Motor Function Assessment for Parkinson's Patients Using Radar Sensor and Machine Learning – Finger Tapping

Cecilia Nordberg
Dimitrios Pediaditis
Manousos Manouras
Wajih Habrah

CHALMERS

Course: SSY267 - Advanced Topics in Biomedical Engineering
Supervisor: Xuezhi Zeng
Date: November 1, 2024

Contents

1 Introduction

Parkinson's disease (PD) is a progressive neurological condition that significantly impairs motor skills, balance and coordination due to uncontrollable movements such as shaking and rigidity. Another characteristic symptom is bradykinesia, which refers to slowness of movement. Beyond motor symptoms, PD can cause a variety of non-motor problems, such as sensory abnormalities, cognitive impairment, and mental health difficulties [1]. Collectively, these symptoms have a significant impact on the patient's quality of life.

PD is the second most common neurodegenerative disorder and the fastest-growing neurological condition in terms of incidence, disability, and mortality. It affects 1-2% of people over the age of 65. With an aging population, he prevalence of this condition is quickly increasing, presenting serious challenges for healthcare systems and society [2]. Currently, no neuroprotective treatments exist to stop or slow down disease progression, which emphasize the need of early detection to allow for personal disease management, especially through motor function assessment. Some medications are available for symptom management, but no cure for PD exists[3].

Motor function assessment are essential for evaluating a patient's ability to perform specific movements. These tests are extremely useful for physiotherapists in identifying disorders, tracking disease progression, developing rehabilitation programs, and evaluating treatment outcomes. Key characteristics of PD, including bradykinesia, are central to these evaluations. Among various tests for evaluating bradykinesia, the finger-tapping test specifically assesses a patient's ability to tap their index finger and thumb together continuously for 10 seconds without interruption. This test aids the physiotherapists in scoring the severity of PD by assessing the patient's motor function.[4].

This project builds on a prior student initiative that used a biomedical radar system to assess the finger tapping test, using traditional signal processing methods. The goal is to improve the accuracy of the method by utilizing machine learning. This study will compare different machine learning algorithms to determine the most effective solution to the challenge at hand. Rather than making diagnoses, the purpose is to help physiotherapists and doctors by providing quantitative data on crucial parameters like frequency of interruptions, amplitude fluctuations, and overall performance throughout the finger-tapping test.

1.1 State-of-the-art

The current gold standard for assessing motor function in PD patients, such as the finger-tapping test, relies on observations by physiotherapists. However, since this approach is subjective and requires high expertise, there is significant room for improvement with the help of new technologies.

One solution, proposed by S. F. Desyansah et al. (2021) [5], is a detection system for bradykinesia by measuring changes in movement while the patients remain still. The model was based on signals from an accelerometer embedded in a bracelet to detect wrist movements. Statistical analysis was then used to interpret the results. A similar approach, collecting movement data through a wearable sensor is described in the study by Lina Tong et al. (2022) [6]. The researchers created a wearable device with sensors placed on the wrist and ankle, which wirelessly collects movement data from Parkinson's patients. In contrast to the statistical analysis done by previous researchers, they used a deep learning model to analyze this data and detect bradykinesia, achieving an accuracy of 98.6%. Ghosh et al. (2023) [7] extended this by placing sensors on the neck, hand, and leg, for detecting bradykinesia using deep learning.

Wearable sensors for motor assessment propose a potential solution to the subjective observations made by physiotherapists. However, a notable limitation is that the recorded signal greatly depends on the sensor placement, as symptoms from PD can be asymmetric [8]. Although more sensors could improve accuracy in characterizing the severity of bradykinesia, increasing the number of sensors can make the system more uncomfortable for patients. A camera-based approach was presented by Abin Ghosh et al.[9]. The researchers used image analysis to measure finger velocity, angles, and amplitude via deep learning. The method had many successes and gave an accurate assessment of the patient's status, but patient discomfort with being filmed limited its clinical applicability.

Biomedical radar is a contactless, non-invasive method that emits electromagnetic waves to monitor health, making it ideal for remote or in-home healthcare settings. This approach shows promise for assessing motor function in PD, particularly in evaluating bradykinesia through the finger-tapping test. Compared to current methods, biomedical radar offers continuous, contact-free monitoring, enhancing patient comfort and providing physiotherapists with a more objective tool for assessment.

1.2 Aim

This project aims to identify a machine learning-based system for assessing and detecting patterns in finger-tapping data, simulating movement patterns of PD patients. Using radar sensors for data collection, the long-term objective is to provide physiotherapists with reliable, quantitative metrics on motor function changes, such as tapping frequency and interruptions. Unlike traditional observational methods, this approach seeks to deliver an objective assessment tool that can support early detection and routine monitoring of PD symptoms.

1.3 Limitations

This project is limited to analyzing simulated finger-tapping data with specific patterns, including few interruptions or freezing, and aim to identify machine learning models that can classify these simpler patterns. While additional data has been recorded to explore more detailed changes, such as frequency and amplitude decrements or combination of patterns, time constraints limit the project's scope to classification of simpler patterns.

The data used for training and testing is simulated and collected by group members based on instructional videos and guidelines from prior physiotherapist-supervised data collection. Evaluation of the model performance on realistic data is limited to a small amount of available physiotherapist data, and no actual patient data is included in the project.

Lastly, the project is limited to evaluating the performance of simple machine learning models types and architectures. This limitation is due to both the limited time frame for the project, access to GPU resources, and the small dataset available, which would not support the complexity of more advanced models.

2 Methods

The following sections provide an outline of the used methods in this project, including an introduction to the radar system used for data collection, processes for data annotation and pre-processing, as well as a description of the machine learning models used to classify the resulting radar signatures from the finger-tapping tests.

2.1 Radar system

In this project, a biomedical radar system is used to capture and record finger-tapping sequences. The system operates by emitting electromagnetic waves toward the subject and analyzing the reflected signals to detect movement. The radar sensor used is a Frequency Modulated Continuous Wave (FMCW) radar manufactured by Texas Instruments.

The radar transmits sinusoidal signals with increasing frequency, called chirps, which reflect back to the radar and get captured by a receiver antenna. The reflected signals are combined with the transmitted signals to generate an intermediate frequency signal, which is processed to extract information about the distance and velocity of the moving object, in this case, the finger-tapping movements. The primary technique used for this signal processing is the Fourier Transform. Specifically, the Fast Fourier Transform (FFT) is used to determine the distance by extracting frequency components, while the Short Time Fourier Transform (STFT) is used to analyze frequency changes over time. [10]

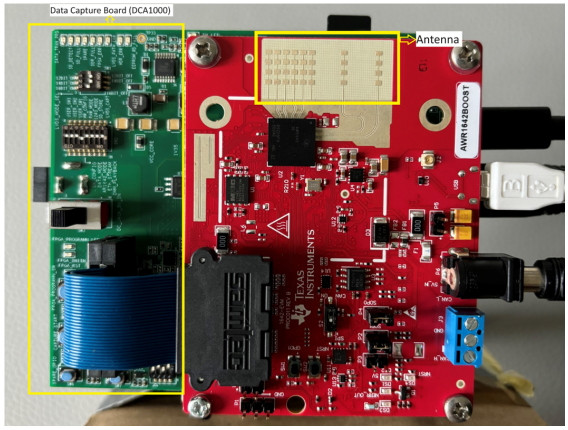

Figure 1: FMCW radar sensor for capturing finger-tapping sequences, with positions of the antenna and the data capture board (DCA1000) highlighted in yellow [4].

The radar system is composed of an integrated circuit and a DCA1000 evaluation board for real-time data capture, as shown in Figure 1. Signals are sampled using analog-to-digital converters and transmitted to a computer for post-processing and data visualization. Communication with the radar sensor is managed by Texas Instruments' mmWave Studio, a graphical user interface that allows configuration of the various parameters. The recommended parameter values for sensor configuration were provided by the supervisor, based on findings from the previous master's thesis. These parameters are detailed in Appendix A.

2.2 Data collection

A large part of the project involved conducting and gathering data from various finger-tapping tests, in order to create a sufficiently large dataset to evaluate machine learning models. Once the radar sensor was configured, the data collection was done through multiple series of finger-tapping tests. For such a test, the radar was positioned on a table next to the seated participant, with their hand placed 10 - 15 cm from the radar. The participant then performed repeated finger-tapping movements within the radar's transmission field, allowing the system to capture detailed motion data over a 10-second sequence.

Data was collected across multiple scenarios, designed to capture a range of finger-tapping behaviors. These different patterns or behaviors can roughly be classified into the following cases:

- Case 1: Normal conditions, with no interruptions or slowing of movement.
- Case 2: Test performed with interruptions during finger-tapping.
- Case 3: Test performed with "freezing" (interruptions longer than 1 second).

Typical finger-tapping patterns in PD patients often exhibit reduced movement frequency, indicating slowed movement, as well as decrements in amplitude. However, this project is limited to focus exclusively on three specific behaviors above. In practice, real-world radar patterns from patients with PD will likely involve combinations of these cases, introducing greater complexity in the received signal. These complex combinations of movement patterns are beyond the current scope of the project, and the main focus is on isolating more simple behaviors to train and test machine learning models effectively.

In total, 530 data samples were collected, covering an equal distribution of different finger-tapping scenarios. The collected data from each test were loaded in Matlab and saved in a '.mat' file containing the test number and test label, for further processing using Python. The labeling of the data is described in following subsections.

2.2.1 Additional test data from physiotherapists

In addition to self-collected data, simulating different finger-tapping behaviors, a total of 10 data samples for these scenarios could be isolated from available physiotherapist data.

Collected data and physiotherapist data for additional testing is publicly available on Kaggle:

`https://www.kaggle.com/datasets/manousos/all-matrices-new`

2.3 Machine learning

2.3.1 Data pre-processing and augmentation

The data matrices from each test are loaded in a Python notebook as a list of matrices. The list contains elements equal to the number of tests performed and each element has a size of (1024,1247) where the columns represent the time axis or frames and the rows represent the velocity values. Additionally, for ease of use, a dictionary containing the test number as key and the test label as value was created.

Data augmentation is a widely used technique in machine learning to pre-process datasets, especially beneficial when working with limited data. Augmentation of data was performed after visualizing the tests and omitting the ones deemed unclear or unable to be classified. Firstly, to account for late starts in some of the performed tests, the first 47 columns were removed from each data matrix making the final dimensions (1024,1200). Moreover, horizontal, vertical, and combined flipping were performed, producing four variations of each test sample. Figure 2 illustrates a comparison between an original radar signature and its augmented version with both horizontal and vertical flipping applied. After some adjustments in the data labels, the data were split into train, validation and test sets with 70% reserved for training and for validation and test sets 15% for each.

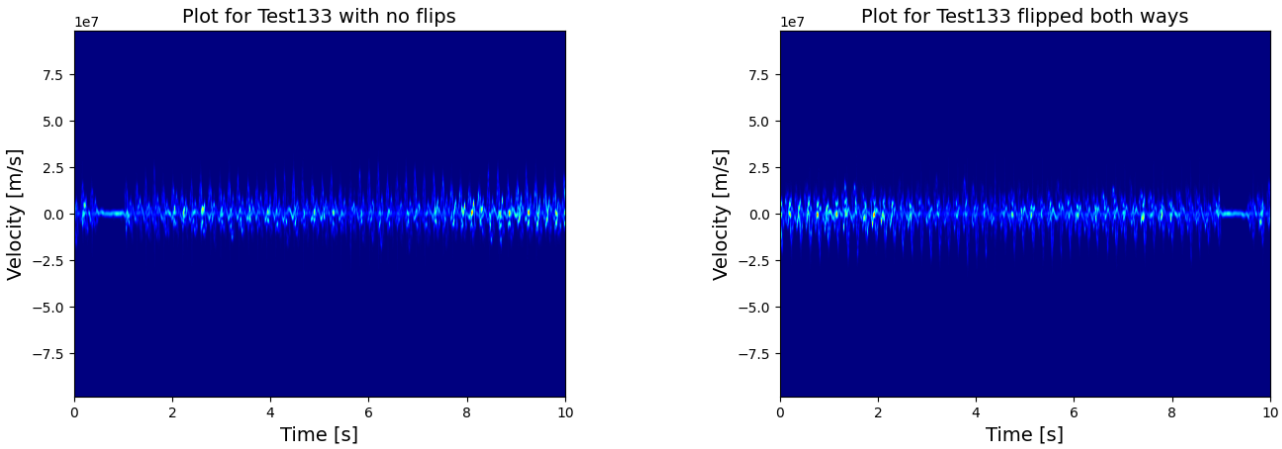

Figure 2: Original radar signatures(left) and augmented radar signatures(right) with both horizontal and vertical flipping.

2.3.2 Data labeling

The labeling of the data was performed during each finger-tapping using the file name as an indication of the label. The tests were split into 6 different classes:

- 0: Normal
- 1: 1 interruption
- 2: 2 interruptions
- 3: 3 interruptions
- 4: 4 interruptions
- 5: Contains freezing (>1 sec interruptions)

These categories aim to identify the number of interruptions during each finger-tapping test. A total of 530 data samples were collected, and evenly distributed across the analyzed classes. Following data augmentation, the total number of samples increased to 2120. The distribution of the augmented data is illustrated in Figure 3.

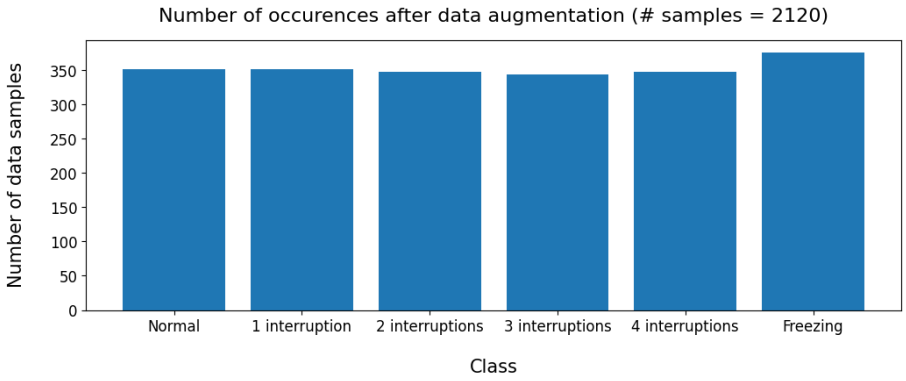

Figure 3: Distribution of 530 data samples, resulting in a total of 2120 samples after augmentations, showing number of samples for each class.

2.3.3 Model descriptions

The problem at hand is relatively straightforward, as the number of interruptions can be identified by examining patterns in the time-velocity plots of the data. Additionally, the dataset is self-collected and relatively small compared to those typically used in machine learning applications. Given the limited size of the dataset and the simplicity of the patterns involved, complex and deep machine-learning algorithms are unsuitable for this problem. Such algorithms typically require more data and are designed to detect more intricate patterns than the ones that are presented here.

Given these constraints, many different models were implemented and tested using an iterative approach. For simplistic sequence problem, convolutional neural networks (CNNs) generally perform well. Additionally, because this is a time-series problem, various types of recurrent neural networks (RNNs) were also evaluated. The models that demonstrated the best performance are summarized below:

- Random Forest classifier
- One-dimensional CNN with 1 to 3 layers (1D-CNN)
- Long Short-Term Memory (LSTM) combined with 1D-CNN (CNN-LSTM)

Random Forest classifier: A 'random forest classifier' is a machine-learning algorithm that creates multiple decision trees with random subsets of the dataset features and uses each tree's prediction to find the best classification for the whole dataset. It is impervious to overfitting (learning noise and irrelevant details that hurt generalization) and effectively handles noise and complex relationships between features.

1D-CNN: One-dimensional CNNs apply filters to sequential data, such as time series, to detect patterns like anomalies or interruptions. They capture local relationships within the data efficiently, using relatively few parameters.

CNN-LSTM: The LSTM layer captures the data's temporal dependencies (usually long-term)[11] and, combined with CNN, which captures spatial dependencies, creates an effective model for complex sequence analysis tasks.

Each model contains a set of hyperparameters that were fine-tuned to optimize their performance. This was performed mainly through trial and error due to the lack of available computational resources that could speed up the training process. For example, some of the hyperparameters of the 1D-CNN model are the filter size, the learning rate (step size) of the optimizer and the dropout rate. For the random forest classifier, parameters that need optimization include the number of trees (estimators) and the maximum depth of each tree.

2.3.4 Evaluation of model performance

The performance of the models was evaluated using cross-entropy loss and accuracy metrics for both training and validation datasets. These metrics were plotted over each epoch to show a comparison of model performance over time, allowing us to identify potential overfitting or underfitting. Following this, the models were assessed on both the test set and test data from physiotherapists. To visualize the predictions, a confusion matrix was used. The diagonal of the matrix illustrates the number of correct predictions for each class while the rest of the cells showcase the number of incorrect predictions. This provides an indication of the overall performance as well as points out if the model fails to differentiate specific classes.

3 Results

This section presents the main results, covering data analysis and performance of machine learning algorithms applied for classification of obtained radar data. This includes an overview of collected data and resulting radar signatures for each class, and a comparison of model performance for different machine learning architectures.

The code is publicly available on GitHub: `https://github.com/ManourasM/finger_tapping_with_ML`

3.1 Radar signatures for different classes

Figure 4 shows obtained radar signatures for the analyzed classes, including normal motor performance, various number of interruptions, and "freezing" (> 1 sec interruptions).

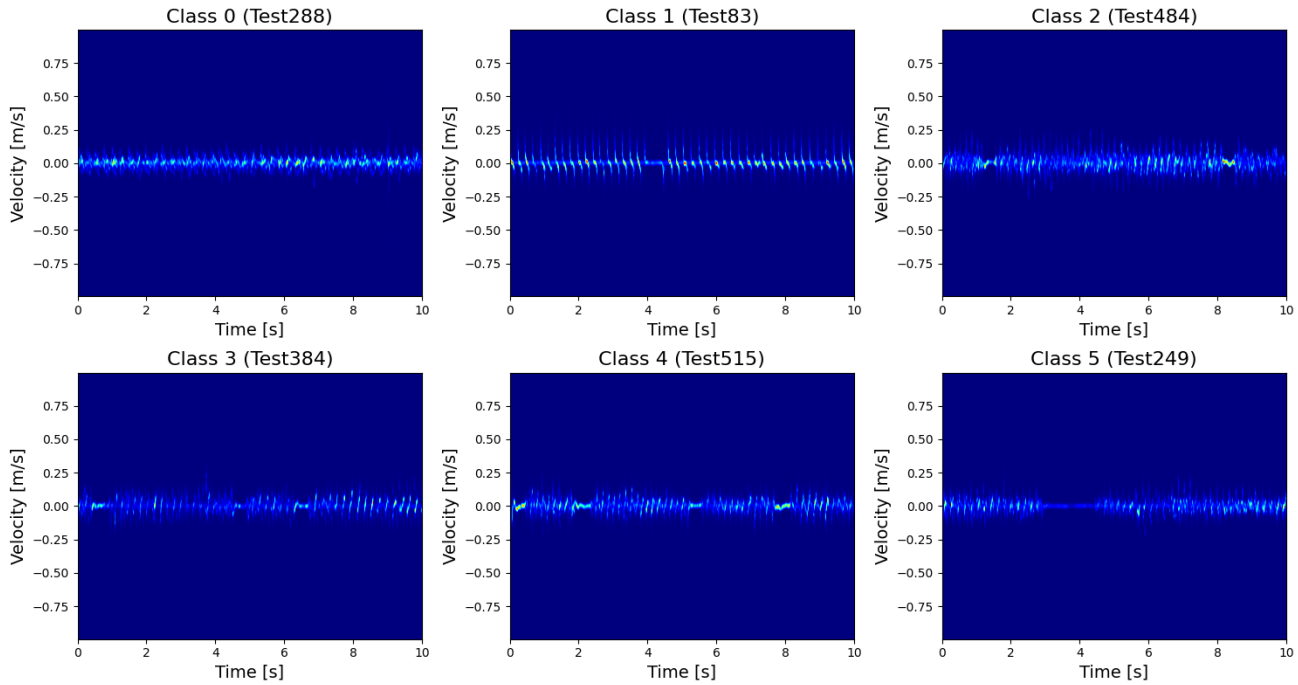

Figure 4: Radar signatures for different finger-tapping tests with class 0: normal motor performance, 1: one interruption, 2: two interruptions, 3: tree interruptions, 4: four interruptions, and 5: "freezing".

3.2 Classification of radar signatures

The training efficiency and performance of the models varied greatly depending on the network architecture, and a lot of hyper-parameter tuning and iterative testing of different architectures was needed. The process of finding a suitable architecture, including parameters such as model depth and number of features, involved a trial-and-error approach while continuously analyzing performance metrics. These metrics served as a basis for determining if the models show potential for this type of classification problem, and if further optimization of the network architecture was necessary. Table 1 presents the training and evaluation metrics for the models that achieved the best performance on our dataset, including their performance on the physiotherapist data.

Table 1: Training and evaluation metrics for the best-performing machine learning models

	Train		**Validation**		**Test**	**Physio test data**
Model	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Accuracy**	**Accuracy**
1D-CNN (1 layer)	0.0750	97.00%	0.8500	92.00%	88.61%	10.0%
1D-CNN (2 layers)	0.1181	96.49%	0.2663	92.45%	96.23%	30.0%
1D-CNN (3 layers)	**0.1400**	**95.00%**	**0.3600**	**92.00%**	**97.47%**	**60.0%**
1D CNN-LSTM	0.0955	96.02%	0.1126	97.17%	93.40%	40.0%
Random Forest	-	100.00%	-	-	82.00%	30.0%

Bold indicates best performing model

The results suggest that the best performing model is a 1D-CNN model with three layers, receiving a test accuracy of 97.47% and an accuracy of 60% on the physiotherapist test data. The performance of this model is further illustrated in Figure 5, showing the progression of the evaluation metrics during training. As can be seen in the figure, the validation loss is considerably higher than the training loss, suggesting that the model overfits to the training data. Despite this, the model performed best on the physiotherapist data and show some potential in classifying these more realistic cases. The confusion matrix, in subfigure 5b, indicate that the model can effectively differentiate both normal and freezing conditions from cases containing only interruptions and that it

is more challenging to detect the precise number of interruptions. The detailed architecture of model is presented in Appendix B.

(a) Loss and accuracy during training and validation.

(b) Confusion matrix for the test set.

Figure 5: Model performance metrics for the best-performing model - 1D CNN (3 layers).

In terms of training efficiency, the 1D CNN models outperformed all other approaches, reaching convergence faster and requiring less time per epoch. For instance, each 1D CNN model converged in under 100 epochs, whereas the 1D CNN-LSTM model, which also had a longer training time per epoch, required at least 300 epochs to achieve comparable performance. Note that the exact training times cannot directly be compared between the individual models, as the models were trained on GPUs with different capacities.

In addition to the presented models, both 2D CNN networks and pure LSTM models were also explored, but without any promising results. The large size of the radar matrices (1024, 1200) led to memory and computational limitations that made training of 2D CNN models infeasible. For LSTM models, despite testing multiple architectures, we were unable to achieve a loss below 1.0 or accuracy above 30%, indicating that these models struggled to capture features in the radar data.

4 Discussion

4.1 Motor function assessment using machine learning

After experimenting with multiple models it was realized that deeper more complex models failed to converge, this is due to the simplicity and limited size of the data where deeper models are unable to extract important features, in comparison with simpler models where they managed to converge and reach high accuracy in the unseen test data. The 1D-CNN with a 3-layer architecture outperformed other models, obtaining 97% accuracy on test data and 60% accuracy on physiotherapist data (see Table 1). In CNN models, adding convolutional, batch normalization, and max-pooling layers increases complexity and allows for more effective feature extraction. Among the models evaluated the 3-layer 1D-CNN generated the most accurate feature maps, resulting in the best performance. Although other models performed similarly on test data, they were unable to grasp the complexity of the physiotherapists' data, resulting in poor results.

To avoid poor generalization in deep machine learning, diverse training and testing data are required. Training exclusively on self-collected data might result in considerable accuracy gaps, as can be seen from the test and physiotherapist data. For medical applications, precision is essential to patient outcomes. As a result, gathering data from field specialists or actual patients is critical for creating a highly accurate deep-learning model.

Previous thesis work, which presented a model based on conventional signal processing techniques rather than machine learning, achieved accuracies of 93.18%, as well as 60.71% on validation data collected by physiotherapists [4]. While these results are similar to the ones presented in this project, a direct comparison is challenging since their approach classified severity rather than quantifying parameters such as the number of interruptions. Additionally, their study included an analysis of frequency and amplitude decrements, which was not included in the scope of this project. Generally, machine learning algorithms are powerful in handling and extracting features from large datasets, which is also reflected in our results. While the machine learning models in this

project managed to achieve comparable accuracy on a larger dataset, they do not show a significant advantage in comparison to the conventional methods. The main reasons for this, are challenges in data collection, especially the difficulty of realistically simulating movement patterns of PD patients without previous experience or input from physiotherapists, as well as the need for large amounts of data to develop a stronger model. Nonetheless, machine learning algorithms show strong potential for assessing finger-tapping patterns, but further improvements and testing with different techniques are needed to reach the accuracy and reliability needed for clinical use.

4.2 Limitations and future perspectives

As mentioned previously, data collection is crucial for ensuring the models perform well. To achieve this, more diverse data are needed either from physiotherapists or real patients, especially for the models to be clinically relevant. Currently, the models are only tested on simulated data from physiotherapists or self-collected data, with no assessments conducted on actual patient data. Therefore, the true performance of the model remains unknown.

For future work, exploring other architectures could be beneficial if more computational resources are available. Specifically, 2D-CNNs or RNN-based models are useful for sequential problems of this nature. For time-series problems, RNNs are often preferred over convolutional networks because they process the data step by step. This makes them more effective to capture long-range dependencies in the time-dependent sensor readings.

The model is designed to identify the number of interruptions, if there is a freezing, or if it is normal. Another implementation that could assist the doctors and the physiotherapists could be to correctly identify decrements in amplitude or frequency that are crucial for a better diagnosis in finger-tapping tests. One suitable method could be an unsupervised clustering algorithm which can help with pattern recognition[12].

5 Conclusion

The finger-tapping test is a motor function assessment used by physiotherapists to diagnose Parkinson's disease and evaluate its severity. In this project, recording of the finger-tapping test is conducted using a radar sensor, with a machine learning algorithm employed to detect collected movements, such as interruptions and freezing. By combining biomedical radar technology with machine learning, the method can provide physiotherapists with reliable, quantitative metrics for monitoring changes in motor function. This would offer an objective assessment tool to support early detection and assessment of bradykinesia.

The machine learning models developed in this project show a strong potential, specifically the model based on a 3-layer 1D-CNN, which achieved the highest performance on both self-collected data and data previously collected by physiotherapists. This suggests that machine learning can be an effective approach for detecting movement patterns. However, data collection presents significant challenges, especially in simulating the realistic behavior of PD patients. Future developments may include collecting data from physicians or real patients to improve model generalization, potentially leading to even more accurate outcomes.

Finally, all radar data collection and model training followed a very specific protocol: the hand of the test subject was positioned 10-15 cm away from the sensor. This configuration is required to ensure accurate model predictions and should be carefully maintained throughout the device setup for achieving consistent results.

References

[1] Parkinson's Disease: Causes, Symptoms, and Treatments; 2022. Available from: https://www.nia.nih.gov/health/parkinsons-disease/parkinsons-disease-causes-symptoms-and-treatments.

[2] Brakedal B, Toker L, Haugarvoll K, Tzoulis C. A nationwide study of the incidence, prevalence and mortality of Parkinson's disease in the Norwegian population. npj Parkinson's Disease. 2022 Mar;8(1):1-8. Available from: https://www.nature.com/articles/s41531-022-00280-4.

[3] What is Parkinson's? | Parkinson's Foundation;. Available from: https://www.parkinson.org/understanding-parkinsons/what-is-parkinsons.

[4] Isaksson Gustav MS. Motor Functions Assessment for Parkinson Disease via Radar Sensors Focusing on Finger Tapping Test. Chalmers University of Technology; 2024.

[5] Desyansah SF, Mohammed MN, Al-Zubaidi S, Syamsudin H, Abdullah I, Yusuf E. Bradykinesia Detection System Using IoT Based Health Care System for Parkinson's Disease Patient. In: 2021 IEEE International Conference on Automatic Control & Intelligent Systems (I2CACIS). Shah Alam, Malaysia: IEEE; 2021. p. 329-33. Available from: https://ieeexplore.ieee.org/document/9495885/.

[6] Tong L, Liu D, Zhang M, Peng L. A Parkinson's Bradykinesia Recognition System Based on Deep Learning Method. In: 2022 12th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER). Baishan, China: IEEE; 2022. p. 1005-8. Available from: https://ieeexplore.ieee.org/document/9907204/.

[7] Ghosh A, Anand N, Kaushik P, Bagrodia V, Pal PK, Yadav R. IOT based Sensor System for 24×7 monitoring movement disorder symptoms using machine learning. In: 2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS). Bangalore, India: IEEE; 2023. p. 66-71. Available from: https://ieeexplore.ieee.org/document/10041330/.

[8] Toth C, Rajput M, Rajput AH. Anomalies of asymmetry of clinical signs in parkinsonism. Movement Disorders. 2004 2;19:151-7.

[9] Islam MS, Rahman W, Abdelkader A, Lee S, Yang PT, Purks JL, et al. Using AI to measure Parkinson's disease severity at home. NPJ digital medicine. 2023 Aug;6(1):156.

[10] Introduction to mmwave Sensing: FMCW Radars;. https://www.ti.com/content/dam/videos/external-videos/zh-tw/2/3816841626001/5415203482001.mp4/subassets/mmwaveSensing-FMCW-offlineviewing_0.pdf.

[11] Jiang H, Fioranelli F, Yang S, Romain O, Kernec JL. Human Activity Classification Using Radar Signal and RNN Networks; 2020.

[12] Sinaga KP, Yang MS. Unsupervised K-Means Clustering Algorithm. IEEE Access. 2020;8:80716-27.

Appendix A Sensor settings

The specifications of the radar sensor configuration, set in the Graphical User Interface (GUI) of mmWave Studio provided by Texas Instruments, are presented in Figure 6 and in the Matlab code snippet below.

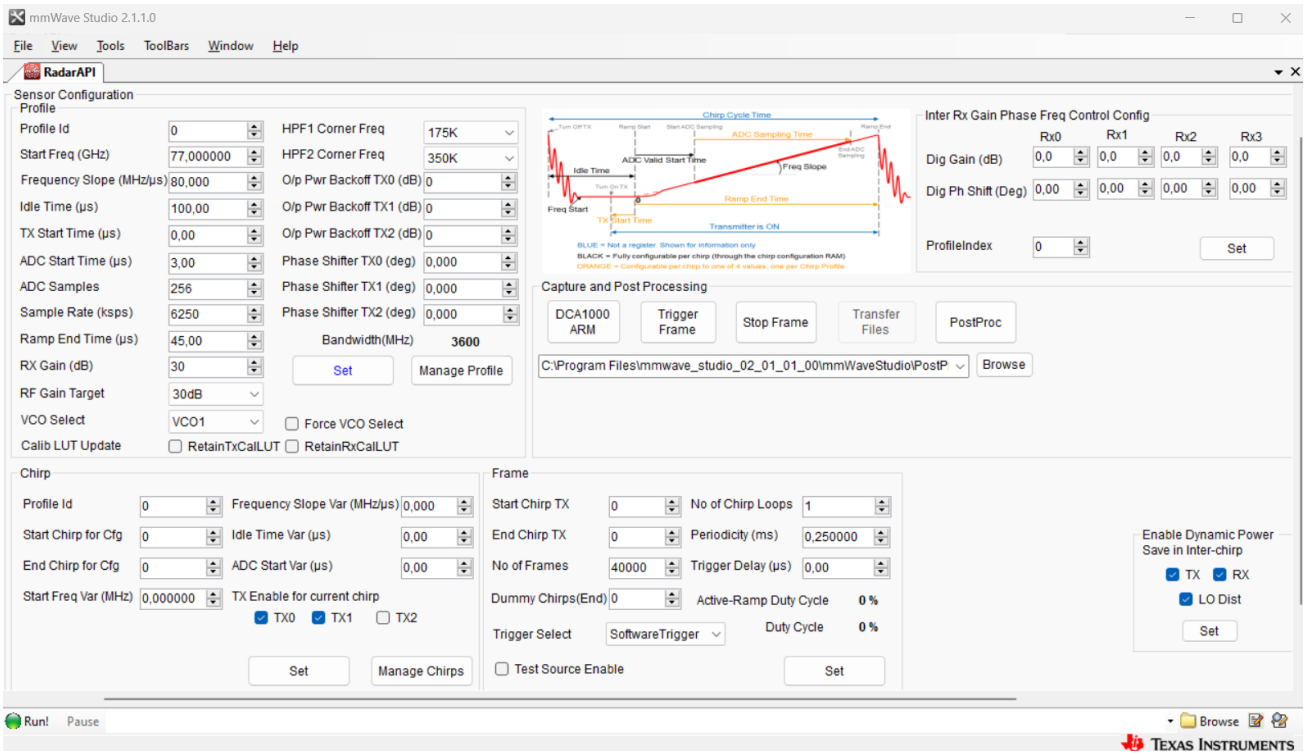

Figure 6: Specifications of the radar sensor configuration set in the mmWave Studio GUI.

```
% Sensor setting parameters
N_frame=40000;           % Number of frames used
N_sample=256;            % Number of samples per frame
periodicity = 0.00025;   % frame-to-frame interval

S=80*10^12;              % Frequency slope [Hz/s]
Fs=6250*10^3;            % ADC Sampling rate [sps]
c=299792458;             % Speed of light [m/s]

f_start = 77*10^9;                % Start frequency of chirp [Hz]
t_window = (N_sample-1)*(1/Fs); % The length of the fast time window
B=S*t_window;                     % Bandwidth
delta_f = 1/t_window;             % Frequency interval of sampled signal
f_range = 0:delta_f:(N_sample-1)*delta_f;   % Frequency axis of the fast
                                            % time signal
d_range = (f_range*c)/(S*2);      % Distance calculated from frequency [m]
lambda = c/f_start;               % Wavelength [m]
```

Appendix B Model architecture of 1D-CNN (3 layer)

The best-performing model for detecting interruptions and freezing in radar data from finger-tapping tests was a 1D CNN with three layers. The architecture of the model can be seen in Figure 7 below.

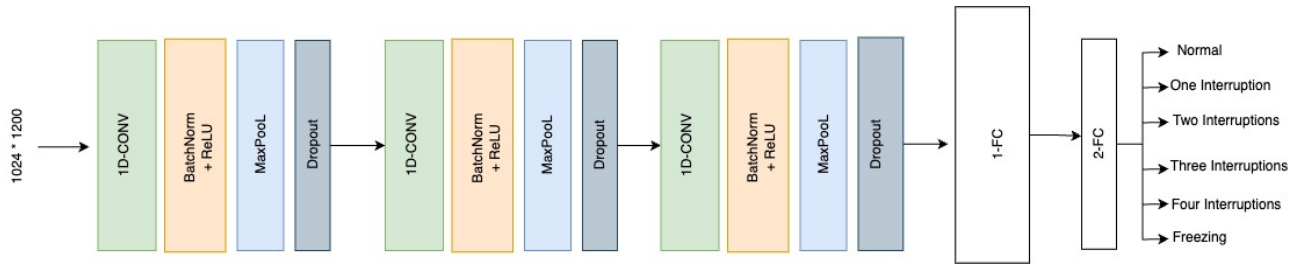

Figure 7: Architecture of the 1D-CNN (3 layer)

1D-CNN (3 layer) - Model class

```python
class CONV1D_3(nn.Module):
    def __init__(self):
        super(CONV1D_3, self).__init__()

        # First convolutional layer
        self.conv1 = nn.Conv1d(in_channels=1024, out_channels=32, kernel_size=3, padding=1)
        self.bn1 = nn.BatchNorm1d(32)

        # Second convolutional layer
        self.conv2 = nn.Conv1d(in_channels=32, out_channels=64, kernel_size=3, padding=1)
        self.bn2 = nn.BatchNorm1d(64)

        # Third convolutional layer
        self.conv3 = nn.Conv1d(in_channels=64, out_channels=128, kernel_size=3, padding=1)
        self.bn3 = nn.BatchNorm1d(128)

        # Pooling and dropout layers
        self.pool = nn.MaxPool1d(kernel_size=2, stride=2)
        self.dropout = nn.Dropout(0.5)

        # Fully connected layers
        self.fc1 = nn.Linear(128 * 150, 256)  # Updated to 9600 (64 * 150)
        self.fc2 = nn.Linear(256, 6)  # 6 classes

    def forward(self, x):
        # First convolution block
        x = F.relu(self.bn1(self.conv1(x)))
        x = self.pool(x)
        x = self.dropout(x)

        # Second convolution block
        x = F.relu(self.bn2(self.conv2(x)))
        x = self.pool(x)
        x = self.dropout(x)

        # Third convolution block
        x = F.relu(self.bn3(self.conv3(x)))
        x = self.pool(x)
        x = self.dropout(x)

        # Flatten the output for the fully connected layers
        x = x.view(x.size(0), -1)

        # Fully connected layers

        x = F.relu(self.fc1(x))
        x = self.fc2(x)

        return x
```

Summary of the model with total number of parameters

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv1d-1             [-1, 32, 1200]          98,336
       BatchNorm1d-2             [-1, 32, 1200]              64
         MaxPool1d-3              [-1, 32, 600]               0
           Dropout-4              [-1, 32, 600]               0
            Conv1d-5              [-1, 64, 600]           6,208
       BatchNorm1d-6              [-1, 64, 600]             128
         MaxPool1d-7              [-1, 64, 300]               0
           Dropout-8              [-1, 64, 300]               0
            Conv1d-9             [-1, 128, 300]          24,704
      BatchNorm1d-10             [-1, 128, 300]             256
        MaxPool1d-11             [-1, 128, 150]               0
          Dropout-12             [-1, 128, 150]               0
           Linear-13                  [-1, 256]       4,915,456
           Linear-14                    [-1, 6]           1,542
================================================================
Total params: 5,046,694
Trainable params: 5,046,694
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 4.69
Forward/backward pass size (MB): 2.64
Params size (MB): 19.25
Estimated Total Size (MB): 26.58
----------------------------------------------------------------
```

Appendix C Contribution report

The first weeks, we did the data collection together to get familiarized with both sensor and software, as well as to understand the different finger-tapping patterns being studied. After this, Wajih was main responsible for collecting enough data for the different classes. Together, we researched the the provided parameters and reviewed the provided code and articles to deepen our understanding of the problem. Every week, we had a supervisor meeting with Xuezhi, as well as two additional meetings together with the group, to discuss progress and planning of the project.

Manousos and Dimitrios were responsible for developing a general code framework for data processing and training the machine learning models. Following this, Manousos, Dimitrios and Cecilia contributed to the final code and to find suitable models and fine-tune parameters of the models to get good results. The training of the models were done individually to utilize the computational resources on everyone's computer and parallelize the development of models.

Table 2 provides an overview of contributions to the different sections of the final report, where the main author(s) were responsible for drafting the original text. All sections were then revised multiple times by all group members to ensure a cohesive and accurate text, as well as a flow throughout the report.

Table 2: Contribution to final report

Section	***Headline***	***Main author(s)***
1	*Introduction*	*Dimitrios*
1.1	*State of art*	*Cecilia, Wajih*
1.2	*Aim*	*Cecilia*
1.3	*Limitations*	*Cecilia*
2	*Methods*	*Cecilia*
2.1	*Radar system*	*Cecilia*
2.2	*Data collection*	*Cecilia, Wajih*
2.3	*Machine learning*	*Manousos*
2.3.1	*Data pre-processing and augmentation*	*Manousos*
2.3.2	*Data labelling*	*Manousos, Cecilia*
2.3.3	*Model descriptions*	*Manousos*
2.3.4	*Evaluation of model performance*	*Cecilia, Dimitrios*
3	*Results*	*Cecilia*
3.1	*Data*	*Cecilia*
3.2	*Radar signatures for different classes*	*Cecilia*
3.3	*Classification of radar signatures*	*Cecilia*
4	*Discussion*	*Dimitrios*
4.1	*Motor function assessment using machine learning*	*Dimitrios, Cecilia*
4.2	*Limitations and future perspectives*	*Dimitrios*
5	*Conclusion*	*Dimitrios*

A documentation of the work and each individual's contributions and time spent throughout the project has been recorded in a shared time log, see Appendix D.

Appendix D Time log

		Time spent per activity / study week									Total spent hours
Person	**Activity**	**SW1**	**SW2**	**SW3**	**SW4**	**SW5**	**SW6**	**SW7**	**SW8**	**SW9**	
Cecilia	Lecture / presentation session	2,0			2,0					4,0	8,0
	Meeting with supervisor		1,0	1,5	1,0	1,0	1,0	1,0	1,0		7,5
	Group meeting / project planning		4,0	4,0	4,0	4,0		4,0	4,0	4,0	28,0
	Report writing			2,5	4,0			1,0	11,0	10,0	28,5
	Research	2,0	4,0	2,5	2,0	1,0	1,0		3,0		15,5
	Data collection / processing		3,5	3,0	3,0	3,5	0,5				13,5
	Model development / evaluation					3,0	7,0	8,0	7,0		25,0
	Other			2,0						8,0	10,0
		4,0	12,5	15,5	16,0	12,5	9,5	14,0	26,0	26,0	**136,0**

		Time spent per activity / study week									Total spent hours
Person	**Activity**	**SW1**	**SW2**	**SW3**	**SW4**	**SW5**	**SW6**	**SW7**	**SW8**	**SW9**	
Dimitrios	Lecture / presentation session	2,0			2,0					4,0	8,0
	Meeting with supervisor		1,0	1,5	1,0	1,0	1,0	1,0			6,5
	Group meeting / project planning		4,0	4,0	4,0	4,0		4,0	4,0	4,0	28,0
	Report writing			3,0	4,0			3,0	10,0	10,0	30,0
	Research	2,5	4,0	2,0	2,0	1,5	1,0		2,0		15,0
	Data collection / processing		5,0	5,0	4,0	3,5					17,5
	Model development / evaluation				3,0	5,0	7,0	6,0	6,0		27,0
	Other									4,0	4,0
		4,5	14,0	15,5	20,0	15,0	9,0	14,0	22,0	22,0	**136,0**

		Time spent per activity / study week									Total spent hours
Person	**Activity**	**SW1**	**SW2**	**SW3**	**SW4**	**SW5**	**SW6**	**SW7**	**SW8**	**SW9**	
Manousos	Lecture / presentation session	2,0			2,0					4,0	8,0
	Meeting with supervisor		1,0	1,5	1,0	1,0	1,0	1,0	1,0		7,5
	Group meeting / project planning		4,0	4,0	4,0	4,0		4,0	4,0	4,0	28,0
	Report writing			2,0	3,5			2,0	8,0	8,0	23,5
	Research	3,0	4,0	3,0		2,0	3,0				15,0
	Data collection / processing		4,0	4,0	4,0	3,5	2,0	1,0	1,0		19,5
	Model development / evaluation				5,0	5,0	7,0	6,0	5,0		28,0
	Other	0,5	0,5	0,5						1,0	2,5
		5,5	13,5	15,0	19,5	15,5	13,0	14,0	19,0	17,0	**132,0**

		Time spent per activity / study week									Total spent hours
Person	**Activity**	**SW1**	**SW2**	**SW3**	**SW4**	**SW5**	**SW6**	**SW7**	**SW8**	**SW9**	
Wajih	Lecture / presentation session	2,0			2,0					4,0	8,0
	Meeting with supervisor		1,0	1,5	1,0	1,0	1,0	1,0	1,0		7,5
	Group meeting / project planning		4,0	4,0	4,0	4,0		4,0	4,0	4,0	28,0
	Report writing			3,0	2,0			2,0	3,0	3,0	13,0
	Research	4,0	12,0	2,0						5,0	23,0
	Data collection / processing		3,0	3,0				10,0	6,0		22,0
	Model development / evaluation							1,0	2,0	3,0	6,0
	Other		1,0				2,0			12,0	15,0
		6,0	21,0	13,5	9,0	5,0	3,0	18,0	16,0	31,0	**122,5**