tanh) or sigmoid functions. ReLU outputs the input directly if it is positive, and zero otherwise. The summary of commonly used activation functions in ANNs is presented in Table 8 .

Table 8 Activation functions commonly used in artificial neural networks

Activation Function	**Equation**	
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	Eq. 1
Tanh	$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	Eq. 2
ReLU	$ReLU(x) = max\,(0, x)$	Eq. 3
Softmax	$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$	Eq. 4

Generally, convolutional layers are separated with pooling layers – usually max-pooling or average-pooling is applied [90]. Their goal is to reduce the spatial dimension of the feature maps outputted by convolutional layers. For every subregion of the data, they summarize the presence of every feature, leading to the down-sampling of the data while maintaining the most important detected features. Max-pooling selects the highest value in every subregion, while average-pooling calculates the average value. The use of these layers allows following convolutions to focus on higher level features. For every pooling layer the size of the subregion, the stride and the padding can be provided.

Due to the big sizes of deep learning models, several approaches have been created to handle the problem of overfitting to the training data [90]. The first technique is using the dropout layer [90]. During the training process, the layer randomly selects a number of neurons that are deactivated, and their outputs will not be passed further. This forces the network to learn robust features and not rely only on a small number of neurons, since they can be deactivated during some iterations. The percentage of neurons that are being deactivated can be defined by the developer.

The data, after being processing with convolutional layers is then usually flattened [90]. This output – representing the high-level features present in the input data is then passed through fully connected layers. Their role is to take the high-level, abstracted features from the preceding layers and combine them to make a final prediction. An interesting, alternative approach is to use the global average pooling layer [93], after the convolutional layers. It can serve as an effective layer to reduce the spatial dimensions of feature maps into a single vector per map by calculating the average of each feature map. This technique not only helps in minimizing the model's complexity by reducing the number of parameters (reducing overfitting), but also provides another approach to transforming the data for further processing, which can also simplify the network architecture by reducing the need for fully connected layers.

The initial structure of the network is based on ideas proposed in the DREAM Challenge [28], tasked with:

- prediction of tremor severity,
- detection of:
    - tremor,
    - bradykinesia,
    - dyskinesia.

The neural network consists of two main branches: convolutional branch – transforms the accelerometer signal and outputs feature values extracted from the signal and the simple input branch providing task and device information. The simple branch output is concatenated with the flattened output of the last convolutional layer. The convolutional branch consists of 8 sequences of convolutional layers, rectified linear unit (ReLU) activation functions and max pooling (size: 2). After passing through convolutional layers the signal is flattened to create a one-dimensional tensor which is supplemented with 20 additional values – tasks and the device side vector after one-hot encoding. The full vector is then processed by two fully connected layers with ReLU activation functions and dropout. The final layer has 1 output – binary classification (bradykinesia, dyskinesia) or 5 (tremor) outputs that correspond with output symptom severity/presence. For tremor severity prediction the values are then passed through the soft max function. The number of filters, their sizes, and the complete network structure have been presented in Figure 11.

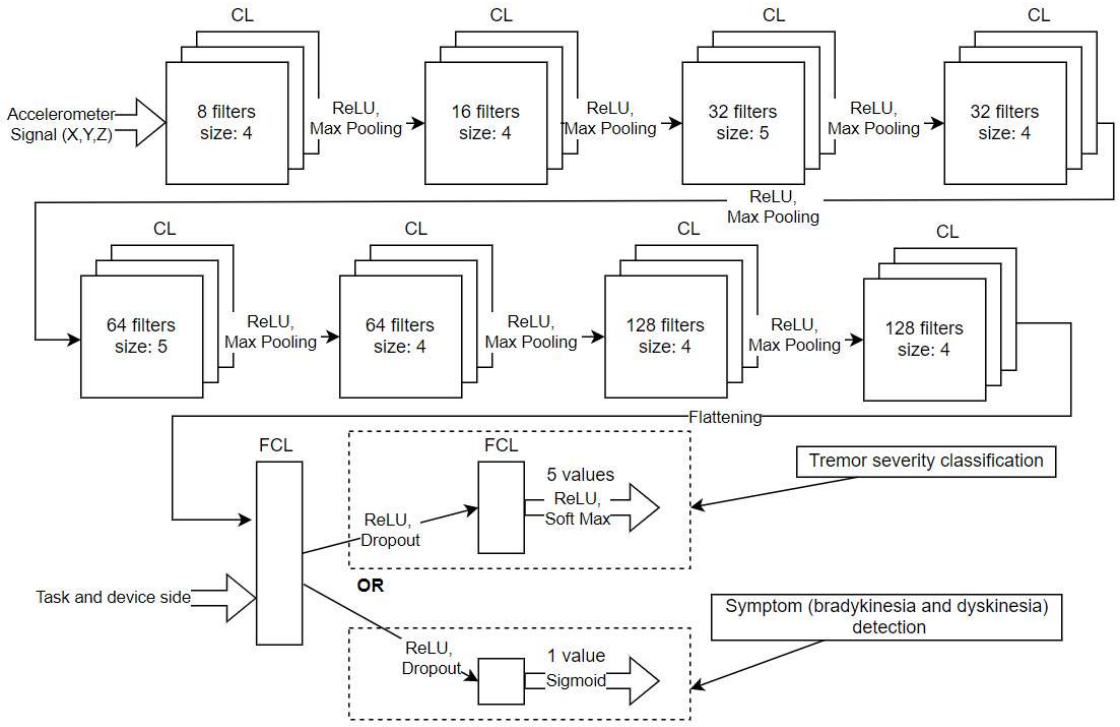

Figure 11 Initial neural network structure for classification of symptom presence and severity [68]

In the initial study [68] the same network structure has been used for all the classification tasks with the only differences being the number of outputs in the last layer, the activation function (Sigmoid for detection, ReLU for severity evaluation) and the use of soft max function. The Adam algorithm [94], chosen for its efficiency and adaptive learning rate, was used for optimization, with a learning rate between 0.0003 and 0.001 depending on the symptom. The cross entropy [95] has been used as a loss function. Due to the big differences in class members number (imbalanced dataset), weights have been provided, in order to improve the separation of class instances. The learning process was run for a different number of epochs depending on the symptom. If the value of loss had not decreased in the last 15 epochs, the training process was stopped – to avoid overfitting. This resulted in 271 epochs for bradykinesia 179 for dyskinesia and 338 for tremor. The model has been created and trained using Python library – PyTorch [91], which is dedicated for solving deep learning problems.

During the revision of the problem more architectures have been investigated to solve the problem, these included recurrent neural networks [96], which are designed to process sequential data by keeping the memory of previous inputs using their inner state. This helps capture dependencies between different parts of the signal and temporal

dynamics. The application of attention layers [97], has also been investigated. It provides a mechanism that allows the model to focus on specific parts of the input sequence when making predictions, improving the handling of long-range dependencies, and enhancing the performance. As an alternative solution to one-hot encoding of the performed task by the patient, the use of an embedding layer [98] was investigated. Instead of creating one column per every possible categorical data value, a representation using 2, 3 or 4 values was tested to account for all the 18 tasks. Embeddings result in mapping similar categories to points close to each other in the vector space, capturing the semantic relationships between them. They are trained by iteratively adjusting initially random vectors to minimize a loss function. The described approaches and layers have been tested to define architectures that provide the best results for both classification and regression tasks defined for patient symptom severity evaluation. Based on the experiments, the best results were acquired using the network presented in Figure 12. It consists of 3 convolutional layers that process the accelerometer input, separate with max-pooling layers. The features found in the data are then processed using the global average pooling layer, which is followed by the concatenation with meta data inputs. The device type and limb are encoded using one-hot encoding. For the task name two approaches were explored, the first one was using one-hot encoding and the second one being embeddings created to represent specific tasks. After concatenation of these two inputs, the data is passed through two fully connected layers, the last one ending with a sigmoid or a SoftMax activation function applied to all of the output. The sigmoid activation function (Eq. 1) restricts the output to the range from 0 to 1 with the value for 0 being 0.5, while the SoftMax function (Eq. 4) converts a list of real numbers into a probability distribution. The number of outputs is defined by the type of classification task, for symptom detection, which is a binary classification, there is just one output. When the severity of the symptom is predicted, the number of outputs is 5, one for each severity level.

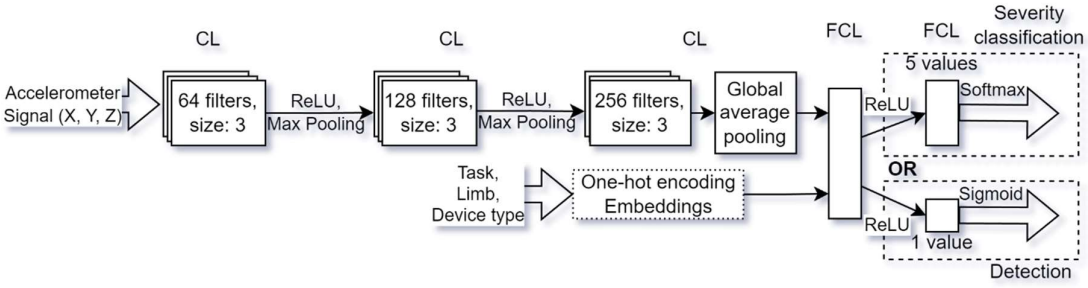

Figure 12 The revised model architecture for prediction of presence or severity of PD symptoms using the MJFF dataset

The training of the revised network was performed with a weighted variant of binary cross entropy function (Eq. 5) [99]. It accounts for the imbalance in the dataset, making predictions of less represented classes possible and reliable. The loss function uses positive (Eq. 6) and negative (Eq. 7) weights defined for every class and based on that, the contribution of specific classes in the loss function is organized. The optimization was performed using Adam optimizer with a learning rate of 0.001, the maximum number of epochs was set to 600. However, there was an early stop callback defined too. These models were constructed and trained using the TensorFlow [92] library.

$$L = -\sum_{i}\left(w_{pos,i}\cdot y_{true,i}\cdot\log\left(y_{pred,i}+\epsilon\right)+w_{neg,i}\cdot\left(1-y_{true,i}\right)\cdot\right.$$
$$\left.\cdot\log\left(1-y_{pred,i}+\epsilon\right)\right) \qquad \text{Eq. 5}$$

$$w_{\mathrm{pos,i}} = \frac{n_{\mathrm{neg,i}}}{n}$$

$n_{\mathrm{neg,i}}$ – number of negative samples (do not belong to the class i), Eq. 6

n – total number of samples.

$$w_{\mathrm{neg,i}} = \frac{n_{\mathrm{pos,i}}}{n} \qquad \text{Eq. 7}$$

$n_{\mathrm{pos,i}}$ – number of positive samples (belong to the class i).

To build a regression model, a similar architecture was used as for classification (Figure 12). Similarly, it includes two inputs – accelerometer signal and meta data input, consists of the same number of layers with equal architectures. The only change is in the last layer. For regression, there is always just one output, and no activation function is applied; the output directly represents the severity of the symptom. Due to the imbalance in the dataset and worse performance, the experiments in building regression models

using datasets individually are not a part of this study, only experiments on the dataset including both Shimmer3 and smartwatches are explored.

For the training process, again, the Adam optimizer was used. However, the learning rate was increased to 0.003, because the previous value, used in classification, allowed the network to improve the value of the loss function result for over 1000 epochs. The regression model utilizes the mean squared error (MSE) (Eq. 8) as a loss function. Experiments were performed with newly defined loss functions for handling the imbalance of the data; however, they did not provide significantly better results.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{\mathrm{true},i} - y_{\mathrm{pred},i}\right)^2 \qquad \text{Eq. 8}$$

4.1.2.1. Results

This part of the study focused on creating classification and regression deep learning models in order to detect and predict the severity of PD symptoms: bradykinesia, dyskinesia, and tremor on a scale from 0 to 4. In the classification task the presence of the symptom was predicted (binary classification) and the severity evaluation (multiclass classification). The study was performed in two stages. The results of the first stage were presented previously at a conference [68], the data collected from smartwatches placed on the wrists were used to predict the presence of each of the 3 symptoms and the severity of tremor. The dataset was expanded with the data collected from Shimmer3 sensors and classification was performed for both datasets and their combination using an improved architecture. The detection and severity evaluation models were created for all of the symptoms.

In order to evaluate the classification models 3 metrics were selected, two of them designed to address the class imbalance of the dataset. The first one is accuracy (Acc), defined as the percentage of correctly classified samples (Eq. 9). The second metric is balanced accuracy (BAcc) (Eq. 10)[100], representing an adjusted version of the accuracy that takes into account the data imbalance by making the samples of classes less represented more important. It can be calculated using recall (Eq. 11) which in this case represents the ability to identify all instances within a particular class. The last metric selected was the area under the precision-recall curve (AUC PR) (Eq. 12). This choice was made to facilitate comparison with the values presented in the DREAM Challenge, where this metric was used to evaluate the models [28,70]. It uses the recall values

calculated for different thresholds of the model outputs (direct output values between 0 and 1, not the class index) and the precision, which represents the proportion of positive identifications that were actually correct. In the case of multiclass classification, the weighted AUC PR for the one-vs-rest (Eq. 13) scenario was used, where the weights represented the contributions of specific classes.

$$Acc(y_{\text{true}}, y_{\text{pred}}) = \frac{1}{\text{n}} \sum_{i=0}^{\text{n}-1} 1(y_{\text{pred},i} = y_{\text{true},i}) \quad \text{Eq. 9}$$

n – number of samples, 1(x) – indicator function

$$BAcc = \frac{1}{C} \sum_{i=1}^{C} recall_i \quad \text{Eq. 10}$$

C – number of classes

$$recall_i = \frac{\sum_{j=1}^{n} 1(y_{\text{pred},j} = i \wedge y_{\text{true},j} = i)}{\sum_{j=1}^{n} 1(y_{\text{true},j} = i)} \quad \text{Eq. 11}$$

$$\text{AUC PR} = \sum_{k=1}^{n} (recall_k - recall_{k-1}) \times \text{p}recision_k \quad \text{Eq. 12}$$

n – number of threshold levels

$$Weighted\ AUC\ PR\ = \frac{1}{C} \sum_{i=1}^{C} \frac{n_{\text{pos,i}}}{n}\ AUC\ PR_i \quad \text{Eq. 13}$$

C – number of classes

The classification results for the whole dataset (Shimmer3 + smartwatches) are presented, in a form of normalized confusion matrices [101] for the revised model (Figure 12). Figure 13 presents the results for bradykinesia, symptom detection and severity evaluation, while Figure 14 presents these results for tremor and Figure 15 for dyskinesia.

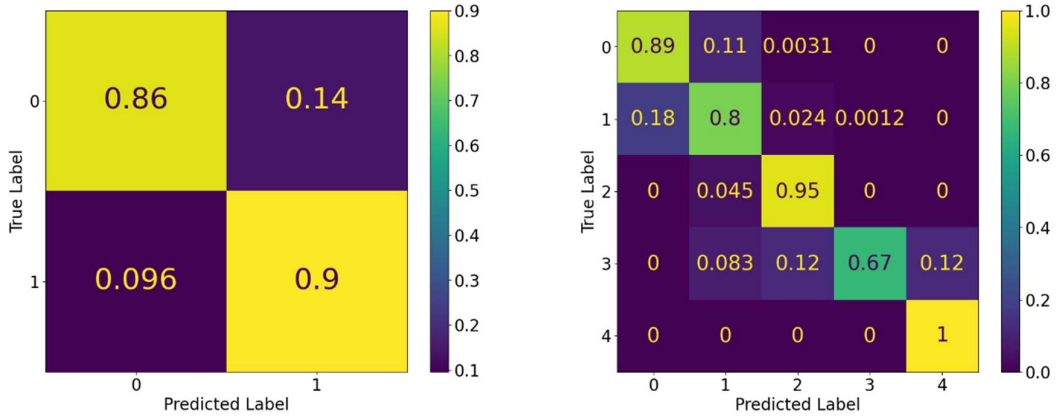

Figure 13 The normalized confusion matrices for bradykinesia detection (left) and severity (right)

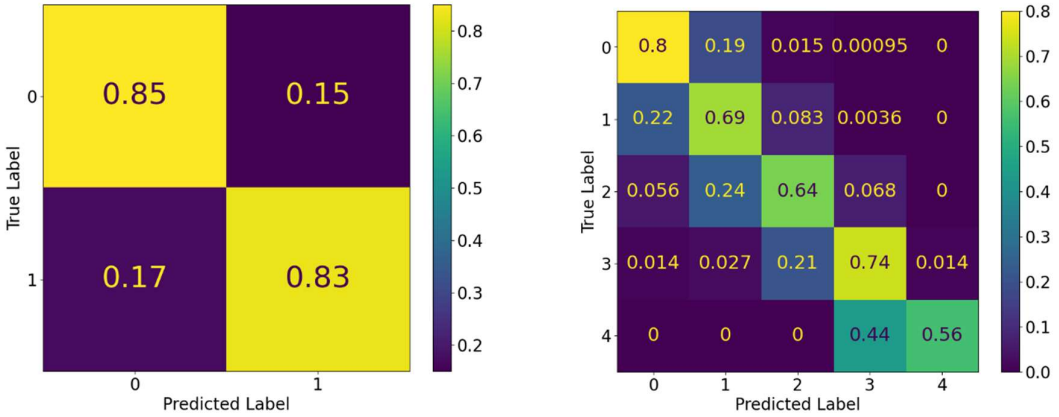

Figure 14 The normalized confusion matrices for tremor detection (left) and severity (right)

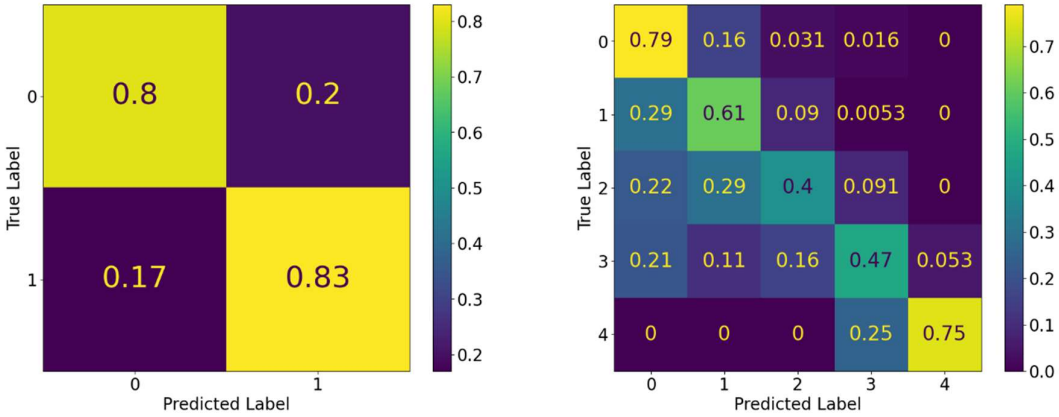

Figure 15 The normalized confusion matrices for dyskinesia detection (left) and severity (right)

The specific results for all the models trained with the initial architecture are showcased in Table 9. Table 10 contains the results (metrics values) for the improved convolutional neural network, along with the architecture parameters, such as number of embeddings or the way of handling 'Yes' values in the severity evaluation task.

Table 9 Prediction results for test sets classification using the initial CNN model

Symptom	Dataset	Mode	BAcc	Acc	AUC PR
Bradykinesia	SW	detection	83.7%	83.1%	0.847
Dyskinesia	SW	detection	70.0%	70.9%	0.279
Tremor	SW	severity	61.1%	60.5%	0.730
Tremor	SW	detection	78.9%	80.9%	0.748

SW – smartwatch dataset, SH – Shimmer3 dataset

Table 10 Prediction results on the test set for the revised CNN model

Symptom	Dataset	Mode	'Yes' mapping	Task name	BAcc	Acc	AUC PR
Bradykinesia	SH	detection	-	2 emb.	89.6%	89.8%	0.921
	SW			3 emb.	85.4%	86.9%	0.895
	SH+SW			2 emb.	88.2%	89.1%	0.882
	SH	severity	1	3 emb.	74.7%	88.0%	0.937
	SH+SW			4 emb.	86.2%	86.5%	0.927
Dyskinesia	SH	detection	-	one-hot	82.0%	81.2%	0.695
	SW			2 emb.	76.9%	83.0%	0.481
	SH+SW			4 emb.	81.6%	82.5%	0.687
	SH	severity		2 emb.	62.3%	71.0%	0.844
	SH+SW		Shimmer data or 1	one-hot	60.5%	76.6%	0.900
Tremor	SH	detection	-	4 emb.	85.4%	85.5%	0.834
	SW			4 emb.	84%	84.7%	0.821
	SH+SW			3 emb.	83.6%	83.2%	0.819
	SH	severity		one-hot	66.9%	81.1%	0.872
	SW			3 emb.	67.9%	71.9%	0.82
	SH+SW			one-hot	68.3%	76.3%	0.852

SW – smartwatch dataset, SH – Shimmer3 dataset, emb. - embeddings

The revised model performed significantly better in all previously explored machine learning tasks. For bradykinesia detection the change resulted in a 1.7% improvement to the balanced accuracy, for dyskinesia the improvement was 6.9% and for tremor 5.1%. The improvement can be seen across all used metrics. For tremor severity

classification the improvement was 6.8% in the balanced accuracy and higher for other metrics. The classification task has also been explored for the dataset created with Shimmer3 sensor recordings. In most cases, the models trained on that data provided better results. This might be caused by the fact that Shimmer3 sensors collected less diverse data, as they were used only on patients from Boston.

Among the 3 symptoms, the best results have been achieved for bradykinesia, in both detection and severity evaluation, all three metrics confirm that the model performs well for detecting the symptom and evaluating its severity. However, due to the low number of samples, especially with higher severities, these results should be further verified on a bigger and more diverse dataset. The evaluation for tremor provided second best results, with high values among all 3 metrics. The severity evaluation demonstrated strong performance, as illustrated by the confusion matrix. Most of the misclassifications are results of measurements assigned to the neighboring class. In case of ordinal classification problems, it is more preferable than assignment to more distant ones. It can also be a result of wrongly assigned labels to the data by the clinician due to the high granulation of severities (which might sometimes be hard to distinguish), especially if the measurements were performed at different sites and supervised by different clinicians. The results for dyskinesia detection are worse than for other symptoms, especially the detection based on the smartwatch dataset. This might have been caused by the difficulty for the model to distinguish between dyskinesias and voluntary movements performed by patients or incorrectly selected task set for detection.

When evaluating machine learning models trained on clinical data, particularly collected with precise sensors like accelerometers, it is crucial to consider the accuracy and consistency of data labels – in this case symptom severities. These data labels are determined by human experts, such as clinicians, whose judgments can introduce subjectivity and potential bias. This human factor can lead to inconsistencies and potential bias in the dataset. Invalid labels might end up in the training or the test set. Having them in the training set, in large quantities, results in models that are seemingly accurate but in reality, only replicate the errors present in the label assignments. Their presence in the test set might result in apparent misclassification and worse prediction results. This is a serious problem especially in case of datasets that are highly imbalanced. Errors in the labels of underrepresented classes might lead to worse results, therefore it is necessary to be aware and handle them appropriately. In case of this dataset, it was only possible to

partially investigate the suitability of labels – for tremor severity. The tremor can be partially visible in the charts of accelerometer signals, unfortunately this dataset is not free from these inconsistencies, which are presented in Figure 16. The chart presents two signals registered from patients, the signal with a higher variability in the PD tremor frequency spectrum was assigned a lower label value (0) than the signal with little variability, for which the clinician provided the label of tremor severity equal to 3.

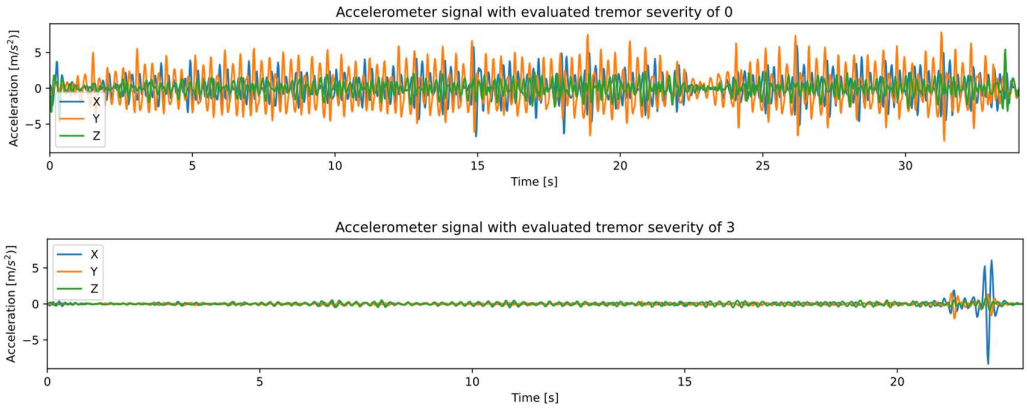

Figure 16 Charts presenting inconsistencies in tremor evaluation by clinicians

Table 11 and Table 12 present the previously presented metrics (accuracy, balanced accuracy, and AUC PR) for each of the performed tasks separately. Due to the lower sample numbers, these have been provided only for the dataset including smartwatch and Shimmer3 sensors together. The metrics are calculated for detection of these 3 symptoms and the severity evaluation of tremor (the severity evaluation scores were present for tremor in both datasets). This could help determine which tasks are more effective than others for predicting tremor severity and detecting bradykinesia, dyskinesia, and tremor.

Table 11 Bradykinesia and dyskinesia presence prediction results for specific tasks

Task	**Bradykinesia**			**Dyskinesia**		
	BAcc	Acc	AUC PR	BAcc	Acc	AUC PR
drawg	77.2%	80.6%	0.695	77.5%	82.9%	0.559
drnkg	57.4%	87.5%	0.137	84.1%	85.4%	0.807
fldng	63.9%	88.0%	0.470	86.0%	86.2%	0.750
ftnl	89.6%	91.2%	0.899	80.3%	80.9%	0.640
ftnr	90.2%	92.0%	0.922	84.6%	82.6%	0.731

ntblt	74.4%	85.5%	0.554	78.9%	81.1%	0.635
orgpa	86.5%	90.9%	0.677	84.8%	86.5%	0.741
raml	88.6%	93.1%	0.944	80.3%	80.5%	0.696
ramr	91.0%	95.4%	0.968	82.7%	82.5%	0.764
sittg	42.1%	78.7%	0.046	62.4%	74.0%	0.224
stndg	-	-	-	77.0%	72.3%	0.607
strsd	76.1%	76.6%	0.726	62.1%	87.9%	0.451
strsu	80.0%	80.0%	0.375	90.5%	82.9%	0.670
ststd	84.0%	92.1%	0.432	70.6%	73.5%	0.483
typng	72.8%	92.9%	0.592	81.1%	83.5%	0.655
wlkgc	89.4%	89.1%	0.936	86.7%	90.1%	0.864
wlkgp	87.7%	86.6%	0.791	88.3%	90.3%	0.818
wlkgs	89.4%	90.1%	0.888	87.9%	89.5%	0.782

Detecting bradykinesia based on accelerometer signal has been most successful when the patients were performing repeated arm movements for both arms – left and right. Excellent results were also achieved for all walking exercises (straight, while counting and through a narrow passage) and finger to nose movements. The worst results were received for the following activities: sitting, drinking, folding towels as well as the tasks represented by the smallest number of samples: going up and going down the stairs. The probable reason for bad performance of the model when the patient was sitting, drinking is the fact that no or little movement is performed during these tasks. Therefore, it is harder to detect the slowness of movement - bradykinesia.

The results of dyskinesia detection are overall worse than for bradykinesia. However, surprisingly, for some of the tasks the model provided better metrics than the model for bradykinesia e.g., going up the stairs. The trained model could have trouble distinguishing the symptom occurrence with voluntary patient movement. The tasks providing the best performance were related to walking, including walking straight, while counting and through a narrow passage, drinking and folding towels. The worst results were achieved for tasks related to no or minimal movement such standing, sitting and sit to stand movements.

Table 12 Tremor severity and presence prediction results for specific tasks

Task	Tremor severity			Tremor presence		
	BAcc	Acc	AUC PR	BAcc	Acc	AUC PR
drawg	36.7%	74.0%	0.804	84.5%	80.1%	0.775
drnkg	60.6%	73.4%	0.842	78.8%	77.8%	0.793
fldng	55.7%	83.4%	0.881	85.7%	85.9%	0.836
ftnl	60.8%	70.1%	0.852	83.1%	81.2%	0.861
ftnr	46.1%	75.3%	0.870	85.0%	83.0%	0.907
ntblt	39.6%	67.0%	0.785	77.4%	75.4%	0.731
orgpa	52.4%	74.4%	0.842	81.3%	82.9%	0.798
raml	68.1%	70.5%	0.817	82.0%	79.7%	0.828
ramr	75.3%	74.6%	0.832	84.7%	83.3%	0.832
sittg	50.0%	66.8%	0.714	69.1%	77.1%	0.538
stndg	43.9%	71.2%	0.772	74.1%	76.4%	0.747
strsd	39.3%	82.8%	0.794	82.5%	87.9%	0.622
strsu	17.9%	82.9%	0.744	79.4%	90.2%	0.405
ststd	46.7%	86.7%	0.801	76.7%	87.8%	0.582
typng	42.3%	70.3%	0.815	80.7%	80.2%	0.788
wlkgc	62.0%	91.1%	0.921	90.1%	93.7%	0.924
wlkgp	54.6%	86.9%	0.839	87.1%	93.1%	0.841
wlkgs	70.5%	94.4%	0.931	94.4%	96.8%	0.936

Due to the small number of class members for higher values of tremor severity classification, when split by task type, the results presented in the left part of Table 12 might be ambiguous. Some of the tasks did not have even one class member for at least one of the classes when split based on task type was performed. However, based on analysis of metric values present in both parts of the table it is possible to notice the tasks that allowed for the best and the worst performance in case of both severity evaluation and detection. Considering the values of balanced accuracy and AUC PR, the activities regarding walking resulted in best performance. These are followed by finger to nose movements, for both hands, folding towels, and repeated arm movements. Similarly to dyskinesia evaluation, this model performed the worst on data collected when the patient was sitting or standing as well as walking up and down the stairs.

To evaluate the performance of regression models, defined metrics are calculated on the prediction results of the test set. Commonly used metrics in evaluation are the coefficient of determination (R^2) (Eq. 14), MSE (Eq. 8), mean absolute error (MAE) (Eq. 15), Pearson's correlation coefficient (r) (Eq. 16) between the true values and predicted outcomes [102]. These give a good overview on the overall performance of the machine learning models. However, in case of highly imbalanced datasets, such as this one, these metrics might now give enough information for model evaluation. Therefore, two additional metrics have been constructed. Since the original labels are discrete values, which were considered classes previously, it is possible to calculate class specific metrics. MAE has been selected to be calculated for every class separately (Eq. 17). This has been used to create a derived metric – bMAE (Eq. 18), which represents the mean absolute error across different classes, similarly bMSE (Eq. 19) has been defined, using the mean squared error calculated for every class.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_{\text{true},i} - y_{\text{pred},i}\right)^2}{\sum_{i=1}^{n}\left(y_{\text{true},i} - \overline{y_{\text{true}}}\right)^2} \quad \text{Eq. 14}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_{\text{true},i} - y_{\text{pred},i}\right| \quad \text{Eq. 15}$$

$$r = \frac{\sum_{i=1}^{n}\left(y_{\text{true},i} - \overline{y_{\text{true}}}\right)\left(y_{\text{pred},i} - \overline{y_{\text{pred}}}\right)}{\sqrt{\sum_{i=1}^{n}\left(y_{\text{true},i} - \overline{y_{\text{true}}}\right)^2 \sum_{i=1}^{n}\left(y_{\text{pred},i} - \overline{y_{\text{pred}}}\right)^2}} \quad \text{Eq. 16}$$

$$\text{MAE}_k = \frac{1}{n_k}\sum_{i \in \text{class } k}\left|y_{\text{true},i} - y_{\text{pred},i}\right| \quad \text{Eq. 17}$$

$$\text{bMAE} = \frac{1}{C}\sum_{k=1}^{C}\text{MAE}_k \quad \text{Eq. 18}$$

$$bMSE = \frac{1}{C}\sum_{k=1}^{C}\frac{1}{n_k}\sum_{i \in \text{class } k}\left(y_{\text{true},i} - y_{\text{pred},i}\right)^2 \quad \text{Eq. 19}$$

For every regression task: the prediction of tremor, dyskinesia and bradykinesia severities, a violin plot has been selected to present the results. Usually, for regression tasks a scatter plot is selected, but due to the big number of samples and the discrete nature of true labels, it would not be clear and would not provide a good overview of model's performance. A violin plot can be used to provide a visual summary of the data's

distribution showing both the spread of the data and the density off data points at different values. In this case, the real symptom severities (classes) appear on the X axis and the Y axis represents the values predicted with the model. To make the presentation more informative, class specific MAE values are presented in the center of each violin. Plots for each symptom, are presented in Figure 17 and the metric values for the models on the validation set are included in Table 13.

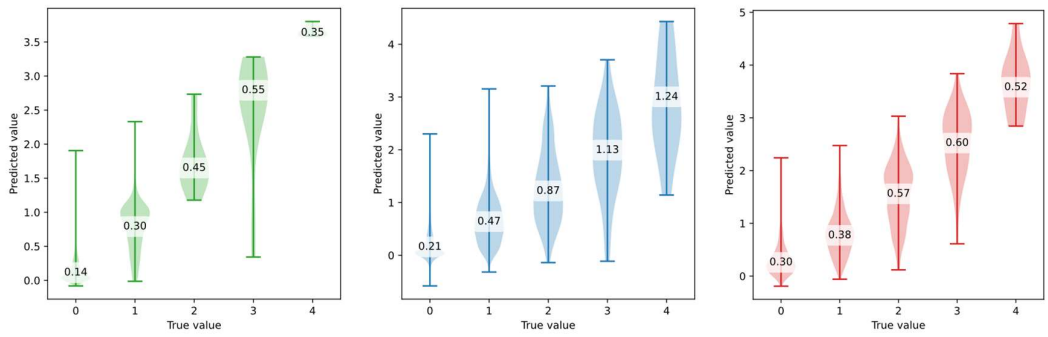

Figure 17 Violin plots presenting regression results with class specific MAE values for bradykinesia (left), dyskinesia (middle), tremor (right)

Table 13 Prediction results of regression models for symptom severities

Symptom	R^2	R	MAE	bMAE	bMSE
Bradykinesia	0.721	0.850	0.204	0.358	0.288
Dyskinesia	0.422	0.675	0.282	0.785	1.10
Tremor	0.686	0.828	0.383	0.473	0.395

The regression models for predicting the symptom severity allow to predict continuous values of the severities. Similarly to classification, the best results are received for bradykinesia, where traditional metrics such as R^2 and r have high values and errors (MAE) have low values. The newly introduced, balanced metrics for bradykinesia are also the lowest among all symptoms. Their values indicate low errors in predictions. The model for dyskinesia performs the worst, especially when considering samples with higher severities. The violins for severities 2, 3 and 4 are tall and wide in most of their height, presenting lower predictive capabilities for these states. The model for tremor prediction demonstrates good predictive performance and provides a strong linear relationship between the true and predicted values. However, its predictive capabilities significantly worsen for higher severities.

4.1.3. Conventional ML models

Deep learning methods, which were used for symptom state evaluation, do not perform well on small datasets. Therefore, the use of conventional machine learning methods was explored too. The MUW dataset contains significantly fewer samples than the MJFF dataset, which reduces the effectiveness of deep learning methods and necessitates the use of classic ML techniques.

Before applying these methods just to the MUW dataset, they were first tested on the MJFF dataset, where performance metrics are available from deep learning for comparison. When working with large datasets containing raw signal data, the training process requires additional preprocessing and feature extraction. The process of preparation the raw sensor signal for ML training/prediction is presented in Figure 18.

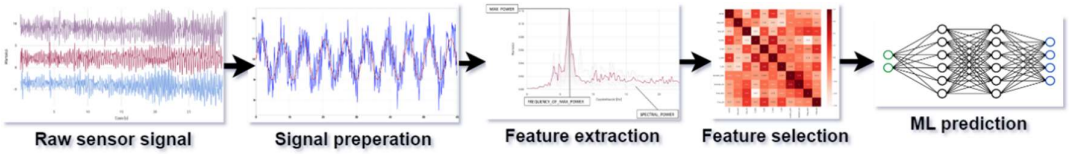

Figure 18 Chart presenting the preparation of raw signal for conventional machine learning models

The raw signal should be prepared first, based on the signal type, preparation should include actions such as filtering the signal to remove some components from the raw signal, calculating the magnitude of the signal, decomposing it into multiple signals. This is then followed by the feature extraction step. Based on the signal type and the purpose of the model (what variable is predicted), a set of features is selected. These features are calculated based on the signal and should provide a good representation of it, considering the expected results of the model. If the number of created features is high, appropriate method are often employed that aim at reducing the number of variables, leaving only those that might provide the most accurate model. The reduced number of features can then be delivered as the inputs to the ML model.

MJFF dataset consists of measurements from accelerometers – inertial sensors, which are used to monitor and record the movements and activity levels of individuals. To prepare these signals for ML models, it is first filtered using a high-pass filter – in order to remove the gravitational acceleration component with a cut-off frequency of 0.1 Hz. All of the signals are also filtered with a low-pass filter, to remove all noise. Since the sampling frequency is 50 Hz, the cut-off frequency was selected as 20 Hz – none of

the tremors and interesting PD-related movement features should surpass this frequency. For filtering the signals, the Butterworth filter [103] was used, which is often utilized for its flat frequency response in the passband, ensuring minimal signal distortion before the cutoff frequency. This approach balances removing unwanted components and retaining crucial movement data, facilitating accurate feature extraction and analysis.

The signals after filtering are used to calculate the magnitude signal (Eq. 20), which represents the movement captured in all directions. This derived signal allows for a more aggregated analysis, disregarding the direction of movement, which might be important, especially in cases where the sensors are not always worn in the same orientation.

$$M = \sqrt{X^2 + Y^2 + Z^2} \quad \text{Eq. 20}$$

4.1.3.1. Feature extraction

Following the description of main PD symptoms such as dyskinesia, bradykinesia, and tremor, along with consultations with neurologists, the signal is decomposed into 3 bands, based on the frequencies of components. These form the 0-3, 3-9 and 9-14 Hz frequency bands. The features are then calculated for each of these frequency bands and the whole signal, for each of the axes and the magnitude. This allows to capture different aspects of the signal, in order to provide accurate and precise representation of the signal. The features selected to be calculated are chosen based on literature review [26,32,35,104–108] regarding the analysis of inertial signals for detecting activities, diagnosing PD and quantification of PD symptoms.

They can be divided into time domain features, which are calculated directly using the signal, and frequency domain features, which provide insights into the signal's frequency content. To extract frequency domain features, the signal undergoes a Fourier Transform [109], a process that decomposes the signal into its constituent frequencies, revealing the spectrum of frequencies present and their relative intensities. Part of this analysis involves computing the Power Spectral Density (PSD), which quantifies the power present within each frequency component of the signal. The PSD is a crucial step in understanding the energy distribution across various frequencies, enabling the identification of dominant frequency bands that may signify the presence of tremors or other PD-related motor symptoms. This helps highlight the specific frequencies contributing to the signal and aids in detecting patterns or abnormalities in the frequency

domain, offering a better representation of how Parkinson's Disease affects motor functions. Table 14 contains a list of features that are calculated based on the signal for each axis and the magnitude.

Table 14 List of features extracted from inertial sensor signals

Feature	Equation/Explanation	
Time domain		
Mean	$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$	Eq. 21
Standard deviation	$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$	Eq. 22
Median	The middle value of the sorted signal samples.	
Skewness	$S = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s}\right)^3$	Eq. 23
Kurtosis	$K = \frac{n\sum_{i=1}^{n}\left(\lvert X(f_i)\rvert - \overline{\lvert X(f)\rvert}\right)^4}{\left(\sum_{i=1}^{n}\left(\lvert X(f_i)\rvert - \overline{\lvert X(f)\rvert}\right)^2\right)^2} - 3$	Eq. 24
Max	The maximum value in the signal.	
Min	The minimum value in the signal.	
Interquartile range	The difference between the 75th and 25th percentiles of the signal.	
Approximate entropy	A measure of the regularity and unpredictability of fluctuations in a time series [110].	
Sample entropy	A measure of the likelihood that similar sequences in time-series data remain similar over time [110].	
Power	$P = \frac{1}{n}\sum_{i=1}^{n} x_i^2$	Eq. 25
Absolute mean difference	$\Delta = \left\lvert \frac{2}{n}\sum_{i=1}^{n/2} x_i - \frac{2}{n}\sum_{i=n/2+1}^{n} x_i \right\rvert$	Eq. 26
Frequency domain		
Max power	Maximum power found in the PSD.	

Max power frequency	The frequency at which the maximum power occurs.	
Spectral power	$P = \frac{1}{N}\sum_{i=0}^{N-1}\lvert X(f_i)\rvert^2$	Eq. 27
Weighted mean power	$WMP = \frac{\sum_{i=0}^{N-1}\lvert X(f_i)\rvert^2 \cdot f_i}{\sum_{i=0}^{N-1} f_i}$	Eq. 28
Kurtosis	$K = \frac{N\sum_{i=1}^{N}\left(\lvert X(f_i)\rvert - \overline{\lvert X(f)\rvert}\right)^4}{\left(\sum_{i=1}^{N}\left(\lvert X(f_i)\rvert - \overline{\lvert X(f)\rvert}\right)^2\right)^2} - 3$	Eq. 29
Skewness	$S = \frac{N}{(N-1)(N-2)}\sum_{i=1}^{N}\left(\frac{\lvert X(f_i)\rvert - \overline{\lvert X(f)\rvert}}{s}\right)^3$	Eq. 30
Interquartile range	Interquartile Range of the PSD values.	
Spectral centroid	$C = \frac{\sum_{i=0}^{N-1} f_i \cdot \lvert X(f_i)\rvert}{\sum_{i=0}^{N-1}\lvert X(f_i)\rvert}$	Eq. 31

n – number of samples, x_i – i-th sample, N - number of frequency bins, f_i - frequency of the i-th bin, $X(f_i)$ - magnitude of the Fourier Transform at the i-th bin.

The features are calculated using functions from the NumPy, SciPy and PyWavelets, EntropyHub Python libraries. Additional custom features were implemented individually in Python. Time domain features were calculated for the entire signal, across all 3 axes and for the magnitude resulting in 48 features. Frequency domain features were computed for 3 previously described frequency bands and the initial signal across all axes (X, Y, Z and magnitude) yielding 128 features. To ensure the signal's characteristics are captured as accurately as possible, additional features were added.

A short-time Fourier transform (STFT) was performed with a window size of 4 seconds and a 2-second overlap. For each window, the mean PSD was calculated, and the following statistics were computed for the vector: mean, standard deviation, skewness, min, and max. This resulted in 5 additional features for each axis and each frequency band, totaling 80 features. Similarly, the raw signal was segmented into windows of this size. For each window, the value range and the entropy were calculated, as described by E. Sejdić et al. [105]. Based on these values, the previously described statistics were calculated, adding 40 more features.

Following the methodology described by Thomas et al. [32] a 3-level Discrete Wavelet Transform was applied using a Daubechies wavelet of order 10. The means and the standard deviations were calculated for first-level high-frequencies, second-level high-frequencies and third-level high-frequencies. These calculations resulted in additional 24 features.

To capture the correlations between different axes, Pearson correlation coefficients have been calculated for each axes pair (X and Y, X and Z, Y and Z), resulting in 3 features. The total number of features extracted from a single accelerometer signal is 323.

4.1.3.2. ML model training

In addition to the 323 extracted features, metadata features were incorporated, similarly to the deep learning approach. These features included one-hot encoded representations of the performed task, device type and the limb the device was worn on. The training-test split was conducted the same way as for the deep learning models, to allow straightforward comparison between deep and shallow ML models. The input data was normalized using the mean and standard deviation of the training set features, after which the training and evaluation process could be started.

The goal of this experiment is to compare performance of deep learning models and classic ML models on the MJFF dataset and to verify the validity of the defined feature set for predicting PD symptoms' severities. This can be beneficial when the methods are later applied on the MUW dataset collected using the designed mobile application. Consequently, this process focuses only on experiments with the whole dataset (Shimmer3 sensors and smartwatches) for severity classification of three symptoms and regression tasks.

Five different machine learning models from the scikit-learn Python library were selected for classification: Logistic Regression [111], Random Forest [112], Extreme Gradient Boosting (XGBoost) [112], Support Vector Machine (SVM) [111], and the previously described Multilayer Perceptron (Figure 9). These models were initialized with default parameters. However, for models that support the 'class_weight' parameter (Logistic Regression, SVM, Random Forest), the parameter was set to 'balanced'. This adjustment allows them to take into account the imbalanced nature of the dataset, paying attention to less represented classes more. For the regression task also five methods were

used. However, the logistic regression model was replaced by Linear Regression [111], because logistic regression is typically used for classification problems rather than regression.

Linear regression [111] is a fundamental ML model, it captures the relationship between a target variable and one or more input variables by fitting a linear equation to observed data. While is easy to interpret, it is sensitive to outliers and can capture only linear relationships between variables.

Logistic regression [111] is used for classification tasks and it uses a logistic function to predict the probability that the given input belongs to a specific class (Eq. 32). The model outputs probabilities, which can be used with a threshold to make a binary decision. During the training process, the model establishes the values of weights used in the equation to increase prediction accuracy.

$$\hat{p}(X_i) = \frac{1}{1 + \mathrm{e}^{-X_i w - w_0}} \qquad \text{Eq. 32}$$

Random forest [112] is an ensemble method that bases its decisions on multiple subordinates, simple ML models, such as decision trees. Each tree is trained on a random subset of the data and features, and the final prediction is made by combining the predictions of all trees. This approach reduces overfitting and improves generalization compared to a single decision tree.

XGBoost [112] is also an ensemble model that uses a collection of decision trees. However, it builds these trees sequentially, where each tree aims to correct the errors of the previous ones. This method, known as boosting, enhances the model's accuracy and robustness by focusing on the mistakes made by prior models, thereby improving predictive performance.

SVM [111] can be used both for classification and regression tasks. For classification, it tries to find the optimal hyperplane that best separates members of different classes in the feature space. SVM supports using kernel functions, which can transform the data into higher dimensions, making the separation process easier and can enable handling non-linear boundaries. The default kernel in scikit-learn is Radial Basis Function [111].

During the training process, the same metrics as for the deep learning model are calculated for the test set. For classification, these metrics include BAcc, Acc and AUC PR. For regression, the metrics include R^2, r, MAE, bMAE, bMSE.

4.1.3.3. Results

The training of all classification models was evaluated on the test set using the defined metrics. The performance of each model was measured and compared to identify which models yielded the best results in terms of symptom severity assessment. The results are presented in Table 15, with the highest metric values for each symptom classification highlighted in bold for clear identification. This comparative analysis helps in understanding which models are more effective in accurately classifying the symptoms of Parkinson's Disease.

Table 15 Classification results for PD symptoms using shallow ML models

Symptom	Model	BAcc	Acc	AUC PR
Bradykinesia	SVM	**64.3%**	83.9%	0.909
	Logistic regression	64.2%	80.2%	0.886
	Random forest	64.1%	83.8%	0.894
	XGBoost	63.7%	**88.3%**	**0.938**
	Multilayer perceptron	61.2%	86.9%	0.926
Dyskinesia	SVM	**56.0%**	76.3%	0.884
	Logistic regression	49.8%	60.9%	0.83
	Multilayer perceptron	46.8%	84.5%	0.881
	Random forest	42.8%	79.9%	0.855
	XGBoost	37.2%	**87.2%**	**0.899**
Tremor	Random forest	**70.4%**	75.5%	0.817
	Multilayer perceptron	63.9%	79.4%	0.838
	Logistic regression	62.8%	68.4%	0.795
	XGBoost	58.4%	**83.6%**	**0.88**
	SVM	57.6%	78.0%	0.857

For each symptom, the model that achieved the highest balanced accuracy was selected to generate a confusion matrix on the test set. These confusion matrices provide a detailed view of the classification performance by illustrating how well the model

distinguished between different classes. The confusion matrices for bradykinesia, dyskinesia, and tremor are presented in Figure 19.

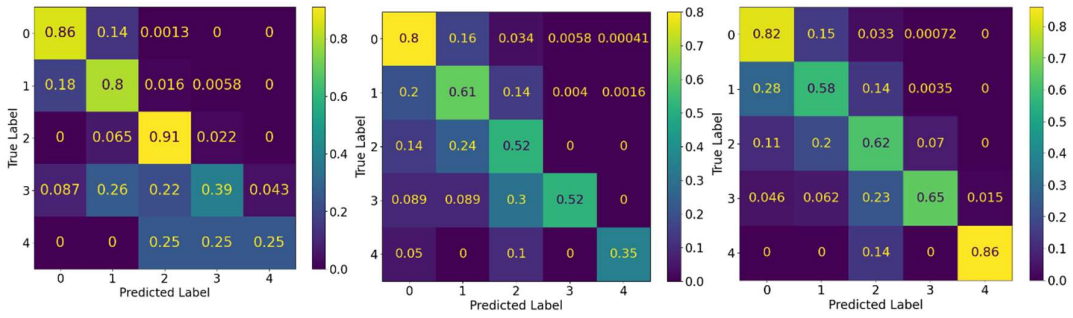

Figure 19 Confusion matrices presenting classification results for bradykinesia (left), dyskinesia (center), tremor (right)

The classification models generally performed well, achieving high values across all metrics. This validates the effectiveness of the selected feature set used to represent the signals in the training process. Although the metrics values were slightly lower than for the deep learning models, the results confirm that conventional ML models can provide comparable results when the data preparation is done properly.

For the accuracy, the XGBoost method consistently yielded the best value, verifying its robustness as an ensemble model. However, due to the dataset imbalance, it did not handle the less represented classes well. Models that allowed assigning appropriate weights to these underrepresented classes, such as SVMs and random forests, achieved better performance for these classes.

The training of regression models for each symptom was also followed by evaluation on the test set. Each model's performance was assessed using previously indicated metrics to ensure accurate predictions. The results for all models are presented in Table 16, with the best values metric values for every symptom highlighted in bold.

Table 16 Regression results for PD symptoms using shallow ML models

Symptom	Model	R^2	r	MAE	bMAE	bMSE
Bradykinesia	XGBoost	**0.653**	**0.808**	0.22	**0.591**	0.703
	Random forest	0.642	0.808	0.23	0.617	0.746
	SVM	0.62	0.79	**0.219**	0.924	1.58
	Linear regression	0.498	0.705	0.297	0.921	1.42

	MLP	0.486	0.739	0.3	0.612	**0.638**
Dyskinesia	XGBoost	**0.419**	0.673	0.277	1.01	1.68
	SVM	0.411	**0.674**	**0.297**	1.29	2.68
	Random forest	0.388	0.645	0.286	1.26	2.56
	Linear regression	0.216	0.467	0.365	1.47	3.3
	MLP	0.201	0.594	0.413	**0.871**	**1.24**
Tremor	SVM	**0.586**	**0.771**	**0.276**	1.07	2.25
	XGBoost	0.583	0.764	0.299	0.588	0.627
	Random forest	0.528	0.735	0.331	0.76	0.943
	MLP	0.446	0.722	0.383	**0.584**	**0.564**
	Linear regression	0.432	0.658	0.376	0.719	0.86

Similar to classification models, the regression models with the best metrics values, particularly those accounting for data imbalance (bMAE and bMSE), were selected to generate a more detailed overview of their performance. This is presented using violin plots, with a MAE value for each class. Figure 20 shows violin plots for bradykinesia, dyskinesia, and tremor regression.

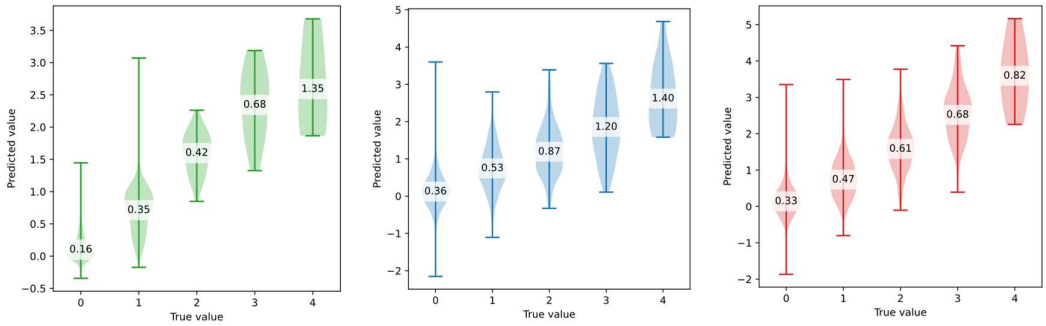

Figure 20 Violin plots presenting regression results with class specific MAE values for bradykinesia (left), dyskinesia (middle), tremor (right)

Similar to classification problems, the shallow ML models for regression performed only slightly worse than deep learning models, proving the validity of this approach. For regression, no specific methods were applied to address the dataset imbalance. As a result, while the overall performance metrics (R^2, r, and MAE) showed good values, the balanced metrics (bMAE and bMSE) were significantly higher (worse) compared to the deep learning models.

Among the shallow models, the MLP models achieved the best results for the balanced metrics. For the remaining metrics, XGBoost typically provided the best performance. However, XGBoost achieved this by yielding smaller errors for symptom severities close to 0 (the most represented), while performing worse for higher symptom severities.

4.1.4. Discussion

The findings of this part of the study confirm that machine and deep learning models are effective in detecting and measuring the presence and severity of Parkinson's disease motor symptoms, using only the data from the task performed, the device's side, and the raw accelerometer signals. This research has highlighted specific tasks that improve the accuracy of predictions, as well as those that lead to lower accuracy, suggesting which activities might be prioritized or excluded in future studies. Notably, actions such as walking and arm movements have been identified as key indicators of symptom severity, proving to be highly valuable for predictive modeling. On the other hand, tasks with little movement like sitting or standing showed the least accurate results across all symptoms, pointing to their limited relevance in assessing symptom severity.

The results presented in this section prove that it is possible to build machine learning models to predict symptom severities, whether a classification task is created to predict exact, discrete values or the model predicts continuous values using regression models. Furthermore, the newly developed model, illustrated in Figure 12, demonstrates superior classification and prediction accuracy compared to the models designed in the DREAM Challenge. The inclusion of shallow machine learning methods in this study revealed that these models also perform well, achieving high values across all metrics. While slightly lower than those for deep learning models, these results confirm that conventional ML models can provide comparable results with appropriate preparation – especially the feature extraction.

Investigating how different tasks impact prediction accuracy, especially where data samples were limited (stair climbing), underscores the need for more research to ensure a diverse and balanced representation of tasks as well as symptom severities. The low numbers of severities higher than 1, made it more difficult for the models to capture symptom-specific features and accurately predict higher severities. This section focuses on prediction of symptoms and their severities using the MJFF dataset. This is a crucial

aspect that could greatly enhance treatment personalization. The improvement in data collection and analysis methods highlighted in this study plays an important role in this research, laying the foundation for the development of advanced monitoring systems. In addition, this study verified the usefulness of known ML methods, identified those with the best properties, and highlighted their limitations related to imbalanced data.

4.2. MUW dataset

The dataset collected at the Medical University in Warsaw consists of significantly less measurements than the MJFF dataset. Furthermore, the samples come from a larger number of patients, this contributes to the higher variability in the dataset.

Due to the small number of samples and a wide range of data the conventional ML approach is applied, without the use of deep learning methods. This includes initial preparation of data such as signal filtering, feature extraction, selection, and additional transformations of the data.

4.2.1. Feature extraction

The most complex part of data preparation is the feature extraction process. These features should be tailored to the signal type and the goal of the model. Therefore, for each of the signal type, the features are discussed separately. Due to the low number of samples and low quality of voice recordings e.g., clinicians and other patients sometimes speaking in the background, it was decided to focus on sensor, writing and reaction exercises only and utilize recordings in future research.

4.2.1.1. Inertial sensors

During sensor examinations the sensors in the mobile phone as well as the wearable devices capture inertial signals, which include accelerometer and gyroscope data. These signals are rich in information about the patient's movements. The process of extracting features from accelerometer signal has already been discussed and validated for the MJFF dataset (Feature extraction p.65). For the MUW dataset the same method is applied to both accelerometer and gyroscope signal due to their similar characteristics – same length, sampling frequency and nature. This approach provides 323 features from each accelerometer/gyroscope signal registered during a single exercise performed by a patient.

4.2.1.2. Drawing and handwriting

The signal registered when the patients are drawing and writing on the screen of the mobile screen have a different nature than inertial sensor signals. The sampling frequency is not constant; a measurement is recorded every time a screen action is performed. Therefore, a timestamp is recorded for every measurement. Instead of treating the data as time series with equal time steps, another approach has been selected, based on conducted research and description of methods previously described in literature [39,40,80,82,116,117].

In this study, the movement of stylus or finger on a screen is captured along two axes – X and Y. Based on these, the magnitude (Eq. 20) is calculated, similarly to inertial sensor signals. For each of the following data series:

- X position,
- Y position,
- magnitude,
- pressure (value representing pressure applied to the screen),

three new time series have been derived: speed (Eq. 33), acceleration (Eq. 34) and jerk - the third derivative of position with respect to time (Eq. 35).

$$v_i = \frac{p_{i+1} - p_i}{t_{i+1} - t_i}$$

p_i – time series value (X_i, Y_i, $magnitude_i$ or $pressure_i$), Eq. 33

t_i – time of i-th action.

$$a_i = \frac{v_{i+1} - v_i}{t_{i+1} - t_i} \quad \text{Eq. 34}$$

$$j_i = \frac{a_{i+1} - a_i}{t_{i+1} - t_i} \quad \text{Eq. 35}$$

These were calculated based on changes in time related to every change in the value of these time series. This resulted in 12 additional series for analysis, 4 representing speed, 4 for acceleration and 4 for jerk. Each of them has been used to calculate a set of features providing a representation of the data, including features form Table 14 and Table 17.

Table 17 Additional features calculated for handwriting time series

Feature	Explanation		Applied to
5^{th} percentile	A measure that indicates the value below which a given percentage k of observations falls.		Speed, acceleration, jerk
10^{th} percentile			
20^{th} percentile			
30^{th} percentile			
90^{th} percentile			
95^{th} percentile			
5% trimmed mean	A measure calculated by removing a specific percentage of the smallest and largest values from a data set and then finding the average of the remaining values.		
10% trimmed mean			
15% trimmed mean			
25% trimmed mean			
Absolute sign changes (ASC)	The total count of changes in the sign of speed or acceleration in the time series.		Speed, acceleration
Relative sign changes (RSC)	$\mathrm{RSC} = \frac{\mathrm{ASC}}{\mathrm{Duration\ of\ Exercise}}$	Eq. 36	

The feature set included also time-related features such as:

- duration of exercise,
- ratio of the writing time to the time in air,
- ratio of time in air to total exercise time,
- ratio of the writing time to total exercise time,
- total time in air,
- total time of writing,
- mean and standard deviation of times for continuous writing.

In the conducted experiments and performed literature review, these features were found to provide a good overview of patient condition.

4.2.1.3. Reaction exercises

While there have been many studies regarding the analysis of inertial sensors signals and of handwriting regarding the assessment of PD patients, there is significantly less research regarding the defined reaction exercises performed using a screen of a mobile device. The most studies exercises presented in the app include finger tapping. However, other studies often used other sources of data. Researchers used cameras and computer vision algorithms to detect the tapping [53], accelerometers and touch sensors placed on fingers [115] and also motion analyzers [116]. Some research was performed regarding tapping the touch screen of smartphone devices [52,117,118], being the closest to the exercises defined in the mobile application.

After experiments and reviewing the literature a set of features was established to capture the performance of the patient. Firstly, the time differences between consecutive clicks were calculated, and similarly, the changes in the pressure applied to the screen – resulting in two new data series. As previously mentioned, for every click the distance from the center of the square was registered using the X and Y coordinates. These were then used to calculate the magnitude of the distance. For the six data series (time differences, pressure, pressure differences, X distances, Y distances and magnitude distances) features have been calculated. These included some of the features previously mentioned in Table 14 and Table 17: mean, median, standard deviation, interquartile range, kurtosis, skewness, sample entropy, approximate entropy, percentiles: 5^{th} , 10^{th} , 20^{th} , 30^{th} , 90^{th} , 95^{th} , trimmed means: 5%, 10%, 15%, 25%.

Additionally, features related to the correctness of actions were considered, which lead to construction of four features: number of correct clicks, number of incorrect clicks, the mean and standard deviation of number of consecutive correct clicks. The correctness of the click depends on the exercise performed:

- For the first exercise (clicking a square) all clicks are considered correct.
- For the second exercise (clicking the highlighted square), clicking the highlighted one is correct, while clicking others is considered incorrect.
- For third and fourth exercises (sorting numbers), clicking the square with the next value is considered correct.
- For the fifth exercise (alternating clicking of two squares), the action is considered correct if the click was on a different square than last click; if

the user clicks the same square twice – the second action is considered incorrect.

This collection result in 112 features derived from every reaction exercise performed by the patient.

4.2.2. Examination metadata

To build appropriate models for predicting patient state, alongside the features extracted from the collected sensor data additional features are added that can improve the quality of the model and its prediction precision. These features are patient characteristics equal among all examinations of that patient as well as characteristics of specific examinations. The full list has been showcased in Table 18.

Table 18 Features created from patient and examination metadata

Name	Description	Source
Affected side	The side of the body more affected by the disease	Patient
Handedness	The dominant hand of the patient	Patient
Groups	Belonging to groups (disease, treatment method)	Patient
Diagnosis	Time since diagnosis to execution of examination	Patient + Exam
Age	Age during examination	Patient + Exam

For categorical features, such as Affected side, Handedness and Groups, one-hot encoding was performed, to ensure correct interpretation of the values by ML models. The remaining features are normalized along with sensor derived features, by subtracting the mean and dividing by standard deviation.

4.2.3. Feature selection

A single examination performed by a patient can consist of 4 types of exercises, with a maximum number of 26 exercises, when both hands are considered. Considering the number of sensors and extracted features, one examination can provide thousands of features, a number that can be easily higher than the number of patients and even the number of total examinations performed. As stated by Guyon and Elisseeff [119] in these situations it is important to consider feature selection methods. These can reduce the number of dimensions and therefore make it easier for ML model to learn the dependencies in data, as well as perform the training process faster.

The process to restrict, the number of features has been performed in two steps. The first step focuses on removing the variables that are highly correlated with each other. Having duplicates features does not improve the performance of ML models, but only slows down the process. Therefore, the Pearson's correlation coefficient (Eq. 16) has been calculated for every pair of features in question. Whenever there was a correlation value above 0.97 between two features, these were excluded from further analysis.

The second step in the reduction of feature dimensions can be performed in one of 3 ways, for each training the solution that provided best results was selected. The first algorithm is Linear Discriminant Analysis (LDA) [120]. It is a method designed to maximize the separation between multiple classes by projecting the data onto a lower-dimensional space. It tries to find a linear combination of features that best separates the classes by maximizing the ratio between the between-class and within-class variance.

An alternative approach designed to reduce the number of features is Principal Component Analysis (PCA) [120]. PCA transforms the original features into a new set of uncorrelated features called principal components, which capture the most variance in the data. By focusing on the directions of maximum variance, PCA reduces the dimensionality of the dataset, simplifying the analysis while retaining essential information. Unlike LDA, PCA does not require class labels to be applied.

The last approach considered for reducing the dimensions of feature space is using scikit-learning's SelectFromModel [121]. This method involves training a model that assigns importance scores to each feature, and based on that selects the best features for the executed ML task using the scores. SelectFromModel can significantly enhance the efficiency and accuracy of machine learning models. In this case, the decision trees were selected to perform the feature selection. They are simple and fast to train, making the selection process not bothersome and easy to apply.

4.2.4. Individual symptom evaluation

The features that have been described for the signals collected from exercises completed by patients create a representation of the patient's condition during the examination. The features provide different aspects of the examination performance and might be important in identifying specific symptoms of PD. At the end of examinations conducted in the presence of the clinician, a state assessment screen is displayed where the overall state evaluation is provided along with individual symptoms, including:

tremor, bradykinesia, muscle stiffness, and dyskinesia. The clinician is asked to evaluate their severity on a scale of 0 (not present) to 4 (very severe). While this evaluation has not been provided in all of the examinations for PD patients, 356 of the patient examinations contain these evaluations. This section focuses on building ML models capable of predicting individual symptom severities (as evaluated by clinicians) based on exercise-derived features.

In this dataset, the problem of imbalance is significant and even more visible than in the MJFF dataset. The total number of samples is lower, and higher symptom severities are poorly represented. For example, there is only one sample for dyskinesia severity of 4, making it impossible to train and evaluate the model for this severity. Other symptoms have better representation, with the most balanced dataset being for tremor prediction – 10 samples for severity of 4. The class distribution for all symptoms (tremor, bradykinesia, muscle stiffness and dyskinesia) is shown in Figure 21.

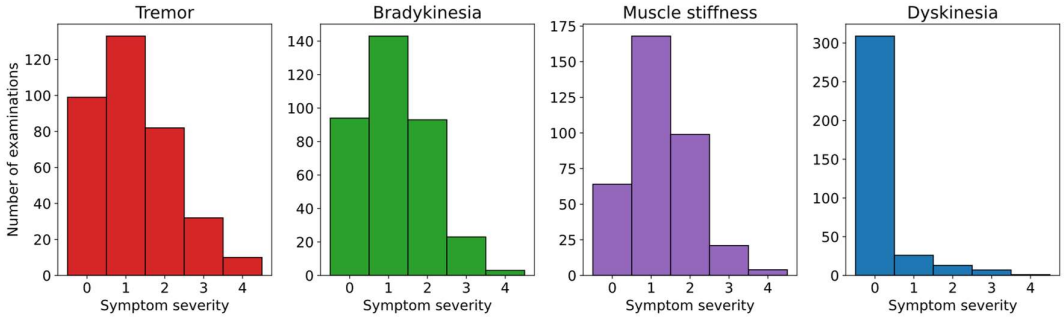

Figure 21 Histogram presenting the distribution of symptom severities for MUW dataset

To compare these results with publicly available studies and results achieved with the MJFF dataset, both classification and regression models can be used to approach that problem. However, due to the limited representation of higher severities, only results for regression are showcased for this dataset, as the balanced metrics for classification are unsatisfactory.

The evaluation metrics used to evaluate the models are the same as for MJFF: R^2, r, MAE, bMAE, bMSE. Similarly, the set of ML models employed for training includes Linear Regression (LR), Random Forest (RF), SVM, XGBoost and MLP. The main difference is in the train-test split methodology.

The MJFF dataset had significantly more data, which allowed for effective performance even when split into training and testing sets. However, due to the smaller

number of examinations in the MUW dataset, a use of different approach was necessary to make sure enough samples were in the training and testing sets. Cross-validation [122] was employed, a technique where the data is randomly split into k disjoint sets. The training process is then performed and evaluated k times, with k-1 subsets used for the training and the remaining subset used for evaluation. This process is repeated k times, ensuring that every subset has been treated as the test set exactly once.

In the simplest version of cross-validation, the split into subsets is performed randomly. However, there are more advanced versions that can be used for specific scenarios. For example, stratified k-fold cross-validation is often used for classification problems. In this method, the partitioning is done so that the distribution of class samples in different subsets is similar.

Another approach, commonly used in medical applications is group k-fold cross-validation [122], which involves assigning groups to specific samples, ensuring that all samples from the same group end up in the same subset. It can be especially useful when data from multiple patients is available, as it prevents data from the same patient from appearing in both the training and testing set, allowing the model's performance to be evaluated on entirely new patients.

Additionally, leave-one-out (LOO) cross-validation [122] can be used. It can be performed either on individual samples or on groups. When performed on samples, each subset contains only one sample. When performed on groups, the number of subsets is equal to the number of groups, with each model evaluated on one group while being trained on the remaining groups.

All of the splits are included in the scikit-learn Python library in the form of the following classes: KFold, StratifiedKFold, GroupKFold, LeaveOneOut and LeaveOneGroupOut, which are used to perform the training process in this part of the study.

During the training process, numerous training processes are executed, they can be grouped into two groups based on the expected goal of the training:

- single exercise – finding which exercise is best at capturing each of the symptoms,

- exercise type – finding which set of exercises (grouped by type) is best at capturing specific symptoms.

Each experiment is performed using all of the previously defined models (ML model training p. 68). For single exercises, no algorithms are used for reducing the dimensions of the feature space; however, for experiments regarding more than one exercise, three of the described methods are used. Each of the experiments is validated using cross-validation with two different splits: 10-fold split (10F) and leave one patient out (LOO) – to see how models perform in these different situations.

4.2.4.1. Results

The goal of the first training process was to perform the training on data from single exercises and single signals from sensors. For each of the training processes all of the suitable discussed ML models were first applied and then set up to solve the regression problem of evaluation symptom severities for tremor, bradykinesia, muscle stiffness and dyskinesia. Each model was evaluated using metrics and the models that performed the best (provided the highest R^2 score) are listed in Table 19 for each of the symptoms. The table showcases 2 models for each symptom, one for the 10-fold cross-validation and the second for leave-one-patient-out cross-validation. The exercise numbers are assigned as presented in Table 5. Comparison of these models helps to notice the impact of individual patient symptom characteristics.

Table 19 Training results for models predicting symptom severities based sensor signals for single exercises

S	Split	Model	Sensor	Ex	R^2	r	MAE	bMAE	bMSE
T	10F	RF	Phone, GYR	1	0.527	0.728	0.585	0.768	0.821
	LOO	RF	Phone, GYR	1	0.501	0.708	0.593	0.775	0.526
B	10F	SVM	Phone, GYR	3	0.229	0.483	0.633	1.12	1.84
	LOO	SVM	MYO, ACC	3	0.184	0.434	0.660	1.17	1.98
S	10F	SVM	MYO, ACC	3	0.176	0.422	0.604	1.16	1.98
	LOO	RF	MYO, ACC	3	0.166	0.408	0.617	1.16	1.94
D	10F	SVM	Phone, GYR	1	0.167	0.443	0.286	1.70	4.48
	LOO	SVM	Phone, GYR	1	0.149	0.403	0.294	1.70	4.40

S – symptom, T – tremor, B – bradykinesia, S – stiffness, D – dyskinesia, Ex – exercise, GYR – Gyroscope, ACC – Accelerometer

For the best-performing models for each symptom (using 10-fold cross-validation), the results are presented in the form of violin plots to make it easier to inspect how the models handle different severity levels. Figure 22 presents viol plots for tremor and bradykinesia, while Figure 23 for muscle stiffness and dyskinesia.

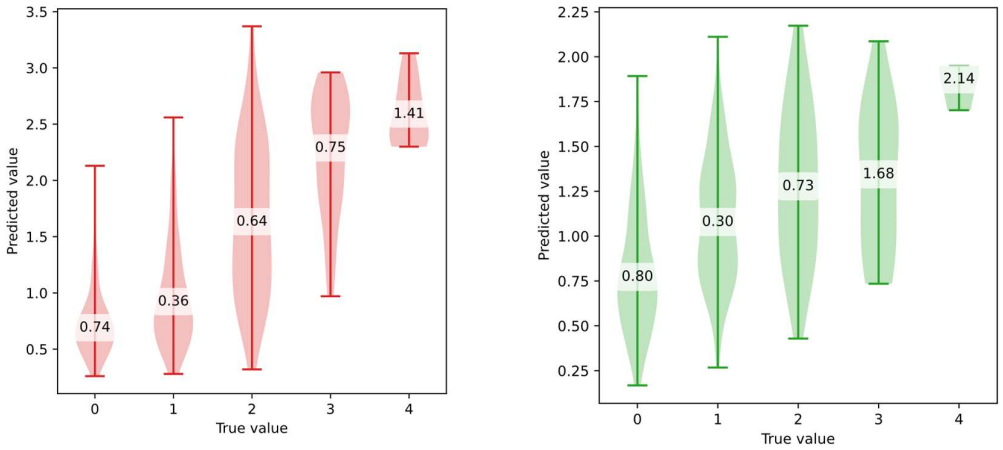

Figure 22 Violin plots presenting regression results with class-specific MAE values for tremor (left) and bradykinesia (right) using best-performing models evaluating based on a single exercise sensor signal

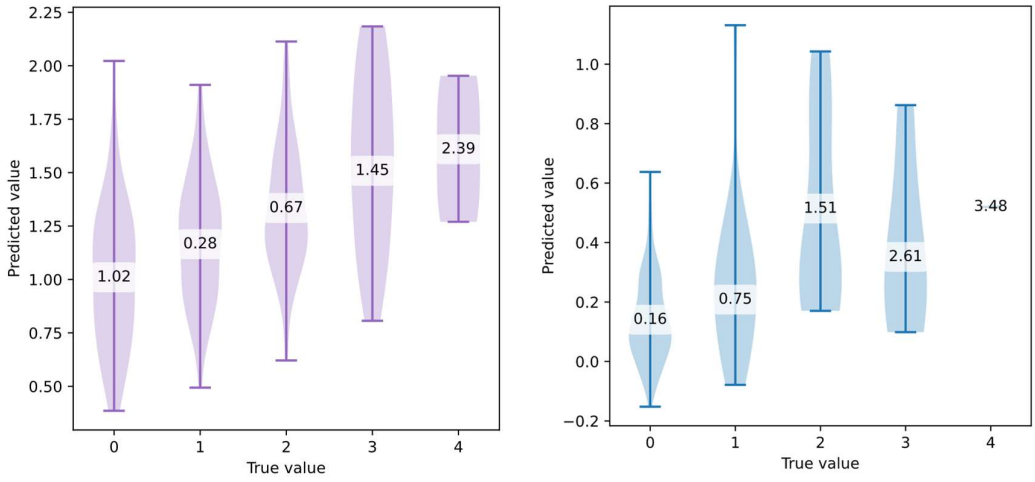

Figure 23 Violin plots presenting regression results with class-specific MAE values for muscle stiffness (left) and dyskinesia (right) using best-performing models evaluating based on a single exercise sensor signal

The training for the tremor has provided the best results, the metrics values are, just slightly worse than for the MJFF dataset, even though the dataset is significantly smaller. As expected, the performance for the higher severities is low, due to the class imbalance and low representation. The models for bradykinesia and muscle stiffness provide similar results, with bradykinesia achieving slightly better metrics values. These models provided worse performance than tremor, and worse than expected from the MJFF dataset study. This could be attributed to the worse distribution of severity values and the difficulty in assigning the labels, which is more difficult than for tremor and can lead to inconsistencies in the clinical evaluation process. The model for dyskinesia performed even worse. To achieve better results, more examinations should be performed on patients with dyskinesia. To do that they could be forced by excessive doses of medication as in a study by Thomas et al. [32].

The results from Table 19 prove that the sensor examinations performed by the patient using mobile phones and sensor armbands provided the best results for predicting the severities. These exercises outperformed the reaction and handwriting exercises. For tremor and dyskinesia, the best exercise was the first one, focused on rest tremor, when the patient keeps their hands on knees or a vertical platform, for 30 seconds. These symptoms are recognized by movements and can be better detected when the patient is not performing any voluntary movements that could interfere with the symptom or make the detection process more difficult. For bradykinesia and muscle stiffness, the most useful was the third sensor exercise, when the patients were performing the pronation-supination movement for 30 seconds. Since these two symptoms affect the mobility of patients, it is best to observe them when movements are quick, and their range is large.

Due to the large number of patients, the differences between classic 10-fold cross-validation and leave-one-patient-out cross-validation are small. This means that the model generalizes well and does not focus that much on patient-specific symptom characteristics. This is the advantage of this dataset, and these models can be applicable in the assessment of state of other patients.

The goal of the next set of training experiments is to verify how the ML models trained on different types of exercises perform. Since the results from Table 19 verified that the differences between 10F and LOO are small, Table 20 presents only the results for 10F.

Table 20 Training results for models predicting symptom severities based on sensor signals for exercises group by types

Symptom	Model	Exercise type	R^2	r	MAE	bMAE	bMSE
Tremor	RF	Sensor	0.532	0.732	0.581	0.742	0.811
	RF	Handwriting	0.227	0.489	0.740	1.09	1.68
	RF	Reaction	0.115	0.339	0.780	1.18	1.96
Bradykinesia	SVM	Sensor	0.240	0.498	0.629	1.11	1.80
	SVM	Handwriting	0.189	0.437	0.650	1.20	2.19
	SVM	Reaction	0.155	0.396	0.665	1.22	2.16
Stiffness	RF	Sensor	0.183	0.429	0.603	1.12	1.83
	RF	Handwriting	0.156	0.398	0.601	1.11	1.69
	RF	Reaction	0.128	0.361	0.623	1.18	1.95
Dyskinesia	LR	Sensor	0.212	0.488	0.370	1.28	2.38
	RF	Handwriting	0.104	0.346	0.349	1.63	4.24
	SVM	Reaction	0.0617	0.255	0.333	1.82	5.04

The results in Table 20 show how well the ML models can perform for predicting symptom severities when more than one exercise is considered. The models were trained on features generated from signals registered during all exercises of the specific type. An improvement can be seen for every symptom if more than one exercise is used for prediction. For some symptoms such as tremor, the improvement is modest. For dyskinesia, the improvement is significant. Providing more data makes it easier to capture specific features, especially those that do not manifest continuously. However, expanding the feature space, especially when limited data is available, makes it more difficult for models to find which features and how impact the severity of the investigated symptom. To reduce the impact of high dimensionality, the described methods were used. The best results that ended up in the table were acquired only with the SelectFromModel approach or when no method was applied. As previously discussed, during the trials the scope of the dataset was expanded, which resulted in missing data in older examinations. This led to less samples in this experiment, where more exercises were used (only the examinations that had all of the required exercises were included in the dataset). Probably,

the results could be even better if more samples were available, and the improvement gained from including multiple exercises would be greater.

The results in Table 20 provide also a comparison between different exercise types, it can help investigate which types are best for each symptom. As suspected from the results in Table 19, the best results are achieved using sensor exercises – for every symptom. Using reaction exercises resulted in the worst performing models. This can be used to create further recommendations regarding the list of performed examinations, for example by restricting the number of reaction and handwriting exercises, which not only performed worse than sensor exercises, but take significantly more time, require more attention from the patient, and are impossible to apply in passive monitoring of patients.

4.2.5. Overall state evaluation

The comprehensive assessment of a patient's overall state plays an important role in understanding the nature of PD. While detailed evaluations of specific symptoms offer valuable insights into the disease's characteristics, severity, and symptom manifestations, they may not fully encompass the impact on a patient's quality of life and daily functioning. To address this, the MDS-UPDRS [76] provides a foundational framework for a more inclusive evaluation. In an effort to simplify the case and represent the therapeutic effect of medication, Westin et al. [57] proposed the TRS scale, optimizing it to capture the spectrum of patient experiences from severe symptoms to severe dyskinesia, with 0 being the optimal state. The TRS used in this study ranges from -4 to +4, as presented in Figure 24. It has been adjusted to allow clinicians to gauge the overall state of PD patients more effectively. Such comprehensive assessment is crucial for monitoring disease progression and customizing treatment plans to align with the dynamic needs of each patient, thereby enhancing therapeutic outcomes and patient well-being.

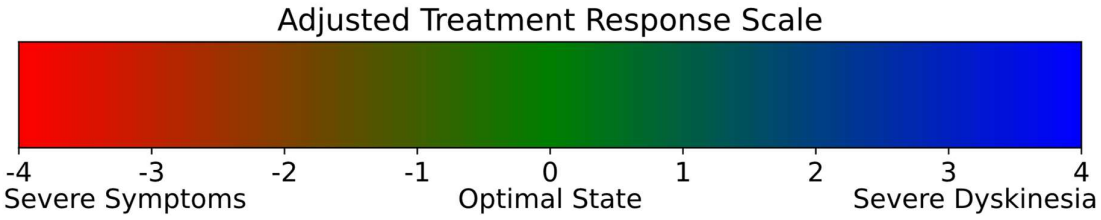

Figure 24 Value range of the adjusted TRS scale

In this section, the focus is on the development of machine learning models capable of predicting the adjusted TRS scale values. The predictions are based on a set of data collected during patient evaluations. These include sensor exercises, screen interactions, handwriting, and vocal exercises. By analyzing a diverse collection of examination data, the models aim to achieve a more accurate and personalized understanding of patient conditions. This approach is designed to enhance the precision of treatment plans, tailoring interventions to meet the unique needs of individuals with PD.

The goal of training ML models is to evaluate the patient state, during examinations. The ground truth values for this were provided both by the patient – their subjective opinion and by their clinician – hopefully, more objective. As for the individual symptom severities, this dataset has also been affected by an imbalance in the label values. Furthermore, the range of values is more than twice as big and the precision is higher, which is shown in Figure 25 along with the number of examinations that had each of the labels assigned. This makes it more difficult to prepare a model which performs well in the range of values.

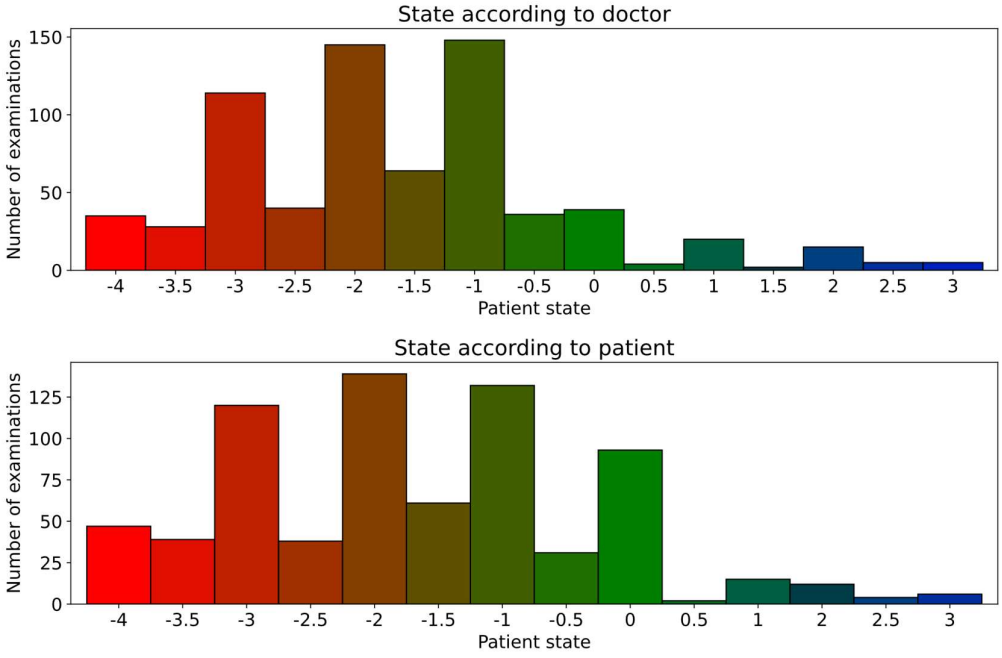

Figure 25 The distribution of label values representing the patient state evaluated by the clinician (top) and by the patient (bottom)

The process to build ML models for predicting the state of the patient is similar to the prediction of specific symptom severities. Regression models are built using sensor

signals registered from different exercises, similarly the 10-fold and leave-one-patient-out cross validation is performed and the previously described metrics (R^2, r, MAE, bMAE, bMSE) are used to evaluate the models. The main difference is the scope of experiments. The results acquired with models for individual symptoms provided some feedback regarding which exercises performed better and which performed worse for capturing different aspects of PD. This information is used for constructing the input data for evaluating the state of the patient. Therefore, the following exercise sets are used for performing experiments (exercises numbered according to Table 5):

- sensor exercises: #1 and #3 (provided best results for each symptom) (SEN:#1,#3),
- all exercises (ALL),
- exercises completed during a short examination (SHORT):
    - sensor exercises: #1 and #3,
    - reaction exercises: #2 and #5,
    - handwriting exercises: #1,
- handwriting exercises: #1 and #2 (HAND:#1,#2),

as presented in MUW dataset description p. 33.

The exercises are conducted using 2 approaches to dimensionality reduction: no reduction and SelectFromModel, which reduces the number of features to 200. Due to the bigger number of possible values than for the symptom severities, the results are presented in a form of scatterplot instead of violin plot. It is used to present machine learning regression results by plotting the true values on the x-axis and the predicted values on the y-axis, allowing for the assessment of the model's performance.

4.2.5.1. Results

The experiments for building ML models to evaluate patient state were run multiple times using different ML methods, different dataset split configurations and with various exercise sets. The goal was to evaluate the examinations to achieve labels similar to those provided by the clinician or by the patient. The results for these experiments are provided in Table 21, where the metrics values are provided acquired through the cross-validation process.

Table 21 Training results for models predicting patient state based on different types of signal input data

Rater	Split	Input data	Model	R^2	r	MAE	bMAE	bMSE
Clinician	10F	SEN:#1, #3	RF	**0.265**	**0.518**	0.781	1.71	4.73
	LOO		RF	0.242	0.492	0.902	**1.69**	**4.43**
	10F	HAND:#1,#2	RF	0.162	0.410	0.918	1.85	5.24
	LOO		RF	0.144	0.382	0.936	1.86	5.35
	10F	ALL	SVM	0.261	0.511	**0.746**	1.75	5.20
	LOO		SVM	0.224	0.476	0.774	1.77	5.30
	10F	SHORT	RF	0.260	0.515	0.781	1.71	4.68
	LOO		RF	0.242	0.495	0.793	1.73	4.73
Patient	10F	SEN:#1, #3	RF	0.267	0.521	0.811	1.63	4.53
	LOO		RF	0.221	0.471	0.844	1.66	4.70
	10F	HAND:#1,#2	RF	0.149	0.387	0.988	1.79	5.22
	LOO		RF	0.103	0.321	1.01	1.85	5.52
	10F	ALL	SVM	0.249	0.501	**0.796**	1.65	4.91
	LOO		SVM	0.181	0.434	0.850	1.72	5.15
	10F	SHORT	RF	**0.278**	**0.533**	0.798	**1.63**	**4.53**
	LOO		RF	0.223	0.473	0.831	1.69	4.85

The models for predicting overall patient state provided metrics values that were between those for individual symptoms, which was expected since the overall state evaluation incorporates these individual symptoms. Good results were obtained from combining sensor exercises: holding hands on a flat surface and the pronation-supination task. Adding more exercises did not significantly improve the performance of the model. The differences between the 10-fold and leave-one-patient-out cross-validation were minimal.

Interestingly, the best model for predicting patient state, according to patient's own assessment, provided better results for R^2 and r metrics than the models based on clinician's assessment. When compared to other studies [32,123] the correlation coefficient (r) values in this study for the leave-one-patient-out split were at a slightly higher level for the RF (0.42) and SVM (0.49) models [32]. Buvarp et al. tested this model, trained on data from [32] on a new dataset [123] and achieved a correlation

coefficient of 0.23. Applying this to the MUW dataset (accelerometer and gyroscope signals from MYO armband) resulted in a slightly higher value of 0.24.

To fully understand the ML models' performance, the results of models with the highest R^2 scores for predicting the score according to the clinician and according to the patient were visualized in a form of scatterplots in Figure 26.

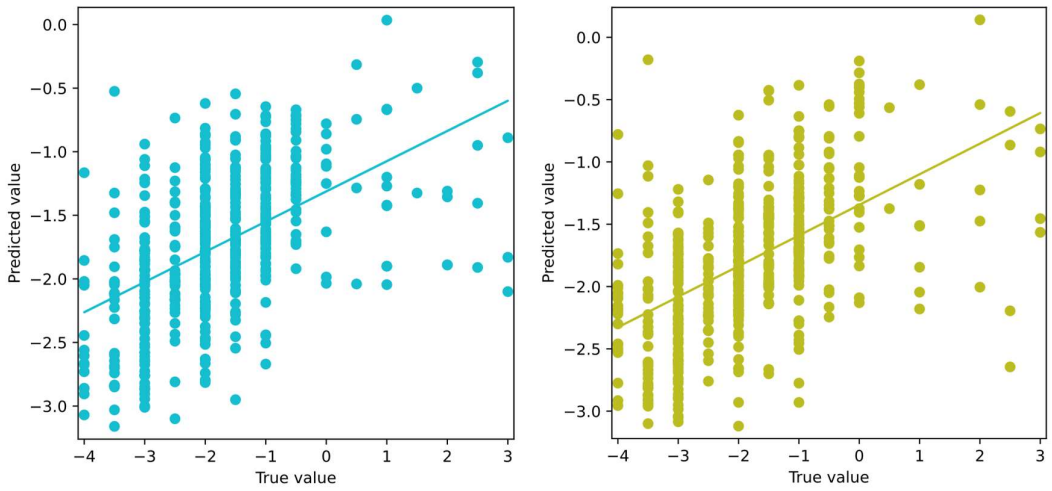

Figure 26 Scatterplots presenting the results for patient state prediction according to clinician (left) and according to patient (right)

The scatterplots for both model types show that the models perform better for negative state values (patients experiencing symptoms) than for positive values (patients experiencing dyskinesias). When the models were restricted to values ranging from -4 to 0, performance improved, with a correlation coefficient of approximately 0.65 between true and predicted values. This is again caused by the drastic imbalance in the dataset. The imbalance is not the only problem, the low number of samples is another. With more samples, methods to handle imbalanced datasets could be employed, potentially reducing the impact of imbalance. However, applying such methods with the few examinations capturing dyskinesias did not yield significant improvements.

4.2.6. Discussion

The experiments in this section focused on building machine learning models to predict both the individual severities of symptoms and the overall state of patients related to PD, as assessed by clinicians and patients. Due to the limited number of samples and imbalanced dataset, traditional machine learning models were used.

The performance of the models trained to evaluate symptom severity varied depending on the symptom, with the best results achieved for tremor. The metrics for tremor were close to those obtained with the MJFF dataset. However, the results were worse for other symptoms: bradykinesia, stiffness, and dyskinesia, for which the results were the worst. The limited number of samples (356) and the high imbalance in the dataset likely contributed to these poorer results.

Experiments with different exercise sets revealed that the sensor exercises performed with a smartphone and an armband were the most effective for evaluating individual symptom severities. Specifically:

- The first exercise (holding hands on a flat surface) was best for evaluating tremor and dyskinesia.
- The third exercise (performing pronation-supination movements) was best for evaluating bradykinesia and muscle stiffness.

Other exercises provided less satisfactory results. The better performance of models trained on the MJFF dataset compared to those trained on the MUW dataset is understandable given the larger size of the MJFF dataset, the evaluation process was conducted by a group of experienced clinicians, evaluations were provided for specific limbs, not like in the MUW dataset where the labels were representing the symptoms for the whole body.

The results for overall patient state evaluation were slightly better than results in other studies. However, the performance was still limited by the imbalance and small size of the dataset. The worst performance was observed for positive states representing dyskinesias. The number of patients with dyskinesias was very low and it was challenging for the model to learn how to detect and evaluate this condition, resulting in most prediction values falling in the range of -4 to 0. This range does not fully capture the full of patient states. Despite the limitations, the models could still provide a good approximation of the patient state, particularly for capturing symptoms such as tremor, bradykinesia, and muscle stiffness.

4.3. Conclusions

In this chapter, models to evaluate patient states using data from both the MJFF and MUW datasets were discussed. The analysis covered various aspects, including

individual symptom evaluation, the application of deep learning models, and conventional machine learning approaches.

The experiments confirm that machine and deep learning models are effective in detecting and measuring the presence and severity of Parkinson's disease motor symptoms using meta data and sensor signals such as accelerometer and gyroscope signals, which are registered using inertial sensors integrated into the developed mobile application.

Experiments with the MJFF dataset demonstrated the superior performance of deep learning models over conventional ML models. However, conventional ML models still achieved high metric values, confirming their usability with appropriate feature extraction. Challenges were noted with imbalanced data, particularly with higher severity levels, highlighting the need for more diverse and balanced task representation.

The MUW dataset analysis revealed variations in model performance based on the symptom, with the best results for tremor. Specific sensor exercises, such as holding hands on a flat surface for tremor and dyskinesia, and pronation-supination movements for bradykinesia and muscle stiffness, were identified as most effective. The limited and imbalanced dataset posed challenges, especially for symptoms like dyskinesias.

Based on these experiments with both the MJFF and MUW datasets, the following recommendations are proposed for further development of methods for assessing patient state:

- Restrict the number of exercises to make the examination shorter, focusing only on those that provided best results (sensor exercises).
- Conduct experiments where data is collected passively in the background to capture more data and reduce the burden on patients.
- Increase the number of examinations on patients in the advanced phase of PD to capture symptoms of higher severities, particularly dyskinesias.
- Consider administering higher doses of medication to patients to induce dyskinesias, as has been done in other studies.
- Ensure more examinations have a consistent scope to reduce the impact of missing data.

- Improve the quality of labels by involving more specialists in the labelling process and verifying labels based on video recordings.

Following these recommendations could significantly improve the quality of models trained to evaluate PD patient states. Future studies will further explore easier-to-implement steps, such as passive data collection, to enhance model performance and patient state assessment.

5. Medicine response model

The treatment of PD relies heavily on the administration of medication sever times a day. Therefore, understanding and predicting a patient's response to medication is necessary for providing appropriate and effective treatment. This chapter focuses on the development of ML models designed to predict patient responses to medication, specifically levodopa. Since individual responses to medication can vary significantly due to disease progression and patient-specific parameters, considering these factors is essential for optimizing treatment schedules.

The chapter begins by discussing PK/PD models, which describe how the body processes levodopa and its effects. These models are used to generate potential patient responses to medication. Following this, the architectures of ML models trained to predict the individual patient responses to medication are presented, aiming to reduce the need for invasive tests and increase the flexibility of medication schedules. While these models are mostly retrained individually for every patient, a variant of a general medicine response prediction model is also introduced, capable of individualization using patient demographic data and results from common PD scales.

These models are explored using two datasets:

- simulated patients with responses to medication generated using PK/PD,
- real patients, combining real and simulated responses to medication using PK/PD models.

5.1. PK/PD model for levodopa

Every medication is presumed to have an effect on the patient's condition. The impact can differ between patients and their characteristics. Patients in the early stages of the disease can exhibit a good response to medication, while patients in the later stages can be barely affected by it. Therefore, prescribing medication is a difficult task and requires the clinicians to gain extensive knowledge about various drugs, their effects and interactions.

The most commonly used drug in PD is levodopa, therefore there has been numerous studies regarding its pharmacokinetics and pharmacodynamics [61,124,125], including research with additional medication such as aromatic L-amino acid decarboxylase (AADC) and catechol-O-methyltransferase (COMT) inhibitors. In a study

by Westin et al. [62] a pharmacokinetic-pharmacodynamic (PK/PD) model has been defined, which is capable of predicting the patient's state after medicine doses using the TRS scale. In Thomas et al. [64] it has been adapted for oral medication of levodopa-carbidopa tablets. The goal of the model is to describe and predict the dynamics of drug absorption, distribution, metabolism, and excretion (pharmacokinetics) along with the drug's effects on the body (pharmacodynamics). It is a two compartment model, the central compartment represents the bloodstream, where levodopa concentration peaks shortly after oral administration, reflecting rapid drug absorption. The "peripheral" compartment encompasses less accessible tissues and organs, capturing the drug's slower distribution and essential action in crossing the blood-brain barrier to alleviate PD symptoms. This distinction is crucial for understanding levodopa's pharmacokinetics and its direct pharmacodynamic effects within the brain. The structure of this PK/PD model is presented in Figure 27.

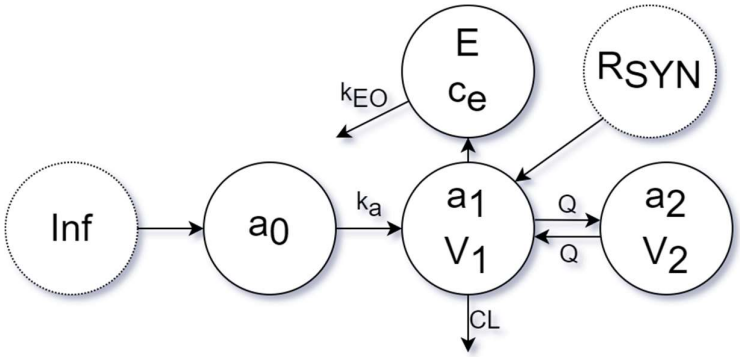

Figure 27 Structure of the PK/PD model for levodopa [62]

The model consist of 5 equations, 3 of them (Eq. 37, Eq. 38 and Eq. 39) are differential equations that focus on the pharmacokinetics of the drug. In these equations a_n represents the drug amount (mg) in the n-th compartment. The remaining two equations (Eq. 40 and Eq. 41) represent the pharmacodynamics, the first one is a differential equation describing the concentration of levodopa in the effect compartment, represented by c_e (mg/L), and the second one is used to calculate the effect (E) - patient state represented in the TRS scale (between -3 and 3).

$$\frac{da_0}{dt} = Inf - k_a \cdot a_0 \qquad \text{Eq. 37}$$

$$\frac{da_1}{dt} = BIO \cdot k_a \cdot a_0 - \left(\frac{Q + CL}{V_1}\right) \cdot a_1 + \left(\frac{Q}{V_2}\right) \cdot a_2 + R_{syn} \qquad \text{Eq. 38}$$

$$\frac{da_2}{dt} = \left(\frac{Q}{V_1}\right) \cdot a_1 - \left(\frac{Q}{V_2}\right) \cdot a_2 \quad \text{Eq. 39}$$

$$\frac{dc_e}{dt} = kEO \cdot \left(\frac{a_1}{V_1} - c_e\right) \quad \text{Eq. 40}$$

$$E = BASE + \frac{E_{MAX}\, c_e^{\gamma}}{c_e^{\gamma} + EC50^{\gamma}} \quad \text{Eq. 41}$$

The parameters of equation these equations (Eq. 37-41) are patient specific. Their meaning and population means are presented in Table 22.

Table 22 PK/PD model parameters with description population means [62]

Symbol	Description	Population mean
Inf	Infusion rate (mg/min), represents the dose size	-
k_a	Absorption rate (1/min)	0.035
BIO	Bioavailability	0.88
Q	Intercompartmental clearance (L/min)	0.58
V_1, V_2	Volume in first/second compartment (L)	11, 27
CL	Clearance rate (L/min)	0.52
R_{syn}	Endogenous levodopa synthesis rate (mg/min)	0.01
kEO	Effect rate (1/min)	0.048
BASE	Baseline effect – lowest effect value	-1.58
E_{MAX}	Maximum change from baseline effect	2.39
EC50	Concentration at 50% effect (mg/L)	1.55
γ	Hill coefficient - quantifies how steeply the response changes with increasing medicine concentration	11.6

PK/PD model, personalized for patients using parameters from Table 22 was used to find the optimal infusion rates of levodopa for PD patients [63] and was later combined with sensor output from the pronation-supination task to create medicine intake schedules for oral levodopa intake [64]. They consisted of two dose sizes (morning and maintenance) and an equal time interval between doses. Both of these methods yielded positive results and demonstrated that computer science can be utilized to improve treatment of PD.

However, these approaches required the performance of invasive tests – collecting blood samples, in order to estimate the model parameters. Furthermore, the predefined

model was very restricting, as it was created with a constant number of parameters and could handle only one medicine. Treatment of PD usually includes more than one drug (polytherapy). The schedules generated in the study considered only two different doses sizes, one taken in the morning and equal dose sizes during the day and only one time interval for doses. While this might be convenient for the patient, advanced stages of PD might require more flexibility in the dose sizes and time intervals between them. Lastly, their method did not allow for any updates to the schedule, it was created once and if it did not fulfill patient needs, it was not possible to improve it.

With the expanding capabilities of machine learning algorithms, they are being applied in new areas, especially in medicine. This has led to the idea of using ML methods to model the patient's response to medication, to train them to do the task of the described PK/PD models and to reduce the need for invasive tests and examinations, as well as allow more flexibility. The goal of such ML models would be to predict patient future condition under the influence of specific medicine doses. The idea is to train the models based on previously captured responses to different medicine doses. After training the model would be able to infer responses to other doses.

5.2. Simulated patients

Before testing such a hypothesis on real patients, it is a good idea to perform preliminary experiments using simulated patients. In a publication by Thomas et al. [63] a method was proposed and validated for generating patients, represented by PK/PD model parameters. The approach involved calculating population means and a covariance matrix for all the parameters of the model using the population characteristics of the patients. This definition allowed for simulating patients – generating them using means and the matrix. After contacting the authors of the publication, it was possible to access the data – the population means and a covariance matrix for the following parameters: k_a, V_1, V_2, CL, Q, R_{syn}, kEO, BASE, E_{MAX}, EC50, γ.

To generate values for individual patients the covariance matrix, which captures the variance and covariance between the PK/PD parameters was utilized. Using this matrix and the population means, sets of parameter values for individual patients using a multivariate normal distribution were generated.

Based on these generated parameters, it was possible to simulate potential reactions to medicine doses using implemented PK/PD models. This method was used to

generate the characteristics (PK/PD parameters) for 50 PD patients, indicating their individual responses to medication. For every generated patient, data spanning 3 days, including wake-up and falling asleep times, was generated using the PK/PD model [62]. The generation algorithm is illustrated in Figure 28.

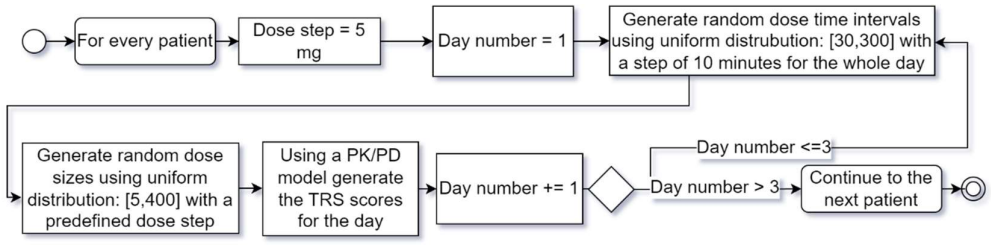

Figure 28 Medication day generation algorithm used to generate 3 days of states for every patient in the dataset

The generation process resulted in TRS values and medication data for 150 days (3 days for each patient) with 10-minute intervals (other time intervals can also be used in the framework), two examples are presented in Figure 29. The dataset was then divided into two sets; data from 40 patients were used for creating the general medication response model. The model would be later fine-tuned using data from the remaining 10 patients individually. The objective of creating the general model is to be able to roughly predict the patient's future states (TRS score) based on the initial state and taken medication, it will be referred to as the patient state prediction model. This model can be easily retrained to match individual patient needs.

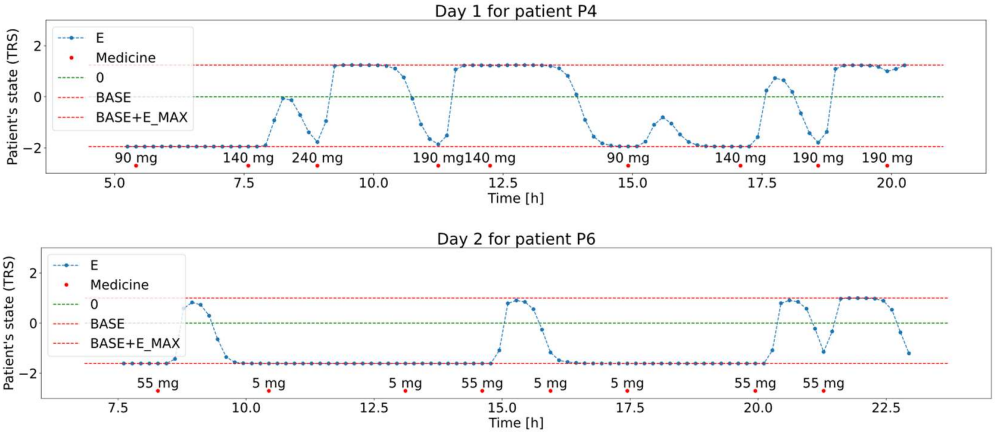

Figure 29 Example medication days generated for patients P4 and P6, medication times (red dots), patient's state – TRS score (blue line) with bounds for the states (red lines) and optimal state (green line)

5.2.1. Patient state prediction model

The model to predict patient's future states is built using machine learning. For that purpose, two types of artificial neural networks architecture have been proposed. Both architectures are capable of predicting the course of patient's states during the day with an equal time step – 10 minutes, which was chosen because it provides sufficient precision for dosing medicine (patients usually do not take medicine exactly to the minute) and it is also enough to monitor the changes in the patients TRS score.

The first architecture is a multilayer perceptron [126] referred to as *history model*. A multilayer perceptron is one of the simplest artificial neural networks. It consists of the input layer – first layer receiving the raw input data features, hidden layers, which are placed between input and output, responsible for transforming input features using weighted computations and activation functions and the output layer, which produces the final prediction of the network. In this problem the network had always one output – to predict the state of the patient. In the multilayer perceptron architecture, each neuron takes input from all neurons in previous layers and computes a weighted sum of these inputs.

The created *history model* considers the history of medication for every state prediction. The network is designed to take into account the previous n states and the last k medication doses to predict the next state to maintain a constant size of input data. The input consists of the n values representing previous states and $2k$ values representing the times since k last doses and k sizes of doses. Table 23 presents example network inputs when $n = 1$, $k = 2$ and two medication doses 100 mg taken at 10 minutes and 150 mg taken at 30 minutes since the beginning.

Table 23 Model input for prediction of future patient's state using two most recent doses and last state of the patient (n=2, k=1).

Current time	Previous state	Time since last dose	Last dose size	Time since previous dose	Previous dose size
0	$s_{initial}$	0	0	0	0
10	s_0	0	100	0	0
20	s_{10}	10	100	0	0
30	s_{20}	0	150	20	100
40	s_{30}	10	150	30	100

The number of network inputs is equal to n + 2k and there is always one output value representing the next state. In this thesis, models with n=1 and k=2 will be considered. This specific combination of n and k values is chosen to balance the trade-off between providing sufficient data for accurate predictions and maintaining a manageable dimensionality for the input data. The inference process for the entire day, based on the initial state and medicine schedule is presented in Figure 30. Two variants have been investigated to handle the TRS score range of -3 to 3. In the first variant, there is no activation function in the output layer and its output values are not directly restricted. In the second variant, the output is passed through the hyperbolic tangent (tanh) activation function and is multiplied by 3, ensuring that all outputs fall within the range of -3 to 3 (the range for the TRS score). During inference, when $n > 1$ the initial state is replicated to fill all the spots for previous values in the input vector.

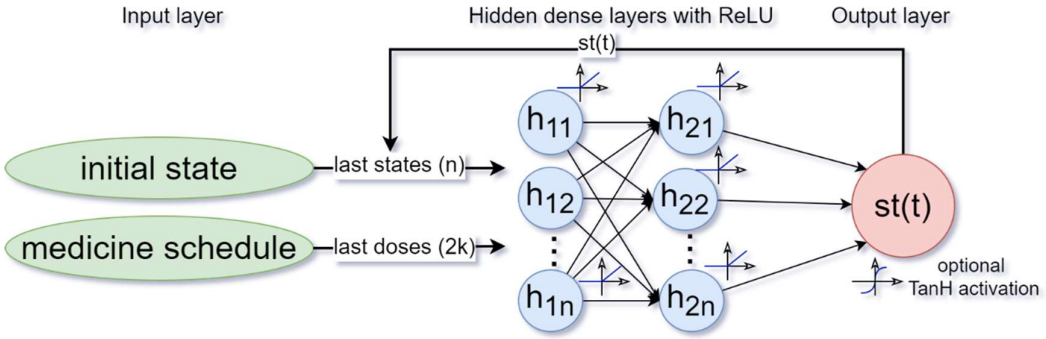

Figure 30 Prediction of states for the entire day provided initial state and medicine schedule using the history model which is based on a multilayer perceptron

The second model is a deep neural network that consists of fully connected and recurrent layers, specifically Long Short-Term Memory (LSTM) layers [96]. Recurrent neural layers are designed to process sequential data, these recurrent layers maintain a memory of previous inputs, employing loops within their architecture to allow information to persist over time. This approach ensures that data from every step of the sequence is processed with a consistent set of weights, enabling the model to effectively capture temporal dependencies and contextual nuances. However, traditional recurrent neural networks face challenges in capturing long-range dependencies, a limitation that LSTMs are specifically engineered to overcome.

LSTMs introduce a sophisticated mechanism consisting of memory cells and a system of gates, including the input, forget, and output gates, to regulate the flow of information. These gates allow LSTMs to selectively learn both long and short-range

dependencies in the data. The input gate controls the inflow of new information into the memory cell. The forget gate decides which information to discard. The output gate determines the information to be output from the cell. This architecture of a memory cell is depicted in Figure 31, illustrating how LSTMs manage information flow and maintain memory over extended sequences using tanh and sigmoid activation functions and capturing short-term (h_t) and long-term (c_t) memory based on the provided input (x_t).

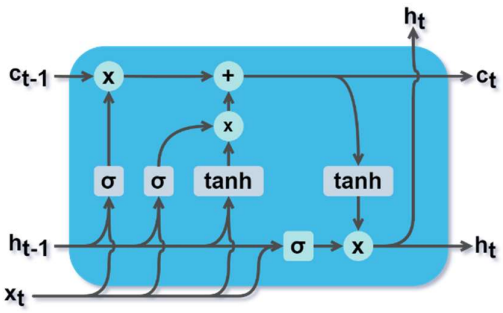

Figure 31 The architecture of an LSTM cell

The ability to learn from data where the context spans across long sequences makes LSTMs valuable for a wide array of applications, including natural language processing, speech recognition, and time series analysis – what is expected to be done with dosing and patient state data. By leveraging LSTMs, the model not only captures the temporal patterns inherent in sequential data, but also provides a robust framework for predicting future events or classifying sequences based on their historical context.

The model created with LSTM cells will be referred to as the *impulse model*. This naming is due to the nature of the input medication vector, which consists mostly of zeros with occasional spikes in value that occur only when medication doses are taken. To predict the TRS score in the next step, the model requires the values representing the previous state and the amount of medicine taken in the previous step. In case no medicine was taken, the value representing the amount of medicine is set to 0. Table 24 presents the input for the impulse model considering the schedule presented in Table 23.

Table 24 Model input for prediction of future patient's state using size of current dose and last state of the patient

Current time	Previous state	Dose size
0	$s_{initial}$	0
10	s_0	0

20	s_{10}	100
30	s_{20}	0
40	s_{30}	150

The defined impulse model allows capturing the complete history of medication and TRS scores in the LSTM cells' states without imposing restrictions on the number of considered doses and states. This may lead to improved performance, particularly when multiple doses are taken with small time intervals. Additionally, the model is expected to be less affected by outliers in the input patient's states. The process of predicting states for the entire day is presented in Figure 32. The initial LSTM cell state value is always 0.

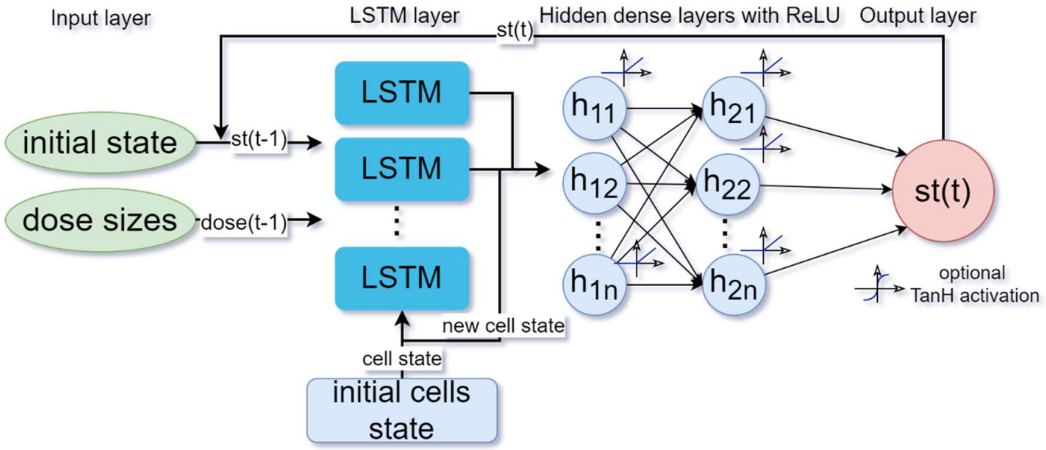

Figure 32 Prediction of states for the whole day provided initial state and medicine doses using the impulse model which uses LSTM cells to keep track of previous states and medicine doses

The training process is performed similarly for both models. Each day is treated as a training sample, where the initial (first) state and the medicine schedule serve as input. The remaining states are the expected outputs (for a 16h day, there would be 95 output states). For batch training, all training days are truncated to the same length, which is determined by the shortest day among the patients. The prediction is performed iteratively starting with the first set of inputs (initial state and medication data at t = 0) and each output state is then concatenated with the medication data for subsequent steps. The loss function value is calculated for the entire daily schedule, rather than for each step individually (Eq. 42). In both cases the mean squared error was used – to penalize significant deviations from the target values the most. The Adam optimizer [94] is used for both prediction models and the networks are implemented using Keras 2.11.0 [127].

$$L_d = \frac{1}{N_d}\sum_{t=1}^{N_d}\left(\widehat{y_t^d} - y_t^d\right)^2$$

N_d – number of output states for day d, Eq. 42

$\widehat{y_t^d}$ – predicted state at time t for day d,

y_t^d – actual state at time t for day d.

5.2.1.1. General model

In situations where the clinician needs to prescribe accurate and efficient medicine intake schedules, training a model from scratch for each patient can be time-consuming and requires a significant amount of collected data. This paper proposes an alternative approach by creating a general model, which after training is familiar with basic concepts of PK/PD modelling for the PD population. Subsequently, this model is personalized for each individual patient to reflect their specific medication response.

To simulate this behavior the dataset has been split into two parts – patients used to train the general model and remaining patients which will be then used to create personalized models. For fitting the general model, data of 40 patients has been used, each consisting of 3 days. In this section only data of these 40 patients will be considered.

All the 120 days were transformed into the input format for each of the introduced models. The data has been randomly split into training (80%) and validation (20%) sets, resulting in a varying number of days for each of the patients in both sets. To facilitate the training process, the data were then standardized separately for dose sizes and times by using the means and standard deviations calculated for each column in the training set.

Several variations of the defined models were explored to assess their effectiveness. These variants differed in the numbers of neurons in the hidden layers, units in the LSTM layer, the target value – which can be the TRS score or the difference between the next TRS score and the current one and the application of the tanh function at the output of the network. The number of neurons and LSTM cells were chosen as powers of 2, which is a common practice in machine learning and the values were ranging from 16 to 128 for the history model. Lowering the number below 16 resulted in worse performance, while increasing the number beyond 128 did not yield better results and increased the risk of overfitting. The variants of the history model used for training are listed in Table 25 and impulse model in Table 26.

Table 25 Variants of history models used for training the general model using data of 40 patients

Abbreviation	Dense hidden layers	Target value	tanh used
H-16,16	16,16	next state	no
H-16,16, tanh	16,16	next state	yes
H-16,16-diff	16,16	difference	no
H-32,32	32,32	next state	no
H-32,32, tanh	32,32	next state	yes
H-32,32-diff	32,32	difference	no
H-64,64	64,64	next state	no
H-64,64, tanh	64,64	next state	yes
H-64,64-diff	64,64	difference	no
H-128,128	128,128	next state	no
H-128,128, tanh	128,128	next state	yes
H-128,128-diff	128,128	difference	no

Table 26 Variants of impulse models used for training the general model using data of 40 patients

Abbreviation	LSTM units	Dense hidden layers	Target value	tanh used
I-(16)16,16	16	16,16	next state	no
I-(16)16,16, tanh	16	16,16	next state	yes
I-(16)16,16-diff	16	16,16	difference	no
I-(32)32,32	32	32,32	next state	no
I-(32)32,32, tanh	32	32,32	next state	yes
I-(32)32,32-diff	32	32,32	difference	no
I-(8)64,64	8	64,64	next state	no
I-(8)64,64, tanh	8	64,64	next state	yes
I-(8)64,64-diff	8	64,64	difference	no
I-(16)8,8	16	8,8	next state	no
I-(16)8,8, tanh	16	8,8	next state	yes
I-(16)8,8-diff	16	8,8	difference	no
I-(16)32,32	16	32,32	next state	no

I-(16)32,32, tanh	16	32,32	next state	yes
I-(16)32,32-diff	16	32,32	difference	no
I-(32)64,64	32	64,64	next state	no
I-(32)64,64, tanh	32	64,64	next state	yes
I-(32)64,64-diff	32	64,64	difference	no
I-(64)64,64	64	64,64	next state	no
I-(64)64,64, tanh	64	64,64	next state	yes
I-(64)64,64-diff	64	64,64	difference	no

For the training process two callbacks have been defined to perform tasks after every epoch. The first one stops the training process if the value of the loss function on the validation set has not decreased for 10 epochs. The second callback is responsible for saving the best model (one with lowest loss on the validation set) achieved during the training process. The networks were trained for 300 epochs with a 0.001 learning rate.

5.2.1.2. Patient specific model

In real-life applications clinicians aim to create accurate medicine schedules with maximum precision while minimizing the amount of data collected from each patient to only what is necessary for the method to be successful. Transfer learning [87] is a machine learning technique that has gained significant attention in recent years due to its ability to improve the performance of a model on a target task by utilizing knowledge learned from a related source task. The basic idea behind transfer learning is to leverage the knowledge and experience gained from solving one problem and apply it to another problem that shows some degree of similarity. Usually, it is performed by first training a network on a similar dataset, which contains more data – for the model to learn the patterns and dependencies in the data. Then the trained network with weights is retrained to adapt more to the target dataset, usually a smaller one, which would not be sufficient to achieve satisfactory results if used individually to train. In transfer learning, all of the weights can be updated during retraining (previously set weights are used as a starting point for training) or some of the first layers might be frozen and weights of only a few last layers are updated during the backpropagation of the training process. In case of the history model the whole network was retrained (only two layers), for the impulse model only the fully connected layers were retrained, and the weights of the LSTM layers were not updated.

In this case, the previously trained general model is retrained to fit the data for individual patients. The remaining 10 patients are treated as new patients requiring the establishment of a medicine schedule. They are observed for 3 days, during that period the sensor (regarding TRS score) and medication data is collected. The data is then split into training (2 days) and validation (1 day) to perform transfer learning on every model presented in Table 3 and Table 4 for each of the 10 patients. The same optimizer and loss functions have been selected (as for training the general model), with a maximum of 100 epochs for training. An early stopping callback has been defined to stop the training if no improved is observed for consecutive 25 epochs. Additionally, another callback is set up to save the model every time an improvement is made.

After training, the best results (lowest loss on validation set) are saved. While the training process and the prediction for a single step are not time-consuming tasks, the prediction for multiply steps (entire day) can take long, especially during optimization when it is performed multiply times. To speed up the performance of the prediction for the entire day, the weights from each neural network are extracted and the networks are implemented manually using NumPy array operations, which resulted in a 10-100 times faster (depending on the model) computation of the TRS scores for the day. The significant improvement is attributed to avoiding the overhead of Keras layers for training and inference for big inputs. Using simple array operations can handle the prediction process faster when processing data of smaller size and computations that must be called iteratively and cannot be parallelized (the state result for each step is needed for the computation of the state in the next step). This method reduces the computational overhead significantly, as Keras layers introduce additional complexity and processing time, particularly when handling large datasets or numerous iterative computations.

5.2.2. Results

5.2.2.1. General model

To select the best model for predicting the patient's medicine response all the models from Table 25 and Table 26 were trained ten times – to minimize the influence of the model's initial weights. The chosen number of epochs – 300 was sufficient for training. In most cases the training process was stopped with the early stop callback, since there was no improvement during last epochs. Despite training the general model on data from many patients, the process of training was fast and resulted in well-fitted models,

considering the diversity of patients. To evaluate and compare the performance of the trained models, the following metrics were calculated:

- mean squared error (MSE), which also serves as the loss function (smaller value is preferred),
- mean absolute error (MAE) (smaller value is preferred),
- coefficient of determination (R^2) (greater value is preferred).

Table 27 presents the results for the history models, displaying the mean values across the 10 training rounds after removing any outliers for both the training and validation sets. When evaluating the model, it is important to observe low loss function values on both the validation and the training set. The models should not be prone to overfitting, which can occur when retraining on a small dataset for individual patients, especially with complex network structures. One potential sign of overfitting is a significantly lower loss on the training set compared to the validation set. To mitigate this, appropriate strategies such as reducing the complexity of the model or applying regularization techniques should be used.

Table 27 Mean metrics values – mean squared error (MSE), mean absolute error (MAE) and coefficient of determination (R^2) for history models sorted by MSE in validation set. Best results for each metric presented in bold

Model info	**Training set**			**Validation set**		
	MSE	MAE	R^2	MSE	MAE	R^2
H-128,128-diff	0.254	0.373	0.683	**0.310**	0.406	0.621
H-32,32-tanh	0.281	0.384	0.666	0.315	0.401	**0.625**
H-64,64-tanh	0.284	0.385	0.660	0.315	0.404	0.624
H-64,64-diff	0.279	0.382	0.668	0.322	0.410	0.620
H-64,64	**0.230**	**0.345**	**0.728**	0.324	**0.395**	0.621
H-128,128-tanh	0.282	0.384	0.661	0.325	0.403	0.614
H-128,128	0.242	0.355	0.713	0.325	0.396	0.618
H-16,16-tanh	0.296	0.398	0.641	0.326	0.410	0.609
H-16,16-diff	0.306	0.398	0.645	0.327	0.408	0.623
H-32,32-diff	0.266	0.370	0.683	0.330	0.401	0.607
H-16,16	0.313	0.406	0.635	0.332	0.415	0.610
H-32,32	0.272	0.388	0.679	0.333	0.417	0.609

To further assess the performance of the models, the Wilcoxon signed-rank test [128,129], a non-parametric statistical test, was employed. This test examines the null hypothesis that two related paired samples (X and Y) are drawn from the same distribution, more specifically if the distribution of the differences (Eq. 43) between two sets of measurements is symmetric around zero. It provides the option to choose from three alternative hypotheses:

- greater – can determine if one set of measurements is stochastically greater than the other set of measurements,
- less – helps determine if one set of measurements is stochastically less than the other set of measurements,
- two-sided - checks whether there is a significant difference between the two distributions.

If the two-sided alternative hypothesis is selected the value of the statistic represents the sum of the ranks of the differences bigger or smaller than 0, whichever is smaller (Eq. 45). When greater or less is selected, the value represents the sum of the ranks of differences above zero. The rank (Eq. 44) in this context refers to the position of each difference when all differences are ordered by absolute value.

$$D_i = Y_i - X_i \quad \text{Eq. 43}$$

$$R_i = \begin{cases} the\ rank\ assigned\ to\ (X_i, Y_i)\ if\ D_i > 0 \\ the\ negative\ of\ the\ rank\ assigned\ to\ (X_i, Y_i)\ if\ D_i < 0 \end{cases} \quad \text{Eq. 44}$$

$$T^+ = \sum_{i=1}^{n} (R_i\ where\ D_i > 0) \quad \text{Eq. 45}$$

In this analysis, the test was conducted using the greater alternative hypothesis. The best performing model was compared with the remaining to determine if it was significantly better – had significantly lower MSE. The hypotheses for the Wilcoxon signed-rank test were formulated as follows:

- Null hypothesis (H0): The MSEs of the compared models are equal to the MSE of the best model.
- Alternative hypothesis (H1): The MSE of the best model is significantly less than the MSEs for other models.

The resulting p-values from the Wilcoxon signed-rank test, comparing the MSE values in 10 trials of the best model with the remaining models, are illustrated in Table

28. With a significance level of 0.05, it was observed that the best model (H-128,128-diff), significantly outperformed 3 models, p-values lower than the significance level resulted in rejecting the null hypothesis, thus accepting that the MSEs of these models are significantly greater than the MSEs of the best model (alternative hypothesis).

Table 28 Statistics and p-values for the Wilcoxon signed-rank test which was performed to verify which models are significantly worse than the best history model. The test was conducted with the alternative hypothesis that the distribution underlying one set of measurements (mean squared errors of the best model) is stochastically less than the distribution underlying the second set of measurements (mean squared errors of other models).

Model	Statistic value	P-value
H-16,16-tanh	17	0.161
H-16,16	20	0.246
H-32,32-diff	3	0.00488
H-32,32-tanh	17	0.161
H-32,32	20	0.246
H-64,64-diff	8	0.0244
H-64,64-tanh	24	0.385
H-64,64	18	0.188
H-128,128-tanh	25	0.423
H-128,128	26	0.461
H-16,16-diff	0	0.000977

For further use, to reduce the training time and focus on getting best results in fitting to individual patients only 5 history models with the lowest MSE were selected. What is noticeable, is that the restriction of the outcome values using the tanh function seems to improve the performance of the network. The number of neurons in the hidden layers does not seem to have an explainable influence on the performance of the history networks. Two patient days have been selected from the validation set to visually compare the outcomes of the top 3 models (with lowest validation MSE) and they are presented in Figure 33.

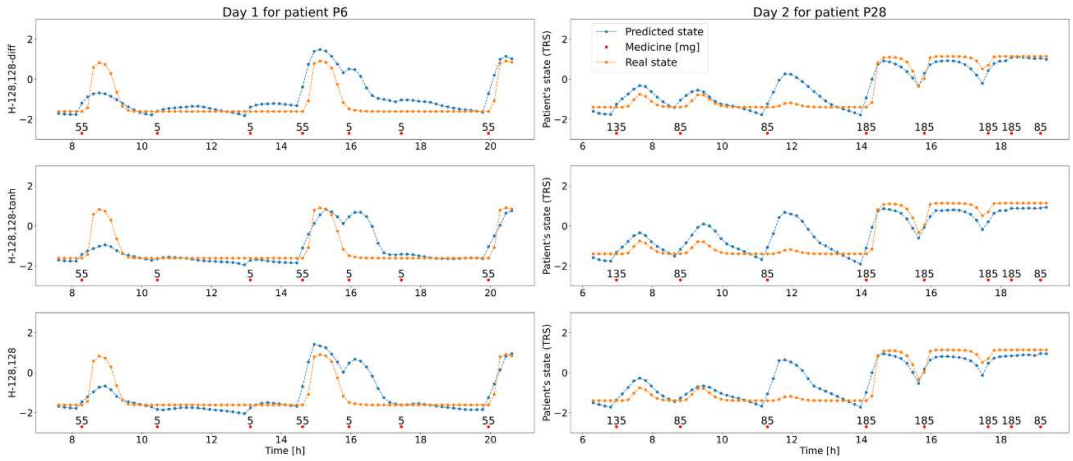

Figure 33 Day 1 for patient P6 and day 2 for patient P28 with medicine doses (red dots) and changing TRS score values during day, the "real" – generated with PK/PD model (orange) and predicted(blue) with top 3 history ML models – H-128,128-diff (top), H-128,128-tanh (middle) and H-128,128 (bottom)

Difficulties with reflecting the individual patient's response to the medication occurred in general models. When the patient is more medicine-resistant and small doses are taken, there should be no drastic change in the patient's state. However, these models seem to react to even small doses such as 5 mg. This proves the need for individualization of medicine schedules, since a dose of some size might have a small impact on one patient and a significant one on the other. Due to the restricted length of the history in the history model, only two last doses and two last states are considered when calculating the next state. The model demonstrates suboptimal performance in situations characterized by frequent doses, small time intervals and significant fluctuations in the patient's recent states.

For the impulse model, a different set of architectures has been chosen. In this case also 10 training cycles have been performed for each of the models (to reduce the influence of randomly initialized weights) and their results – the means of MSE, MAE and R^2 have been presented in Table 29.

Table 29 Mean metrics values – mean squared error (MSE), mean absolute error (MAE) and coefficient of determination (R^2) for impulse models sorted by MSE in the validation set. Best results for each metric presented in bold

Model info	**Training set**			**Validation set**		
	MSE	MAE	R^2	MSE	MAE	R^2
I-(32)32,32-tanh	0.283	0.378	0.671	**0.255**	0.355	**0.686**
I-(8)64,64-diff	0.258	0.368	0.698	0.257	0.358	0.680

I-(16)16,16	0.240	0.353	0.708	0.258	**0.349**	0.680
I-(32)32,32-diff	0.214	0.326	0.742	0.260	0.360	0.676
I-(8)64,64	0.260	0.372	0.685	0.261	0.360	0.660
I-(32)32,32	0.242	0.348	0.704	0.262	0.364	0.663
I-(16)32,32-tanh	0.249	0.357	0.711	0.264	0.361	0.672
I-(32)64,64-tanh	0.251	0.359	0.704	0.268	0.370	0.666
I-(32)64,64-diff	0.198	0.321	0.765	0.270	0.353	0.677
I-(64)64,64-diff	0.219	0.334	0.736	0.271	0.369	0.666
I-(16)8,8	0.282	0.384	0.665	0.273	0.374	0.653
I-(16)16,16-diff	0.213	0.335	0.739	0.274	0.351	0.664
I-(16)8,8-tanh	0.242	0.359	0.706	0.274	0.353	0.671
I-(16)32,32-diff	**0.171**	**0.299**	**0.793**	0.274	0.361	0.664
I-(64)64,64	0.196	0.316	0.761	0.275	0.376	0.658
I-(16)32,32	0.282	0.390	0.670	0.275	0.383	0.660
I-(64)64,64-tanh	0.260	0.366	0.689	0.275	0.374	0.652
I-(32)64,64	0.280	0.382	0.661	0.276	0.382	0.649
I-(8)64,64-tanh	0.255	0.366	0.694	0.277	0.371	0.637
I-(16)16,16-tanh	0.254	0.361	0.702	0.278	0.372	0.658
I-(16)8,8-diff	0.251	0.368	0.663	0.292	0.374	0.572

The Wilcoxon signed-rank was also applied to compare the performance of impulse models and evaluate the significance of the observed differences in MSE. The resulting p-values from comparing the best model (I-(32)32,32-tanh) with other models are presented in Table 30. Using a significance level of 0.05, it was found that the model significantly outperforms 7 models.

Table 30 Statistics and p-values for the Wilcoxon signed-rank test performed to verify the models that are significantly worse than the best impulse model. The test was conducted with the alternative hypothesis that the distribution underlying one set of measurements (mean squared errors of the best model) is stochastically less than the distribution underlying the second set of measurements (mean squared errors of other models)

Model	Statistic value	P-value
I-(16)16,16-tanh	7	0.0186
I-(16)16,16	7	0.0186
I-(16)32,32-diff	9	0.0322

I-(16)32,32-tanh	7	0.0186
I-(16)32,32	10	0.0420
I-(16)8,8-diff	0	0.000977
I-(16)8,8-tanh	8	0.0244
I-(16)8,8	15	0.116
I-(32)32,32-diff	16	0.138
I-(32)32,32	30	0.615
I-(32)64,64-diff	14	0.0967
I-(32)64,64-tanh	19	0.216
I-(32)64,64	15	0.116
I-(64)64,64-diff	22	0.313
I-(64)64,64-tanh	22	0.313
I-(64)64,64	11	0.0527
I-(8)64,64-diff	17	0.161
I-(8)64,64-tanh	20	0.246
I-(8)64,64	11	0.0527
I-(16)16,16-diff	11	0.0527

The impulse models, due to the use of LSTM cells, were able to capture the whole available history, all the provided previous states, previously administered doses. This resulted in a significantly better performance on the validation set, the worst performing model from Table 29 has lower MSE and MAE than the best model from Table 27. This proves the advantage of using LSTMs for analyzing time series. However, this approach allows handling only doses that were taken at time steps – times of doses taken between them would have to be rounded to match a value provided by the time step (e.g., dose taken at 8:17 would be treated as taken at 8:20). This might lead to some inconsistencies in training the model on real samples, when the patient might take the medicine at a different time. To avoid this lower time steps could be chosen, or another feature could be added to the vector representing minutes since the dose was taken.

Using the best 3 models, days for two patients from the validation set have been created. The results for each of the networks do not differ as much as in the case of history models and they match the target values better. The results are presented in Figure 34.

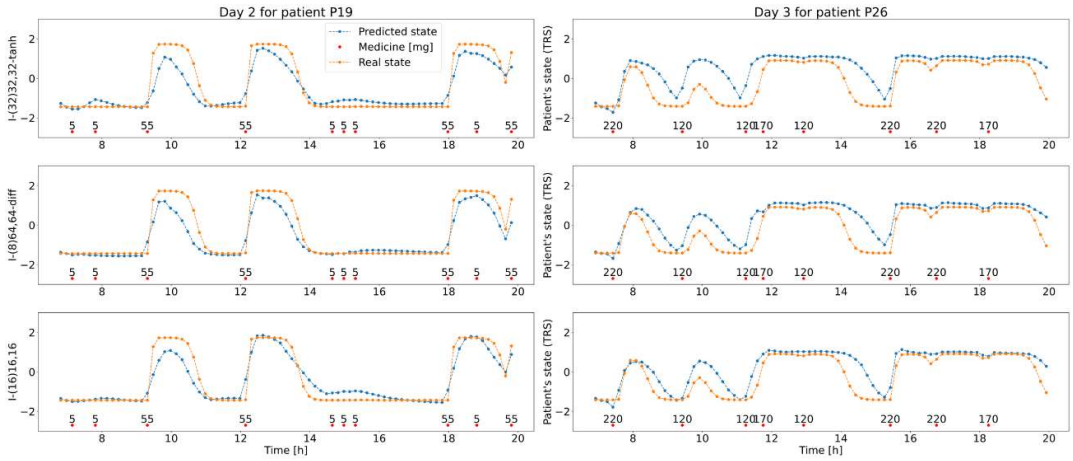

Figure 34 Day 2 for patient P19 and day 3 for patient P26 with medicine doses (red dots) and changing TRS score values during day, the "real" – generated with PK/PD model (orange) and predicted(blue) with top 3 impulse ML models - I-(32)32,32-tanh (top), I-(8)64,64-diff (middle) and I-(16)16,16 (bottom)

5.2.2.2. Patient specific model

To retrain and create patient specific models, only top 5 history (Table 27) and top 5 (Table 29) impulse models were used. For each of the 10 remaining patients, each of these models was trained with 2 generated days and the third day was treated as the validation set. Table 31 presents the results of the training process providing the metrics for the general model (before retraining) and individualized model (after retraining) for best performing models (lowest MSE for training and validation sets) for every patient. Values outside parentheses are for the training set, while values in parentheses are for the validation set.

Table 31 Metrics values – mean squared error (MSE), mean absolute error (MAE) and coefficient of determination (R2) for individual patients' training (day 1 and 2) and validation data (day 3) before and after retraining

P	**Best model**	**Before retraining – training (validation)**			**After retraining – training (validation)**		
		MSE	MAE	R^2	MSE	MAE	R^2
41	I-(32)32,32-diff	0.136 (0.296)	0.296 (0.390)	0.688 (0.539)	0.000314 (0.00683)	0.0124 (0.0467)	0.999 (0.989)
42	I-(8)64,64	0.0969 (0.500)	0.186 (0.513)	0.870 (0.716)	0.00262 (0.0274)	0.0264 (0.0841)	0.996 (0.984)
43	I-(32)32,32-tanh	0.244 (0.188)	0.321 (0.270)	0.734 (0.786)	0.0000342 (0.0100)	0.00454 (0.0586)	0.999 (0.989)

44	I-(16)16,16	0.452 (0.313)	0.485 (0.351)	-1.56 (0.255)	0.00144 (0.00763)	0.0200 (0.0469)	0.992 (0.982)
45	I-(32)32,32-tanh	0.331 (0.0911)	0.447 (0.270)	0.518 (0.828)	0.00235 (0.00982)	0.0314 (0.0679)	0.997 (0.981)
46	I-(8)64,64	0.145 (0.126)	0.323 (0.298)	0.774 (0.846)	0.00324 (0.0110)	0.0388 (0.0596)	0.995 (0.987)
47	I-(16)16,16	1.55 (0.157)	0.837 (0.300)	-0.866 (0.821)	0.000639 (0.00412)	0.0193 (0.0372)	0.999 (0.995)
48	I-(8)64,64	0.441 (0.505)	0.540 (0.543)	0.590 (0.648)	0.000855 (0.00853)	0.0227 (0.0529)	0.999 (0.994)
49	I-(8)64,64	0.397 (0.622)	0.405 (0.548)	0.456 (-0.03)	0.000302 (0.0214)	0.0132 (0.0735)	0.999 (0.964)
50	I-(8)64,64	0.961 (1.07)	0.706 (0.870)	0.174 (-2.31)	0.00128 (0.00579)	0.0278 (0.0483)	0.999 (0.982)

The metric values for the general model for patients that have never been seen are not satisfying. The errors are significantly higher than for the validation set in the previous training and in some cases the R^2 has even negative values, which concludes that this model does not reflect the medication response for specific patients well. After retraining the models on two days for every patient, the results have greatly improved with the worst results for patient 49, with MSE=0.0214 and R^2=0.964 and they are expected to be effective in predicting the patient's response to medication. Figure 35 presents the best model's performance for validation days for patients 46 and 50 showing how the model's output has improved after retraining.

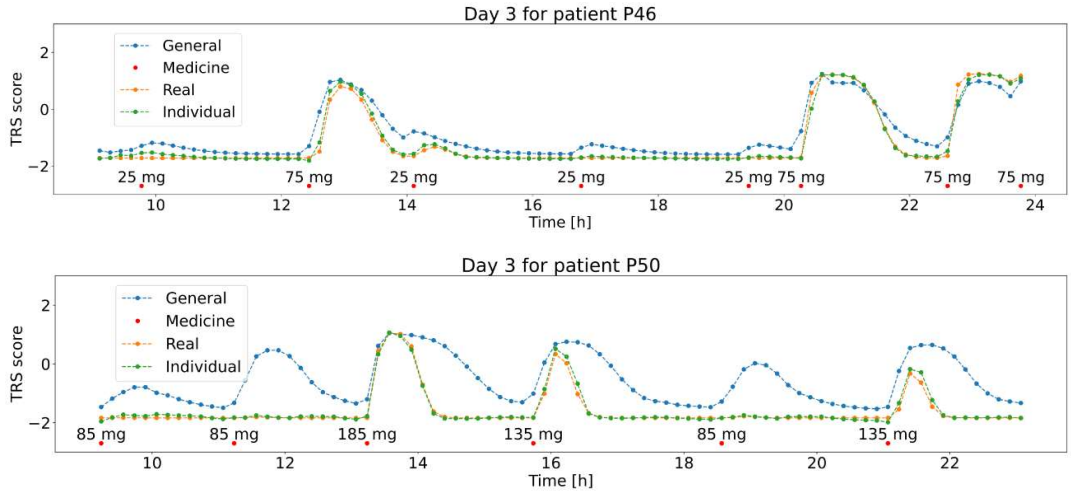

Figure 35 Day 3 for patients P46 (top) and P50 (bottom) with medicine doses (red dots) and changing TRS score values during day – the "real" – generated with PK/PD model (orange), predicted before retraining(blue) and after retraining(green) with the top ML model.

Table 31 shows that different models have been selected as best for each of the patients and a method was selected for choosing the best one overall – the mean metrics among patients have been calculated and the model with lowest mean validation MSE was considered the best, mean MSE and other metrics for these models are presented in Table 32.

Table 32 Mean metrics values – mean squared error (MSE), mean absolute error (MAE) and coefficient of determination (R^2) for individual models sorted by validation MSE. Best results for each metric presented in bold

Model info	**Training set**			**Validation set**		
	MSE	MAE	R^2	MSE	MAE	R^2
I-(8)64,64	0.00756	0.0423	0.984	**0.0215**	**0.0795**	0.965
I-(8)64,64-diff	0.0076	0.05	0.989	0.0239	0.0933	**0.967**
I-(16)16,16	0.0154	0.0716	0.979	0.0438	0.115	0.957
I-(32)32,32-tanh	0.00621	0.0427	0.983	0.0466	0.116	0.918
I-(32)32,32-diff	**0.00434**	**0.0359**	**0.995**	0.05	0.123	0.906
H-64,64-tanh	0.0226	0.0921	0.965	0.0853	0.193	0.858
H-32,32-tanh	0.0524	0.141	0.897	0.0915	0.208	0.854
H-64,64	0.0368	0.122	0.935	0.0967	0.21	0.832
H-128,128-diff	0.0727	0.163	0.826	0.107	0.226	0.804
H-64,64-diff	0.119	0.237	0.834	0.133	0.263	0.809

Presented results show that the impulse model with 8 units in the LSTM cell and 2 hidden fully connected layers provided the best results on the validation dataset for each of the patients, this suggests that this model might be used for new patients. The table also shows the advantage of impulse models compared to history models which had the lowest results for all the metrics. Considering the results presented in this table, model I-(8)64,64 was chosen for optimization, since it provided the best performance among the patients.

To verify the goodness of fit of the selected best models, the Wilcoxon signed-rank test was used to compare the target values (generated from the PK/PD model) with the ML model predictions on the validation set. This test is recommended in the performance validation of simulation models [128]. The Wilcoxon signed-rank test is a non-parametric statistical test that examines the null hypothesis for the test assumes that two related paired samples (target and predicted values) come from the same distribution. In this case the test was conducted with the two-sided alternative hypothesis. The resulting statistic and p-values for the test are presented in Table 33. Across all patients, the null hypothesis was accepted at a significance level of 0.05, confirming that the models are well-fitted.

Table 33 Statistics and p-values for the Wilcoxon signed-rank test performed to verify that the individual models are well-fitted (same distributions of target and predicted values)

Patient	**P41**	**P42**	**P43**	**P44**	**P45**	**P46**	**P47**	**P48**	**P49**	**P50**
Statistic value	1885	1741	1689	2530	1410	1595	2243	2422	2078	1656
P-value	0.323	0.464	0.149	0.878	0.642	0.0955	0.631	0.990	0.0919	0.565

5.2.3. Discussion

This part of the study focused on modelling the PD patient medicine response (levodopa/carbidopa) using machine learning – artificial neural networks.

The success of modelling the individual response to medication using shallow machine and deep learning methods is notable, especially considering the limited number of training samples. These models performed particularly good on the validation set, however the acquired performance differed between patients, due to the randomness of dataset generation and individual responses to medication. After performing experiments with two proposed architectures (history and impulse), the advantage of the LSTM

architecture was demonstrated, with the best model achieving MSE value of 0.0215 and R^2 value of 0.965. The history model was able to capture most recent history regarding the taken doses and previous steps, the number of recent considered doses and previous patient steps had to be set at the beginning, even before the training process. However, the impulse model was unable to precisely process medicine doses, since all of times had to be rounded to a 10-minute step.

The experiments in this section were performed using generated data that closely reflected real conditions, following a validated generation method [64]. Data collection was limited to a maximum of 3 days, resembling typical hospital visits, with measurements recorded at 10-minute intervals.

Further experiments considering the ML models will include trials with real patients focusing on the use of LSTMs to predict their response to levodopa.

5.3. Swedish dataset patients

During the course of PhD studies, it was possible to participate in an Erasmus+ Traineeship Program, which made it possible to go to Dalarna University, where researchers engaged in the projects regarding the Swedish dataset worked. During the traineeship it was possible to get access to the dataset and using the patients considered in their study, machine learning models were trained to reflect their individual responses to medication. Using this dataset will make it possible to compare the results of creating medicine intake schedules with neurologists' prescriptions.

The initial study included data of 25 patients. However, the study focused on 19 patients after excluding six due to various reasons, including inability to perform specific tasks and limited medication response. For this study, the following patient data was used:

- clinician's state evaluations using the TRS scale during the pronation-supination tasks at roughly 20-minute intervals, for 170-320 minutes depending on the patient, sometimes involving two medication doses.
- the timestamps and sizes of taken levodopa and carbidopa doses.
- PK/PD model parameters estimated in the PK/PD model study [64].
- clinician-created medicine schedules: the interval between doses, morning, and maintenance dose sizes.
- demographic and clinical data of the patients (Table 34).

- clinical scales results (Table 3).

Table 34 Patient onboarding data represented by medians and interquartile ranges (IQR)

Name	Description	Median (IQR)
Age	Age in years	68.0 (9.0)
Weight	Weight in kilograms	74.5 (16.4)
Height	Height in meters	1.73 (0.130)
Sex	Male or female	-
BMI	Body Mass Index	25.2 (4.58)
Onset	Years since the onset of disease	11.0 (10.0)
Diagnosis	Years since the diagnosis of disease	9.5 (8.25)
Motor fl.	Years since the start of motor fluctuations	4 (4.5)
Af. side	Most affected side by the disease	-
Blood pressure	Diastolic and systolic, when sitting and standing	80 (10), 126.5 (21.5), 117 (27.8), 78 (11)

5.3.1. Data preparation

The state of each patient, using TRS, has been evaluated only a limited number of times in the study, 8-13 times per patient. Unfortunately, this is not enough to train the ML models. To augment the dataset, the PK/PD models were used to simulate additional measurements. For each patient, an additional 3 days of patient state evaluations were created (each consisting of 960 minutes). For each of the days, random intervals between doses and dose sizes are generated using a uniform distribution with the bounds defined as 50% and 150% of the value (dose size or interval) suggested for the patient by the clinician. These bounds were selected to allow the model to learn how the patient reacts to different doses of medication. This is done to improve the results of the training process, to have enough data to train the neural networks.

To be able to use the general population models that were previously trained by Gutowski et. al [69] and were presented in Simulated patients section (p. 97), the PK/PD models were used to generate data with 10-minute intervals. The clinician's evaluations performed in the study were not performed at equal time intervals. However, in most cases, they were approximately 20 minutes apart. To make this data useful for training

the models, linear interpolation was used to generate approximate TRS values with 10-minute intervals. This interpolation involved calculating TRS values at 10-minute intervals based on the TRS values recorded at the closest clinician evaluation times. For any time t between two clinician evaluations at times t_1 and t_2 with corresponding TRS values TRS_1 and TRS_2, the interpolated TRS value TRS_t is given by Eq. 46.

$$\mathrm{TRS}_t = \mathrm{TRS}_1 + \left(\frac{\mathrm{TRS}_2 - \mathrm{TRS}_1}{t_2 - t_1}\right) \times (t - t_1) \qquad \text{Eq. 46}$$

The patient's state assessments in the final dataset are a combination of interpolated clinician evaluations and data generated by the PK/PD model. Specifically, the interpolation provides TRS values at 10-minute intervals, while the PK/PD model generates additional synthetic data. This dataset integrates real clinician evaluations with synthetic data generated by the PK/PD models, providing a more realistic representation of patient states.

The defined ML models require that the inputs, for all the training/validation samples, are of the same length. Since all the generated days consist of 960 minutes (96 values with 10-minute intervals), it was necessary to extend the vectors representing clinician's state evaluations. They were prepended with minimal patient state values to make sure they all represent a day of 960 minutes. This resulted in a dataset of 4 days (each 960 minutes long) with 10-minute for each of patients.

The dataset for each patient was then split into the training and validation set. Two of the three days generated using PK/PD models were put into the training set, and the remaining generated day, along with the day with clinician's real evaluations, were put into the validation set. This approach ensures that the model is validated on real patient data, providing a clear distinction from purely synthetic patient datasets.

5.3.2. Patient state prediction model

5.3.2.1. Patient medicine response prediction model

In the section regarding simulated patients, the retraining for individual patients was performed for 5 best-performing history (based on multilayer perceptron) and 5 best-performing impulse (based on LSTMs) models. In all of the cases – for training the general model and individual patient models the best-performing was always an impulse model. Therefore, for predicting medicine dose responses for real patients only 3 best-performing variants of the impulse model were investigated presented in Table 35.

Table 35 Impulse models used to predict real patients future states based on previous states and medication

Abbreviation	Network type	LSTMs	Hidden layer sizes	Prediction result
I-(8)64,64	LSTM	8	2 layers, 64 neurons	next state
I-(8)64,64-diff	LSTM	8	2 layers, 64 neurons	difference
I-(16)16,16	LSTM	16	2 layers, 16 neurons	next state

These models were not trained from the beginning; instead, the initial weights obtained during training on the general population were used as a starting point. This transfer learning approach allows the models to leverage the pre-existing knowledge from the general population and adapt it to each patient's specific data. They were then retrained separately for each patient to accurately reflect their individual medicine (levodopa) response.

The training process used mean squared error (MSE) as the loss function and the Adam optimizer (learning rate of 0.005) to update the weights of the networks. The training process was set up to run for up to 500 epochs. However, if there was no improvement to the MSE on the validation set for 40 consecutive epochs, the training was stopped, and the best model was saved and evaluated. Similarly, like for the simulated patients, the models were evaluated using 3 metrics: R^2, MAE, and MSE. In this case, the Wilcoxon signed-rank test was used and only models that were well-fitted (significance level of 0.05) were considered for further research.

5.3.2.2. Correlation analysis

During the onboarding, apart from medication information and TRS scores, patient demographic data and other patient features were collected. These characteristics, such as weight, height, age, etc. might have an impact on how the body responds to medication, how big doses of levodopa the patient should take, and how often. To determine if these patient features influence the medication process, a correlation analysis was conducted. Selected patient features from Table 34 and Table 3 are checked for correlation against medication parameters in Table 36. These medication parameters include the medicine dosing schedule parameters based on a schedule prescribed by the neurologist, others, derived from them, and PK/PD model patient specific parameters that were fitted in a previous study [64]. The patient features were selected based on analysis of literature and possible correlations between these features and disease progression

related characteristics. Features such as diagnosis date, onset of the disease and motor fluctuations are commonly known to be related to the severity of the disease and medicine dosing. Recent research shows also correlations between the age at onset and symptom profiles [130], height and substantia nigra neuron density [131]. Further research identifies a relationship between obesity (related to BMI) and the degeneration process of dopaminergic neurons [132] and between blood pressure and disease progression [133].

The schedules constructed by the neurologists consider only two different dose sizes – the morning dose – usually bigger, taken to make the patient reach the desired state after night and a maintenance dose, taken multiple times during the day to keep the patient in the optimal state. The clinician also assigns a time interval between taking subsequent medicine doses. Other parameters in Table 36 are derived from these 3 parameters such as total daily dose size or the dose ratio between the morning and maintenance dose.

Table 36 Medication parameters created based on neurologist's medication suggestions (ground truth) and fitted PK/PD models

Name	Description
Dosing parameters	
Interval	Time interval between doses suggested by clinician
Number of doses	Day length / interval
Morning dose	Size of the levodopa dose taken after wakeup suggested by clinician
Maintenance dose	Size of the levodopa dose taken during the day suggested by clinician
Dose ratio	Morning dose/Maintenance dose
Daily dose	Morning dose + maintenance dose * (number of doses -1)
PK/PD model parameters – II compartment model [63,64]	
V1	Volume of the first compartment
BASE	Minimum TRS score for the patient
E_MAX	Maximum change of the TRS score from baseline (BASE)
CL	Clearance rate
TKEO	Effect time constant

EC_50	Concentration at 50% effect (mg/L)
γ	Hill's coefficient
α	Elimination rate constant

Performing the correlation analysis allows capturing the dependencies between the data and helps identify patient features that have most impact on planned levodopa dose sizes and their time intervals. The results could be then used as a starting point to build a machine learning model for creating basic medicine schedules, based only on patient metadata.

5.3.2.3. General medicine response prediction model with correlations

After identification of the patient-specific features that have the highest/lowest correlation with treatment-related features, it is possible to create an additional ML model for predicting future patient's state. Apart from using the medicine doses taken and previous states as input, it would accept patient features as well. In this approach there would be only one general model which would be tailored to the patient's needs just by providing the patient-specific parameters. There would be no need for individualized models retrained for specific patients, but only one for all of them. This model could be used especially in situations when there is not enough data for the patient to build individualized models. It might be a good starting point for the treatment, which can be adjusted later.

To build this model all the features that had an absolute correlation value above 0.5 with the dosing parameters (regarding dose sizes and time intervals) and all the parameters that had an absolute correlation value above 0.6 with the PK/PD model parameters were selected. These features were normalized (mean = 0, standard deviation = 1) and then used to perform Principal Component Analysis (PCA), and a subset of the first principal components was selected, that explained at least 80% of the variance (no more than 5). This resulted in the construction of 2 additional approaches:

1. Model accepting additional patient-specific parameters as input to the network for state prediction.
2. Model accepting the first few principal components based on patient-specific parameters as input to the network for state prediction.

These architectures differ only in the number of additional inputs to the network and the way to generate state values is presented in Figure 36.

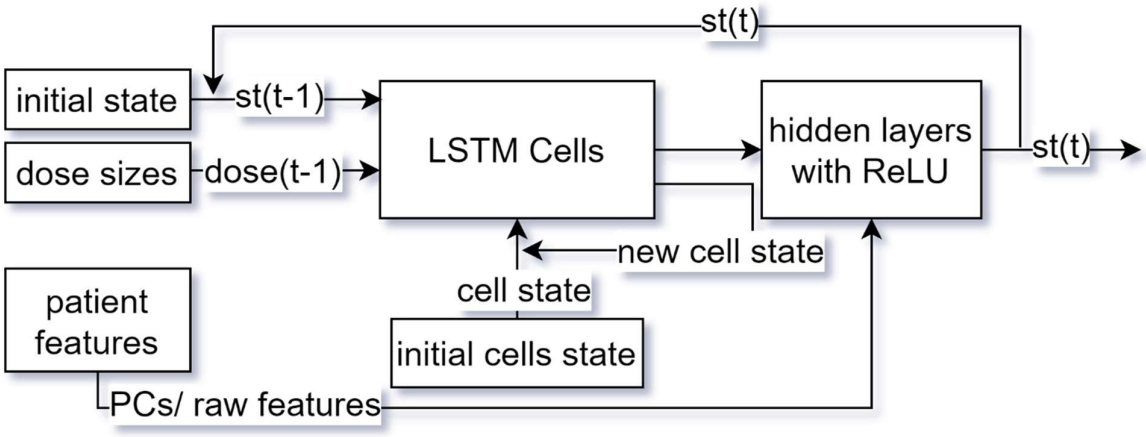

Figure 36 Generating patient's states for the day using the LSTM model, based on initial patient state, sizes of taken doses, and additional patient-specific parameters. The st(t) represents the state at time t and dose(t) represents the size of the dose taken at time t

The training process for these models runs for a maximum of 1000 epochs (the training is performed from scratch) with an early stop callback. The MSE is used as the loss function and the model is evaluated using MSE, MAE and R^2. After finishing the training process the model with the lowest MSE on the validation set is used, out of the 3 previously proposed architectures (I-(8)64,64, I-(8)64,64-diff, I-(16)16,16). The training was performed on data of 18 patients, since patient-specific parameters were missing for one of the patients.

5.3.3. Results

5.3.3.1. Patient medicine response prediction model

For each of the 19 patients, three models from Table 35 were trained to reflect the patient's response to levodopa, as indicated by the TRS scale. Each model was trained using 4 days of the patient, 2 in training and 2 in the validation dataset. This resulted in 57 models trained, 3 for each patient. The best model for each patient was chosen, based on the highest R^2 value on the validation set. The results of the training process – selected models and the metrics considered calculated on the validation set are presented in Table 37. Most models resulted in R^2 values above 0.8, with exceptions for patients P3 and P10. Lower values of R^2 and higher values of errors (MSE and MAE) are a result of validation of the model using real patient data which in some cases significantly differed from values that would be generated with the PK/PD model for the same dosing schedule. The

Wilcoxon test confirmed the validity of these ML models for simulation of patient's response to levodopa, as evidenced by p-values higher than the selected significance level (0.05).

Table 37 Metrics calculated on the validation set for best patient-specific models for each of the patients.

Patient	Selected model	MSE	MAE	R^2	Wilcoxon p-value
1	I-(8)64,64-diff	0.110	0.149	0.873	0.898
2	I-(8)64,64-diff	0.0517	0.113	0.89	0.0624
3	I-(8)64,64-diff	0.0635	0.099	0.791	0.867
4	I-(8)64,64	0.0359	0.0651	0.863	0.203
5	I-(8)64,64	0.0097	0.0578	0.864	0.587
6	I-(16)16,16	0.00984	0.0513	0.884	0.0505
7	I-(8)64,64-diff	0.0171	0.068	0.876	0.274
8	I-(16)16,16	0.0458	0.102	0.835	0.0658
9	I-(16)16,16	0.0474	0.113	0.807	0.0503
10	I-(16)16,16	0.0711	0.109	0.71	0.0502
11	I-(8)64,64-diff	0.0708	0.177	0.915	0.588
12	I-(16)16,16	0.0841	0.144	0.875	0.242
13	I-(16)16,16	0.123	0.180	0.852	0.459
14	I-(8)64,64-diff	0.0141	0.057	0.893	0.0506
15	I-(8)64,64	0.0023	0.0292	0.957	0.0604
16	I-(8)64,64	0.0972	0.235	0.875	0.359
17	I-(8)64,64-diff	0.0276	0.0566	0.869	0.549
18	I-(8)64,64-diff	0.00561	0.0391	0.874	0.884
19	I-(8)64,64-diff	0.0188	0.0623	0.855	0.152

To further assess the models' performance, the data from the validation set can be graphically compared against the predictions generated by the ML model. Figure 37 illustrates this for two patients, P4 and P11, presenting their day 3 and day 4 respectively. These two line charts present the timing and sizes of medicine doses alongside their observed effects reflected on the validation dataset based on the individual patient model, as well as the general model – before retraining on that patient's data. The individual

models closely match the actual data, while the general model presents a different reaction to levodopa doses.

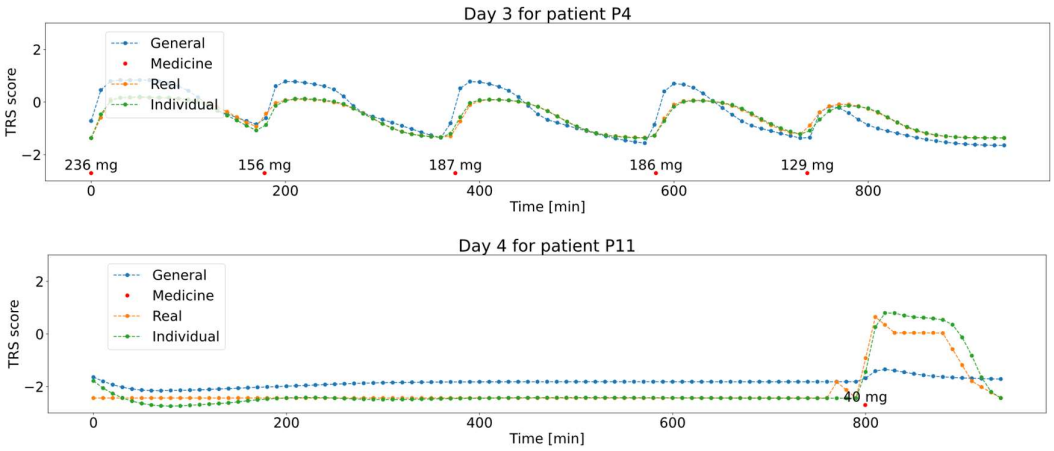

Figure 37 Charts presenting the performance of individual medicine response models for patients P4 and P11, with medicine doses (red dots), patient state curves based on validation data (orange), individual ML model (green) and the general ML model (blue) - before retraining.

5.3.3.2. Correlation analysis

To identify potential influences of patient-specific characteristics on medication response, a correlation analysis was conducted for 18 patients, excluding one due to incomplete data. This analysis aimed to uncover any significant relationships between patient demographic information and medication parameters. The correlations were calculated for basic demographic patient data and MDS-UPDRS scale scores (Figure 38) and outcomes from specific medical scales related to PD (Figure 39). The scales consist of questions with answers that have assigned scores. Correlations are calculated for both the total score of the scale and for specific parts focusing on different aspects of PD to capture the extent and burden of the disease.

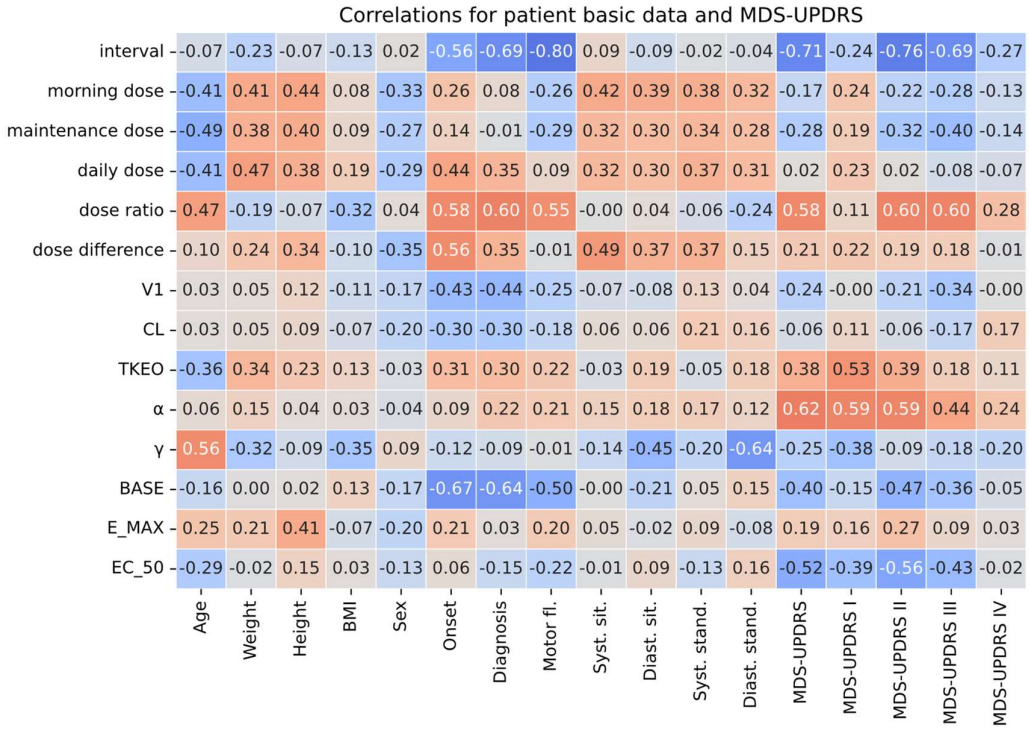

Figure 38 Correlation matrix presenting correlation values between patient basic data, MDS-UPDRS results and medication parameters

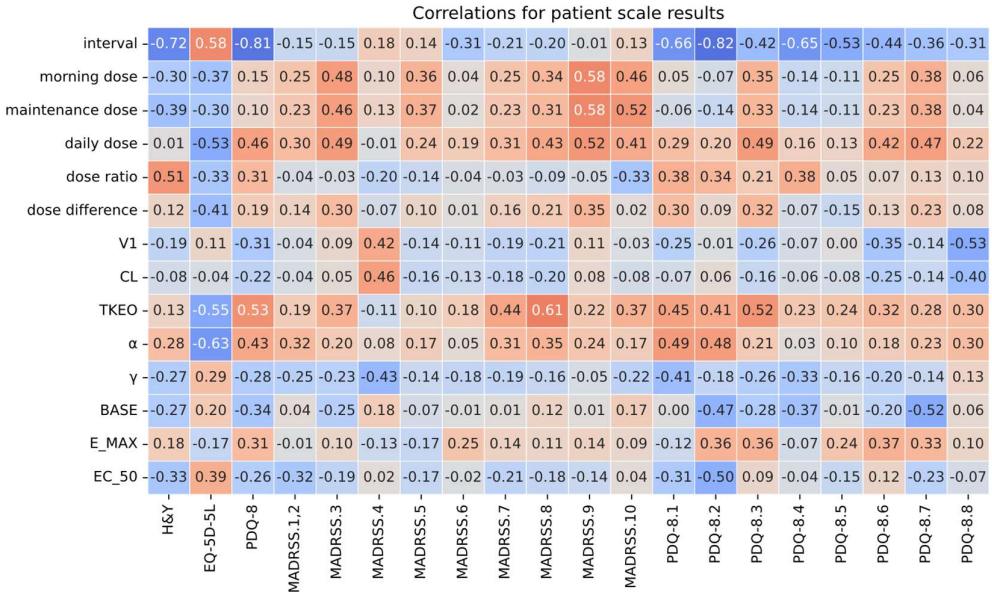

Figure 39 Correlation matrix presenting correlation values between patient scale results and medication parameters

The analysis of the values presented in these matrices identified high correlations between patient features and medication parameters. Specifically, dose time intervals are highly correlated with the onset of motor fluctuations, the time since diagnosis of the disease, and PDQ-8 and MDS-UPDRS scale results. Parameters related to dose size show the strongest correlations with results from the MDS-UPDRS, MADRSS and EQ-5D-5L scales.

These correlations are crucial as they indicate how different aspects of the patient's condition and medication dosing parameters interact, providing insights into the personalized adjustment of treatment regimens. In particular, understanding the correlations with MDS-UPDRS scales is vital as they directly reflect the severity and progression of PD symptoms. Medication dosing parameters show the interaction between the medicine’s effects on the patient and the patient's influence on the drug's efficacy. Analyzing their correlation with patient information and scale results can help explore whether these factors can be used to predict patient responses to medication.

To develop a general levodopa-response model applicable to all patients, parameters with the highest absolute correlation values (0.5 for dosing/interval features and 0.6 for PK/PD model parameters) were selected. This approach, utilizing 32 identified parameters, aims to minimize the training data required for the model.

5.3.3.3. General medicine response prediction model with correlations

Before initiating the training of the general model, the 32 additional features identified from the correlation analysis were normalized (mean = 0 and standard deviation = 1). This was performed as preparation for PCA. PCA was then applied to condense these features into 5 principal components, accounting for 86.2% of the data variance, satisfying the pre-set threshold of 80%. Each variant presented in Table 35 was trained using 2 distinct approaches: one utilizing additional 32 patient-specific features selected from the correlation matrix, and the other with 5 principal components derived from PCA. Table 38 presents the metrics computed on the validation set for these models.

Table 38 Values of selected metrics for 3 ML architectures using 2 different sets of additional input features. The best performing models for each approach are presented in bold

Approach	Model	MSE	MAE	R^2
32 features	I-(8)64,64	**0.113**	**0.183**	**0.905**
32 features	I-(8)64,64-diff	0.134	0.210	0.888
32 features	I-(16)16,16	0.176	0.239	0.852
5 PCs	I-(8)64,64	**0.144**	**0.217**	**0.879**
5 PCs	I-(8)64,64-diff	0.166	0.248	0.861
5 PCs	I-(16)16,16	0.229	0.309	0.808

The best-performing models, highlighted in bold, are selected for developing levodopa intake schedules through optimization algorithms.

5.3.4. Discussion

This part of the study focused on application of ML models for prediction of the response to medication (changes in patient state) for real patients examined and available in the Swedish dataset. It is the next step, after building such models on generated patients in the previous section and in Gutowski et. al [69]. It validates the application of proposed patient-specific ML models for real PD patients, trained on clinician assessments and PK/PD model data The metrics achieved for real patients are slightly worse than the ones calculated for synthetic patients before. However, their values still confirm the good fit of the person-specific models. The worse performance can be attributed to the fact that in this case the training process used also real measurements which are not as predictable as PK/PD generated outputs.

Using a dataset of real patients allowed to introduce models using patient clinical and demographic data, new general ML models for predicting patient reactions to levodopa medication. These can be used as a tool for initial dosing recommendations, especially in situations where detailed patient-specific data, regarding medication history, is not available. While these models did not perform as well as patient-specific ones, they still provided reasonable predictions that could serve as a starting point for treatment. The personalization of the models' outputs was achieved by integrating patient demographic and clinical data into the modeling process through correlation analysis, allowing the models to account for various factors that could influence medication response, such as

disease history and results of commonly used state evaluation scales in PD. The biggest advantage is that these models can produce patient-specific response to levodopa based on only PD scale results and demographic data of the patient in question, which was not the case for PK/PD and patient-specific ML model approaches.

When working with machine learning it is also necessary to remember how these models are trained. The algorithms try to detect and capture the relationship between the input and output data and if the dataset is small, or not representative, the models can learn to capture some dependencies, which do not truly describe the researched phenomenon – in this case the response to medication. Therefore, especially in early phases of developing such solutions, it is necessary for the clinician to inspect and validate the outputs of these models, to ensure the well-being of the patient at all cost.

5.4. Conclusions

In this chapter the modelling of PD patient medication response using ML models was explored, using data from both simulated and real patients. The study covered various aspects, including the application of artificial neural networks, the comparison of different model architectures, and the integration of clinical and demographic data.

The experiments confirm the effectiveness of machine and deep learning models in modelling individual responses to medication. The LSTM architecture demonstrated a clear advantage over other models, achieving god metrics values. However, challenges such as the limitations of the impulse model and the need for diverse and representative datasets were identified.

Future research regarding medicine response models, should focus on expanding the dataset with more diverse patient data and exploring additional patient-specific factors that could further refine the treatment models including eating habits, and physical activity as well as dosing other medications used in PD management such as dopamine agonists, MAO inhibitors and amantadine derivatives.

Based on the experiments with both simulated and real patient data, the following recommendations are proposed for further development of methods for assessing patient medication response:

- Focus on using LSTM models for real patient data to improve prediction accuracy.

- Expand datasets to include a more diverse range of patient data.
- Explore additional patient-specific factors, such as diet, physical activity, and other medications.
- Consider using general ML models for initial dosing recommendations when detailed patient-specific data is not available.

Further research will continue to explore these areas, aiming to enhance the accuracy and applicability of ML models in predicting PD patient responses to medication.

6. Medicine schedules creation

The previous chapter described the construction of a function capable of predicting the future states of patients based on their previous states and applied medicine (levodopa) doses. This provides the ability to predict response to specific doses and can be used to generate the values of patient state when a defined medication intake schedule is applied. With the optimal state defined it is possible to compare different medicine intake schedules and find the best one for every patient. This can be done through optimization.

In this chapter, the following topics are covered:

- various options for defining the dosing optimization problem,
- three optimization methods:
  - two heuristic methods based on evolutionary algorithms,
  - one method using a reinforcement learning approach,
- results for different variants of the optimization problem,
- comparison of achieved results with these presented in [64].

6.1. Medicine schedule optimization methods

6.1.1. Conventional optimization

Designing an optimal medicine schedule is a challenging task, as there is no universally defined objective function in literature, which can accommodate the needs of all patients. During the optimization process, the decision variables represent the times and sizes of all the doses taken throughout the day. In the simplest scenario this would result in *2n* decision variables, with *n* representing the number of doses:

$$T = [t_1, \Delta t_2, \dots, \Delta t_n], \qquad D = [d_1, d_2, \dots., d_n], t_i \in R_+, d_i \in Z_+ \qquad \text{Eq. 47}$$

T – intake times (time of first dose and time intervals between next doses),

D – dose sizes,

$$\Delta t_i = t_i - t_{i-1},\ i = 2, .., n. \qquad \text{Eq. 48}$$

Instead of defining the T vector as times of consecutive doses, it is defined to include the time of the first dose and the intervals between consecutive doses. This allowed the optimization algorithms to converge faster, approximately 2.3 times faster,

due to the changes of individual time variables, which made it easier to define the variable bounds.

It is important to consider discrete values for the times and sizes of doses. Typically, the size of a dose is defined by the size of a single pill or its fractions, requiring discrete steps. This approach reflects practical dosing and simplifies the optimization process. Although times can be continuous, in practice, doses are not administered at exact seconds. Using discrete time steps (e.g., every 10 minutes) reduces the number of possible solutions without significantly compromising accuracy.

For the purpose of optimization, a framework has been developed that prepares the optimization tasks. Firstly, it allows to enforce some constraints on the decision variables:

- forcing the intake times to have discrete values with a defined step e.g., 10 minutes,
- ensuring that the first dose is taken at wake-up (omitting t_1 from optimization, as it is fixed),
- restricting the optimization to 2 dose sizes – a morning dose (d_{mor}) and equal maintenance doses (d_{main}), simplifying the dose size variables,
- enforcing equal time intervals between doses, using a single interval Δt for all doses after the first.

These listed constraints can be combined, leading to a modification in the definition and number of the decision variables. The impact of each constraint, analyzed individually, is presented in Table 39.

Table 39 Definition of decision variables, when each of the described constraints is applied. The differences in the dose times and sizes from the original decision variables are written in bold

Const. number	**Dose times**	**Dose sizes**
None	$T = [t_1, \Delta t_2, \cdots, \Delta t_n],\ t_i \in R_+$	$D = [d_1, d_2, \cdots, d_n],\ d_i \in Z_+$
1	$T = [t_1, \Delta t_2, \cdots, \Delta t_n],\ t_i \in \boldsymbol{Z}_+$	$D = [d_1, d_2, \cdots, d_n],\ d_i \in Z_+$
2	$T = [\Delta \boldsymbol{t_2}, \cdots, \Delta \boldsymbol{t_n}],\ t_i \in R_+$	$D = [d_1, d_2, \cdots, d_n],\ d_i \in Z_+$
3	$T = [t_1, \Delta t_2, \cdots, \Delta t_n],\ t_i \in R_+$	$D = [\boldsymbol{d_{mor}}, \boldsymbol{d_{main}}],\ d_i \in Z_+$
4	$T = [t_1, \Delta \boldsymbol{t}],\ t_i \in R_+$	$D = [d_1, d_2, \cdots, d_n],\ d_i \in Z_+$

Algorithms that are used for optimization also require defining the bounds for each variable, following previous publications [64], the smallest time interval is set to 90 minutes (Eq. 49), and the maximum value is defined by the number of doses and the day length for the patient (Eq. 50). The time of the first should not also be smaller than the wake-up time (Eq. 51). The dose sizes can vary between 0 and 400 mg (Eq. 52).

$$\Delta t_i \geq 90\ [min], \qquad i = 2 \ldots n \qquad \text{Eq. 49}$$

n – number of doses

$$t_1 + \sum_{i=2}^{n} \Delta t_i \leq t_d,\ t_d - falling\ asleep\ time \qquad \text{Eq. 50}$$

$$t_1 \geq t_u,\ t_u - wake - up\ time \qquad \text{Eq. 51}$$

$$0 \leq d_i \leq d_{max} = 400\ [mg], \qquad i = 1 \ldots n \qquad \text{Eq. 52}$$

To optimize the medicine schedule the framework establishes the following criteria:

- the sum of doses

$$\sum_{i=1}^{n} d_i \qquad \text{Eq. 53}$$

- the sum of squares of difference between the states and optimal state

$$\int_{t_u}^{t_d} (st(t,T,D) - \theta)^2 dt \qquad \text{Eq. 54}$$

- the area outside the target range

$$\int_{t_u}^{t_d} max^2(0, \theta_{min} - st(t,T,D),\ st(t,T,D) - \theta_{max}) dt \qquad \text{Eq. 55}$$

- The area below the threshold

$$\int_{t_u}^{t_d} max^2\big(0, \theta_{th} - st(t,T,D)\big) dt \qquad \text{Eq. 56}$$

n – number of doses,

d_i – size of dose *i*,

θ – optimal patient state,

$\theta_{min}, \theta_{max}$ – the lower and upper bound of the target patient range,

θ_{th} – the threshold patient state,

st(t, T, D) – patient state function, represents the state of the patient at time *t* under the schedule represented by *T* (intake times) and *D* (dose sizes). This function can be used to generate the trajectory of patient's state over a specified period by supplying different *t* values. It can be defined using previously established PK/PD models and ML models to predict the patient's medicine response as discussed in Medicine response model chapter (p. 94) – which includes the definition of history and impulse models.

The criteria can be used individually or combined with custom weights, providing greater flexibility during the optimization process and potentially leading to better results. When used as an objective function, the optimization aims to minimize the criteria, finding decision variables that provide the lowest possible value.

Given the non-linear character of the objective functions (Eq. 54 - Eq. 56) (the value is calculated using an artificial neural network or PK/PD model and the functions have a quadratic nature) and the discrete nature of decision variables (which generally increases the computational complexity of the problem), only a subset of optimization algorithms is suitable. In this case, heuristic algorithms from the evolutionary algorithms class have been selected. The genetic algorithm (GA) [134] and differential evolution (DE) [135] have been chosen as they provided satisfactory results in solving this problem. These algorithms can handle non-linear objective functions and constraints, they also do not fall into local minimums, due to the use of mutation operators. Used algorithms iteratively try to improve candidate solutions to improve the value of the objective function. It is performed using evolutionary algorithm operators that are inspired by natural selection.

The GA was implemented in Python using the DEAP library [136]. It offers convenient utilities to perform the optimization and includes operators for selection, crossover, and mutation. The library is easily extensible and provides a simple interface for modifying or creating custom operators. It also allows for constructing individuals consisting of variables of different types, which is necessary to handle discrete dose sizes and continuous intake times.

The framework that has been created has an interface supporting DEAP and directly provides the fitness function (in this case it is defined as the negative value of the objective function) and the definition of the decision variables – their datatype and bounds, which indicate the minimum and maximum values for each variable. As explained before the datatype and meaning of the decision variables is dependent on the selected constraints from Table 39. The number of decision variables can be *2n* (where *n* represents the number of doses) – when no additional constraints are imposed. However, when all constraints are applied, it can be reduced to just 3 – $[\Delta t, d_{mor}, d_{main}]$. Adding these constraints makes the dosing process easier and simpler to follow, but the created schedule will not be as flexible, and the patient might experience more PD symptoms and levodopa induced dyskinesia during the treatment. When determining the dosing schedule, a compromise must be made between the usability of the suggested therapy and the patient's condition throughout the day.

To create a genetic algorithm individual the decision variables arrays – intake times and dose sizes are concatenated. This means that the times will always be placed in the beginning of the array representing the individual and the dose size information will be in the back e.g. $[t_1, \Delta t_2, d_1, d_2]$, $[\Delta t, d_{mor}, d_{main}]$. The composition of the individual (dose schedule) varies based on the number of constraints applied. It can consist solely of discrete variables (when constraint 1 is applied) or a combination of continuous variables (first part of the array) and discrete variables (second part of the array). This distinction should be considered when selecting and applying the operators. The optimization task utilizes the following operators:

- Crossover – two-point crossover – the algorithm randomly chooses two indexes in the individual that indicate where two individuals should be crossed over producing two new individuals.
- Mutation – applied to each variable in the individual with a predefined probability:
    - Gaussian mutation for continuous variables (intake times) – the value of the variable is updated with the value generated with the Gaussian distribution (specified mean and standard deviation). To ensure that the value remains within predefined bounds, a check is performed after each mutation. If the new value exceeds the bounds, it is set to the nearest bound.

- Integer uniform mutation for discrete variables – the value of the decision variable is replaced with a new integer value generated from a uniform distribution within the specified bounds.

- Selection – Tournament selection – randomly picks *k* individuals and selects the best individual based on the value of the fitness function.

The implemented GA utilizes operators outlined in Figure 40.

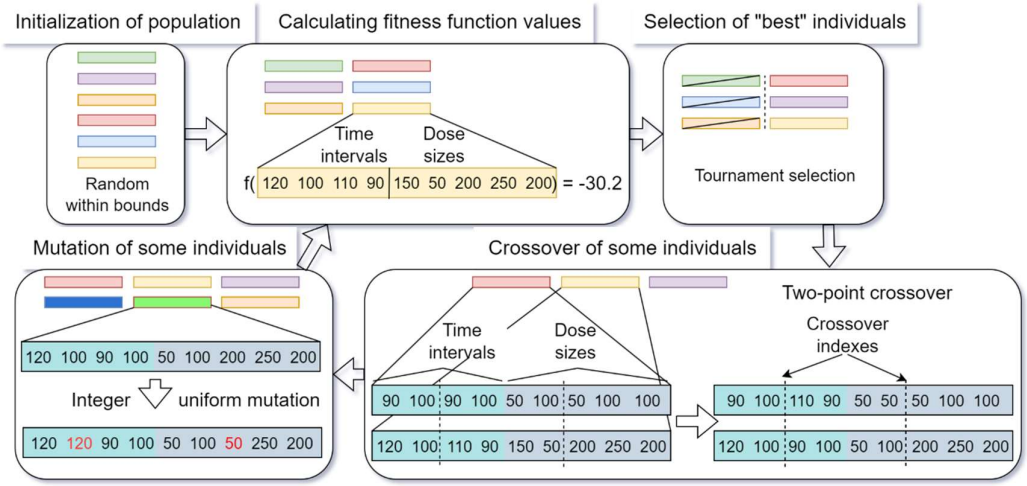

Figure 40 Steps (application of operators) in the GA used to optimize the medicine intake schedules

The crossover operation is performed with a probability (CXPB) of 0.5, while the mutation operation has a probability (MUTPB) of 0.2. Each variable undergoes mutation with probability of 0.1. These values were determined through literature review and experiments. The selected operators are simple, fast to apply and allow the algorithm to quickly find optimal and suboptimal solutions, thus outperforming more complex operators. The overall execution of the algorithm is presented in Figure 41.

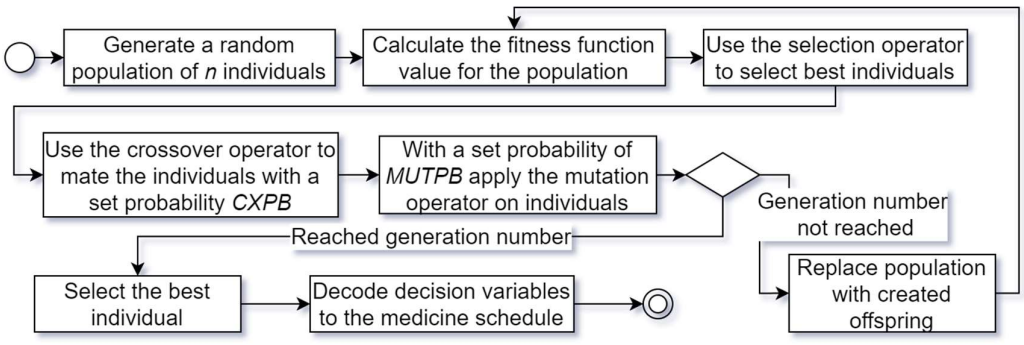

Figure 41 Execution of the GA for medicine schedule optimization in PD

The size of the initial population and the number of generations for optimizing the medicine intake schedules depended on the number of decision variables. However, the population size never exceeded 200 and the number of generations was limited to 100. These limits were chosen to ensure computational efficiency while maintaining solution quality.

Differential evolution, which is also an example of evolutionary algorithms, is a powerful optimization algorithm that has gained widespread popularity due to its simplicity, efficiency, and effectiveness in solving complex optimization problems. The algorithm is based on the principle of recombination and selection, where candidate solutions are perturbed by a set of differential vectors to generate a trial solution. The trial solution is then compared with the current best solution, and the fitter of the two is selected as the parent for the next generation. The algorithm requires the specification of 3 parameters: the population size (n), crossover probability (CXPB) and the differential weight (F). The steps of the algorithm are outlined in Figure 42, while Figure 43 presents how each step is executed.

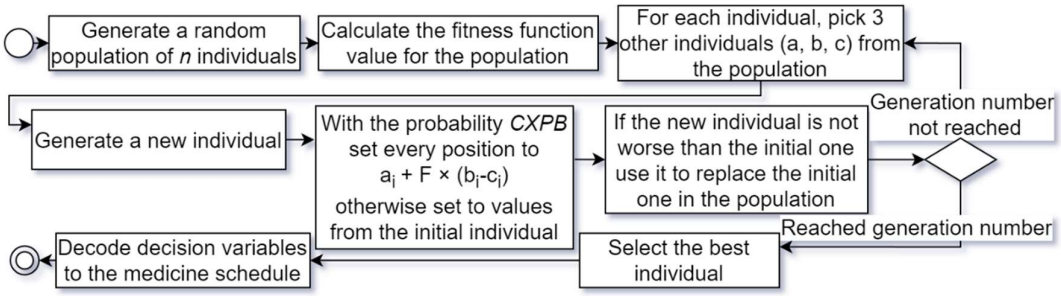

Figure 42 Execution of the DE algorithm for medicine schedule optimization in PD

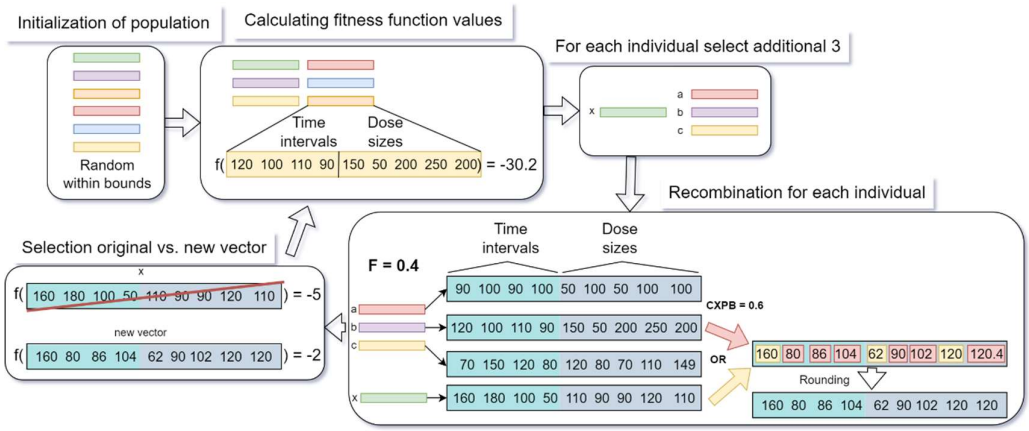

Figure 43 Examples demonstrating each step of DE

The medicine schedule optimization utilized the *differential_evolution* implementation from Python's *SciPy 1.10.1* library. It provides a simple interface, requiring the objective function callback and the decision variable bounds and also accepts other parameters including the *integrality* constraint, determining which of the decision variables are constrained to integers. By combining the *integrality* and *bounds* parameters the constraints from Table 5 were applied. The parameters to the algorithm were supplied by the previously designed framework, which just like in the case of GA takes care of coding and decoding decision variables. The pseudocode for both, the GA (Pseudocode 1) and DE (Pseudocode 2), has been presented below.

```
Input: n > 2, g_max
Initialize: the population x with n vectors sampled at random;
Set: g = 0
While g ≤ g_max Do
  x = select(x)
  For i = 1 to n/2 Do
    If rand(0, 1) < CXPB Then
      crossover(x[2i], x[2i+1])
    End If
  End For
  For i = 1 to n Do
    If rand(0, 1) < MUTPB Then
       mutate(x[i])
    End If
  End For
  g = g + 1
End While
```

Pseudocode 1 Genetic algorithm pseudocode

Input: F > 0, n > 2, g_{max}, m > 1, integrality
f – objective function
Initialize: the population *x* with *n* vectors of length *m* sampled at random;
Set: g = 0
While g ≤ g_{max} **Do**
  For j = 1 **to** n **Do**
    Select distinct individuals *a, b, c* from *x* at random, $a \neq b \neq c \neq x[j]$
    For i = 1 **to** m **Do**
      $v[i] = \begin{cases} a[i] + F * (b[i] - c[i]) & \text{if } rand(0,1) < CXPB \\ x[j,i] & \text{otherwise} \end{cases}$
      If integrality[i] **Then**
        v[i] = round(v[i])
      End If
    End For
    If f(v) ≤ f(x[j]) **Then**
      x[j] = v
    End If
  End For
  g = g + 1
End While

Pseudocode 2 Differential evolution pseudocode

Various experiments were conducted with other algorithms and combinations of them; however, they did not provide significantly better results or performance and the libraries providing these algorithms were less clear and documented. Both algorithms were used for schedule optimization; however, only the best result is showcased in the paper.

6.1.2. Reinforcement learning

Reinforcement learning [137] is a subfield of machine learning that deals with how an agent should learn to make decisions through trial and error. In this system, RL is employed to adaptively learn the most efficient medication schedules for patients based on real-time data and predictive models. In reinforcement learning, an agent interacts with an environment and learns by receiving feedback in the form of rewards or penalties for its actions. The agent's goal is to maximize its cumulative reward over time, which is often referred to as the "return."

The agent in this RL model is trained to make decisions on the optimal time and dose size of medication based on patient-specific parameters. RL algorithms typically use a policy, which maps states to actions, to make decisions. The policy is learned through

exploration of the environment and exploitation of past experiences. The agent's actions in the environment are guided by the policy, which is updated based on the feedback received from the environment.

The application of RL in creating individualized medicine intake schedules has been previously presented in a publication [65]. This paper applies RL to learn the optimal strategies to take medications – time and size of doses. One of the main challenges in using RL for this task is correctly defining the environment. In this case, the environment is constructed based on two types of models: PK/PD model, ML model, both created to predict the patient's state in future under a specific medication schedule. In case of optimization the course of the TRS scores for the whole day was created at once, for RL, the models have a wrapper which allows generating values step by step (step size equal to 10 minutes). The environment was created using the *Gymnasium* library, a fork of the OpenAI's Gym library [138].

The library required the definition of reset() and step() methods, which are called to reset the state of the environment and make a next step by taking an action, as well as the observation and action space. The action space was defined as single integers from 0 to n_{max} representing dose sizes of medicine. The number of available actions is patient specific and depends on the dose step, e.g., a patient with maximum dose of 400 mg and a dose step of 25 mg would have 17 available actions: 0 – no medication, 1- 25 mg, 2 - 50 mg, …, 16 – 400 mg. The shape and interpretation of the observation space varied depending on the model that was used. In the PK/PD model, observations were represented by the amount/concentration of medicine in different compartments a_0, a_1, a_2, c_e, resulting in a 4-variable environment state. For the history model, the state of the environment consisted of $2k+1$ variables, where k represents the number of recent doses considered. The variables captured the resent patient's state, the sizes of k previous doses and the elapsed time since their administration. For the impulse model, the LSTM cells capture and process temporal patterns in the patient's condition. The LSTM's internal states serve as the observation space for the RL model. This allows the RL agent to base its decisions on a richer, time-aware representation of patient conditions. Due to the use of tanh function in LSTM cells, the bounds for the variables were set -1 to 1.

The reset method is expected to reset the state of the environment and return the initial observation and the auxiliary info dictionary which helps in result interpretation.

In the presented implementation, the environment collects the information about past patient's states and administered medication doses since last reset. These values, along with previously defined environment state and time are reset upon invoking the reset method.

The step method takes an action as an argument, which is then translated into the appropriate dose and then the input for the state prediction model is prepared. The TRS score for the next step is calculated and if it exceeds a specified maximum value for the patient the episode is terminated – the value is patient specific. For every patient, a target TRS score range has been selected and the reward is the negative value of the objective function from the optimization task, to make these approaches comparable. The episode is terminated when the patient's state exceeds the maximum value or after reaching 110 steps, equivalent to a duration of over 18 hours (typically longer than the time patient is awake. The method returns the current observation, the reward for the step, Boolean values indicating whether the episode was terminated or truncated and additional information.

Depending on the selected dose step, the RL agent may generate medicine intake schedules that expect the user to take small medicine doses at each time step (e.g., every 10 minutes). To make the model's suggestions more applicable in real-life situations, we have implemented a constraint that imposes a minimum time interval between medication doses. After a dose is taken, the agent must wait a predefined number of time steps before taking the next action. In the environment implementation, this is handled by a loop simulating several steps in the step() function when a dose is taken (assuming the agent selects an action other than 0).

The network training process utilized the Stable Baselines3 library [139]. It offers implementations of multiply reinforcement learning algorithms, including both on-policy and off-policy approaches. The applications of algorithms differ and depend on the definition of the action space. When handling discrete action spaces, the following algorithms can be applied: A2C (on-policy), DQN (off-policy) and Proximal Policy Optimization (PPO) (on-policy). After conducting trials, it was determined that PPO provided the best results. Therefore, it was the only algorithm chosen to create schedules for all the patients. The process of training the RL agent is presented in Figure 44.

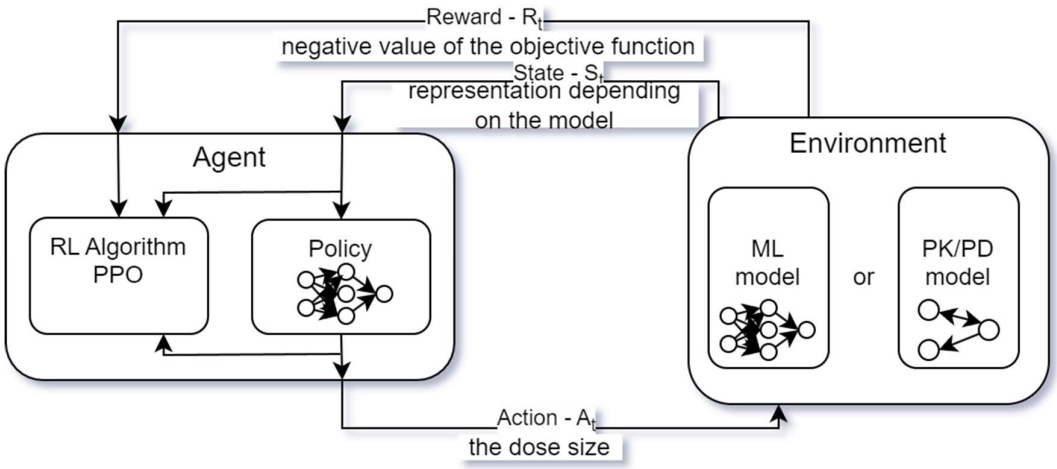

Figure 44 The process of training the RL agent to suggest appropriate medicine doses in different patient's conditions using the PK/PD or ML model as the environment

PPO is a RL algorithm that updates the policy in a controlled approach, preventing drastic changes. PPO optimizes the agent's policy and its evaluation of different states (value function) to maximize expected rewards. PPO balances exploration (trying new actions to gather more information) and exploitation (using known information to make the best decisions).

This is achieved by using a clipped objective function (Eq. 57) that limits how much the policy can be changed with each update, providing a stable and reliable learning process. The algorithm works by collecting data for the agent's interaction with the environment. It computes the advantages and targets for these actions and then adjusts the policy based on the results. Advantages represent the difference between the observed return (cumulative reward) and the expected return. Targets are the actual returns the value function aims to predict, derived from the observed rewards and the estimated future rewards.

$$L^{CLIP}(\theta) = E_t\left[\min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}A_t,\ \text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)A_t\right)\right]$$

Eq. 57

θ – parameters of current policy, a_t – the action taken at time t,

s_t – the state at time t, A_t – the advantage estimate at time t,

ϵ – a hyperparameter that controls the clipping range,

$\pi_\theta(a_t|s_t)$ and $\pi_{\theta old}(a_t|s_t)$ – probabilities of taking action a_t in state s_t under the current and old policies, respectively,

clip(x,1−ϵ,1+ϵ) clips the value of x to be within the range [1−ϵ,1+ϵ]

The specific steps of the PPO algorithm are outlined in Pseudocode 3.

Input: θ_initial, epochs, steps_per_epoch, minibatch_size, ε, α, β
Initialize: policy π with weights $\theta = \theta_{initial}$, value function V with weights ϕ
Set: g = 0
While g ≤ epochs **Do**
  Initialize buffer B
  For step = 1 to steps_per_epoch **Do**
    a_t = sample action from $\pi(s_t, \theta)$
    $s_{\{t+1\}}$, r_t = environment step with action a_t
    Store (s_t, a_t, r_t, $s_{\{t+1\}}$) in buffer B
  End For
  Compute returns R_t using rewards r_t
  Compute advantage estimates A_t using rewards r_t and value function V
  Set $\theta_{old} = \theta$
  For optimization_step = 1 to minibatch_size **Do**
    Sample minibatch from B
    For each (s_t, a_t, A_t) in minibatch **Do**
      $r_t(\theta) = \pi_\theta(a_t|s_t) / \pi_{\theta old}(a_t|s_t)$
      $L_{clip} = \min(r_t(\theta) * A_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon) * A_t)$
      Update rule for θ: $\theta = \theta + \alpha * \nabla_\theta(L_{clip})$
      $V_{loss} = (V_\phi(s_t) - R_t)^2$
      Update rule for ϕ: $\phi = \phi - \beta * \nabla_\phi(V_{loss})$
    End For
  End For
  g = g + 1
End While

Pseudocode 3 PPO algorithm

6.2. Simulated patients

To compare results obtained from optimization and reinforcement learning with previously published results, the algorithm introduced in [64] was used to find medicine intakes schedules for each of the patients. This required the specification of 2 parameters for each patient:

- target range – range defining the minimum and maximum target state, defining the desired patient's state,
- threshold – the minimum patient's state that should not be exceeded before the next dose is taken.

The objective was to minimize the area outside the target range while ensuring the patient's state did not fall below the threshold after taking the first dose. The threshold and the target range are individual for every patient, the threshold being 10% of the maximum patient's state (BASE + E_{MAX}) and target ranging from 20% to 40% of the maximum value for patient's state.

The range for doses sizes was specified to [0, 400] for morning dose and [0, 300] for maintenance doses. The schedules considered in this paper used two dose step values: 5 mg, to reflect the microtablets used in [64] and 50 mg, to reflect traditionally available levodopa pills. This specification, along with all four constraints from Table 39, resulted in the formulation of the following optimization task, based on previously defined criteria (Eq. 55 and Eq. 56):

$$\min_{(T,D)} \int_{t_u}^{t_d} max^2(0, \theta_{min} - st(t,T,D),\ st(t,T,D) - \theta_{max})dt$$
$$\theta_{min} = 0{,}2 \cdot (BASE + E_{MAX}), \qquad \theta_{max} = 0{,}4 \cdot (BASE + E_{MAX}) \qquad \text{Eq. 58}$$
$$(T = [\,\Delta t], D = [d_{mor}, d_{main}])$$

with the following constraints:

$$\int_{t_u}^{t_d} max^2\big(0, \theta_{th} - st(t,T,D)\big)dt = 0 \qquad \text{Eq. 59}$$
$$\theta_{th} = 0.1 \cdot (BASE + E_{MAX})$$

$$\Delta t\ \epsilon\ \{90, 100, .., 90 + k \cdot 10, .., 90 + n \cdot 10\},\ \ t_u + (n-1) \cdot \Delta t \le t_d \qquad \text{Eq. 60}$$

$$d_{mor} \epsilon \left\{k \cdot d_s : k = 1, .., \frac{400}{d_s}\right\},\ d_s\ \epsilon\ \{5, 50\} \qquad \text{Eq. 61}$$

$$d_{main} \epsilon \left\{k \cdot d_s : k = 1, .., \frac{300}{d_s}\right\},\ d_s\ \epsilon\ \{5, 50\} \qquad \text{Eq. 62}$$

A number of experiments have been performed on simulated patients to create medication intake schedules, as listed in Table 40.

Table 40 Optimization experiments performed on simulated patients to create medicine intake schedules

Optimization	**State function**	**Constraints**	**Dose steps**
Exhaustive search	PK/PD model	1, 2, 3, 4	5 mg, 50 mg
Evolutionary algorithms	PK/PD model	1, 2, 3, 4	5 mg, 50 mg
		1, 2, 3	50 mg
		1, 2, 4	50 mg
		1, 2	5 mg, 50 mg
	ML model	1, 2, 3, 4	5 mg, 50 mg
RL	PK/PD model	1, 2, 3, 4	5 mg, 50 mg
	ML model	1, 2, 3, 4	5 mg, 50 mg

The exhaustive search method was used in a study by Thomas et al. [64] was employed here to create a benchmark and find the best possible solutions. Furthermore, the evolutionary algorithms were tested to evaluate their performance and the time to find the optimal or satisfactory solutions under different sets of constraints. The experiments with ML models were conducted to assess their effectiveness in approximating patient states when used by an optimization algorithm. Finally, both of these approaches for predicting the response to medication were used by RL.

6.2.1. Results

6.2.1.1. Exhaustive search

The exhaustive search algorithm generated all possible schedules, considering all the constraints described in Table 39. The best schedules generated using the algorithm for the patients are presented in Table 41 and Table 42, where score column represents the values of the objective function (Eq. 58).

Table 41 Selected medicine intake schedules for each of the patients represented by: the dose time interval (minutes), morning and maintenance dose sizes (mg). The schedules created with the 5 mg dose step. The score column represents the values of the objective function (Eq. 58)

Patient	Dose interval	Morning dose size	Maintenance dose size	Score
41	90	210	90	30.6
42	90	90	40	83.8
43	90	270	150	27.3
44	90	145	75	25.3
45	100	285	160	15.8
46	90	105	65	42.9
47	90	365	65	29.1
48	90	145	85	56.4
49	90	280	135	22.6
50	90	400	265	40.4

Table 42 Selected medicine intake schedules for each of the patients represented by: the dose time interval (minutes), morning and maintenance dose sizes (mg). The schedules created with the 50 mg dose step. The score column represents the values of the objective function (Eq. 58)

Patient	Dose interval	Morning dose size	Maintenance dose size	Score
41	90	200	100	38.9
42	100	100	50	96.0
43	90	300	150	27.9
44	110	200	100	28.1
45	110	300	200	17.1
46	170	350	200	50.6
47	120	400	100	29.7
48	100	200	100	65.3
49	130	400	250	25.9
50	90	400	300	43.9

Finding these schedules using exhaustive search ensures optimality; however, it takes a lot of computational time (480 s) and can be only performed with many constraints, which leads to excluding some solutions, that might represent better schedules.

6.2.1.2. Optimization

To evaluate the patient state prediction model further, optimalization was performed using not only the ML models, but also using the PK/PD model that was used to train the networks, allowing for comparison of their outputs. Two algorithms and a set of constraints from Table 39 were utilized in this evaluation. The following constrains combinations were explored:

- constraints 1, 2, 3 and 4 with 5 mg and 50 mg doses,
- constraints 1, 2 and 3 with 50 mg doses,
- constraints 1, 2 and 4 with 50 mg doses,
- constraints 1 and 2 with 5 mg and 50 mg doses.

Constraint 1 was consistently applied using a time step of 10 minutes, as the impulse models cannot handle doses not taken with a 10-minute step. Constraint 2 ensured that the first dose is taken immediately after wake-up. Constraints 3 and 4 reduced the number of accepted solutions but simplified the dosing process. Optimization with 5 mg doses was performed in only 2 cases, because the small dose size step allowed for precise dosing and removing a single constraint does not significantly improve the solution. In each of the experiments the (Eq. 58) objective function was used with an added penalty for every state value below the threshold (Eq. 59).

Initially, the optimization process using the PK/PD model was performed, to create a benchmark for ML generated models. The first set of constraints (1, 2, 3 and 4) allowed comparing the results with the exhaustive search results (Table 41 and Table 42), as proposed by Thomas et al. [64], to validate the optimization methods implementations used and their usability for solving this task. Both algorithms successfully found the optimal solution for all patients with the mean time of 4.96 s for DE and 21.9 s for GA, demonstrating significantly faster performance than the exhaustive search. The results for other constraints and dose step combinations are presented in Table 43.

Table 43 Optimization results for dose step and constraint combinations (based on PK/PD model). The score column represents the values of the objective function (Eq. 58)

Patient	**Score**			
	5 mg- 1, 2	50 mg – 1, 2	50 mg – 1, 2, 3	50 mg – 1, 2, 4
41	29.1	32.5	33.8	36.4
42	76.4	92.5	96.0	92.5
43	25.7	25.8	27.9	27.9
44	22.7	27.5	28.1	28.1
45	14.6	16.0	16.3	16.3
46	41.8	48.5	50.5	48.5
47	25.9	28.1	29.7	28.6
48	50.8	63.1	63.1	65.3
49	22.0	24.0	24.6	25.4
50	38.4	39.0	41.7	39.0

Removing constraints for the dose sizes and time intervals between doses led to an improvement in the suggested schedules, the value of the objective function (area outside the target range) decreased on average by 7%. In case of dosing with 5 mg microtablets the result of removing the constraints was not as significant as in case of 50 mg step doses. The ability to precisely set the medicine dose reduces the need for different maintenance dose sizes and different time intervals between doses.

After receiving the expected optimization results for medicine intake schedules using the PK/PD models, optimization was performed using the trained ML model (I-(8)64,64) to evaluate its applicability in optimization of medicine schedules. The achieved schedules slightly differed from the ones achieved using the PK/PD model, usually in the morning dose size. The Table 44 and Table present score values for the schedules calculated using the PK/PD model, to be able to compare them with previously received results. The schedules were generated when all 4 constraints were applied.

Table 44 Medicine intake schedules for 10 patients acquired through optimization using individual ML models with a 5 mg dose step. The score column represents the values of the objective function (Eq. 58) calculated using PK/PD models

Patient	Dose interval	Morning dose size	Maintenance dose size	Score
41	90	180	95	32.4
42	90	90	45	93.6
43	90	230	145	28.6
44	90	145	65	26.2
45	90	220	140	17.0
46	90	110	75	52.5
47	90	105	65	29.8
48	90	110	75	61.2
49	90	245	145	24.0
50	90	335	270	42.4

Table 45 Medicine intake schedules for 10 patients acquired through optimization using individual ML models with a 50 mg dose step. The score column represents the values of the objective function (Eq. 58) calculated using PK/PD models

Patient	Dose interval	Morning dose size	Maintenance dose size	Score
41	100	200	100	31.2
42	100	100	50	96.0
43	90	250	150	27.8
44	110	200	100	28.3
45	90	250	150	17.8
46	100	200	150	59.0
47	100	350	150	35.0
48	90	150	100	76.2
49	90	250	150	25.3
50	90	300	300	47.0

The high objective function values observed in the results acquired with the trained ML model can be caused by the imprecision of the trained model. Due to the small training (2 days) and validation (1 day) sets it was difficult for the model to learn the individual patient responses to medication, considering the fact that the 3 days were generated and included doses that would not typically be taken by the patient. Nevertheless, both the schedules and their evaluation in most cases reflected the sensitivity of the patients to medication, demonstrating the usability of selected ML models. A comparison of schedules generated using the PK/PD and ML models is presented in Figure 45 for a 5 mg dose step and Figure 46 for a 50 mg dose step.

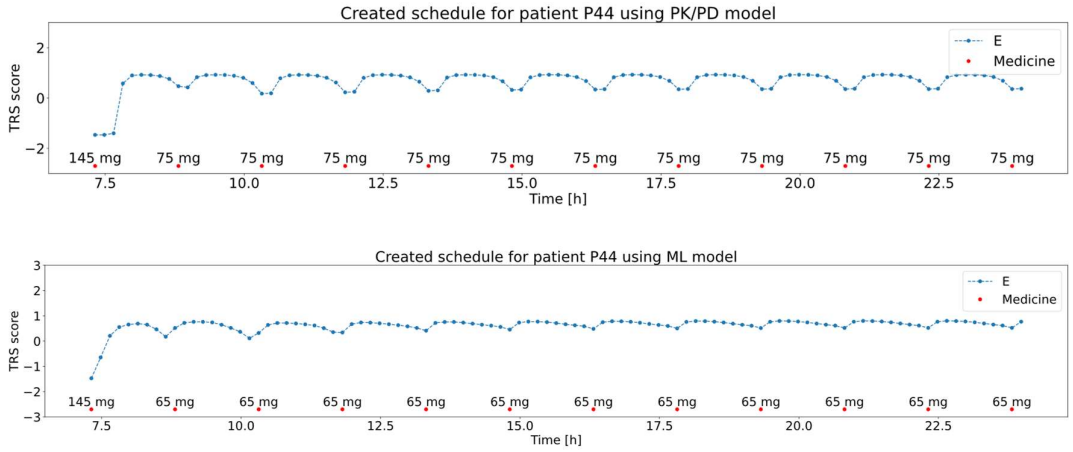

Figure 45 Generated medicine schedule (red dots) using optimization with patient's states generated with PK/PD model (top) and ML model (bottom) for patient P44 with a dose step 5 mg and patient TRS scores (blue line)

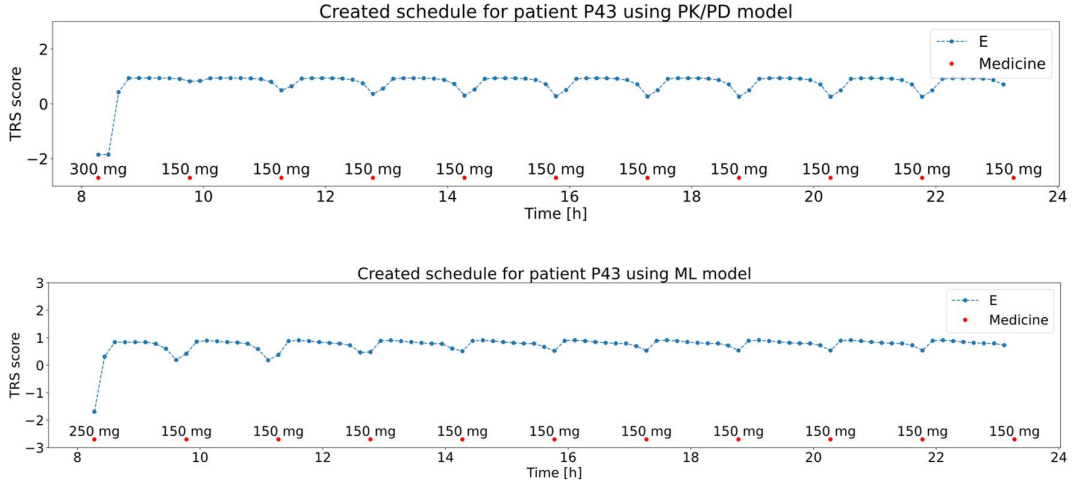

Figure 46 Generated medicine schedule (red dots) using optimization with patient's states generated with PK/PD model (top) and ML model (bottom) for patient P43 with a dose step 50 mg and patient TRS scores (blue line)

6.2.1.3. Reinforcement learning

The reinforcement learning agent was trained using 2 types of constructed environments: first based on PK/PD model and second based on the trained ML model (I-(8)64,64). Two dose steps were used (5 mg and 50 mg) and a minimum time interval of 90 minutes between doses was set to prevent overly frequent doses. The training process was performed with PPO's default parameters [139]; however, the network architecture was modified. A multilayer perceptron with 2 hidden layers was used, each consisting of 400 neurons for the 5 mg dose step and 200 neurons for the 50 mg dose step. The ReLU

activation function was selected, and the training was performed for 500 000 timesteps, resulting in similar outcomes to those achieved by classical optimization algorithms. Every 50 000 timesteps a callback was called and if there was no improvement the training process was stopped. After completing, the model was not only able to create a schedule using the initial patient's state, but also adapt and adjust medication in response to different states. The reward for each schedule was designed to be the negative value of the objective function (Eq. 58) from the optimalization task. After training, the agents were able to create schedules similar to the ones acquired using previously described optimization methods. The scores (negative rewards) of created schedules are presented in Table 46, while Figure 47 and Figure 48 present examples of generated schedules. For ML, the reward was recalculated using the PK/PD model to create comparable data.

Table 46 The scores (objective function values) acquired when generating individual schedules for each of the patients using PPO trained agents with PK/PD and ML environments and 2 dose steps – 5 mg and 50 mg

Patient	**Score (negative reward)**			
	PK/PD – 5 mg	PK/PD – 50 mg	ML – 5 mg	ML – 50 mg
41	28.2	38.2	40.5	37.7
42	76.5	86.2	94.9	96.0
43	27.3	27.8	34.8	34.3
44	22.1	28.0	32.3	31.1
45	15.5	16.9	18.9	17.8
46	38.8	46.9	39.6	55.4
47	26.4	28.9	34.4	35.4
48	48.7	63.6	73.1	74.3
49	26.7	25.3	30.0	30.8
50	36.8	38.6	41.3	48.0

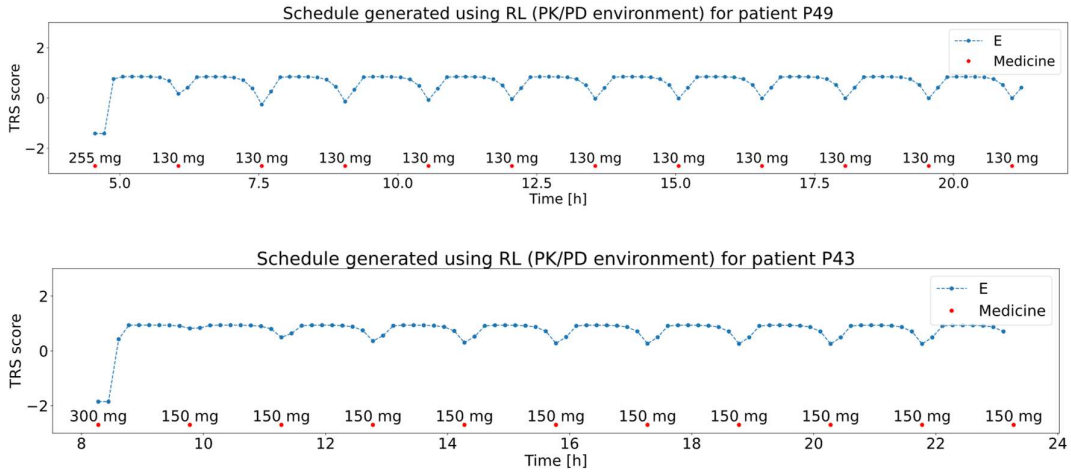

Figure 47 Generated medicine schedule (red dots) using RL with PK/PD environment for patients P49 and P43 with a dose step 5 mg (top) and 50 mg (bottom) with patient TRS scores (blue line)

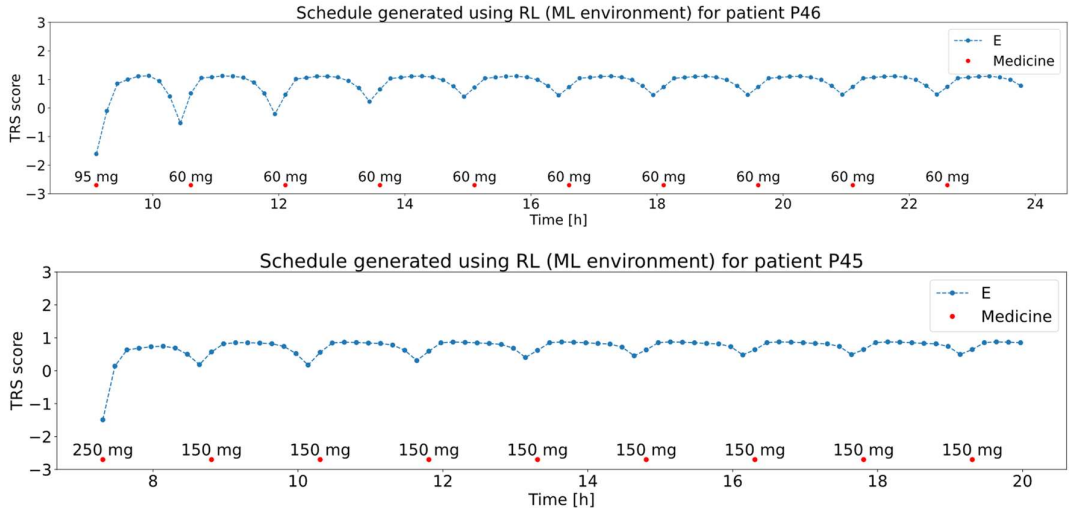

Figure 48 Generated medicine schedule (red dots) using RL with ML environment for patients P46 and P45 with a dose step 5 mg (top) and 50 mg (bottom) with patient TRS scores (blue line)

The reinforcement learning agent successfully learned the policies that maximized the obtained reward in both the PK/PD and ML environments. Both models enabled creation of schedules that minimized the area outside the target range. However, it should be noted, that in many cases, the results acquired with the ML environment are worse due to the inaccuracy of the trained model.

6.2.2. Discussion

Building medicine intake schedules can be performed using optimization once the objective function is defined. This section focuses on the optimization of individual

levodopa intake schedules for 10 simulated patients, which as a result provided the times and sizes of specific doses during the day.

A previous study [64] explored the possibility of selecting the best schedule out of all possible schedules (constrained to equal time intervals between doses and just 2 types of dose size using outputs created with PK/PD models. However, this approach uses heuristic optimization methods to improve the computation time allowing also to explore less constrained approaches – different dose sizes and different time intervals, which lead to more flexibility and result in lower values of the objective function. This might lead to the improvement of the quality of patient life and more personalization of schedules, making them more fitting to patient's daily schedules and decreasing the symptom severities.

Furthermore, the optimization was performed not only using PK/PD models to predict the patient state. The ML models described in the previous chapter were used too. This allowed for an additional verification of these models. During the optimization process, the models received inputs of the scope of the training and validation sets, these values were suggested by optimization algorithms. Comparing the optimal solutions generated using PK/PD models and ML models gives further insight into the performance of the models and their applicability. In most cases, the results acquired with the PK/PD and ML models were close. The differences could be attributed to the low number of samples selected for training. The representativeness of the dataset could be improved as well, to include samples that are close to the optimal solutions too. This could be achieved easily with synthetic data and is possible with real datasets too.

Using optimization for creating medicine intake schedules is a great approach when the schedule is generated once, before application, and is then prescribed to the patient. With the expanding impact of mobile technologies adaptive approaches should be considered too. These would allow to update the medicine intake schedule once it has been applied by the patient. The updates could happen if the patient's state deviates form the initial predictions, which could happen due to the model inaccuracies or various factors that are not initially considered or expected by the model. Due to the flexible nature of the optimization task, the optimization could be run again for the remaining part of the day, to update the schedule. However, this would require running the whole algorithm again. Therefore, the RL algorithm has been introduced to solve this problem

with ease. The trained RL agent is capable of providing actions to take (sizes of medicine doses) that will keep the patient in the desired state. RL can be applied during the day and also it can handle more complex models such as stochastic models, which could be created in the future to represent the response to medication. In this case, the trained agents were only used to create schedules for the whole day, to be able to compare the results with the once acquired using conventional optimization methods. While in most cases the RL approach provided worse results it is a different approach that can be applied in more diverse scenarios.

Both methods – conventional optimization and RL provided good results for creating medicine schedules.

6.3. Swedish dataset patients

In the previous chapter (Medicine response model), ML models were built to predict the response to levodopa of patients in the Swedish dataset. Two types of models were described: patient-specific models - retrained individually for every patient and general models that used additional inputs about the patient. These models, just like the models for synthetic patients, can be used to perform optimization in order to find the best medicine intake schedules. Conventional optimization and RL could be used, but since there is access to medicine schedules prescribed by the neurologist, the goal will be to compare the results of the proposed method with neurologist's suggestions, which in this case will be treated as the ground truth. Since the neurologist's suggestions are prescribed in advance, only the conventional optimization methods are used, because they provided better results in that case. RL finds its application, when the patient's state deviates from initial prediction. Every neurologist's prescription consists of the sizes of morning, maintenance doses and the time interval. Comparing schedules defined like this can be problematic (could be done using the objective function from the optimization task). Therefore, the optimization, in this case, focuses only on the size of the doses and the time intervals are set equal to those suggested by the neurologists. This approach simplifies the comparison to just comparing the sizes of morning and maintenance doses of levodopa/carbidopa pills.

6.3.1. Optimization

The optimization for simulated patients used an objective function consisting of two parts: minimizing the area outside the target range (Eq. 58). However, after getting

access to the Swedish dataset and consulting the researchers, it was discovered, that their study [64] defined the objective function based on patient characteristics. Additionally, only a portion of the day was considered when calculating the objective function's value. Specifically, for optimizing the morning dose, only the patient's state between the 22nd minute and the time of the first maintenance dose was considered. For optimizing the maintenance dose size, the states after the 10th hour were considered to reduce the impact of the morning dose. This methodology has also been applied in defining objective functions in this study. The following guidelines were established to create objective functions based on the value of the maximum state experienced by the patient:

1. If the maximum state of the patient is above 0.05 on the TRS scale:
    a. maintenance dose:
        i. minimize the differences between the target (defined as a percentage of maximum state) and the experienced states after intake,
    b. morning dose:
        i. minimize the falling out of the target range (defined as percentages of maximum state) of experienced states after intake.
2. If the maximum state is lower than 0.05 on the TRS scale:
    a. maintenance dose:
        i. minimize the falling out of the target range (defined as absolute values close to the maximum state) of experienced states after intake,
    b. morning dose:
        i. minimize the falling out of the target range (defined as absolute values close to the maximum state) of experienced states after intake.
3. The patient's state should also not fall below a defined threshold (0.5 below the maximum value) after the doses are absorbed.

These guidelines result in the objective function consisting of three components for the predicted TRS scores during the day:

- ensuring the patient state does not fall below a threshold (Eq. 56),
- minimizing the falling out of the target range for the morning dose (Eq. 55),
- minimizing the difference from the target state (Eq. 54) or the falling out of the target range (Eq. 55) for the maintenance dose.

The complete form of the objective function for patients whose maximum state is above 0.05 is presented in Eq. 63, while for those with a maximum state below 0.05, it is presented in Eq. 64. The constant k in the equations represents a high-value

number, ensuring that the optimization primarily focuses on meeting the requirement for values to remain above the threshold.

$$\min_{T,D} \int_{t_u+22\,min}^{t_1} max^2(0, \theta_{min} - st(t,T,D),\ st(t,T,D) - \theta_{max})dt + \\ + \int_{t_u+\,10h}^{t_d} max^2\big(0, \theta_{th} - st(t,T,D)\big)dt\ + \\ + k \cdot \int_{t_u+10h}^{t_d} (st\,(t,T,D) - \theta)^2 dt \qquad \text{Eq. 63}$$

$$\min_{T,D} \int_{t_u+22\,min}^{t_1} max^2(0, \theta_{min} - st(t,T,D),\ st(t,T,D) - \theta_{max})dt + \\ + \int_{t_u+10h}^{t_d} max^2(0, \theta_{min} - st(t,T,D),\ st(t,T,D) \qquad \text{Eq. 64} \\ - \theta_{max})dt + + k \cdot \int_{t_u+10h}^{t_d} (st\,(t,T,D) - \theta)^2 dt$$

When performing the optimization for simulated patients, it was noticed that, in most cases the DE performed better (gave faster results and closer to the optimal solution) than the GA. This influenced the decision to continue with using just DE for optimization.

The optimization is performed for all the 19 patients separately, which resulted in 19 sets of morning and maintenance doses. These are then evaluated against the doses suggested by the neurologist that used the PKG device. For each dose type, Pearson's correlation (r) is calculated and mean relative errors (RE) – these are the metrics calculated in the PK/PD model study [64] and they can be used to compare and assess the validity of presented models. They provide values allowing to evaluate the quality of acquired results.

This evaluation is performed for 3 model types:

1. Patient-specific ML models – models tailored to each patient based on their individual data.
2. General ML model, personalized with additional input (32 features) - model personalized by incorporating an additional 32 patient-specific features identified from the correlation analysis.
3. General ML model, personalized based on PCA results (5 PCs) - model personalized using the first five principal components resulting from PCA.

6.3.2. Results

The process of creating medicine intake schedules was conducted using three approaches to predict patient's future states: the patient-specific medicine response model and 2 general models with correlations, which use the most important patient-specific features as network input.

The differences between the morning and maintenance doses suggested by the neurologists and the optimization results created using ML models are shown in Figure 49.

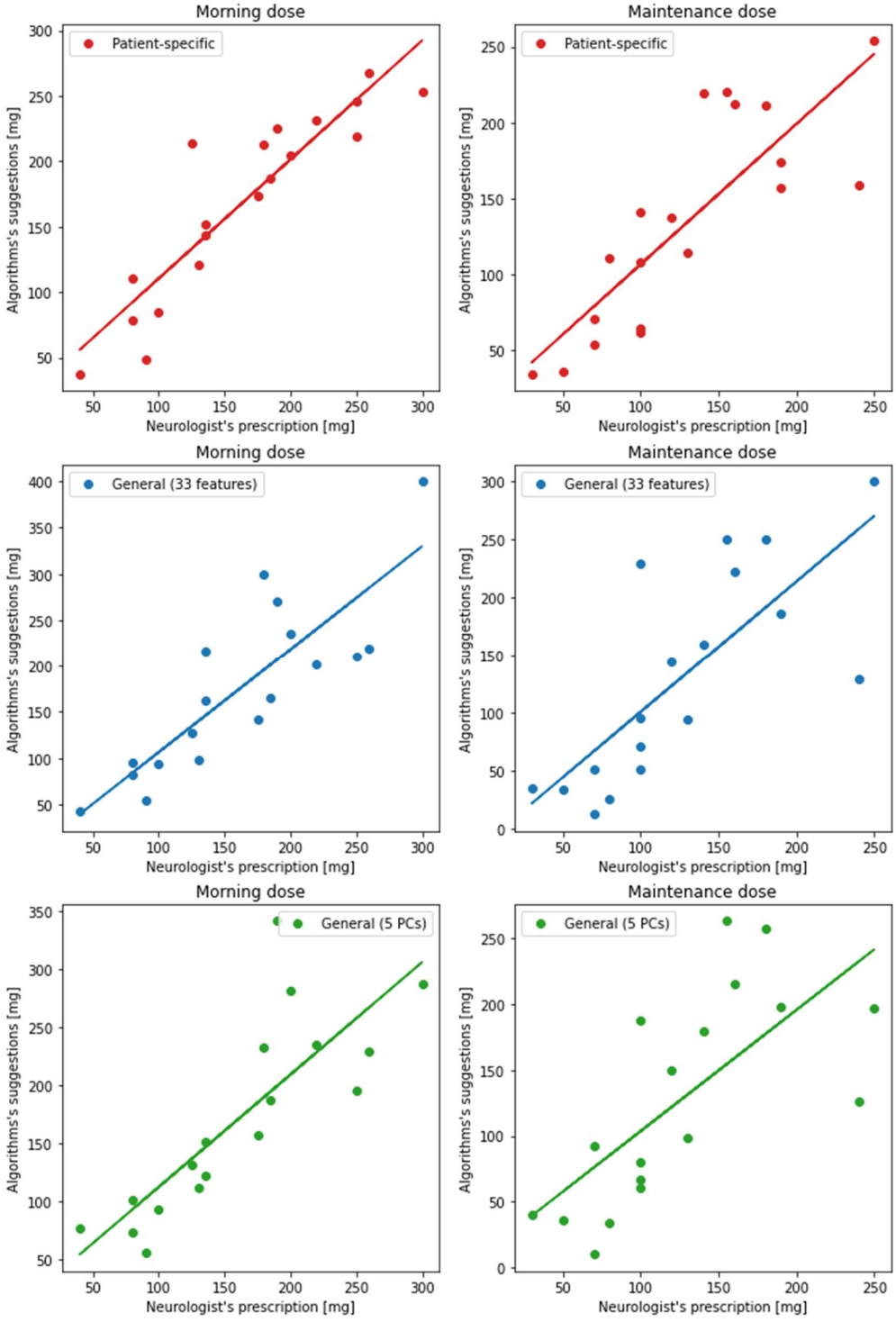

Figure 49 Comparison of algorithmic (optimization + 3 ML models) dose suggestions with neurologist's suggestions.

The results of each method are summarized in Table 47, which include Pearson's correlation coefficients and relative errors (based on neurologist's suggestions). The

metrics are provided for both the previously published PK/PD models study [64] and the three methods discussed in this study.

Table 47 Pearson's correlation coefficients and relative errors for each of the described methods for morning and maintenance doses with regards to the neurologist's suggestions.

	Morning dose		**Maintenance dose**	
Model	r	RE	r	RE
PK/PD study	0.95	12.5%	0.8	21%
Patient-specific model	0.92	14.9%	0.82	24.1%
General model (32 features)	0.844	22.8%	0.758	39.0%
General model (5 PCs)	0.818	23.2%	0.698	39.6%

Among the three proposed approaches, the patient-specific models provide the best results, demonstrating the highest correlation values and the lowest relative errors for both the morning and maintenance dose sizes. Nonetheless, they still showed marginally less accuracy than the outcomes derived from the PK/PD models. The slightly lower performance of the patient-specific models is attributed to their training on data from PK/PD models, limiting their potential to surpass these results. With adequate real-world data, these models have the potential for enhanced accuracy by adapting to more complex features and phenomena not reflected in PK/PD models. The general models, despite their worse performance, are still a good choice for initial dosing recommendations in scenarios when patient-specific data is not available. Using neural networks with optimization in place of the PK/PD model approach allowed to significantly decrease the duration of schedule generation. The approaches described in this study were able to find the optimal doses in approximately 2.97 seconds for one patient, while the PK/PD model approach took approximately 39.4 seconds using the same machine.

6.3.3. Discussion

The previously presented method for creating levodopa intake schedules, which was tested on simulated patients has now been validated using real patients. Each of these patients had a schedule prescribed by a neurologist which gives the possibility of comparing algorithmically generated schedules. In this case, only conventional optimization algorithms were used, without RL, since it provided worse results for

generating whole daily schedules. The optimization was performed for 19 patients using previously trained ML models for predicting medicine response and in order to compare the result with neurologist's prescriptions and results of previous methods [64], only the sizes of morning and maintenance doses were optimized – the time intervals between doses were adapted from neurologist's suggestion. This allowed for easier comparison of neurologist's prescriptions (treated as ground truth in this case) and optimization results.

For each levodopa-response model the correlation and mean relative errors have been calculated for the population of 19 patients. In all cases, the correlation was high, with the best values achieved for patient-specific models. This proves the model's and optimization's applicability for creating medicine intake schedules. The general models, which used patient clinical data to personalize the response performed significantly worse, but still provided a good representation of patient's state.

When evaluating medicine intake schedules by comparing them with neurologist's suggestions it is also necessary to remember that the schedules created by clinicians do not have to represent the best possible schedule for the patient, they are created based on current knowledge of the patient and of the disease and it might be possible for schedules generated using proposed methods to suggest better results. Therefore, further studies should include the application of generated schedules in patients. Having the patient evaluate the treatment using different schedules might provide insight into which schedule is subjectively the best and provides the highest quality of life. This can be also verified using objective measures, such as comparing the patient's state using sensor evaluations under different medicine schedules (created by neurologists and by the proposed method), using the described solution which uses sensor data and ML method for current state assessment.

6.4. Conclusions

In this chapter the optimization of PD patient medication intake schedules was discussed, using both, evolutionary algorithms and RL. The study focused on analyzing data from simulated and real patients to create the schedules.

The experiments confirm the effectiveness of using optimization methods for developing individualized levodopa intake schedules. The use of heuristic optimization methods allowed for flexible scheduling, leading to low objective function values, what might improve the quality of life for patients. Furthermore, the objective function values

were lower than those for the schedules designed by the clinician. The RL approach, while not outperforming conventional optimization methods, showed promise for dynamic schedule updates and applicability in more diverse scenarios.

Future research should focus on expanding datasets with more diverse patient data and exploring additional patient-specific factors that could further improve the treatment models. This includes collecting more data on factors such as diet, physical activity, and other medications used in PD management, such as dopamine agonists, MAO inhibitors, and amantadine derivatives. Larger clinical trials should also include more data for each patient. These would include more state assessments to build levodopa response models based solely on real data.

Based on the experiments with both simulated and real patients, the following recommendations are proposed for further development of methods for optimizing patient medication intake schedules:

- Focus on expanding and diversifying datasets to enhance the robustness of the optimization models.
- Incorporate additional patient-specific factors, such as diet, physical activity, and concurrent medications, into the optimization process.
- Conduct larger clinical trials to validate the optimization methods and models in real-world settings.
- Explore the potential of reinforcement learning for real-time schedule adjustments and handling more complex scenarios.
- Apply the generated schedules to verify their performance compared to clinician-recommended schedules.

Incorporating these recommendations in future studies could significantly improve the applicability and performance of the methods in creating medicine intake schedules and provide new ways of evaluating them, thereby reducing clinician involvement.

7. System for tracking PD patients' therapy

7.1. The need for system

Monitoring the health status of PD patients during treatment requires a different approach than a singular diagnosis. It should be repeatable, easy to follow and not prone to errors or missing data. To achieve this, a system has been created to collect sensor and meta-data from patients, simplifying this process. It is designed for use by both the patient and the clinician supervising the patient.

Collecting data from PD patients is a challenging task, as they are mostly elderly individuals who are not accustomed to mobile devices, often dislike new technology advancements, and may resist using them. Even though mobile phones and smartphones have been around for many years, many elderly people are not proficient in their use. As people age, many suffer from presbyopia [140] and other sight disorders, which decrease their ability to use smartphones without eyewear.

Additionally, PD patients have to face the symptoms of the disease, which affect their mental abilities and physical movement. This makes their movements imprecise and prone to errors, especially in the OFF state. Therefore, a system dedicated for PD patients has to be simple to use. The font sizes should be large enough for elderly people to read, high-contrasts colors should be used, and all clickable areas should be spaced apart. Every important decision or step should require confirmation to avoid mis-clicks.

However, clinicians require a comprehensive overview of the patient's examinations and results to precisely monitor the condition and treatment. By enabling remote data collection and analysis, the system can potentially reduce the time required to determine the appropriate medicine dosage, leading to faster optimization of treatment plans. This means that the designed system must provide separate interfaces: a simple one for patients and a more complex, but still intuitive one for clinicians. This approach also supports less experienced doctors by providing them with detailed patient data, thereby improving their ability to make informed decisions regarding patient care. To meet the needs of both patients and clinicians, two applications were designed for these two user groups.

7.2. Architecture

To track and monitor the patients, a system consisting of 2 applications has been designed:

- Mobile application – primarily used by patients to perform examinations.
- Web application – used by clinicians to schedule and store examinations, monitor, and review completed tasks. It also contains the data analysis and machine learning module.

Figure 50 presents the structure of the system consisting of these two applications, including the technologies and main libraries used in the implementation. The mobile application communicates with the web application to send and receive data using the HTTP protocol following the REST style [141].

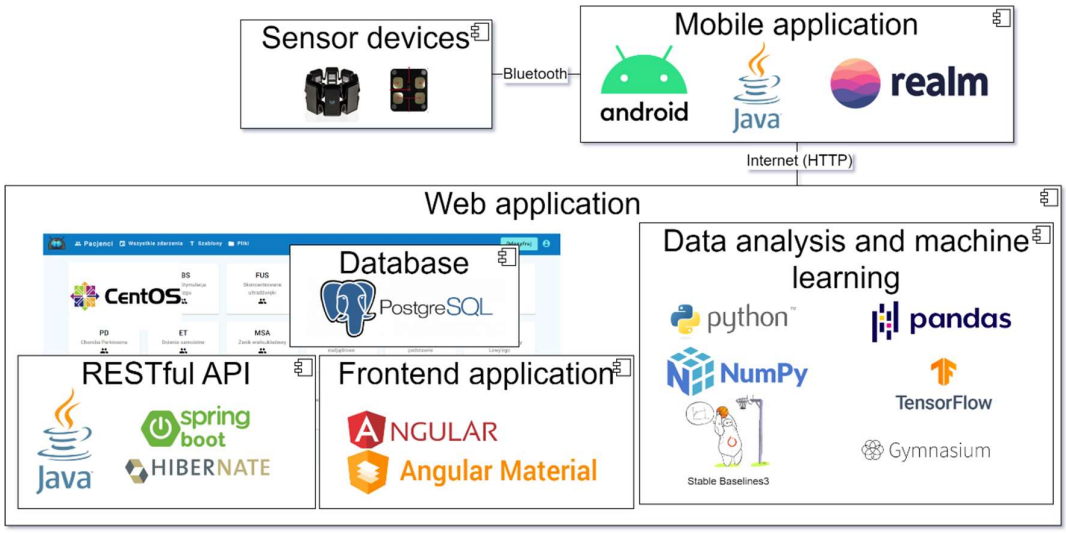

Figure 50 Architecture of the system for tracking PD patients' therapy

7.3. Mobile application

The mobile application was designed for devices with the Android operating system and was written in Java using Android Studio. The application's data is stored in a Realm object database, which provides a simple interface for storing data. The current application version supports devices with Android version at least 7, which means it can be used by over 95% of users [142]. However, to use all the functionalities of the application the device should also have Bluetooth and GPS, inertial sensors: accelerometer and gyroscope (supporting streaming data with a frequency >= 50 Hz), a touch screen with a minimum size of 6 cm x 6 cm supporting getting the size of the

finger touching the screen. The device should also be equipped with a back camera, a microphone, and a stylus should be provided for writing examinations. Not fulfilling these requirements might result in restrictions regarding the scope of data collected during examinations and scales. An Internet connection is required only for registration, sending, and receiving data from the web application.

The application is designed to be used by both the patient and the clinician. The basic capabilities it provides include:

- user registration,
- sensor pairing,
- performing examinations,
- performing state evaluations using common scales,
- medicine intake notification.

Apart from these main features the application provides additional functionalities. For example: a daily schedule view, where patients can see all the tasks to be completed during the current day. It also allows patients to redo or continue unfinished tasks. To monitor completed tasks, the dashboard and history screens have been created. The dashboard provides a summary of all completed events, while the history screen offers detailed information about the completion of every event. The application also features a settings screen where both patients and clinicians can modify application capabilities, including examination preferences, data synchronization settings, and graphical interface options.

7.3.1. User registration

When new patients join the trials, they are required to sign up in the system (web or mobile application). This can be performed individually or from the clinician's account. To register a new patient the following data should be provided:

- first and last name,
- username and password,
- date of birth/age,
- date of diagnosis / time since diagnosis,
- sex,
- dominant side of the disease,

- handedness – is the person left or right-handed,
- other diseases,
- assigned groups – based on this and the current year an identifier is created.

The groups that the patient can be assigned to represent different diseases and therapy types. A patient can be assigned to more than one group. Currently, the following groups are available:

- diseases:
    - PD – Parkinson's disease,
    - ET – essential tremor,
    - PSP - progressive supranuclear palsy,
    - CBS - corticobasal syndrome,
    - DLB - dementia with Lewy bodies,
    - MSA – multiple system atrophy,
- therapy types:
    - BMT – blood marrow transplant,
    - DBS – deep brain stimulation treatment,
    - FUS – focused ultrasounds,
    - GK – Gamma knife,
    - APO – apomorphine treatment,
    - DUO – duodenal pump treatment.

Once all the data is provided and submitted a new account is created with a default, empty clinical schedule.

7.3.2. Sensor pairing

In order to perform examinations using sensors connected via Bluetooth, it is necessary to complete the pairing process first. For this purpose, a dedicated view has been created. After turning on the Bluetooth and GPS, it allows users to search for available/discoverable devices. More than one device can be paired to collect data from multiple limbs simultaneously.

The application currently supports two external sensor devices: Thalmic Labs' Myo Gesture Control Armband (Figure 51) and SiFi Labs' Biopoint (Figure 52). These

are placed on the limbs, preferably arms, and are equipped with sensors to monitor the patient's condition.

The Myo armband is built of 8 segments, each with EMG electrodes to measure muscle activity. It is equipped with a 3-axes accelerometer and a 3-axes gyroscope, placed in the main unit [74]. It uses a vibration motor and two lights to alert the user. Using Bluetooth Low Energy (BLE), it is capable of transmitting sensor data from 8 EMG units collected at a frequency of 200 Hz and accelerometer and gyroscope data at 50 Hz frequency. The device is equipped with two lithium batteries 3.7 V – 260 mAh and according to the documentation should be able to work for a single day without recharging. However, experiments proved this is not the case when the data is collected and transmitted via Bluetooth continuously. The armband does not have memory storage; all data should be transmitted during a connection with the main device.

The Biopoint is a recently developed solution by SiFi Labs, resulting from previously conducted research regarding a Myo armband successor [143]. It captures all of the functionalities of the Myo armband and provides additional sensors and capabilities [75]. Biopoint has one EMG sensor, collecting data at 2000 Hz, accelerometer, and gyroscope with a frequency of 100 Hz. Additionally, it contains an electrodermal activity sensor, electrocardiograph (500 Hz), skin thermometer and 4 photoplethysmography (PPG) sensors using blue, red, green and infrared light. The battery is expected to be sufficient for the device to collect data for more than 18 hours. It can transmit data online using BLE, but it is also equipped with 2GB of memory which can be used to store data. Like the Myo armband, it uses a vibration motor and LEDs to notify the user. The only disadvantage of the Biopoint is that it uses just one EMG sensor; SiFi Labs is currently developing a solution with 8 such sensors – BioArmband.

Figure 51 Thalmic Labs' Myo Armband

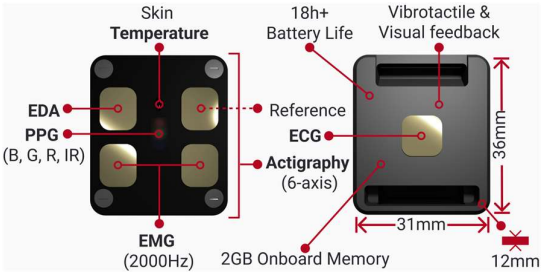

Figure 52 SiFi Labs' Biopoint

7.3.3. Performing examinations

Completing sensor examinations is the main goal of the mobile application. These can be completed in three modes:

- on demand – when the user chooses to perform an examination,
- based on a schedule – at a time selected by the clinician; the patient is notified by the device to perform an examination,
- in the background – the device collects data in the background, during daily activities.

Examinations performed on demand are available only in the clinician mode. These can be performed during the onboarding process of new patients and whenever the patient is available at the clinic and can be supervised.

The application allows performing 4 types of examinations three modes:

- sensor examinations,
- screen exercises,
- writing exercises,
- voice exercises.

Before the examination is started the scope of the examination can be selected. Every type of examination is available for selection, as well as specific examinations. The user can choose which sensors are used – embedded in the mobile device, Myo armband, Biopoint and which upper limbs will be examined. Such a configuration is available only in on-demand examinations. When they are scheduled, the configuration is predefined – only changes in the sensors used can be made when there is a problem with Bluetooth connection with the external sensors. The configuration process is visible in Figure 53.

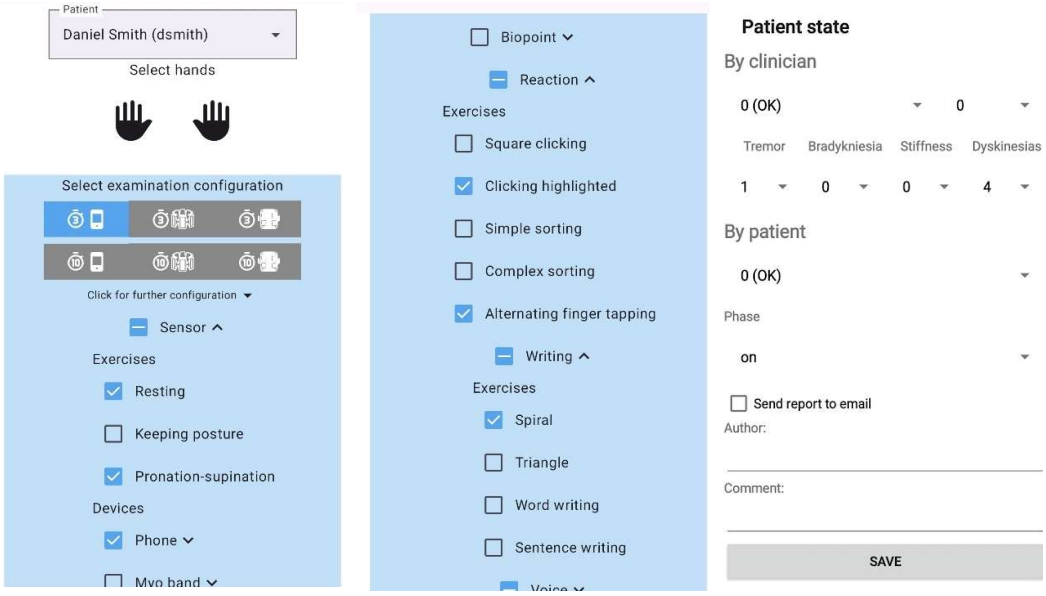

Figure 53 Examination configuration screen in the application (left) and state evaluation screen (right)

After completing every examination, the patient or clinician is asked to answer a few questions regarding the condition during the examination. The form for the patient includes an individual subjective state evaluation on a scale from -4 (severe symptoms) to +4 (severe dyskinesias) with 0 being the optimal state, and a textbox to place additional comments regarding the condition. The form for the clinician additionally requires providing the following data based on the clinician's knowledge:

- overall state evaluation on a scale from -4 to +4,
- overall state evaluation on a scale from -10 (severe symptoms) to +10 (severe dyskinesia),
- individual symptom evaluation on a scale from 0 to 4 for bradykinesia, tremor, dyskinesia, and stiffness,
- current phase of the patient: ON/OFF,
- clinician initials.

The clinician's evaluation is utilizes two scale, one ranging from -4 to +4 for overall functional status and another from -10 to +10, providing different precisions for a comprehensive assessment of the patient's condition and improving the detection of assessment errors.

The form for inputting this data is presented in Figure 53. This data is then sent along with the examination to the server.

7.3.4. Performing state evaluations using common scales

The application allows evaluating the state of the patient using commonly used scales in Parkinson's disease (Table 4). These state evaluations can be scheduled by the clinician or performed on demand. The device displays each question at a time, allowing the following types of questions:

- single choice,
- multiple choice,
- number,
- text.

These questions can be grouped into sections, with an instruction view added before each. For each question, the text and description can be defined. Single and multiple-choice types require defining options, and for each option, a text and a description can be provided. Some of the scales calculate a score based on selected options, this makes providing a score for every option mandatory. In that case, once the user completes the questions, the overall score is displayed. While the questions are being answered, it is possible to collect sensor data from armbands worn by the patient. Along with timestamps of question answers, this can be used to identify movement while each question is being answered.

When the clinician performs the evaluation, the application allows video recording of the evaluation process. This recording is then saved and sent along with the completed scale answers to the server. Figure 54 presents screens from the mobile application of scale completion.

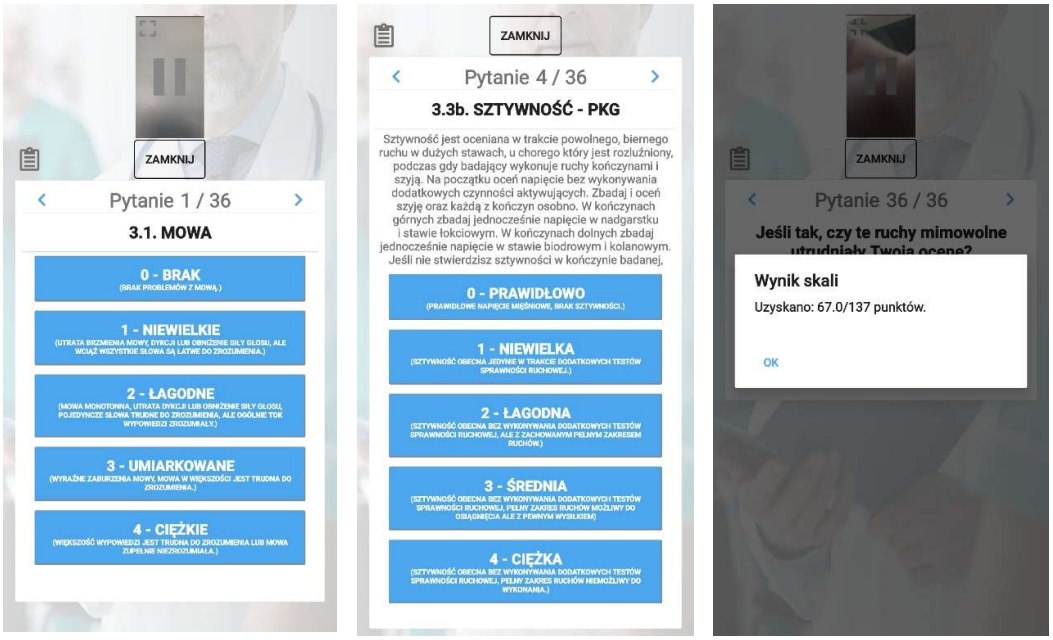

Figure 54 Scale completion screens in the mobile application

7.3.5. Medicine intake notifications

Remembering to take medication can sometimes be challenging for PD patients. The application supports notifications about medicine intakes. The phone rings, and the patient is asked to confirm if they took the medication and at what time. This not only reminds patients to take their medicine but also helps keep track of the exact times at which the medication was taken. This is essential for monitoring the medicine response and planning future doses of medicine. The schedule of medicine doses is provided by the clinician. The information on whether the medication was taken, and the exact time are sent to the server. The application supports inputting additional doses on demand in the clinician mode. The form requires specifying the time and size of the dose as well as the name of the drug. The medicine notification and the on-demand form are presented in Figure 55.

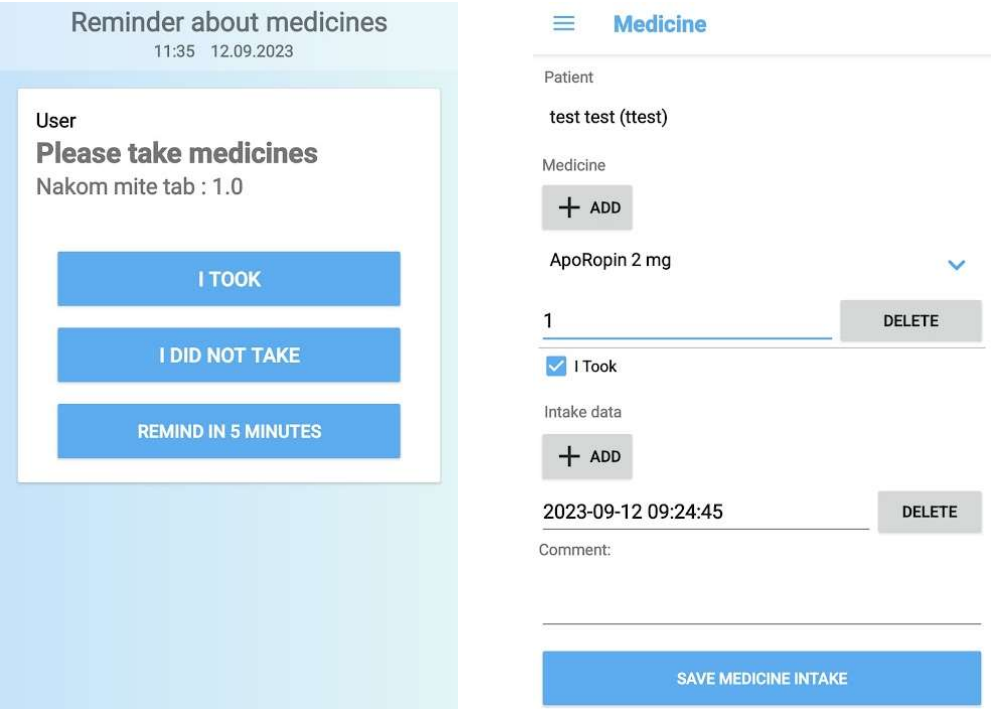

Figure 55 Medicine notification and medicine input screens in the mobile application

7.4. Web application

The web application is designed to be primarily used by clinicians. It consists of three main parts presented in Figure 50: the frontend application created using Angular framework, the backend application written using Spring framework in Java, and the data analysis and machine learning module written in Python.

The frontend application provides the interface for users – clinicians, it serves the data provided by the backend application and the data analysis and machine learning module. The backend application is also used by the mobile application to register new accounts, provide dictionary data (surveys, medicine, patient groups), and to receive performed tasks – medicine intakes, examinations, and surveys. Besides providing logic to the app, the backend application also serves as a gateway to the data stored in the PostgreSQL database.

The data analysis and machine learning module is used by both the backend and frontend application. It has a direct access to the database. Based on the stored data, it provides data analysis features for completed events by the patients. It is also responsible for training and inference of machine learning models used for predicting patient's current

state based on sensor signals, predicting future patient states, and building/modifying medicine intake schedules.

The web application is accessed by the clinician through the frontend single page application created using the angular framework. After logging into the portal, the user is presented with the navigation bar at the top and a list of patient groups in the center of the screen. The navigation bar is available on every page and allows the user to easily navigate the application. All navigation options are presented in Figure 56. The "Decrypt" button on the right asks the user to provide a password to decrypt user data – first and last name using AES (Rijndael) [144] encryption algorithm.

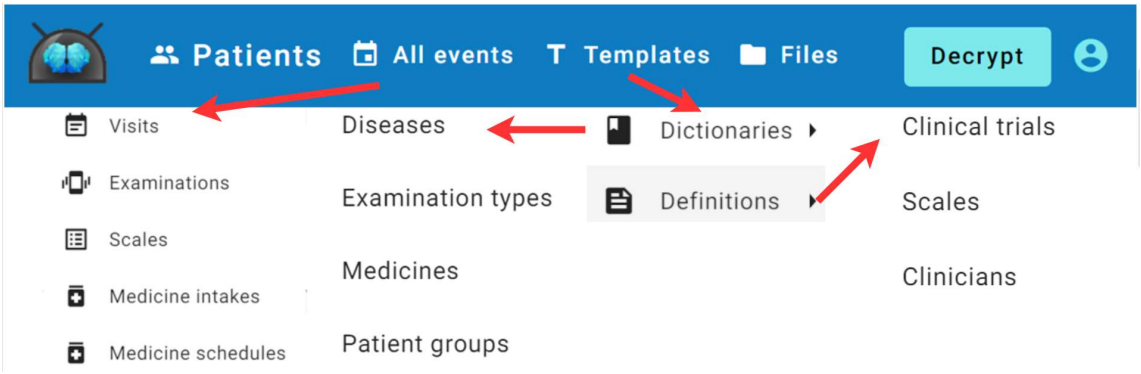

Figure 56 Navigation options in the web application

The first tab – "Patients" – navigates to the list of patient groups, as displayed in Figure 57. Here the clinician can choose a specific patient group or navigate to the list of all patients, where they see the usernames, identifiers, and registration dates of patients. If the user provided a decryption password, the first and last names are displayed as well. The list view is commonly used in the application, and it supports pagination filtering and sorting. From the list of patients, it is possible to remove a patient or navigate to their clinical data.

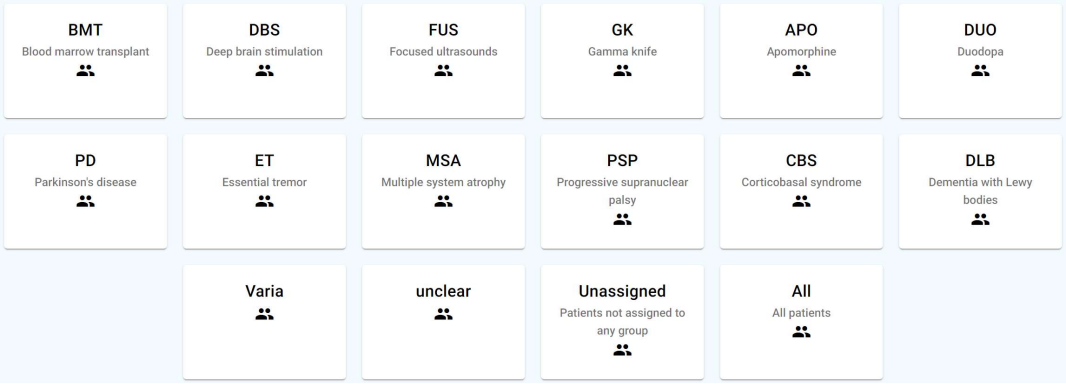

Figure 57 The patient group view in the web application

The patient view consists of 4 tabs: Patient, Visits, Examinations, Scales, and Medicine. The first tab is for editing patient data, including their name, groups, other diseases, birth date, diagnosis date, etc. The following tabs contain list views of events for this patient in the current clinical schedule. Using these views, each of these components can be added or edited: visit, examination, scale, medicine intake.

Adding and editing visits can be performed only by the clinician using the web application. A visit is a single occasion when a patient arrives at the clinic. The data model allows entering data regarding the patient and performed tests in the following sections:

- epidemiology,
- medicine,
- DBS programming,
- posturography, treadmill,
- speech,
- neuropsychology,
- neuroimaging and neurophysiology,
- blood lab tests.

Each of the sections requires providing data regarding the patient, as presented in Figure 58 for Epidemiology. The system also allows adding new section definitions and properties.

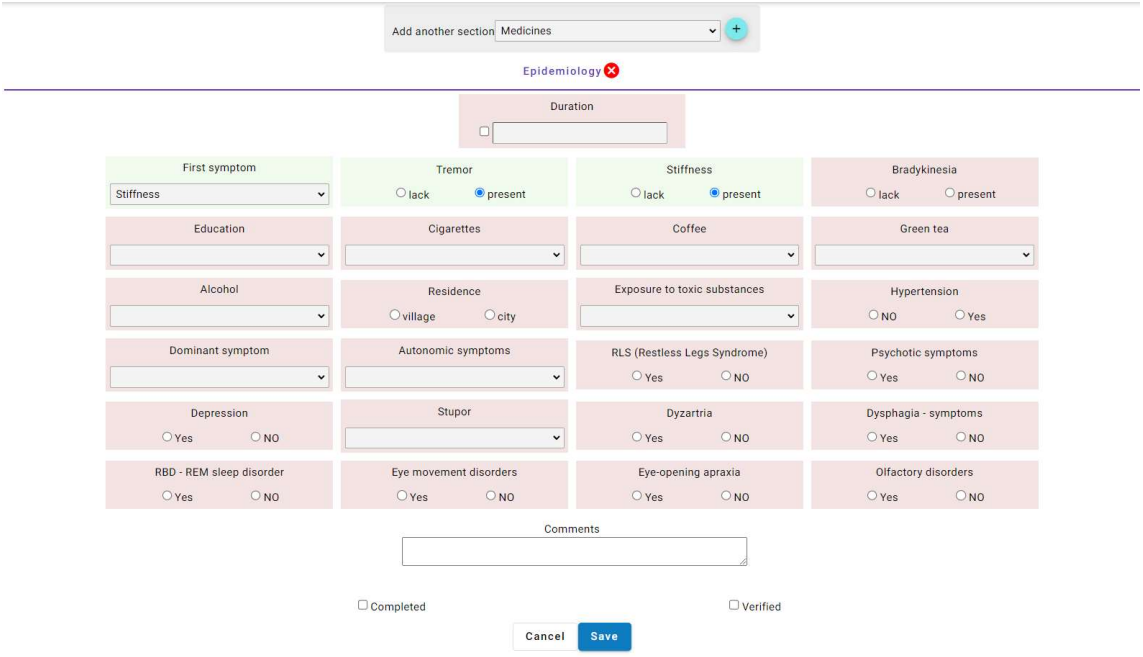

Figure 58 The visit edit view in the web application

In the Examination tab, the clinician can schedule new examinations for the patient to perform at specified times. It is also the place to view completed examinations. After clicking the “Details” button a list of all performed measurements is displayed, allowing the clinician to view specific sensor signals, their analysis, and other presentation forms of the results. For completed examinations, the measurements cannot be changed in any way; only updates to the fields regarding patient’s state, phase and comments are allowed. However, all previous values are still retained in the database. In the case of examinations performed on both hands, buttons to view results for each of them are available. Voice recordings are currently only available for download through the web application. A view of a completed examination is presented in Figure 59.

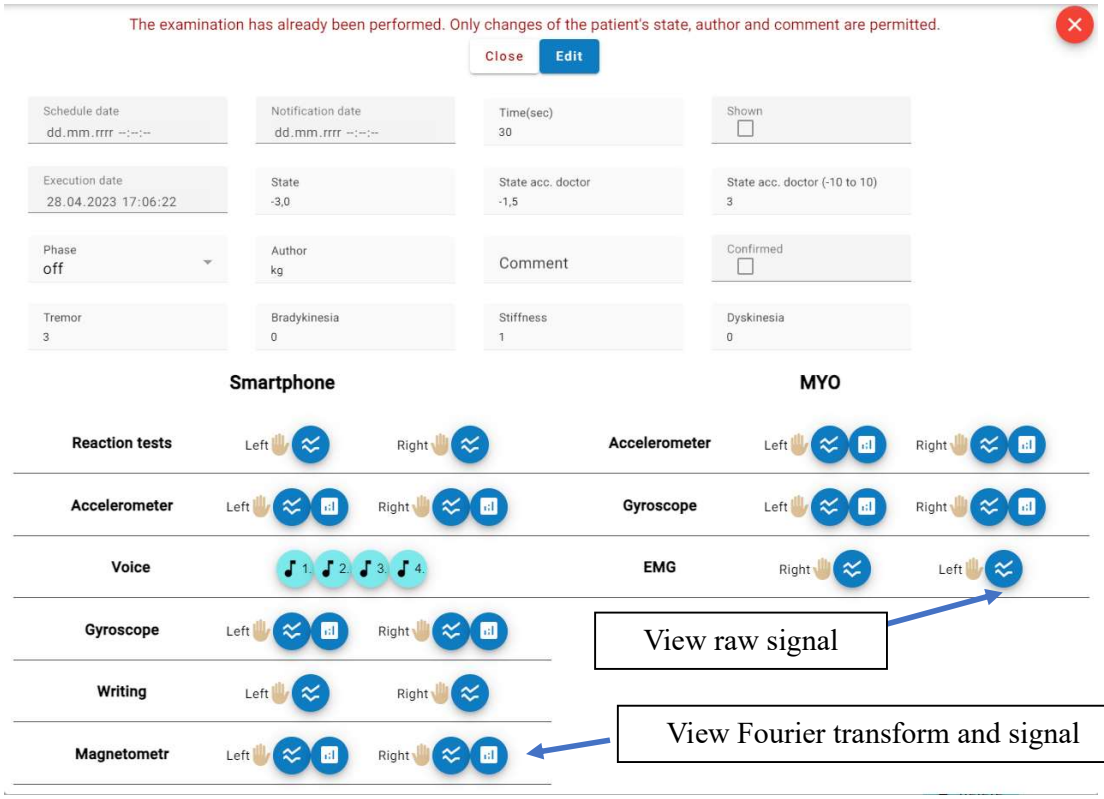

Figure 59 View of a completed examination in the web application

Surveys are by default completed in the mobile application, which allows recording the evaluation process both in video and using inertial sensors. However, the web application also supports filling/updating surveys. The survey's detailed view supports also reviewing completed surveys (in case of scored surveys, the result is displayed too) and viewing the collected measurements and videos.

The Medicine tab displays the medicine schedules the patient is following and information regarding specific medicine intakes, such as whether the dose was taken and the exact time it was taken. Adding medicine using the web application can be performed in two ways:

- Adding every intake separately – for each dose, the medicine, intake time, and dose size must be provided.
- Adding in bulk – a daily schedule is defined for each medicine and is then replicated for the specified number of days. Adding in bulk makes the process faster and requires less duplication. For every bulk schedule, an additional comment can also be provided.

Apart from viewing events for each patient separately, it is possible to view all of them grouped by type in a paginated list. This is available from the second option in the navigation bar – “All events”. The “Templates” tab provides navigation to editing dictionaries used in patient and event definitions such as patient groups, medicines, and diseases, as well as editing surveys/scales and clinical trials definitions. Using the app to create survey definitions is simple and intuitive. The user adds questions, and can specify the main text, description, and question type for each.

When conducting clinical trials using the system, many patients are expected to follow the same schedules regarding examinations, scales/surveys, and medication. To avoid adding these individually for every patient, clinical schedule definitions have been created. These definitions can include specific schedules for examinations (specification of measurement types, scope of exercises, duration), scales, and medication. These schedules can be defined by a start and end date and times of day of each event. Based on these definitions, uniform clinical trials can be generated for many patients by filling the form only once.

The last link from the navbar – “Files” is dedicated to data analysis. Its main goal is to prepare data files for analysis tools. It supports downloading completed scales, visits, and medicine intakes of patients. However, the primary feature is downloading parameters of sensor signals collected during examinations. To download these parameters, the user selects devices, sensors, chooses patient groups, and how the data from both hands should be handled (selects one hand, parameters for both hands as one row, parameters for each hand separated with an additional column to distinct the hands) and which exercises should be included. Once the “Generate” button is clicked the data analysis and machine learning module generates a coma-separated value (CSV) file with parameters predefined for the selected sensor/measurement type (the parameters were discussed in Feature extraction p. 65).

The data analysis and machine learning module performs other tasks as well. It is written in Python and uses data processing libraries such as NumPy and Pandas to generate examination reports, which are sent to the clinician after an examination is completed. It contains implementations of numerous signal parameters for each measurement type and provides tools to analyze the data.

These parameters, along with implemented machine learning and optimization methods, offer an interface for evaluating patient's state based on sensor data, forecasting patient's future states, evaluating medicine intake schedules, and creating new schedules. The machine learning capabilities are delivered by the TensorFlow library – for supervised learning, Gymnasium and Stable Baselines 3 – for reinforcement learning, and the DEAP and SciPy libraries – for optimization. The data analysis and machine learning models were discussed in detail in following chapters:

- Patient state evaluation – p. 41,
- Medicine response model – p. 94,
- Medicine schedules creation – p. 131.

7.5. Data model

To fulfill the requirements for the application, a data storage has been created. It stores data regarding patients, doctors, and their performed actions. This data is stored in both the mobile application and web applications. To maintain data consistency, the data is synchronized whenever an Internet connection is available. The data models of databases in the mobile and web applications are similar. However, in some cases the model in the mobile application is simpler, with less data stored regarding some of the entities. In this section, the most important parts of the data model are presented. Figure 60 presents the most significant entities stored in the database of the web application.

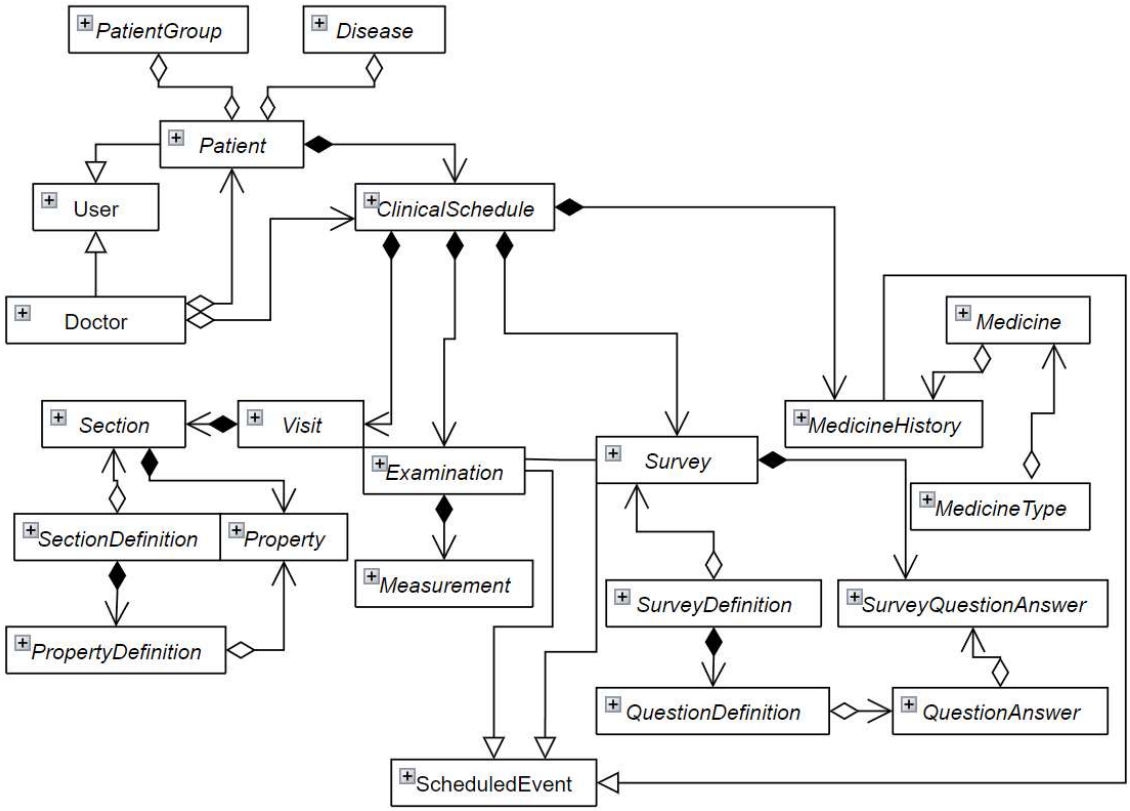

Figure 60 Entities stored in the database of the web application

To keep the figures simple, the content of specific entities presented in Figure 60 has been organized into 3 additional figures – Figure 61, Figure 62, and Figure 63. These entities are represented by Java classes and are mapped to database tables using an Object-Relational Mapping (ORM) library called Hibernate. Instead of providing table definitions in the diagrams, class diagrams have been selected to increase clarity.

Figure 61 presents the entities related to users – clinicians and patients. These two classes extend the user class containing basic user data such as names, usernames, and passwords (an encrypted password is stored using Bcrypt [145]). Every clinician working with the system is represented by the doctor class – they have a list of patients and clinical schedules assigned. The database stores all patient data provided during the registration process such as sex, assigned groups, disease information, and a generated identifier that allows clinicians to identify patients. The first name and last name of the patient are not stored directly in the database, to keep them protected, an encrypted version of them using AES [144] is stored.

The ClinicalSchedule entity represents the participation of a patient in a clinical study. It is a container of examinations (Examination class), surveys/scales (Survey class), medicine intakes (MedicineHistory class), and registered visits (Visit class). It is assigned to a specific patient and clinician.

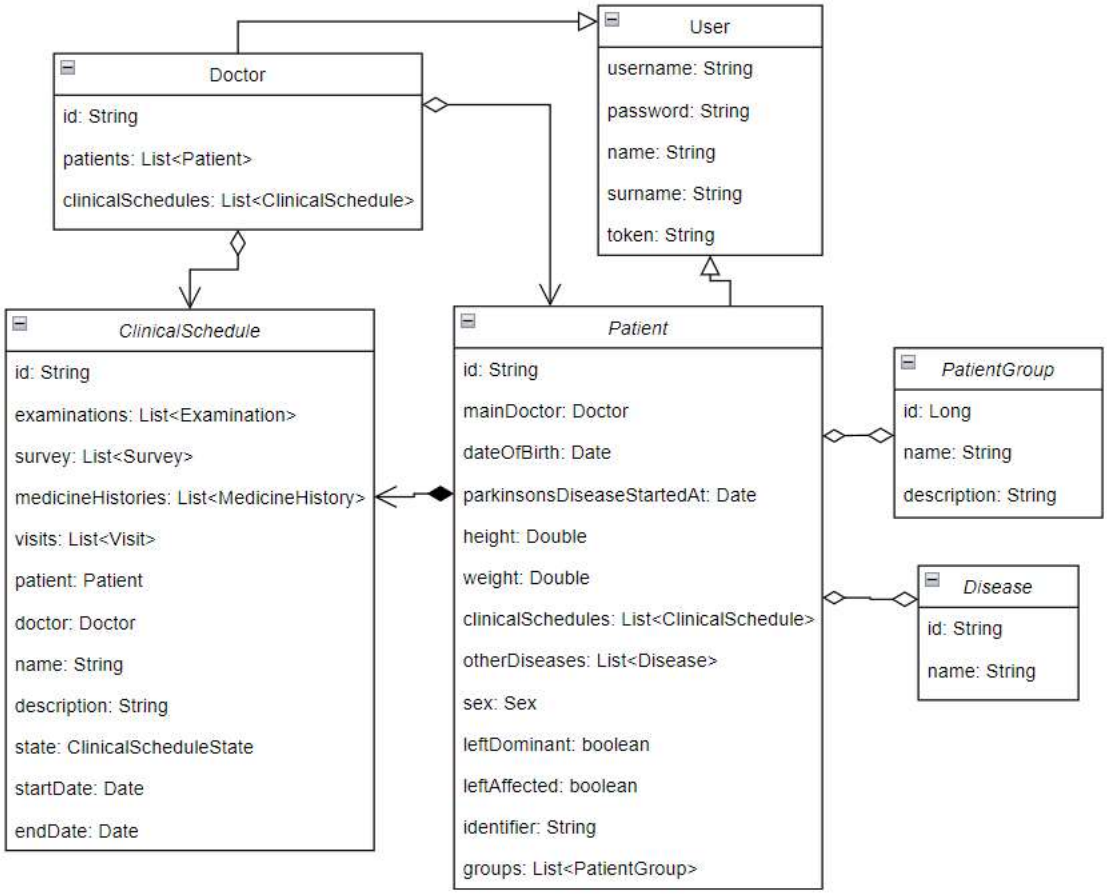

Figure 61 User-related entities stored in the database

In Figure 62, the structure of entities regarding registered visits and medicine intakes is presented. Each registered visit consists of sections where predefined properties can be defined, following the definition of sections and respective property definitions. These include the labels, data types, and in case of choice properties, available options. Each visit may consist of multiply sections with many defined properties, for each visit, a clinician should be selected.

The MedicineHistory, Examination and Survey entities all extend the ScheduledEvent class. This class contains the definition for attributes regarding

scheduled events for the mobile application. The user is informed about them using notifications. The time of the notification is defined by the notificationDate (which can be updated if the user postponed it). The scheduledDate represents the initial scheduled date, and executionDate represents the real date and time the event was performed.

Medicine intakes are registered using the MedicineHistory class. Each object represents one medicine intake with specified medicine, dose, and time. These objects make it possible to track all the medicine doses taken by the patient.

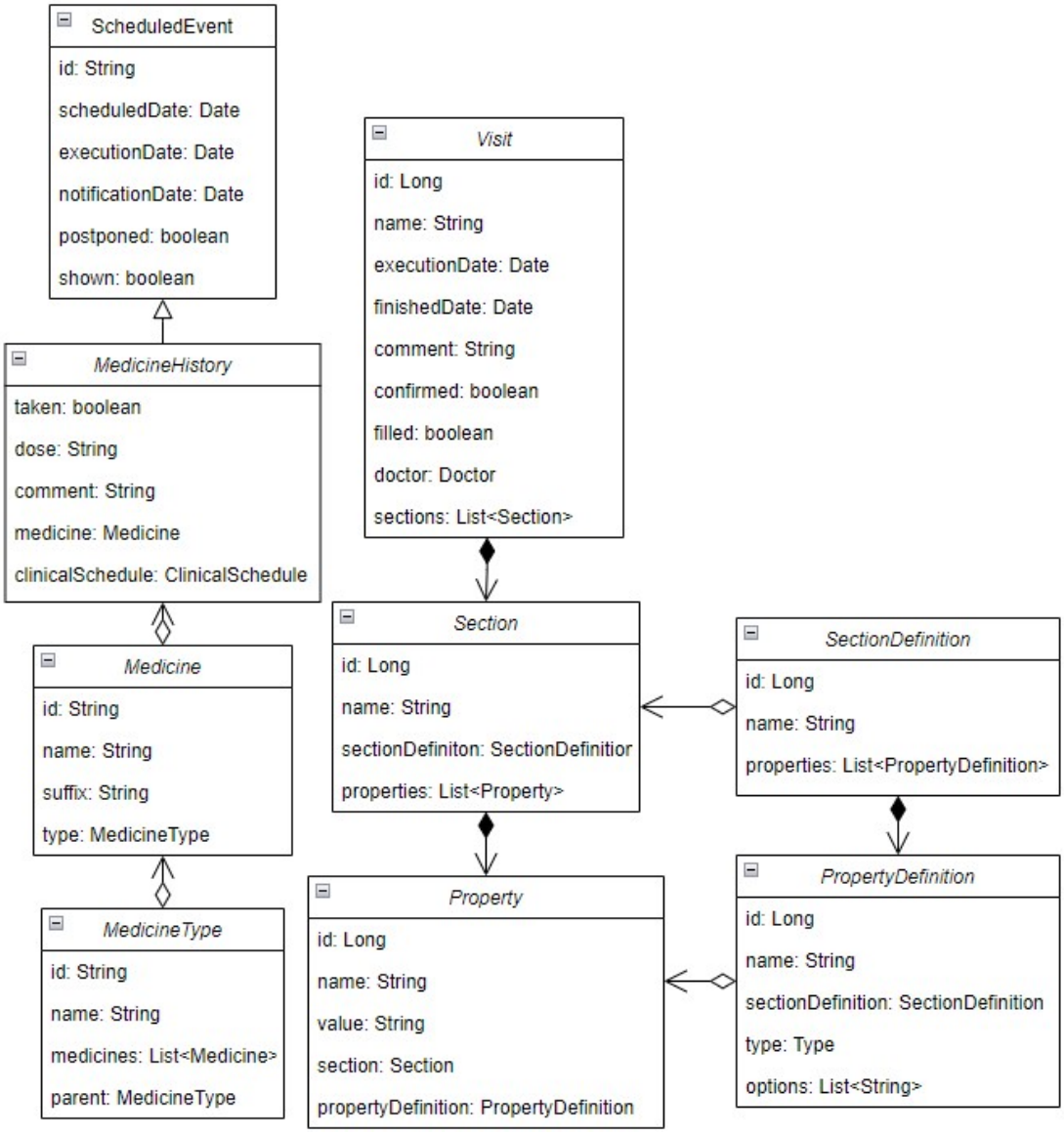

Figure 62 Visit and medicine intake related entities in the database

Figure 63 presents the structure of entities related to examinations and completed state evaluation scales. The examination object is created during scheduling of

examinations and is later completed or can be created during examinations on-demand. It stores data inherited from the ScheduledEvent class regarding the scheduling of the event. In the list of measurements, all measurement data is saved from completing specific exercises. Metadata regarding the sensors and devices used, sampling frequency, measured hand, and competed exercises is stored along with the list of saved values. In the case of voice exercises, the names of saved audio files are saved in the fileAttachments list. The examination entity also contains information regarding the evaluation of patient's state – according to the patient and according to the clinician (in two scales -4 to 4 and -10 to 10). Four symptoms – bradykinesia, tremor, dyskinesia, and stiffness are additionally evaluated and saved in the symptomString attribute. An examination can also be performed during a survey completion; these examinations have a filled value for the survey property.

Evaluation of the state using scale is carried out using the definitions saved in the SurveyDefinition entity. In the case of scales providing an overall score, the maxScore attribute contains the maximum possible score. The specific questions are stored in QuestionDefinition objects. There are five supported types of question definitions:

- Yes/No – The user chooses from yes or no options.
- Single choice – The user selects one of the available options.
- Multiple choice – The user can choose many out of the available options.
- Text – The user is required to provide text.
- Information/Instruction – Only an instruction is displayed, the patient is expected to click the "Next" button.

For yes/no, single choice, or multiple-choice type, a set of answers is assigned with an answer text and a description. A score can be assigned to every answer, which is then used to calculate the final score of the scale. For each question a main text is required; however, an additional description can be provided as well.

To save a completed survey/scale an object of the Survey class is used. It stores notification, scheduled, execution, and finished times and dates, the definition of the survey, the list of answers along with their times, and the name of the video file that can be recorded during completion of the survey. During the survey, if the patient is wearing the armbands, measurements can be collected. This data can be saved in the examination attribute.

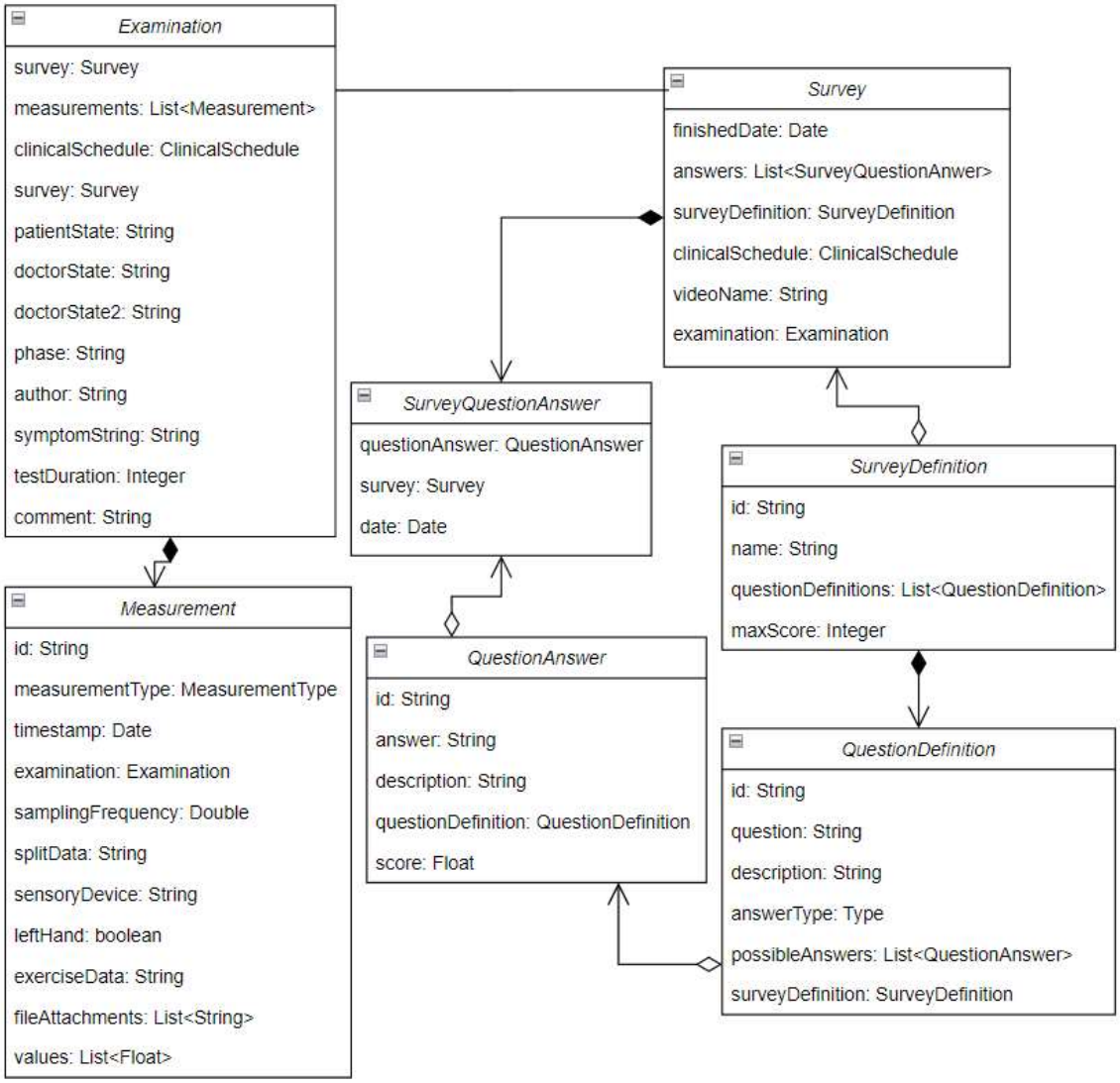

Figure 63 Examination and survey related entities stored in the database

7.6. Conclusions

The presented system has been developed for monitoring and tracking the health status of PD patients. The system presents a high level of maturity, has been successfully implemented in a real-word research setting and has been necessary in collecting data for the MUW dataset. The system played a crucial role in conducting the research presented in this thesis, particularly in training the models for identifying patients' states.

The system has successfully collected data from a significant number of patients, providing valuable information regarding the progression of disease and treatment effectiveness. To date, the application's database includes examinations of over 350

patients with various diseases, offering the dataset for training and validating the methods proposed in this thesis.

During clinical trials and the development of the system, both applications were continuously expanded to include new features and incorporate feedback from clinicians. In the future, the application should be redesigned to fully incorporate new requirements and expectations, which have been formed also based on results showcased in this thesis. The scope of exercises should be adjusted to utilize the findings regarding the efficiency in capturing the patient state. Furthermore, there should be greater focus on supporting passive monitoring of patients to reduce the burden of instrumental exercises. This could involve tracking regular phone usage and collecting data in the background from wearable sensors, providing more data for ML models.

Data security is an important factor, when creating systems for processing patient clinical data. In this application, multiple security measures have been implemented, including AES (Rijndael) for encrypting sensitive patient information and Bcrypt for hashing user passwords. Most importantly, access to both applications is password-restricted, and by establishing distinct patient and clinician roles, the system ensures limited access to the data, protecting it from unauthorized access and breaches, and maintaining the integrity and confidentiality of the collected information.

The designed system for tracking PD patients' therapy not only meets the current needs of patients and clinicians but also lays a solid foundation for future innovations and improvements in the management and treatment of PD.

8. Discussion and conclusions

This dissertation addresses significant challenges in the management of Parkinson's disease (PD) by using machine learning and optimization techniques to create personalized medication schedules. The research covers multiple aspects, from symptom severity evaluation to medicine response modeling and optimization of intake schedules. Below, the key findings, contributions, and implications of the study are discussed, followed by recommendations for future research.

The findings of this study confirm the efficacy of machine and deep learning models in detecting and measuring the severity of PD motor symptoms using data from tasks performed on mobile devices and wearable sensors. Analyzing the results of experiments with the MJFF dataset allowed to discover that, tasks involving significant movement, such as walking and arm movements, were identified as key indicators of symptom severity, whereas tasks with minimal movement, like sitting or standing, provided less accurate results, especially for symptoms that manifest in the movements of patients e.g., dyskinesia. The study demonstrated that both shallow machine learning models and deep learning models could achieve high accuracy, with deep learning models performing slightly better. The results also highlighted the importance of diverse and balanced task representation to improve model accuracy, particularly for severe symptoms.

In evaluating the MUW dataset, traditional machine learning models were effective despite the dataset's limitations. The models performed the best for predicting tremor severity, with results comparable to those obtained using the MJFF dataset. However, the models performed worse for bradykinesia and muscle stiffness, and the worst for dyskinesia, due to the limited and imbalanced sample size. Sensor exercises, especially those involving holding hands on a flat surface and performing pronation-supination movements, were most effective in evaluating symptom severities.

Based on these findings, several recommendations are proposed for further development of patient state assessment methods (Discussion p. 90). These recommendations focus on selecting only the most promising tasks and mitigating the impact of data imbalance, such as performing passive examinations and capturing higher symptom severities by focusing on the patients in the advanced phase of PD.

The modeling of patient responses to levodopa using machine learning, particularly artificial neural networks, demonstrated notable success. Despite the limited number of training samples, the models performed well on validation sets, with LSTM architectures showing the best performance.

In the Swedish dataset, machine learning models validated the application of patient-specific models for real PD patients, achieving good fit metrics. Although the performance was slightly worse than for synthetic patients, the models still provided reasonable predictions. The integration of patient demographic and clinical data into the modeling process allowed for personalization of medication response predictions.

Building medication intake schedules using optimization methods proved effective for simulated patients. The optimization allowed for flexibility in dose sizes and intervals, leading to more personalized schedules. Comparison of optimization results using both PK/PD and ML models showed close alignment, indicating the applicability and validity of the models. The study also explored reinforcement learning (RL) for dynamic schedule updates, though conventional optimization methods generally provided better results.

The validation of the proposed method on real patients showed high correlation and low relative errors for optimized dose sizes, confirming the method's applicability. However, further research should include patient trials to evaluate the generated schedules subjectively and objectively. Comparing patient states under different schedules, as assessed by sensors and ML models, could provide valuable insights.

To further enhance this research, larger clinical trials should be conducted to build levodopa response models based solely on real data. Collecting more data on additional factors influencing medication response, such as diet and physical activity, could improve model accuracy. Real-life applications may also require refining the objective function for optimization tasks and exploring alternative approaches like dynamic RL.

Performing this research was possible due to the development of a comprehensive information system for monitoring patients' condition. Consisting of both a mobile and mobile application, it provides clinicians an overview of patients' changing conditions and how they are impacted by the medication. The mobile application captures sensor signals and serves as the interface for patients, enabling them to register and send information about their condition and treatment progress. The web application, equipped

with the data analysis and machine learning module, offers clinicians methods for accessing patients' conditions, building medicine response ML models, and optimizing intake schedules. This system meets the current needs of patients and clinicians and creates a solid foundation for future innovations and improvements in the treatment of PD.

The research presented in this dissertation has led to the following advancements:

- Methods to assess the symptoms and the state of a PD patients:
  - Formulated and implemented new exercises and measurement strategies using a mobile platform.
  - ML methods for detection and evaluation of disease symptoms.
- ML methods for predicting medication response.
- Optimization methods for creating personalized medicine intake schedules.
- An information system comprising mobile and web applications to monitor patients' states.

In conclusion, this dissertation presents a comprehensive approach to personalized PD treatment through machine learning and optimization. The findings demonstrate the potential for significant improvements in symptom management and patient quality of life, thereby confirming the hypothesis stated in the Hypothesis and method overview section (p. 24). The integration of sensor data, ML models, and optimization algorithms offers a promising direction for future PD treatment strategies.

9. Bibliography

1. Ou, Z.; Pan, J.; Tang, S.; Duan, D.; Yu, D.; Nong, H.; Wang, Z. Global Trends in the Incidence, Prevalence, and Years Lived With Disability of Parkinson's Disease in 204 Countries/Territories From 1990 to 2019. *Front Public Health* **2021**, *9*, 776847, doi:10.3389/FPUBH.2021.776847.

2. Bloem, B.R.; Okun, M.S.; Klein, C. Parkinson's Disease. *Lancet* **2021**, *397*, 2284–2303, doi:10.1016/S0140-6736(21)00218-X.

3. Sveinbjornsdottir, S. The Clinical Symptoms of Parkinson's Disease. *J Neurochem* **2016**, *139*, 318–324, doi:10.1111/JNC.13691.

4. Raju, V.R.; Konda, S.; Balmuri, K.R.; Balabhadra, A.; Raju, B.; Rao, G.D. MER Based Analysis of Local Field Potentials with Deep Brain Stimulation Subthalamic Nucleus in Parkinson's Disease Using Coherence and Entropy Techniques. *IP Indian J Neurosci* **2021**, *6*, 202–219, doi:10.18231/J.IJN.2020.041.

5. Tolosa, E.; Wenning, G.; Poewe, W. The Diagnosis of Parkinson's Disease. *Lancet Neurol* **2006**, *5*, 75–86, doi:10.1016/S1474-4422(05)70285-4.

6. Poewe, W.; Wenning, G. The Differential Diagnosis of Parkinson's Disease. *Eur J Neurol* **2002**, *9*, 23–30, doi:10.1046/J.1468-1331.9.S3.3.X.

7. Lee, T.K.; Yankee, E.L. A Review on Parkinson's Disease Treatment. *Neuroimmunol Neuroinflamm* **2021**, *8*, 222–244, doi:10.20517/2347-8659.2020.58.

8. Bhidayasiri, R.; Tarsy, D. Parkinson's Disease: Hoehn and Yahr Scale. In *Movement Disorders: A Video Atlas*; Bhidayasiri, R., Tarsy, D., Eds.; Humana Press: Totowa, NJ, 2012; pp. 4–5 ISBN 978-1-60327-426-5.

9. Goetz, C.C. The Unified Parkinson's Disease Rating Scale (UPDRS): Status and Recommendations. *Mov Disord* **2003**, *18*, 738–750, doi:10.1002/MDS.10473.

10. Gasser, T. Genetics of Parkinson's Disease. *J Neurol* **2001**, *248*, 833–840, doi:10.1007/S004150170066.

11. Di Monte, D.A.; Lavasani, M.; Manning-Bog, A.B. Environmental Factors in Parkinson's Disease. *Neurotoxicology* **2002**, *23*, 487–502, doi:10.1016/S0161-813X(02)00099-2.

12. Schapira, A.H.V.; Emre, M.; Jenner, P.; Poewe, W. Levodopa in the Treatment of Parkinson's Disease. *Eur J Neurol* **2009**, *16*, 982–989, doi:10.1111/J.1468-1331.2009.02697.X.

13. Stocchi, F.; Tagliati, M.; Olanow, C.W. Treatment of Levodopa-Induced Motor Complications. *Mov Disord* **2008**, *23*, S599–S612, doi:10.1002/mds.22052.

14. Louis, E.D. Diagnosis and Management of Tremor. *Continuum (Minneap Minn)* **2016**, *22*, 1143–1158, doi:10.1212/CON.0000000000000346.

15. Bologna, M.; Leodori, G.; Stirpe, P.; Paparella, G.; Colella, D.; Belvisi, D.; Fasano, A.; Fabbrini, G.; Berardelli, A. Bradykinesia in Early and Advanced

Parkinson's Disease. *J Neurol Sci* **2016**, *369*, 286–291, doi:10.1016/J.JNS.2016.08.028.

16. Delis, D.; Direnfeld, L.; Alexander, M.P.; Kaplan, E. Cognitive Fluctuations Associated with On-off Phenomenon in Parkinson Disease. *Neurology* **1982**, *32*, 1049, doi:10.1212/WNL.32.9.1049.

17. Cabestany, J.; López, C.P.; Sama, A.; Moreno, J.M.; Bayes, A.; Rodriguez-Molinero, A. REMPARK: When AI and Technology Meet Parkinson Disease Assessment. In Proceedings of the Proceedings of the 20th International Conference Mixed Design of Integrated Circuits and Systems - MIXDES 2013; 2013; pp. 562–567.

18. Dhall, R.; Kreitzman, D.L. Advances in Levodopa Therapy for Parkinson Disease: Review of RYTARY (Carbidopa and Levodopa) Clinical Efficacy and Safety. *Neurology* **2016**, *86*, S13–S24, doi:10.1212/WNL.0000000000002510.

19. Giugni, J.C.; Okun, M.S. Treatment of Advanced Parkinson's Disease. *Curr Opin Neurol* **2014**, *27*, 450, doi:10.1097/WCO.0000000000000118.

20. Szlufik, S.; Kloda, M.; Potrzebowska, I.; Jaros, K.; Gędek, A.; Przybyszewski, A.; Mandat, T.; Koziorowski, D. STN DBS Improves Balance Disorders in Parkinson's Disease Patients and Impacts the Disease Progression. *Parkinsonism Relat Disord* **2023**, *113*, doi:10.1016/J.PARKRELDIS.2023.105579.

21. Rong, P.; Baig, F.; Marsili, L.; LeMoyne, R.; Fröhlich, H.; Valero, M.M.; J-c, C.; Gyseghem J-M, V.; Bontridder, N.; Petrovska-Delacréta, D.; et al. Leveraging the Potential of Digital Technology for Better Individualized Treatment of Parkinson's Disease. *Front Neurol* **2022**, *13*, 788427, doi:10.3389/FNEUR.2022.788427.

22. Yanase, J.; Triantaphyllou, E. A Systematic Survey of Computer-Aided Diagnosis in Medicine: Past and Present Developments. *Expert Syst Appl* **2019**, *138*, 112821, doi:10.1016/J.ESWA.2019.112821.

23. Sun, W.; Cai, Z.; Li, Y.; Liu, F.; Fang, S.; Wang, G. Data Processing and Text Mining Technologies on Electronic Medical Records: A Review. *J Healthc Eng* **2018**, *2018*, doi:10.1155/2018/4302425.

24. Shehab, M.; Abualigah, L.; Shambour, Q.; Abu-Hashem, M.A.; Shambour, M.K.Y.; Alsalibi, A.I.; Gandomi, A.H. Machine Learning in Medical Applications: A Review of State-of-the-Art Methods. *Comput Biol Med* **2022**, *145*, 105458, doi:10.1016/J.COMPBIOMED.2022.105458.

25. Oung, Q.W.; Hariharan, M.; Lee, H.L.; Basah, S.N.; Sarillee, M.; Lee, C.H. Wearable Multimodal Sensors for Evaluation of Patients with Parkinson Disease. *Proceedings - 5th IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2015* **2016**, 269–274, doi:10.1109/ICCSCE.2015.7482196.

26. Patel, S.; Lorincz, K.; Hughes, R.; Huggins, N.; Growdon, J.; Standaert, D.; Akay, M.; Dy, J.; Welsh, M.; Bonato, P. Monitoring Motor Fluctuations in Patients with Parkinsons Disease Using Wearable Sensors. *IEEE Trans Inf Technol Biomed* **2009**, *13*, 864–873, doi:10.1109/TITB.2009.2033471.

27. MJFF Levodopa Response Study - Syn20681023 - Wiki Available online: https://www.synapse.org/#!Synapse:syn20681023/wiki/594678 (accessed on 7 March 2022).

28. Sieberts, S.K.; Schaff, J.; Duda, M.; Pataki, B.Á.; Sun, M.; Snyder, P.; Daneault, J.F.; Parisi, F.; Costante, G.; Rubin, U.; et al. Crowdsourcing Digital Health Measures to Predict Parkinson's Disease Severity: The Parkinson's Disease Digital Biomarker DREAM Challenge. *npj Digital Medicine 2021 4:1* **2021**, *4*, 1–12, doi:10.1038/s41746-021-00414-7.

29. Lee, S.I.; Daneault, J.F.; Golabchi, F.N.; Patel, S.; Paganoni, S.; Shih, L.; Bonato, P. A Novel Method for Assessing the Severity of Levodopa-Induced Dyskinesia Using Wearable Sensors. In Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS; Institute of Electrical and Electronics Engineers Inc., November 4 2015; Vol. 2015-November, pp. 8087–8090.

30. Tsipouras, M.G.; Tzallas, A.T.; Rigas, G.; Bougia, P.; Fotiadis, D.I.; Konitsiotis, S. Automated Levodopa-Induced Dyskinesia Assessment. In Proceedings of the 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC'10; 2010; pp. 2411–2414.

31. Rodríguez-Molinero, A.; Pérez-López, C.; Samá, A.; De Mingo, E.; Rodríguez-Martín, D.; Hernández-Vara, J.; Bayés, Á.; Moral, A.; Álvarez, R.; Pérez-Martínez, D.A.; et al. A Kinematic Sensor and Algorithm to Detect Motor Fluctuations in Parkinson Disease: Validation Study Under Real Conditions of Use. *JMIR Rehabil Assist Technol* **2018**, *5*, doi:10.2196/REHAB.8335.

32. Thomas, I.; Westin, J.; Alam, M.; Bergquist, F.; Nyholm, D.; Senek, M.; Memedi, M. A Treatment-Response Index from Wearable Sensors for Quantifying Parkinson's Disease Motor States. *IEEE J Biomed Health Inform* **2018**, *22*, 1341–1349, doi:10.1109/JBHI.2017.2777926.

33. Sotirakis, C.; Su, Z.; Brzezicki, M.A.; Conway, N.; Tarassenko, L.; FitzGerald, J.J.; Antoniades, C.A. Identification of Motor Progression in Parkinson's Disease Using Wearable Sensors and Machine Learning. *npj Parkinsons Dis* **2023**, *9*, 1–8, doi:10.1038/s41531-023-00581-2.

34. Griffiths, R.I.; Kotschet, K.; Arfon, S.; Xu, Z.M.; Johnson, W.; Drago, J.; Evans, A.; Kempster, P.; Raghav, S.; Horne, M.K. Automated Assessment of Bradykinesia and Dyskinesia in Parkinson's Disease. *J Parkinsons Dis* **2012**, *2*, 47–55, doi:10.3233/JPD-2012-11071.

35. San-Segundo, R.; Zhang, A.; Cebulla, A.; Panev, S.; Tabor, G.; Stebbins, K.; Massa, R.E.; Whitford, A.; de la Torre, F.; Hodgins, J. Parkinson's Disease Tremor Detection in the Wild Using Wearable Accelerometers. *Sensors 2020, Vol. 20, Page 5817* **2020**, *20*, 5817, doi:10.3390/S20205817.

36. Papadopoulos, A.; Kyritsis, K.; Bostanjopoulou, S.; Klingelhoefer, L.; Chaudhuri, R.K.; Delopoulos, A. Multiple-Instance Learning for In-The-Wild Parkinsonian Tremor Detection. *Annu Int Conf IEEE Eng Med Biol Soc* **2019**, 6188–6191, doi:10.1109/EMBC.2019.8856314.

37. Palmerini, L.; Klenk, J.; Becker, C.; Chiari, L. Accelerometer-Based Fall Detection Using Machine Learning: Training and Testing on Real-World Falls. *Sensors 2020, Vol. 20, Page 6479* **2020**, *20*, 6479, doi:10.3390/S20226479.

38. Pereira, C.R.; Weber, S.A.T.; Hook, C.; Rosa, G.H.; Papa, J.P. Deep Learning-Aided Parkinson's Disease Diagnosis from Handwritten Dynamics. *Proceedings - 2016 29th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2016* **2017**, 340–346, doi:10.1109/SIBGRAPI.2016.054.

39. Rios-Urrego, C.D.; Vásquez-Correa, J.C.; Vargas-Bonilla, J.F.; Nöth, E.; Lopera, F.; Orozco-Arroyave, J.R. Analysis and Evaluation of Handwriting in Patients with Parkinson's Disease Using Kinematic, Geometrical, and Non-Linear Features. *Comput Methods Programs Biomed* **2019**, *173*, 43–52, doi:10.1016/j.cmpb.2019.03.005.

40. Drotár, P.; Mekyska, J.; Rektorová, I.; Masarová, L.; Smékal, Z.; Faundez-Zanuy, M. Analysis of In-Air Movement in Handwriting: A Novel Marker for Parkinson's Disease. *Comput Methods Programs Biomed* **2014**, *117*, 405–411, doi:10.1016/J.CMPB.2014.08.007.

41. Mucha, J.; Zvoncak, V.; Galaz, Z.; Faundez-Zanuy, M.; Mekyska, J.; Kiska, T.; Smekal, Z.; Brabenec, L.; Rektorova, I.; Lopez-De-Ipina, K. Fractional Derivatives of Online Handwriting: A New Approach of Parkinsonic Dysgraphia Analysis. *2018 41st International Conference on Telecommunications and Signal Processing, TSP 2018* **2018**, doi:10.1109/TSP.2018.8441293.

42. Galaz, Z.; Drotar, P.; Mekyska, J.; Gazda, M.; Mucha, J.; Zvoncak, V.; Smekal, Z.; Faundez-Zanuy, M.; Castrillon, R.; Orozco-Arroyave, J.R.; et al. Comparison of CNN-Learned vs. Handcrafted Features for Detection of Parkinson's Disease Dysgraphia in a Multilingual Dataset. *Front Neuroinform* **2022**, *16*, 877139, doi:10.3389/FNINF.2022.877139.

43. Little, M.A.; McSharry, P.E.; Hunter, E.J.; Spielman, J.; Ramig, L.O. Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease. *IEEE Trans Biomed Eng* **2009**, *56*, 1015–1022, doi:10.1109/TBME.2008.2005954.

44. Bayestehtashk, A.; Asgari, M.; Shafran, I.; McNames, J. Fully Automated Assessment of the Severity of Parkinson's Disease from Speech. *Comput Speech Lang* **2015**, *29*, 172–185, doi:10.1016/j.csl.2013.12.001.

45. Rueda, A.; Krishnan, S. Feature Analysis of Dysphonia Speech for Monitoring Parkinson's Disease. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* **2017**, 2308–2311, doi:10.1109/EMBC.2017.8037317.

46. Frid, A.; Hazan, H.; Hilu, D.; Manevitz, L.; Ramig, L.O.; Sapir, S. Computational Diagnosis of Parkinson's Disease Directly from Natural Speech Using Machine Learning Techniques. In Proceedings of the International Conference on Software Science, Technology and Engineering, SWSTE 2014; IEEE Computer Society, 2014; pp. 50–53.

47. Wodzinski, M.; Skalski, A.; Hemmerling, D.; Orozco-Arroyave, J.R.; Noth, E. Deep Learning Approach to Parkinson's Disease Detection Using Voice

Recordings and Convolutional Neural Network Dedicated to Image Classification. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* **2019**, 717–720, doi:10.1109/EMBC.2019.8856972.

48. Rehman, A.; Saba, T.; Mujahid, M.; Alamri, F.S.; ElHakim, N. Parkinson's Disease Detection Using Hybrid LSTM-GRU Deep Learning Model. *Electronics 2023, Vol. 12, Page 2856* **2023**, *12*, 2856, doi:10.3390/ELECTRONICS12132856.

49. Orozco-Arroyave, J.R.; Vásquez-Correa, J.C.; Vargas-Bonilla, J.F.; Arora, R.; Dehak, N.; Nidadavolu, P.S.; Christensen, H.; Rudzicz, F.; Yancheva, M.; Chinaei, H.; et al. NeuroSpeech: An Open-Source Software for Parkinson's Speech Analysis. *Digit Signal Process* **2018**, *77*, 207–221, doi:10.1016/j.dsp.2017.07.004.

50. Szymański, A.; Szlufik, S.; Koziorowski, D.M.; Przybyszewski, A.W. Building Classifiers for Parkinson's Disease Using New Eye Tribe Tracking Method. *Lecture Notes in Computer Science* **2017**, *10192 LNAI*, 351–358, doi:10.1007/978-3-319-54430-4_34.

51. Zhao, S.; Dai, G.; Li, J.; Zhu, X.; Huang, X.; Li, Y.; Tan, M.; Wang, L.; Fang, P.; Chen, X.; et al. An Interpretable Model Based on Graph Learning for Diagnosis of Parkinson's Disease with Voice-Related EEG. *npj Digit Med* **2024**, *7*, 1–12, doi:10.1038/s41746-023-00983-9.

52. Lee, C.Y.; Kang, S.J.; Hong, S.K.; Ma, H. Il; Lee, U.; Kim, Y.J. A Validation Study of a Smartphone-Based Finger Tapping Application for Quantitative Assessment of Bradykinesia in Parkinson's Disease. *PLoS One* **2016**, *11*, e0158852, doi:10.1371/JOURNAL.PONE.0158852.

53. Khan, T.; Nyholm, D.; Westin, J.; Dougherty, M. A Computer Vision Framework for Finger-Tapping Evaluation in Parkinson's Disease. *Artif Intell Med* **2014**, *60*, 27–40, doi:10.1016/J.ARTMED.2013.11.004.

54. Adams, W.R. High-Accuracy Detection of Early Parkinson's Disease Using Multiple Characteristics of Finger Movement While Typing. *PLoS One* **2017**, *12*, e0188226, doi:10.1371/JOURNAL.PONE.0188226.

55. Matarazzo, M.; Arroyo-Gallego, T.; Montero, P.; Puertas-Martín, V.; Butterworth, I.; Mendoza, C.S.; Ledesma-Carbayo, M.J.; Catalán, M.J.; Molina, J.A.; Bermejo-Pareja, F.; et al. Remote Monitoring of Treatment Response in Parkinson's Disease: The Habit of Typing on a Computer. *Mov Disord* **2019**, *34*, 1488–1495, doi:10.1002/MDS.27772.

56. Aghanavesi, S.; Westin, J.; Bergquist, F.; Nyholm, D.; Askmark, H.; Aquilonius, S.M.; Constantinescu, R.; Medvedev, A.; Spira, J.; Ohlsson, F.; et al. A Multiple Motion Sensors Index for Motor State Quantification in Parkinson's Disease. *Comput Methods Programs Biomed* **2020**, *189*, 105309, doi:10.1016/J.CMPB.2019.105309.

57. Bot, B.M.; Suver, C.; Neto, E.C.; Kellen, M.; Klein, A.; Bare, C.; Doerr, M.; Pratap, A.; Wilbanks, J.; Dorsey, E.R.; et al. The MPower Study, Parkinson Disease Mobile Data Collected Using ResearchKit. *Scientific Data 2016 3:1* **2016**, *3*, 1–9, doi:10.1038/sdata.2016.11.

58. Schwab, P.; Karlen, W. PhoneMD: Learning to Diagnose Parkinson's Disease from Smartphone Data. *Proceedings of the AAAI Conference on Artificial Intelligence* **2019**, *33*, 1118–1125, doi:10.1609/AAAI.V33I01.33011118.

59. Zhan, A.; Mohan, S.; Tarolli, C.; Schneider, R.B.; Adams, J.L.; Sharma, S.; Elson, M.J.; Spear, K.L.; Glidden, A.M.; Little, M.A.; et al. Using Smartphones and Machine Learning to Quantify Parkinson Disease Severity: The Mobile Parkinson Disease Score. *JAMA Neurol* **2018**, *75*, 876–880, doi:10.1001/JAMANEUROL.2018.0809.

60. Elm, J.J.; Daeschler, M.; Bataille, L.; Schneider, R.; Amara, A.; Espay, A.J.; Afek, M.; Admati, C.; Teklehaimanot, A.; Simuni, T. Feasibility and Utility of a Clinician Dashboard from Wearable and Mobile Application Parkinson's Disease Data. *npj Digit Med* **2019**, *2*, 1–6, doi:10.1038/s41746-019-0169-y.

61. Chan, P.L.S.; Nutt, J.G.; Holford, N.H.G. Importance of Within Subject Variation in Levodopa Pharmacokinetics: A 4 Year Cohort Study in Parkinson's Disease. *J Pharmacokinet Pharmacodyn* **2005**, *32*, doi:10.1007/s10928-005-0039-x.

62. Westin, J.; Nyholm, D.; Pålhagen, S.; Willows, T.; Groth, T.; Dougherty, M.; Karlsson, M.O. A Pharmacokinetic-Pharmacodynamic Model for Duodenal Levodopa Infusion. *Clin Neuropharmacol* **2011**, *34*, 61–65, doi:10.1097/WNF.0B013E31820B570A.

63. Thomas, I.; Alam, M.; Nyholm, D.; Senek, M.; Westin, J. Individual Dose-Response Models for Levodopa Infusion Dose Optimization. *Int J Med Inform* **2018**, *112*, 137–142, doi:10.1016/J.IJMEDINF.2018.01.018.

64. Thomas, I.; Alam, M.; Bergquist, F.; Johansson, D.; Memedi, M.; Nyholm, D.; Westin, J. Sensor-Based Algorithmic Dosing Suggestions for Oral Administration of Levodopa/Carbidopa Microtablets for Parkinson's Disease: A First Experience. *J Neurol* **2019**, *266*, 651–658, doi:10.1007/S00415-019-09183-6.

65. Watts, J.; Khojandi, A.; Vasudevan, R.; Ramdhani, R. Optimizing Individualized Treatment Planning for Parkinson's Disease Using Deep Reinforcement Learning. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* **2020**, *2020-July*, 5406–5409, doi:10.1109/EMBC44109.2020.9175311.

66. Isaacson, S.H.; Boroojerdi, B.; Waln, O.; McGraw, M.; Kreitzman, D.L.; Klos, K.; Revilla, F.J.; Heldman, D.; Phillips, M.; Terricabras, D.; et al. Effect of Using a Wearable Device on Clinical Decision-Making and Motor Symptoms in Patients with Parkinson's Disease Starting Transdermal Rotigotine Patch: A Pilot Study. *Parkinsonism Relat Disord* **2019**, *64*, 132–137, doi:10.1016/J.PARKRELDIS.2019.01.025.

67. Gutowski, T.; Chmielewski, M. An Algorithmic Approach for Quantitative Evaluation of Parkinson's Disease Symptoms and Medical Treatment Utilizing Wearables and Multi-Criteria Symptoms Assessment. *IEEE Access* **2021**, *9*, 24133–24144, doi:10.1109/ACCESS.2021.3056629.

68. Gutowski, T. Deep Learning for Parkinson's Disease Symptom Detection and Severity Evaluation Using Accelerometer Signal. **2022**, 271–276, doi:10.14428/ESANN/2022.ES2022-107.

69. Gutowski, T.; Antkiewicz, R.; Szlufik, S. Machine Learning with Optimization to Create Medicine Intake Schedules for Parkinson's Disease Patients. *PLoS One* **2023**, *18*, e0293123, doi:10.1371/JOURNAL.PONE.0293123.

70. Boyd, K.; Eng, K.H.; Page, C.D. Area under the Precision-Recall Curve: Point Estimates and Confidence Intervals. *Lecture Notes in Computer Science* **2013**, *8190 LNAI*, 451–466, doi:10.1007/978-3-642-40994-3_29.

71. Senek, M.; Hellström, M.; Albo, J.; Svenningsson, P.; Nyholm, D. First Clinical Experience with Levodopa/Carbidopa Microtablets in Parkinson's Disease. *Acta Neurol Scand* **2017**, *136*, 727–731, doi:10.1111/ANE.12756.

72. Pahwa, R.; Lyons, K.E. Essential Tremor: Differential Diagnosis and Current Therapy. *Am J Med* **2003**, *115*, 134–142, doi:10.1016/S0002-9343(03)00259-6.

73. Mark, M.H. Lumping and Splitting the Parkinson plus Syndromes: Dementia with Lewy Bodies, Multiple System Atrophy, Progressive Supranuclear Palsy, and Cortical-Basal Ganglionic Degeneration. *Neurol Clin* **2001**, *19*, 607–627, doi:10.1016/S0733-8619(05)70037-2.

74. Visconti, P.; Gaetani, F.; Zappatore, G.A.; Primiceri, P. Technical Features and Functionalities of Myo Armband: An Overview on Related Literature and Advanced Applications of Myoelectric Armbands Mainly Focused on Arm Prostheses. *Int. J Smart Sens Intell Syst* **2018**, *11*, 1–25, doi:10.21307/IJSSIS-2018-005.

75. Home - SiFi Labs Available online: https://sifilabs.com/ (accessed on 26 July 2023).

76. Goetz, C.G.; Tilley, B.C.; Shaftman, S.R.; Stebbins, G.T.; Fahn, S.; Martinez-Martin, P.; Poewe, W.; Sampaio, C.; Stern, M.B.; Dodel, R.; et al. Movement Disorder Society-Sponsored Revision of the Unified Parkinson's Disease Rating Scale (MDS-UPDRS): Scale Presentation and Clinimetric Testing Results. *Mov Disord* **2008**, *23*, 2129–2170, doi:10.1002/MDS.22340.

77. Gutowski, T.; Chmielewski, M. Sensor and User Interaction-Based Evaluation of Clinical Trials Implemented in A Form of a Mobile System Assisting Parkinson's Disease Treatment. In Proceedings of the Proceedings of the 36th International Business Information Management Association (IBIMA); Granada, Spain, November 4 2020; pp. 10723–10731.

78. Drotár, P.; Mekyska, J.; Rektorová, I.; Masarová, L.; Smékal, Z.; Faundez-Zanuy, M. Evaluation of Handwriting Kinematics and Pressure for Differential Diagnosis of Parkinson's Disease. *Artif Intell Med* **2016**, *67*, 39–46, doi:10.1016/J.ARTMED.2016.01.004.

79. Toffoli, S.; Lunardini, F.; Parati, M.; Gallotta, M.; De Maria, B.; Longoni, L.; Dell'Anna, M.E.; Ferrante, S. Spiral Drawing Analysis with a Smart Ink Pen to Identify Parkinson's Disease Fine Motor Deficits. *Front Neurol* **2023**, *14*, doi:10.3389/FNEUR.2023.1093690.

80. Drotar, P.; Mekyska, J.; Smekal, Z.; Rektorova, I.; Masarova, L.; Faundez-Zanuy, M. Contribution of Different Handwriting Modalities to Differential Diagnosis of Parkinson's Disease. *2015 IEEE International Symposium on Medical Measurements and Applications, MeMeA 2015 - Proceedings* **2015**, 344–348, doi:10.1109/MEMEA.2015.7145225.

81. Bocklet, T.; Steidl, S.; Nöth, E.; Skodda, S. Automatic Evaluation of Parkinson's Speech - Acoustic, Prosodic and Voice Related Cues. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* **2013**, 1149–1153, doi:10.21437/INTERSPEECH.2013-313.

82. Orozco-Arroyave, J.R.; Hönig, F.; Arias-Londoño, J.D.; Vargas-Bonilla, J.F.; Daqrouq, ; K; Skodda, ; S; Rusz, ; J; Nöth, ; E; Arias-Londo~, J.D.; Daqrouq, K.; et al. Automatic Detection of Parkinson's Disease in Running Speech Spoken in Three Different Languages. *J Acoust Soc Am* **2016**, *139*, 481–500, doi:10.1121/1.4939739.

83. Cantürk, İ.; Karabiber, F. A Machine Learning System for the Diagnosis of Parkinson's Disease from Speech Signals and Its Application to Multiple Speech Signal Types. *Arab J. Sci Eng* **2016**, *41*, 5049–5059, doi:10.1007/S13369-016-2206-3.

84. Bocklet, T.; Nöth, E.; Stemmer, G.; Ruzickova, H.; Rusz, J. Detection of Persons with Parkinson's Disease by Acoustic, Vocal, and Prosodic Analysis. *2011 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2011, Proceedings* **2011**, 478–483, doi:10.1109/ASRU.2011.6163978.

85. Chromiec, P.A.; Urbaś, Z.K.; Jacko, M.; Kaczor, J.J. The Proper Diet and Regular Physical Activity Slow Down the Development of Parkinson Disease. *Aging Dis* **2021**, *12*, 1605, doi:10.14336/AD.2021.0123.

86. Spyros, S.(; Papapetropoulos,) Patient Diaries As a Clinical Endpoint in Parkinson's Disease Clinical Trials. *CNS Neurosci Ther* **2012**, *18*, 380–387, doi:10.1111/J.1755-5949.2011.00253.X.

87. Weiss, K.; Khoshgoftaar, T.M.; Wang, D.D. A Survey of Transfer Learning. *J Big Data* **2016**, *3*, 1–40, doi:10.1186/S40537-016-0043-6.

88. Potdar, K. A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers. *Int J Comput Appl* **2017**, *175*, 975–8887.

89. Wang, S.-C. Artificial Neural Network. *Interdisciplinary Computing in Java Programming* **2003**, 81–100, doi:10.1007/978-1-4615-0377-4_5.

90. Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a Convolutional Neural Network. *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017* **2017**, *2018-January*, 1–6, doi:10.1109/ICENGTECHNOL.2017.8308186.

91. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Curran Associates, Inc., 2019; pp. 8024–8035.

92. Martín~Abadi; Ashish~Agarwal; Paul~Barham; Eugene~Brevdo; Zhifeng~Chen; Craig~Citro; Greg~S.~Corrado; Andy~Davis; Jeffrey~Dean; Matthieu~Devin; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems 2015.

93. Hsiao, T.Y.; Chang, Y.C.; Chou, H.H.; Chiu, C. Te Filter-Based Deep-Compression with Global Average Pooling for Convolutional Networks. *J. Syst Archit* **2019**, *95*, 9–18, doi:10.1016/J.SYSARC.2019.02.008.

94. Kingma, D.P.; Ba, J.L. Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* **2014**.

95. Bishop, C.M. Linear Models for Classification. In *Pattern Recognition and Machine Learning*; Bishop, C.M., Ed.; Springer New York: New York, NY, 2006; pp. 179–224 ISBN 978-0-387-45528-0.

96. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput* **1997**, *9*, 1735–1780, doi:10.1162/NECO.1997.9.8.1735.

97. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. *Adv Neural Inf Process Syst* **2017**, *2017-December*, 5999–6009.

98. Dahouda, M.K.; Joe, I. A Deep-Learned Embedding Technique for Categorical Features Encoding. *IEEE Access* **2021**, *9*, 114381–114391, doi:10.1109/ACCESS.2021.3104357.

99. Stone, S.; Spector, E. Deep Neural Network for Multi-Pitch Estimation Using Weighted Cross Entropy Loss. *2021 IEEE Western New York Image and Signal Processing Workshop, WNYISPW 2021* **2021**, doi:10.1109/WNYISPW53194.2021.9661285.

100. García, V.; Mollineda, R.A.; Sánchez, J.S. Index of Balanced Accuracy: A Performance Measure for Skewed Class Distributions. *Lecture Notes in Computer Science* **2009**, *5524 LNCS*, 441–448, doi:10.1007/978-3-642-02172-5_57.

101. Susmaga, R. Confusion Matrix Visualization. *Intelligent Information Processing and Web Mining* **2004**, 107–116, doi:10.1007/978-3-540-39985-8_12.

102. Naser, M.Z.; Amir, ·; Alavi, H. Error Metrics and Performance Fitness Indicators for Artificial Intelligence and Machine Learning in Engineering and Sciences. *Archit Struct Constr* **2021**, *3*, 499–517, doi:10.1007/S44150-021-00015-8.

103. Erer, K.S. Adaptive Usage of the Butterworth Digital Filter. *J Biomech* **2007**, *40*, 2934–2943, doi:10.1016/J.JBIOMECH.2007.02.019.

104. Bazgir, O.; Frounchi, J.; Habibi, S.A.H.; Palma, L.; Pierleoni, P. A Neural Network System for Diagnosis and Assessment of Tremor in Parkinson Disease Patients. In Proceedings of the 2015 22nd Iranian Conference on Biomedical Engineering, ICBME 2015; Institute of Electrical and Electronics Engineers Inc., February 9 2016; pp. 1–5.

105. Sejdic, E.; Lowry, K.A.; Bellanca, J.; Redfern, M.S.; Brach, J.S. A Comprehensive Assessment of Gait Accelerometry Signals in Time, Frequency and Time-Frequency Domains. *IEEE Trans Neural Syst Rehabil Eng* **2014**, *22*, 603–612, doi:10.1109/TNSRE.2013.2265887.

106. Tsipouras, M.G.; Tzallas, A.T.; Rigas, G.; Tsouli, S.; Fotiadis, D.I.; Konitsiotis, S. An Automated Methodology for Levodopa-Induced Dyskinesia: Assessment Based on Gyroscope and Accelerometer Signals. *Artif Intell Med* **2012**, *55*, 127–135, doi:10.1016/j.artmed.2012.03.003.

107. Alam, M.N.; Johnson, B.; Gendreau, J.; Tavakolian, K.; Combs, C.; Fazel-Rezai, R. Tremor Quantification of Parkinson's Disease - A Pilot Study. In Proceedings of the IEEE International Conference on Electro Information Technology; IEEE Computer Society, August 5 2016; Vol. 2016-August, pp. 755–759.

108. Eskofier, B.M.; Lee, S.I.; Daneault, J.F.; Golabchi, F.N.; Ferreira-Carvalho, G.; Vergara-Diaz, G.; Sapienza, S.; Costante, G.; Klucken, J.; Kautz, T.; et al. Recent Machine Learning Advancements in Sensor-Based Mobility Analysis: Deep Learning for Parkinson's Disease Assessment. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* **2016**, *2016-October*, 655–658, doi:10.1109/EMBC.2016.7590787.

109. Duhamel, P.; Vetterli, M. Fast Fourier Transforms: A Tutorial Review and a State of the Art. *Signal Process* **1990**, *19*, 259–299, doi:10.1016/0165-1684(90)90158-U.

110. Delgado-Bonal, A.; Marshak, A. Approximate Entropy and Sample Entropy: A Comprehensive Tutorial. *Entropy* **2019**, *21*, doi:10.3390/E21060541.

111. Bishop, C.M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*; Springer-Verlag New York, Inc.: Secaucus, NJ, USA, 2006; ISBN 0387310738.

112. Bentéjac, C.; Csörgő, A.; Martínez-Muñoz, G. A Comparative Analysis of Gradient Boosting Algorithms. *Artif Intell Rev* **2021**, *54*, 1937–1967, doi:10.1007/S10462-020-09896-5/TABLES/12.

113. Impedovo, D.; Pirlo, G.; Vessio, G. Dynamic Handwriting Analysis for Supporting Earlier Parkinson's Disease Diagnosis. *Information (Switzerland)* **2018**, *9*, doi:10.3390/info9100247.

114. Tucha, O.; Mecklinger, L.; Thome, J.; Reiter, A.; Alders, G.L.; Sartor, H.; Naumann, M.; Lange, K.W. Kinematic Analysis of Dopaminergic Effects on Skilled Handwriting Movements in Parkinson's Disease. *J Neural Transm* **2006**, *113*, 609–623, doi:10.1007/s00702-005-0346-9.

115. Yokoe, M.; Okuno, R.; Hamasaki, T.; Kurachi, Y.; Akazawa, K.; Sakoda, S. Opening Velocity, a Novel Parameter, for Finger Tapping Test in Patients with Parkinson's Disease. *Parkinsonism Relat Disord* **2009**, *15*, 440–444, doi:10.1016/J.PARKRELDIS.2008.11.003.

116. Jobbágy, Á.; Harcos, P.; Karoly, R.; Fazekas, G. Analysis of Finger-Tapping Movement. *J Neurosci Methods* **2005**, *141*, 29–39, doi:10.1016/J.JNEUMETH.2004.05.009.

117. Prince, J.; De Vos, M. A Deep Learning Framework for the Remote Detection of Parkinson'S Disease Using Smart-Phone Sensor Data. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* **2018**, *2018-July*, 3144–3147, doi:10.1109/EMBC.2018.8512972.

118. Chen, O.Y.; Lipsmeier, F.; Phan, H.; Prince, J.; Taylor, K.I.; Gossens, C.; Lindemann, M.; Vos, M. De Building a Machine-Learning Framework to Remotely Assess Parkinson's Disease Using Smartphones. *IEEE Trans Biomed Eng* **2020**, *67*, 3491–3500, doi:10.1109/TBME.2020.2988942.

119. Guyon Isabelle; Elisseeff André An Introduction to Variable and Feature Selection. *J Mach Learn Res* **2003**, *3*, 1157–1182, doi:10.5555/944919.944968.

120. Archana, T.; Sachin, D. Dimensionality Reduction and Classification through PCA and LDA. *Int J Comput Appl* **2015**, *122*, 4–8, doi:10.5120/21790-5104.

121. 1.13. Feature Selection — Scikit-Learn 1.5.0 Documentation Available online: https://scikit-learn.org/stable/modules/feature_selection.html#selectfrommodel (accessed on 14 June 2024).

122. Wong, T.T. Performance Evaluation of Classification Algorithms by K-Fold and Leave-One-out Cross Validation. *Pattern Recognit* **2015**, *48*, 2839–2846, doi:10.1016/J.PATCOG.2015.03.009.

123. Johansson, D.; Thomas, I.; Ericsson, A.; Johansson, A.; Medvedev, A.; Memedi, M.; Nyholm, D.; Ohlsson, F.; Senek, M.; Spira, J.; et al. Evaluation of a Sensor Algorithm for Motor State Rating in Parkinson's Disease. *Parkinsonism Relat Disord* **2019**, *64*, 112–117, doi:10.1016/J.PARKRELDIS.2019.03.022.

124. Nutt, J.G. Pharmacokinetics and Pharmacodynamics of Levodopa. *Mov Disord* **2008**, *23*, S580–S584, doi:10.1002/MDS.22037.

125. Lewitt, P.A.; Library, W.O. Levodopa Therapy for Parkinson's Disease: Pharmacokinetics and Pharmacodynamics. *Mov Disord* **2015**, *30*, 64–72, doi:10.1002/MDS.26082.

126. Murtagh, F. Multilayer Perceptrons for Classification and Regression. *Neurocomputing* **1991**, *2*, 183–197, doi:10.1016/0925-2312(91)90023-5.

127. Chollet, F.; others Keras 2015.

128. Hora, J.; Campos, P. A Review of Performance Criteria to Validate Simulation Models. *Expert Syst* **2015**, *32*, 578–595, doi:10.1111/EXSY.12111.

129. Raket, L.L.; Oudin Åström, D.; Norlin, J.M.; Kellerborg, K.; Martinez-Martin, P.; Odin, P. Impact of Age at Onset on Symptom Profiles, Treatment Characteristics and Health-Related Quality of Life in Parkinson's Disease. *Sci Rep* **2022**, *12*, 1–13, doi:10.1038/s41598-021-04356-8.

130. Saari, L.; Backman, E.A.; Wahlsten, P.; Gardberg, M.; Kaasinen, V. Height and Nigral Neuron Density in Parkinson's Disease. *BMC Neurol* **2022**, *22*, doi:10.1186/S12883-022-02775-2.

131. Batarseh, N.; Al Thaher, Y. High-Fat Diet and Related Obesity Provoke Neurotoxins and Alter Neuro-Biomarkers Involved in Parkinson's Disease. *Obes Med* **2023**, *41*, 100500, doi:10.1016/J.OBMED.2023.100500.

132. Mollenhauer, B.; Zimmermann, J.; Sixel-Döring, F.; Focke, N.K.; Wicke, T.; Ebentheuer, J.; Schaumburg, M.; Lang, E.; Friede, T.; Trenkwalder, C. Baseline Predictors for Progression 4 Years after Parkinson's Disease Diagnosis in the De Novo Parkinson Cohort (DeNoPa). *Mov Disord* **2019**, *34*, 67–77, doi:10.1002/MDS.27492.

133. Mirjalili, S. Evolutionary Algorithms and Neural Networks. **2019**, *780*, doi:10.1007/978-3-319-93025-1.

134. Zelinka, I.; Snasel, V.; Abraham, A. Handbook of Optimization: From Classical to Modern Approach. *Intelligent Systems Reference Library* **2013**, *38*, doi:10.1007/978-3-642-30504-7.

135. Fortin, F.-A.; Marc-André Gardner, U.; Parizeau, M.; Gagné, C. DEAP: Evolutionary Algorithms Made Easy François-Michel De Rainville. *J Mach Learn Res* **2012**, *13*, 2171–2175.

136. Marco Wiering; Martijn van Otterlo *Reinforcement Learning State-of-the-Art (Adaptation, Learning, and Optimization, 12)*; Springer, 2012; ISBN 9783642015267.

137. Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. OpenAI Gym 2016.

138. Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *J Mach Learn Res* **2021**, *22*, 1–8.

139. Adler-Grinberg, D. Questioning Our Classical Understanding of Accommodation and Presbyopia. *Am J Optom Physiol Opt* **1986**, *63*, 571–580, doi:10.1097/00006324-198607000-00012.

140. Rodríguez, C.; Baez, M.; Daniel, F.; Casati, F.; Trabucco, J.C.; Canali, L.; Percannella, G. REST APIs: A Large-Scale Analysis of Compliance with Principles and Best Practices. *Lecture Notes in Computer Science* **2016**, *9671*, 21–39, doi:10.1007/978-3-319-38791-8_2.

141. Android Distribution Chart – Composables.Com Available online: https://www.composables.com/tools/distribution-chart (accessed on 19 June 2024).

142. Côté-Allard, U.; Gagnon-Turcotte, G.; Laviolette, F.; Gosselin, B. A Low-Cost, Wireless, 3-D-Printed Custom Armband for SEMG Hand Gesture Recognition. *Sensors* **2019**, *19*, 2811, doi:10.3390/s19122811.

143. Daemen, J.; Rijmen, V. The Design of Rijndael. **2002**, doi:10.1007/978-3-662-04722-4.

144. Zhao, D.; Gao, X.-Z.; Pan, Y.; -, al; Chen, J.; Bao, N.; Zhu -, Z.; Pangidoan Batubara, T.; Efendi, S.; Budhiarti Nababan, E. Analysis Performance BCRYPT Algorithm to Improve Password Security from Brute Force. J Phys Conf Ser 2021, 1811, 012129, doi:10.1088/1742-6596/1811/1/012129.

10. List of Tables

11. List of Figures

12. Abstract

Optimization of Medicine Dosing in Parkinson's Disease, Based on Signals from Sensor Measurements

Tomasz GUTOWSKI

Keywords: machine learning, Parkinson’s disease, optimization, artificial intelligence, signal processing

Parkinson's disease (PD) presents significant challenges in management, requiring precision, and a deep understanding of each patient's experience with the disease. This thesis explores the application of machine learning (ML) and optimization techniques to enhance the treatment of PD, focusing on creating personalized medication schedules. The primary aim is to develop a method that suggests optimal doses and intake times for medication, specifically levodopa, the main medication used in PD treatment, to maintain patients in an optimal state throughout the day.

The research presented in the thesis is divided into several key areas:

- symptom severity evaluation,
- medicine response modelling,
- optimization of medication schedules,
- implementation of the patient monitoring system.

Symptom severity evaluation involved developing machine and deep learning models to predict the severity of PD motor symptoms using data collected from mobile devices and wearable sensors. Experiments focused on determining how different exercises could be used to predict the severity of individual symptoms and the overall state of the patient were the main part of the chapter. The best results were obtained from inertial sensor signals such as accelerometers and gyroscopes. The study highlights the effectiveness of both machine and deep learning models, with the latter showing slightly better performance but requiring significantly more data.

Medicine response modeling included building predictive models to understand individual patient responses to medication. These models were based

on neural networks particularly on Long short-term memory cells and demonstrated success in predicting patient states after medication intakes. The study validated these models on both synthetic and real patients, showing that integration of patient demographic and clinical data allows for personalized medication response predictions.

Optimization of medication schedules employed optimization algorithms and reinforcement learning to create personalized levodopa intake schedules. These methods provided flexibility in dose sizes and intervals, leading to more personalized treatment plans. Comparison of optimization results using both pharmacokinetic/pharmacodynamic and ML models showed close alignment, confirming the applicability of the proposed methods.

The last part of the thesis presents a system implemented to support real-time data collection and patient monitoring. This system includes a mobile application for patients and a web platform for clinicians. The mobile application allows patients to easily record their symptoms, medication intake, and other relevant data in real-time. This data is then synchronized with the web application, where clinicians can monitor patient progress, and make decisions about treatment adjustments. The integration of these tools improves real-time data collection and continuous patient monitoring, ensuring that any changes in the patient's condition can be promptly addressed. By providing a simple interface for both patients and clinicians, this system supports continuous patient care and enables the development and implementation of personalized treatment strategies that are tailored to the individual needs of each patient.

The findings of this thesis demonstrate the potential for significant improvements in symptom management and patient quality of life through personalized treatment approaches. Recommendations for future research include conducting larger clinical trials, exploring additional patient-specific factors, and updating optimization tasks to further enhance model accuracy and applicability.

In conclusion, this dissertation presents a comprehensive approach to personalized PD treatment, integrating ML models and optimization algorithms to offer a promising direction for future PD treatment strategies.

13. Abstract in Polish

Optymalizacja dawkowania leków w chorobie Parkinsona na podstawie sygnałów z pomiarów sensorowych

Tomasz GUTOWSKI

Słowa kluczowe: uczenie maszynowe, choroba Parkinsona, optymalizacja, sztuczna inteligencja, przetwarzanie sygnałów

Choroba Parkinsona (PD) stwarza wiele trudności w terapii wymagając dokładności oraz głębokiego zrozumienia indywidualnych potrzeb pacjentów z chorobą. W tej pracy zbadano zastosowanie metod uczenia maszynowego i metod optymalizacji w celu wsparcia terapii PD skupiając się na budowaniu zindywidualizowanych harmonogramów przyjmowania leków. Głównym celem pracy jest przygotowanie metody, które będzie w stanie sugerować optymalne dawki leków oraz czasy ich przyjęcia, w szczególności dla lewodopy - głównego leku stosowanego w terapii PD, tak aby utrzymać pacjenta w optymalnym stanie jak najdłużej w ciągu dnia.

Zagadnienia zawarte w pracy podzielono na cztery główne obszary:

- ocena intensywności objawów,
- modelowanie reakcji na lek,
- optymalizacja harmonogramów przyjmowania leków,
- implementacja systemu do monitorowania pacjentów.

W ramach oceny intensywności objawów zbudowano modele uczenia maszynowego oraz głębokiego, których celem jest dokonanie oceny aktualnego stanu pacjenta w oparciu o dane zebrane z sensorów wbudowanych w urządzenia mobilne oraz opaski. Obiektem badań niniejszej pracy było określenie, jak zróżnicowane ćwiczenia mogłyby zostać wykorzystane do predykcji intensywności poszczególnych objawów i ogólnego stanu pacjenta. Najlepsze wyniki uzyskano korzystając z sygnałów z sensorów inercyjnych takich jak akcelerometry i żyroskopy. Ponadto badania wykazały również efektywność klasycznych metod uczenia maszynowego oraz metod głębokich. Metody uczenia głębokiego radziły sobie lepiej wymagając jednocześnie więcej danych.

Modelowanie odpowiedzi na lek obejmowało budowanie modeli predykcyjnych w celu zrozumienia indywidualnych reakcji pacjentów na leki. Modele te zostały oparte na sieciach neuronowych, w szczególności wykorzystano komórki LSTM, które wykazały wysoką dokładność w predykcji stanów pacjenta po przyjęciu dawek leków. Podczas badań modele te zostały zweryfikowane na danych pacjentów syntetycznych oraz rzeczywistych, pokazując, że integracja danych demograficznych i klinicznych wspiera personalizację predykcji reakcji na lek.

Optymalizacja harmonogramów dawkowania leków wykorzystała algorytmy optymalizacji i uczenie ze wzmocnieniem w celu zbudowania zindywidualizowanych harmonogramów przyjmowania leków. Metody te zapewniły elastyczność w rozmiarach dawek i okresach między nimi, pozwalając na personalizację planów leczenia. Porównanie wyników optymalizacji uzyskanych z wykorzystaniem modeli farmakokinetyczno-farmakodynamicznych i modeli uczenia maszynowego pokazało, że różnice są niewielkie, potwierdzając zastosowalność zaproponowanych metod.

W ostatniej części rozprawy zaprezentowano system zaimplementowany w celu wsparcia zbierania danych i monitorowania stanu pacjentów. System składa się z aplikacji mobilnej dla pacjenta oraz aplikacji internetowej dla klinicystów. Aplikacja mobilna pozwala pacjentom na rejestrowanie objawów, przyjętych dawek leków oraz innych istotnych informacji w czasie rzeczywistym. Dane te następnie są synchronizowane z aplikacją internetową, gdzie klinicyści mogą monitorować postęp pacjentów oraz podejmować decyzje co do terapii. Integracja tych dwóch narzędzi ułatwia zbieranie danych w czasie rzeczywistym oraz ciągłe monitorowanie stanu pacjentów, co pozwala na szybkie reagowanie na zmiany w stanie pacjenta. Dzięki przyjaznemu interfejsowi użytkownika dla pacjentów i klinicystów, system ten wspiera ciągłą opiekę nad pacjentem i umożliwia rozwój zaawansowanych, zindywidualizowanych strategii leczenia, które dostosowane są do indywidualnych potrzeb pacjentów.

Wyniki badań przedstawione w pracy pokazują możliwości istotnej poprawy w zarządzaniu objawami oraz jakości życia pacjentów poprzez większą indywidualizację leczenia. Przyszłe badania powinny uwzględnić większą liczbę

pacjentów oraz badań, zbadać wpływ innych czynników na stan pacjenta oraz aktualizację zadań optymalizacyjnych, aby zwiększyć dokładność oraz zastosowalność metody.

Podsumowując, praca ta prezentuje kompleksowe podejście do indywidualizacji leczenia PD, poprzez wykorzystanie modelu uczenia maszynowego i optymalizacji, oferując obiecujący kierunek rozwoju przyszłych terapii w PD.