AMP®-Parkinson's Disease Progression Prediction

Daisy Adhikari, Lohitha Vanteru, Mahe Jabeen Abdul, Pranavi Sandrugu

Department of Applied Data Science, San Jose State University

DATA-255

Simon Shim

May 9, 2023

Abstract

Parkinson's disease is a long-term neurological condition that impairs movement, affecting more than 10 million individuals worldwide. People start to have trouble speaking, writing, walking, or performing other basic skills as the dopamine-generating neurons in certain areas of the brain are impaired or perish. As a result, the intensity of the symptoms in the patients increases over time. It is frequently diagnosed using clinical assessments and a progression scale, which typically depend on the skill of the medical professional. Accuracy varies widely across different examiners, and it also takes a long time to diagnose correctly. According to research, this disease's development and progression are significantly influenced by anomalies in proteins or peptides. A thorough evaluation of both motor and non-motor symptoms related to Parkinson's disease is provided by the Movement Disorder Society-Sponsored Revision of the Unified Parkinson's Disease Rating Scale (MDS-UPDRS). Deep Learning can help us understand the intrinsic patterns from protein and peptide data measurements and forecast MDS-UPDR values, which represent the disease's development in Parkinson's patients which could lead to the discovery of novel therapeutic interventions that can either stop the course of Parkinson's disease or treat it. The proposed Deep Neural Network model is then fed with the condensed input feature space and an addition of dense connections and batch normalization. The model can be used to identify people who are at high risk of developing Parkinson's disease and will offer insightful information about the underlying causes of the disease. The team is planning to utilize deep learning techniques covered in the course to analyze the complex relationship between protein and peptide levels and disease progression. The results of this project can have significant implications for the early detection and treatment of Parkinson’s disease, which can improve the quality of life of many people affected by this disease.

1. Introduction

Parkinson's disease (PD) is a progressive neurological disorder that affects millions of people worldwide. It is characterized by the death of dopamine-producing neurons in the brain, leading to motor symptoms such as tremors, rigidity, and impaired balance and coordination. PD also causes non-motor symptoms, including cognitive impairment, sleep disturbances, and depression, which can have a significant impact on a patient's quality of life. Despite decades of research, the exact cause of PD is still unknown, and there is no cure. Currently, the diagnosis of PD relies on clinical symptoms, which can be subjective and may vary from person to person. Moreover, by the time the symptoms appear, significant neuronal damage has already occurred, making early diagnosis and intervention critical for effective treatment. Although PD is currently incurable, early detection and accurate prediction of disease progression can facilitate better management of symptoms and improve treatment outcomes.

Recent studies have shown that changes in the levels of certain proteins and peptides in biological fluids such as blood, cerebrospinal fluid (CSF), and urine can provide valuable information for the early diagnosis and prognosis of PD. In particular, studies have shown that the levels of alpha-synuclein, tau protein, and neurofilament light chain in CSF are potential biomarkers for PD progression. However, the identification of such biomarkers and the interpretation of their significance require sophisticated analytical tools that can handle large and complex datasets. A thorough evaluation of both motor and non-motor symptoms related to Parkinson's disease is provided by the Movement Disorder Society-Sponsored Revision of the Unified Parkinson's Disease Rating Scale (MDS-UPDRS). Deep learning algorithms have recently emerged as powerful tools for analyzing complex datasets and making predictions in various fields, including healthcare. In this project, we aim to explore the potential of deep

learning algorithms in predicting the progression of PD using protein and peptide data measurements. Specifically, we will use data from CSF samples collected from PD patients at different stages of the disease and healthy controls. We will use state-of-the-art deep learning techniques, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to develop models that can predict PD progression based on the protein and peptide data measurements.

The results of this project have the potential to contribute to the development of a non-invasive and accurate biomarker-based tool for the early diagnosis and prognosis of PD. Furthermore, the deep learning-based approach developed in this project can be applied to other complex datasets in healthcare and beyond, contributing to the advancement of data-driven solutions to complex problems. Ultimately, this work could have significant implications for the development of personalized treatments for PD patients and could help to improve the quality of life for those affected by this devastating disease.

2. Literature review/related work

The theoretical underpinning of this project is described in this section of the dissertation, beginning with an explanation of Parkinson's disease and progressing through overviews of machine learning, deep learning, related work, and PD diagnosis issues.

2.1 Overview of Parkinson's disease

One of the disorders caused by damaged or malfunctioning cells in the substantia nigra pars compacta, a region of the brain, is Parkinson's disease (PD). The production of dopamine, a crucial molecule, is carried out by the cells in this area. Dopamine, which is in charge of regulating bodily movements like walking, writing, and even smiling, is lost as a result of the gradual loss of these cells in the brain (Vallejo et al., 2016).This ailment typically worsens with

time and primarily impacts adults between the ages of 50 and 70. There is still no cure for Parkinson's disease (PD), which was first identified by British physician James Parkinson in 1817 (Lones et al., 2014).

Four major indicators are regarded as cardinal symptoms for motor complaints: rest tremor, stiffness, bradykinesia, and occasionally postural instability. A resting tremor between 3-5 HZ that is described as asymmetrical tremor is present in about 70% of PD patients. Cogwheel rigidity, or the sensation of resistance during joint motion, is the second indication of PD .The third symptom, bradykinesia, slows down movement and enlarges with simple handwriting movements. The fourth symptom is postural instability, which is connected to balance and does not occur in the early stages of Parkinson's disease, especially in younger patients. It makes the patient unsteady on their feet and increases the risk of falls (Samii et al., 2004).

Some non-motor symptoms of Parkinson's disease (PD), such as hyposmia, rapid eye movement (REM), sleep behavior disorder, constipation, and depression, may manifest years before any motor symptoms. According to Litvan et al. (2012), many patients also display cognitive dysfunction, which can vary from moderate cognitive impairment (PD-MCI) to PD dementia (PDD). PD-MCI frequently appears in the early stages of the disease, whereas PDD typically happens 20 years after the onset of PD. Thinking and memory issues that are outside of what is typical with aging are referred to as PD-MCI and do not interfere with daily activities. A diagnosis of PD-MCI is crucial since PDD could develop from it (Meireles et al., 2012). Figure 1 depicts the progression of the symptoms gradually up until the dementia stage.

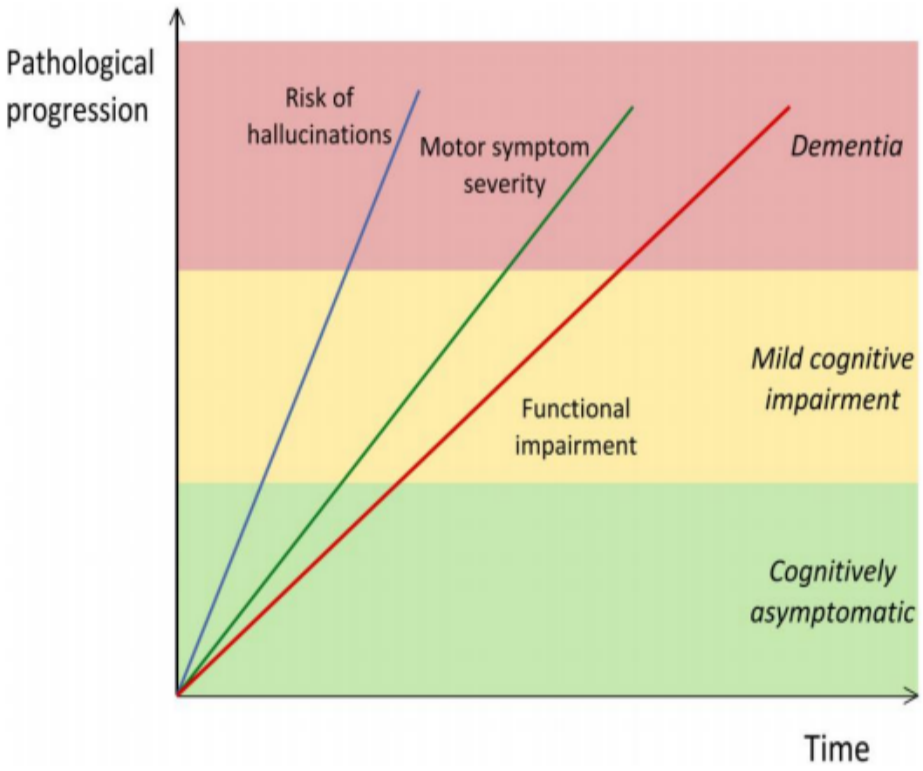


Figure 1: PD Symptoms development over time

2.2 Diagnosis of Parkinson's disease

To start, it's critical to distinguish between Parkinson's disease (PD) and parkinsonism. Tremors, stiffness, and bradykinesia are just a few of the symptoms that could be associated with Parkinson's disease (PD) and are clinically referred to as Parkinsonism. Since PD is only one type of parkinsonism, not all individuals who exhibit parkinsonism symptoms will also have PD (Parkinson's Disease Foundation, 2015). Instead, it may be caused by vascular problems or other neurodegenerative disorders.

Historically, motor symptoms have been used to make the diagnosis of PD. Most of the scales of assessment used to determine the severity of the disease have not yet been thoroughly examined and verified (Jankovic, 2008), despite identifying rating systems used to determine the

severity of the disease have not yet been thoroughly examined and verified (Jankovic, 2008), despite the identification of the cardinal indications of PD in clinical examinations. Although non-motor symptoms (such as cognitive changes, such as difficulties with attention and planning, sleep disorders, and sensory abnormalities, such as olfactory dysfunction) are frequently present in patients before the onset of PD, they lack specificity, are challenging to assess, and/or vary from patient to patient (Zesiewicz et al., 2006). Non-motor symptoms have therefore only recently made it possible to diagnose Parkinson's disease (PD) through independent means, despite some of them having been employed as supportive diagnostic criteria (Postuma et al., 2015). By merely observing the motor characteristics of PD patients, it can be challenging to differentiate PD from the other kinds of parkinsonism. As a result, other criteria must be included to improve the clinical diagnostic precision, such as asymmetry and a strong response to levodopa therapy. This increases the specificity of the diagnosis. Additionally, as revealed by follow-up investigations, the diagnostic specificity increased from what early research estimated was 76% to 90% specificity at death. (Berg et al., 2013).

2.3 Parkinson's Disease Prediction with machine learning algorithms

The Parkinson illness was automatically identified using machine learning, according to Indira R. et al. (2014), who used the speaker's speech and voice to do so. For the purpose of differentiating between persons who are healthy and those who have Parkinson's disease, the author used fuzzy C-means clustering and a pattern recognition-based technique. The accuracy, sensitivity, and specificity of this study's authors were 68.04%, 75.34%, and 45.83%, respectively. PD patients could be monitored remotely using SVM and k-Nearest Neighbor (k-NN) voice recordings. Age, gender, and voice recordings from the baseline, three- and six-month points are utilized to evaluate the attributes. Support Vector Machine had greater success in

spotting major declines in patients' UPDRS scores. In order to distinguish PD from healthy controls, A.Tsanas et al. (2011) offered feature selection, random forest, and support vector machines. Using just ten dysphonia features, the author's classification accuracy was 99% overall.

Saad et al. (2013) proposed a Bayesian Belief Network (BBN) for locating Parkinson disease patients' freezing. In experiments, a video dataset obtained from actual Parkinson's disease patients is used. This dataset is accessible online. Each file contains a matrix with three sensors' measurement data in the x, y, and z dimensions. Weather-related freezing of gait (FoG) incidence is tracked. The Bayesian Naive Classifier (BNC) classifier is utilized to test the models while labeling these annotations via video, which captures every patient's run and visible results.By using real-time acquisition data from healthy and PD participants combined with unique feature extraction parameters, such as tunable - Q factor wavelet transform and signal processing methods, Sakar et al. (2018) showed the early prediction based on speech modality. Principal Component Analysis mixed with Support Vector Machine and Auto Sparse Encoder combined with SVM are two hybrid machine learning models based on speech features that were proposed. A PD detection system that incorporates voice feature data, different ML algorithms, and feature selection strategies was described by Iqra Nissar et al .(2019) K Nearest Neighbour, Decision Tree, Logistic Regression, Multi-Layer Perceptron, Naive Bayes, SVM, and Ensemble Classifier were among the different ML approaches employed in the system. For PD prediction, an ensemble ML model was based on the voice measurements.

T. Swapna et al. () proposed a paper that deals with the application of seven classification algorithms on the acquired data set, drawing comparisons between the results, and also forecasting whether the person is healthy or affected by Parkinson disease from the given data.

The outputs of the chosen algorithms—Naive Bayes, Random Forest, Neural Networks, Decision Trees, AdaBoost, SVM, and KNN—were tabulated and compared. Using Python to implement Scikit Libraries produced the results that were expected. These parameters were used to calculate the final accuracy. The best accuracy is provided by the Random Forest algorithm (78.56%), which is closely followed by the best accuracy provided by the Decision Tree algorithm (77.63%). According to Dragana Miljkovic et al., the Predictor portion was able to anticipate each of the 15 distinct Parkinson's symptoms based on the medical tests the patients underwent. The application of machine learning and data mining approaches to various symptoms individually results in an accuracy range of 57.1% to 77.4%, with tremor detection having the best accuracy.

2.4 Parkinson's Disease Prediction with deep learning algorithms

Zineddine (1 C.E.) proposed a new method for measuring digital PPMI (Parkinson's Progression Markers Initiative) and combines it with spiral drawings to enhance the accuracy of a neural network. The findings demonstrate a high-performing CNN (convolutional neural network) model with an accuracy rate of 100%, indicating that users can have greater confidence in assessing the development of Parkinson's disease. After being trained, validated, and tested, the model was capable of accurately categorizing the progression of Parkinson's disease as High, Medium, or Low with a high level of certainty. The study utilized three datasets: the first dataset consisted of profile data obtained from the measurements provided by PPMI, the second dataset comprised of spiral drawings generated from the collected data, and the third dataset combined the profiles and spiral drawings to train, validate, and test the developed model. Performance metrics such as accuracy and loss were then evaluated to verify the efficacy of the model. Various techniques to enhance the performance of neural networks have been used including

hyper-parameter tuning, data manipulation, data augmentation, and model optimization. In this particular study, data augmentation was employed by rotating profile images to alter their orientation. Additionally, commonly used CNN optimizers such as sgd, rmsprop, adagrad, and adam were employed to identify the most effective one. The epoch number, which refers to the number of times the network's weights were adjusted, was determined by monitoring the behaviors of validation and training errors. The study's innovative approach of combining numerical measurements of disease indicators with spiral drawings of patients in a periodic manner proved successful in assessing the progression of Parkinson's disease. The use of constructed graphical profiles resulted in a high accuracy rate (100%) when categorizing the disease progression using the designed CNN. The variety of measured indicators was a significant factor in achieving an accurate disease progression profile. Neither the profiles without spiral drawings (with an accuracy of 80%) nor the spiral drawings by themselves (with an accuracy of 83%) achieved the high accuracy rate (100%) attained when both types of profiles were combined by superimposition. Therefore, this novel approach is a significant contribution to evaluating the progression of Parkinson's disease.

Grover et al. (2018) proposed a methodology for predicting the severity of Parkinson's disease utilizing deep neural networks on the UCI Parkinson's Telemonitoring Voice Data Set. The 'TensorFlow' deep learning library in Python was utilized to implement the neural network for predicting the disease severity. The authors mentioned the accuracy values achieved by this method were superior to those obtained in previous research works. The methodology includes collecting the voice data from Parkinson's disease patients for analysis then the collected data is then normalized using min-max normalization. Next, a deep neural network is created with an input layer, hidden layers, and an output layer. The number of neurons in the input layer is

equivalent to the number of attributes in the input data. The output layer includes two neurons representing the two classes - "severe" and "non-severe". The normalized data is input into the constructed deep neural network for training and testing purposes. The authors utilized the Parkinson's Telemonitoring Voice Data Set from the UCI Machine Learning Repository, which contains biomedical voice measurements of 42 Parkinson's disease patients. The dataset includes various attributes such as subject number, subject age, subject gender, time interval, Motor UPDRS, Total UPDRS, and 16 biomedical voice measures. The dataset comprises 5,875 voice recordings of these patients, with an average of approximately 200 recordings collected from each patient (identified through the first attribute - subject number). The data is in ASCII CSV format. The researchers discovered that using the motor UPDRS score for classification yields better results than using the total UPDRS score. As a result, it can be concluded that the motor UPDRS score is a more effective metric for predicting the severity of Parkinson's disease.

Ahmed et al. (2022) developed a model for predicting the progression of Parkinson's disease in patients using longitudinal RNA-Seq data. The data used in the study was obtained from the Parkinson Progression Marker Initiative (PPMI) and consisted of 423 patients with varying numbers of visits and 34,682 predictor variables over 4 years. The proposed model is based on a deep Recurrent Neural Network (RNN) with the addition of dense connections and batch normalization into RNN layers. The results show that the model can predict the progression of PD using high dimensional RNA-Seq data with an RMSE of 6.0 and a rank-order correlation of ($r = 0.83$, $p < 0.0001$) between the predicted and actual disease status. To form the target variable for the disease status, only the samples of patients who had undergone MDS-UPDRS-III in the "OFF" medication state were included in the analysis. The developed predictive model is flexible and can accommodate varying visit time intervals and the number of

visits, making it adaptable over time. In this research, they utilized a densely connected RNN model with 256 Vanilla RNN cells in each composite block. The network consisted of three dense blocks, each composed of four composite blocks. They applied L2 regularization on the weights in the RNN layers, with the loss function being mean square error and the optimizer being Nadam. To reduce the learning rate, they employed a learning rate schedule that decreased the learning rate by a factor of 1/5 every 10 epochs if there was no improvement in the validation loss. They conducted training in mini-batches, with a batch size of 16 subjects, where each subject had the same number of time sequence/visits data. To compare the performance between the proposed model and baseline models, They performed fivefold Cross-Validation (CV) on the test dataset. The study found that including Batch Normalization and Dense Connections in the multi-layered RNN significantly improved its ability to learn features from high dimensional gene expression data.

The authors (Aşuroğlu & Oğul, 2022) propose a multistage deep learning approach that uses Ground Reaction Force (GRF) sensors to forecast exact UPDRS (Unified Parkinson's Disease Rating Scale) values for PD severity assessment. The approach involves extracting frequency and time domain features from the GRF signals and combining Convolutional Neural Networks (CNN) and Locally Weighted Random Forest (LWRF) architectures to predict UPDRS values. They propose a multistage deep learning approach in this manner. Several frequency and time domain features are extracted from GRF signals for the first stage. After that, they combined Convolutional Neural Networks (CNN) deep learning architecture with a Locally Weighted Random Forest (LWRF) architecture to predict UPDRS values. LWRF architecture is a locally weighted Random Forest approach to reduce interpatient variability in GRF signals. The authors highlight the contribution of their approach as being the first to use deep learning

regression architectures to predict exact values of PD symptom severities. Additionally, their model outperformed previous studies that used LWRF models to predict UPDRS values.

3. Real World Application of the Project Concept

The development of a predictive model for Parkinson's disease progression using protein and peptide data measurements with deep learning has significant real-world applications in healthcare. Therefore, the ability to predict the progression of the disease could be crucial for developing effective treatments and improving patient outcomes. Firstly, it could enable early diagnosis and treatment of PD, which is crucial for improving patient outcomes. Early detection and intervention have been shown to slow down disease progression and improve quality of life for PD patients. Secondly, the project's concept could aid in the development of personalized treatment plans for PD patients. Currently, treatment options for PD are limited and often involve a trial-and-error approach to finding the most effective medication and dosage By predicting disease progression, doctors can tailor treatment plans to individual patients, optimizing their care and improving outcomes.

Additionally, the concept could have significant implications for drug development. With an accurate predictive model for disease progression, researchers can more effectively test potential treatments and evaluate their efficacy. This could potentially accelerate the drug discovery process, leading to the development of more effective treatments for PD. In addition to these medical applications, predictive models for Parkinson's disease progression could also have implications for insurance and disability claims. PD is a chronic, progressive disease, and many patients struggle to access insurance or disability benefits due to the difficulty in predicting disease progression. Predictive models could potentially provide a more accurate picture of disease progression, allowing patients to access the support and resources they need.

Furthermore, the project's concept could be applied to other neurodegenerative diseases such as Alzheimer's disease and Huntington's disease. Overall, the project's concept has significant real-world applications in the diagnosis, treatment, and drug development of Parkinson's disease and other neurodegenerative diseases. By predicting disease progression using protein and peptide data measurements with deep learning, the project could help improve patient health and accelerate the development of effective treatments.

4. About Data

The dataset for the current study was obtained from a Kaggle competition called "AMP®-Parkinson's Disease Progression Prediction." The cerebrospinal fluid (CBF) samples taken from several hundred patients make up the main dataset, which comprises protein abundance measurements obtained from mass spectrometry analysis. The files and descriptions that Kaggle provided for this challenge are included below.

- Train_peptides.csv: Peptide-level mass spectrometry data, where each peptide is a subunit part of a protein. The file contains details about the month of the visit, relative to the first visit by the patient, and the amount of the peptide and the sequence and frequency of amino acids included in the peptide.
- Train_proteins.csv: Protein expression frequencies compiled from peptide level data including visit and patient details, the related protein's UniProt ID code, and normalized protein expression and the frequency of the protein's occurrence in the sample.
- Train_clinical_data.csv: Clinical information on each patient, such as details about the appointment and the patient as well as results from the various sections of the Unified Parkinson's Disease Rating Scale, which rates the severity of PD symptoms, where higher numbers indicate more severe symptoms along with details like clinical state of mind

whether or not the patient was taking medication such as Levodopa during the UPDRS assessment.

- Supplemental_clinical_data.csv: Clinical data without any corresponding CSF samples that are meant to offer context to the normal development of Parkinson's disease. This information is meant to give further background on how Parkinson's disease typically progresses. similar columns to train_clinical_data.csv are used.

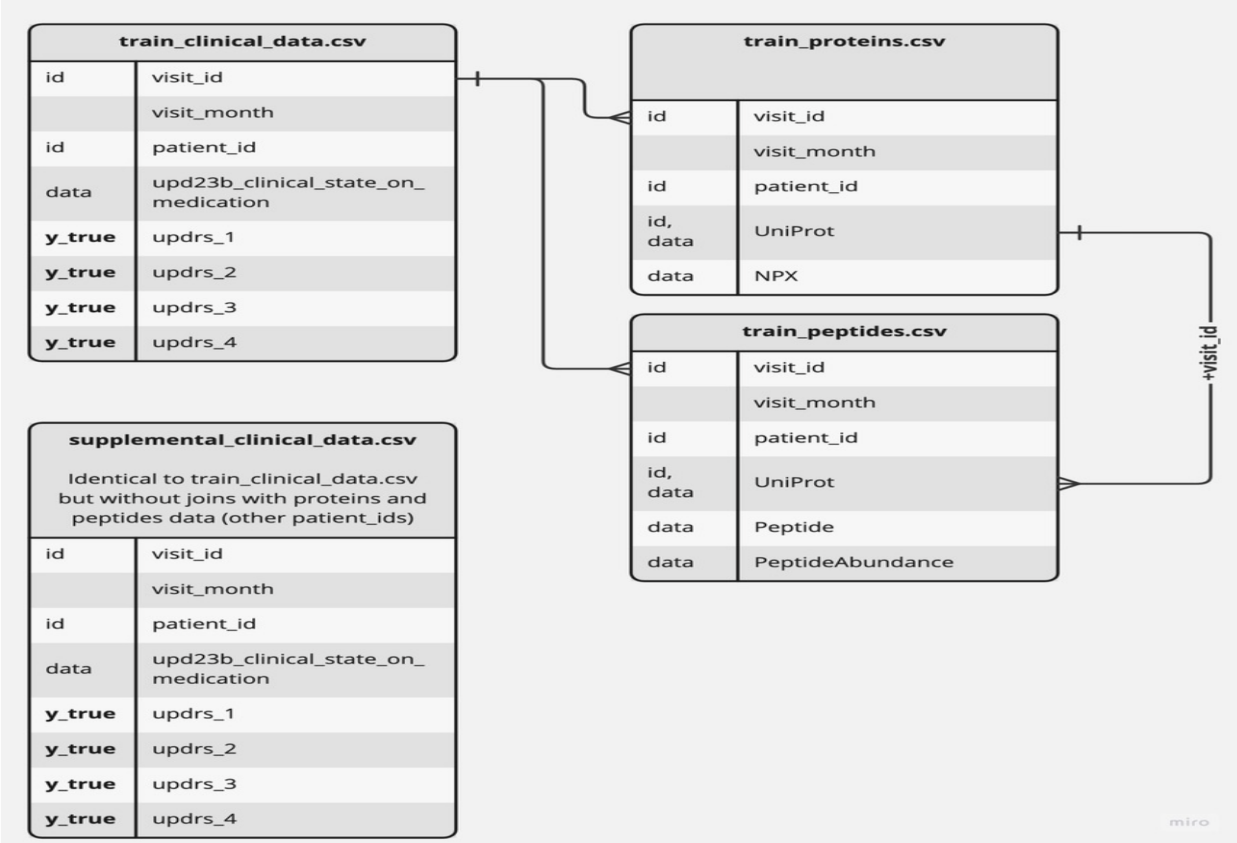


Figure 2: ER Diagram of Our Training Data

The unifed Parkinson's disease rating scale (MDS-UPDRS), which is sponsored by the Movement Disorder Society, provides a good representation of the clinical progression of Parkinson's disease. Its rating scale ranges from 0 to 272, with 0 representing normal and 272 representing severe motor and non-motor decline. Following enrolment, patients were followed

for 4 years, with clinical data being gathered at regular intervals of 3 months, 6 months, 9 months, 12 months, 18 months, 24 months, 30 months, 36 months, 42 months, and 48 months. By taking into account the protein abundance data from numerous prior visits, we hope to estimate this patient's MDS-UPDRS score for the upcoming hospital visit such as in the 6th month ,12th month and 24th month visit. All of the responses to the MDS-UPDRS Questionnaire (Parts I, II, III, and IV) for a subject's visit were combined to construct the target variable.

5. Data Preprocessing

Data preprocessing is a crucial step in building any machine learning model, as it involves preparing and cleaning the data to make it suitable for analysis. Furthermore, the quality of the data has a significant effect on the accuracy and reliability of the model's predictions. Data preprocessing includes several techniques, such as data cleaning, feature selection/extraction, feature scaling, and data transformation. The training data contains three data files: proteins data, peptides data, and clinical data. Some of the processes we have done in the data preprocessing are discussed below:

5.1 Data Cleaning

Data cleaning includes checking for missing values, duplicated data, or incorrect data from the dataset. It also includes removing or imputing missing values and duplicates. The ‘isnull()’ method is used to calculate the total number of missing values for each column of the dataset. This information can help determine the dataset's quality and decide how to handle missing values. In addition, we can ensure the dataset is clean and ready for further analysis by checking for nulls in below Figure 3 and Figure 4.

```
proteins_train.isnull().sum() #checking for nulls
```

```
visit_id       0
visit_month    0
patient_id     0
UniProt        0
NPX            0
dtype: int64
```

Figure 3: Code snippet for checking the nulls on Proteins training dataset

```
proteins_train.duplicated().sum() #checking for duplicates
```

```
0
```

Figure 4: Code snippet for checking the duplicates on Proteins training dataset

The clinical dataset has the feature updrs_[1-4], is the patient's score for part N of the Unified Parkinson's Disease Rating Scale (UPDRS). The severity of PD symptoms is measured using the Unified Parkinson's Disease Rating Scale (UPDRS), which is divided into four parts (UPDRS_1-4), each focusing on specific symptoms of the disease. Part 1 measures non-motor experiences of daily living, Part 2 measures motor experiences of daily living, Part 3 measures motor function, and Part 4 measures complications of therapy. We have created the new columns representing the UPDRS scores at different time intervals. The sample data from the clinical dataset is shown in Figure 5.

```
clinicals_train
```

	visit_id	patient_id	visit_month	updrs_1	updrs_2	updrs_3	updrs_4	upd23b_clinical_state_on_medication
0	55_0	55	0	10.0	6.0	15.0	NaN	NaN
1	55_3	55	3	10.0	7.0	25.0	NaN	NaN
2	55_6	55	6	8.0	10.0	34.0	NaN	NaN
3	55_9	55	9	8.0	9.0	30.0	0.0	On
4	55_12	55	12	10.0	10.0	41.0	0.0	On
...	...	...	...	...	...	...	...	...
2610	65043_48	65043	48	7.0	6.0	13.0	0.0	Off
2611	65043_54	65043	54	4.0	8.0	11.0	1.0	Off
2612	65043_60	65043	60	6.0	6.0	16.0	1.0	Off
2613	65043_72	65043	72	3.0	9.0	14.0	1.0	Off
2614	65043_84	65043	84	7.0	9.0	20.0	3.0	Off

2615 rows × 8 columns

Figure 5: Sample data of the Clinical training dataset

5.2 Data Merging

We have merged two datasets, proteins_train, and peptides_train, into a single dataset called proteins_peptides based on four features: 'visit_id,' 'visit_month,' 'patient_id,' and 'UniProt.' By merging these two datasets, we have obtained the protein and peptide measurements information for each patient's visit into a table shown in Figure 6. It can be used for analyzing the relationship between protein and peptide levels and other clinical features, such as disease progression or response to treatment.

Furthermore, we have merged the dataset, proteins_peptides, with the clinicals_train dataset, based on three columns: 'visit_id,' 'visit_month,' and 'patient_id.' By merging the clinical data with the protein and peptide data, we can combine the clinical features with the molecular features of the patients, which can provide more insights into the relationship between disease progression and biomarkers. The sample data of the final merged data is shown in the Figure 7.

```
# Merging the proteins and peptides data
proteins_peptides = pd.merge(proteins_train, peptides_train, on = ['visit_id', 'visit_month', 'patient_id', 'UniProt'])
proteins_peptides
```

	visit_id	visit_month	patient_id	UniProt	NPX	Peptide	PeptideAbundance
0	55_0	0	55	O00391	11254.3	NEQEQPLGQWHLS	11254.30
1	55_0	0	55	O00533	732430.0	GNPEPTFSWTK	102060.00
2	55_0	0	55	O00533	732430.0	IEIPSSVQQVPTIIK	174185.00
3	55_0	0	55	O00533	732430.0	KPQSAVYSTGSNGILLC(UniMod_4)EAEGEPQPTIK	27278.90
4	55_0	0	55	O00533	732430.0	SMEQNGPGLEYR	30838.70
...	...	...	...	...	...	...	...
981829	58648_108	108	58648	Q9UHG2	369437.0	ILAGSADSEGVAAPR	202820.00
981830	58648_108	108	58648	Q9UKV8	105830.0	SGNIPAGTTVDTK	105830.00
981831	58648_108	108	58648	Q9Y646	21257.6	LALLVDTVGPR	21257.60
981832	58648_108	108	58648	Q9Y6R7	17953.1	AGC(UniMod_4)VAESTAVC(UniMod_4)R	5127.26
981833	58648_108	108	58648	Q9Y6R7	17953.1	GATTSPGVYELSSR	12825.90

981834 rows × 7 columns

Figure 6: Sample data of merged datasets

```
#Merging further with clinical data

merged_data = pd.merge(proteins_peptides, clinicals_train, on = ['visit_id', 'visit_month', 'patient_id'])
merged_data
```

	visit_id	visit_month	patient_id	UniProt	NPX	Peptide	PeptideAbundance
0	55_0	0	55	O00391	11254.3	NEQEQPLGQWHLS	11254.30
1	55_0	0	55	O00533	732430.0	GNPEPTFSWTK	102060.00
2	55_0	0	55	O00533	732430.0	IEIPSSVQQVPTIIK	174185.00
3	55_0	0	55	O00533	732430.0	KPQSAVYSTGSNGILLC(UniMod_4)EAEGEPQPTIK	27278.90
4	55_0	0	55	O00533	732430.0	SMEQNGPGLEYR	30838.70
...	...	...	...	...	...	...	...
941739	58648_108	108	58648	Q9UHG2	369437.0	ILAGSADSEGVAAPR	202820.00
941740	58648_108	108	58648	Q9UKV8	105830.0	SGNIPAGTTVDTK	105830.00
941741	58648_108	108	58648	Q9Y646	21257.6	LALLVDTVGPR	21257.60
941742	58648_108	108	58648	Q9Y6R7	17953.1	AGC(UniMod_4)VAESTAVC(UniMod_4)R	5127.26
941743	58648_108	108	58648	Q9Y6R7	17953.1	GATTSPGVYELSSR	12825.90

941744 rows × 12 columns

Figure 7: Sample data of the final merged data

5.3 Data Transformation

In this step, we have used pivot on the merged_data dataset on the ‘UniProt’ column, with the ‘Peptide’ column as the new columns and the ‘PeptideAbundance’ column as the new values. The resulting dataset will have each unique ‘visit_id’ as a row index, with the

corresponding Peptide values as the columns. The ‘PeptideAbundance’ values will be the entries of the dataset. This data transformation helps organize the data to facilitate analysis. The sample data is shown in the Figure 8.

```
#Pivot on UnitProtein
merged_data_pivotted = merged_data.pivot(index='visit_id', columns = ['Peptide'], values = 'PeptideAbundance')
```

```
merged_data_pivotted
```

Peptide	AADDTWEPFASGK	AAFGQGSGPIMLDEVQC(UniMod_4)TGTEASLADC(UniMod_4)K	AAFTEC(UniMod_4)C(UniMod_4)QAADK	AANEVSSADVK	AATGEC(UniMod_4)TATVGKR	AATVGS
visit_id						
10053_0	6580710.0	31204.4	7735070.0	NaN	NaN	
10053_12	6333510.0	52277.6	5394390.0	NaN	NaN	
10053_18	7129640.0	61522.0	7011920.0	35984.7	17188.00	
10138_12	7404780.0	46107.2	10610900.0	NaN	20910.20	
10138_24	13788300.0	56910.3	6906160.0	13785.5	11004.20	
...	...	...	...	...	...	
8699_24	6312970.0	44462.7	12455000.0	11051.3	1163.18	
942_12	11289900.0	46111.7	11297300.0	NaN	13894.10	
942_24	10161900.0	32145.0	12388000.0	25869.2	17341.80	
942_48	8248490.0	30563.4	11882600.0	NaN	19114.90	
942_6	6177730.0	42682.6	3596660.0	25698.8	17130.60	

1068 rows × 968 columns

Figure 8: Sample data of the Pivotted data

Now, we have merged the merged_data_pivotted dataset (which contains the pivoted protein and peptide abundance data) with the clinicals_train dataset (which contains the clinical assessment data) based on the common visit_id column is shown in Figure 9. This include the UPDRS scores (updrs_1, updrs_2, updrs_3, and updrs_4) and create a new dataset with relevant features for analysis.

Next, we have created a function that is defined to extract target values based on UPDRS scores. First, it initializes an empty dictionary where each key is a patient ID, and each value in columns represents UPDRS scores for each event at each time. Then, it iterates through each unique patient in clinicals_train, extracts their data, and creates a list of visit months for each time point.

It extracts UPDRS scores for that patient and assigns them to the corresponding column in patient_data for each time point. Finally, patient_data is added to the targets with the patient ID.

```
#Merging dataset to add updrs_1,2,3,4

new_merged = pd.merge(clinicals_train, merged_data_pivotted, on="visit_id", how="left")
new_merged = new_merged.set_index('visit_id')
```

```
new_merged
```

visit_id	patient_id	visit_month	updrs_1	updrs_2	updrs_3	updrs_4	upd23b_clinical_state_on_medication	AADDTWEPFASGK	AAFGQGSGPIMLDEVQC(UniMod_4)TGTE
55_0	55	0	10.0	6.0	15.0	NaN	NaN	8984260.0	
55_3	55	3	10.0	7.0	25.0	NaN	NaN	NaN	
55_6	55	6	8.0	10.0	34.0	NaN	NaN	8279770.0	
55_9	55	9	8.0	9.0	30.0	0.0	On	NaN	
55_12	55	12	10.0	10.0	41.0	0.0	On	8382390.0	
...	...	...	...	...	...	...	...	...	
65043_48	65043	48	7.0	6.0	13.0	0.0	Off	7187220.0	
65043_54	65043	54	4.0	8.0	11.0	1.0	Off	NaN	
65043_60	65043	60	6.0	6.0	16.0	1.0	Off	NaN	
65043_72	65043	72	3.0	9.0	14.0	1.0	Off	NaN	
65043_84	65043	84	7.0	9.0	20.0	3.0	Off	NaN	

2615 rows × 975 columns

Figure 9: Sample data of new merged dataset

Then we combine data from the targets dictionary and return a new data frame that includes the UPDRS scores for each patient at specific time intervals. The resulting dataset, formatted_clin, contains columns for the UPDRS scores, with each column containing the score for a different UPDRS event (updrs_1, updrs_2, updrs_3, updrs_4) and time interval (updrs_1, updrs_2, updrs_3, updrs_4). Finally, the index of the data frame is set to the visit_id is shown in Figure 10. By combining the information from the clinical data with the UPDRS scores, this new dataset can be used for further analysis to investigate relationships between the UPDRS scores and other features in the data, such as the protein and peptide abundance values.

```
formatted_clin = pd.concat(targets.values(), ignore_index=True).set_index('visit_id').iloc[:, 7:]
formatted_clin.head()
```

visit_id	updrs_1_plus_0_months	updrs_1_plus_6_months	updrs_1_plus_12_months	updrs_1_plus_24_months	updrs_2_plus_0_months	updrs_2_plus_6_months	upd
55_0	10	8	10	16	6	10	
55_6	8	10	7	14	10	10	
55_12	10	7	16	17	10	13	
55_18	7	16	14	12	13	9	
55_24	16	14	17	17	9	13	

Figure 10: Sample data of UPDRS scores for each patient at time intervals

5.4 Feature Preparation using UnitProt:

We have used pivot to proteins_train dataset by the 'UniProt' column, with 'visit_id' as the index and NPX as the values. Each distinct visit_id and each distinct UniProt value from the previous dataset will have a row and a column, respectively, in the final dataset, portfeatures. The values in each profeatures correspond to the matching UniProt's NPX value for the specified visit_id.

In the next step, we have merged the protein features (protfeatures) and formatted clinical data (formatted_clin) datasets using the visit_id. Furthermore, we have calculated the percentage of missing values in the merged data frame for the protein feature columns. Then, we created a visit_month column in the merged data frame by extracting the visit month from the visit_id column is shown in Figure 11.

```
df = protfeatures.merge(formatted_clin, left_index=True,right_index=True,how='right')
print(f'\nNA values: {df[protfeatures.columns].isna().sum().sum()/(len(df)*len(protfeatures.columns)):.2%}')
df['visit_month'] = df.reset_index().visit_id.str.split('_').apply(lambda x: int(x[1])).values
df.head()
```

```
NA values: 53.64%
```

visit_id	O00391	O00533	O00584	O14498	O14773	O14791	O15240	O15394	O43505	O60888	...	updrs_2_plus_
55_0	11254.3	732430.0	39585.8	41526.9	31238.0	4202.71	177775.0	62898.2	333376.0	166850.0	...	
55_6	13163.6	630465.0	35220.8	41295.0	26219.9	4416.42	165638.0	62567.5	277833.0	170345.0	...	
55_12	15257.6	815083.0	41650.9	39763.3	30703.6	4343.60	151073.0	66963.1	332401.0	151194.0	...	
55_18	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
55_24	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	

5 rows × 244 columns

Figure 11: Sample data of merged datasets- protfeatures and formatted_clin

6. Data Visualization

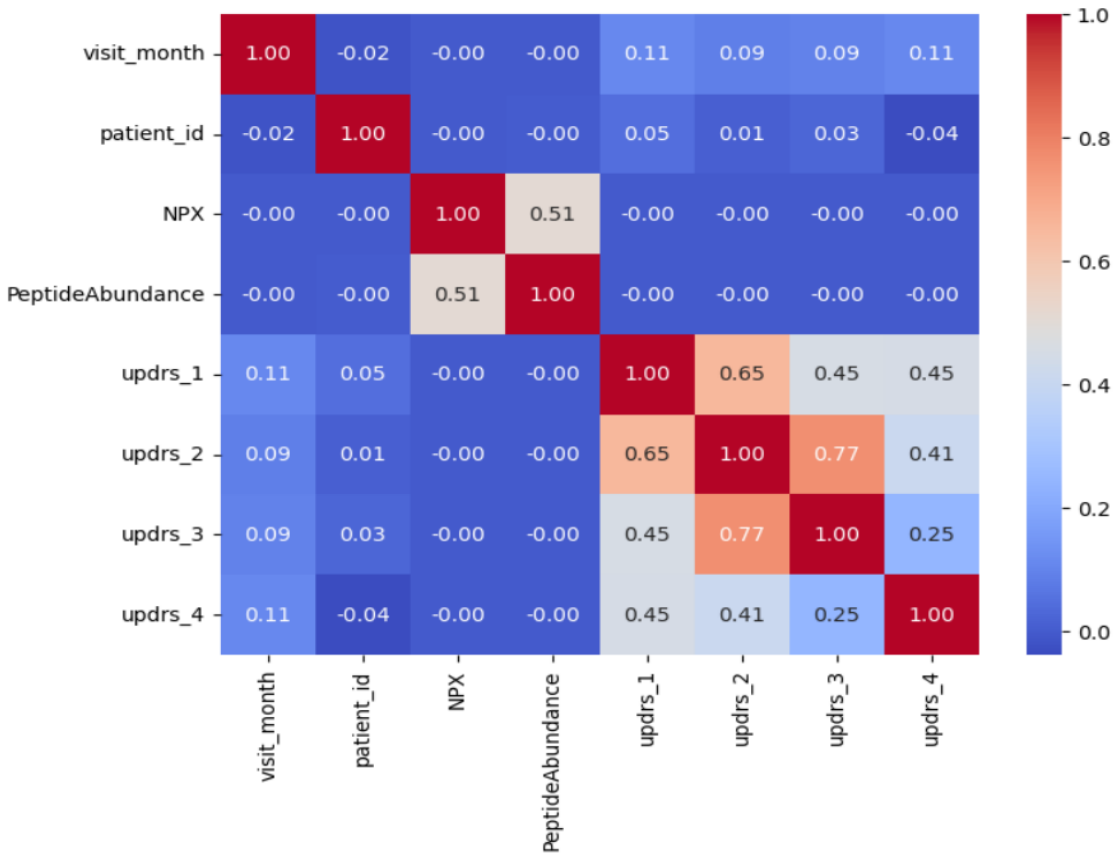


Figure 12: Heatmap to get better understanding of the correlated data

Here we have created a heatmap shown in Figure 12, to help visualize the correlation between the numerical features in the merged_data Dataset. For example, we can see that the columns updrs_2 and updrs_3 and updrs_1 and updrs_2 are highly correlated with each other compared to the other features. Furthermore, we cannot see any correlation between the other columns in the dataset, in which some have no correlation or are negatively correlated. This visualization can help identify patterns and relationships between features, guiding further analysis or informing feature selection for training the models.

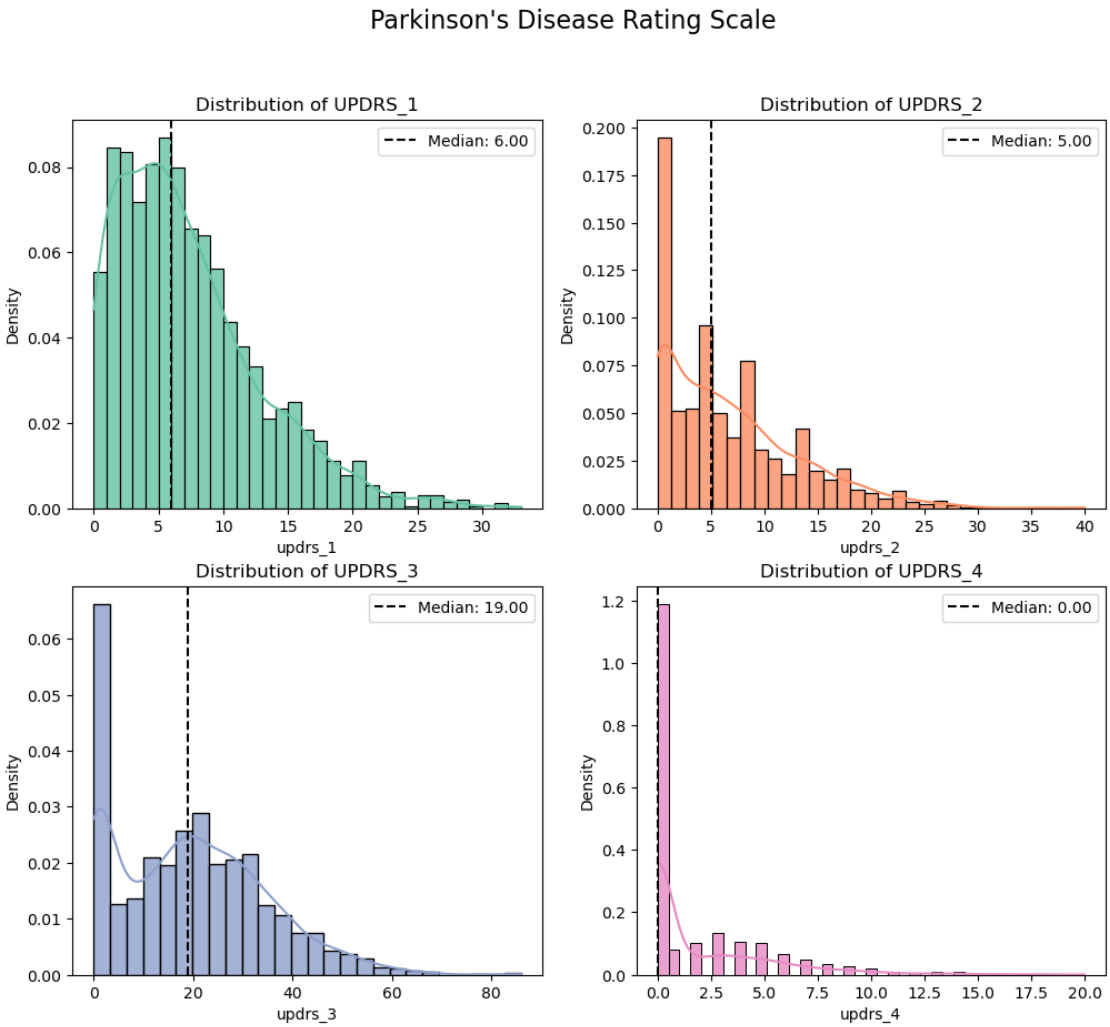


Figure 13: Parkinson's disease Rating Scale

Here we have created four subplots and generated a set of boxplots displaying the distribution of data points for each variable of updrs[1-4] shown in Figure 13. We can see that the plots are highly skewed, and most of the data lies in the left part of the plot showing the ratings below around 10. The vertical line in each subplot indicates the median value of the distribution. The plots visualize the distribution of the Parkinson's Disease Rating Scale scores for each of the updrs[1-4] features.

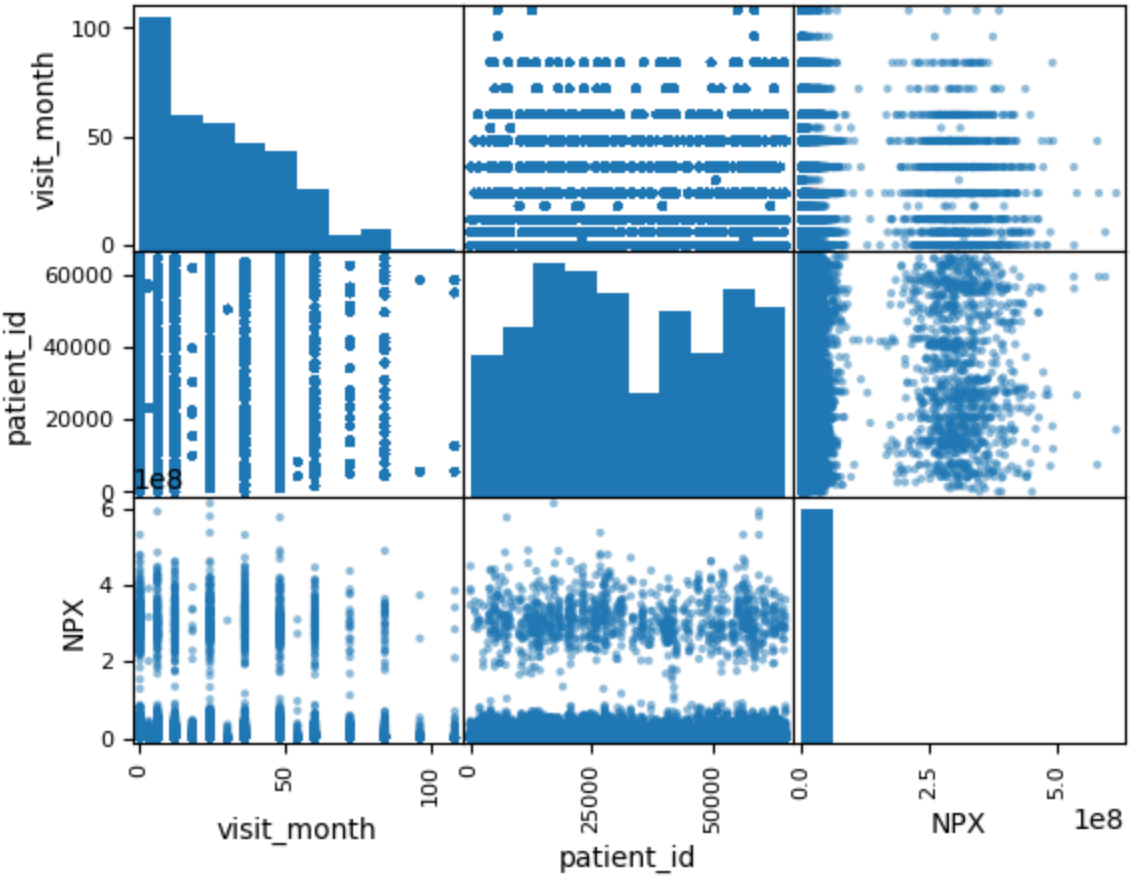


Figure 14: Scatter plot of the proteins_train data

Here we have created a scatter plot shown in Figure 14, to understand the visual representation of the data for the proteins_train dataset, to observe the relationship between the variables.

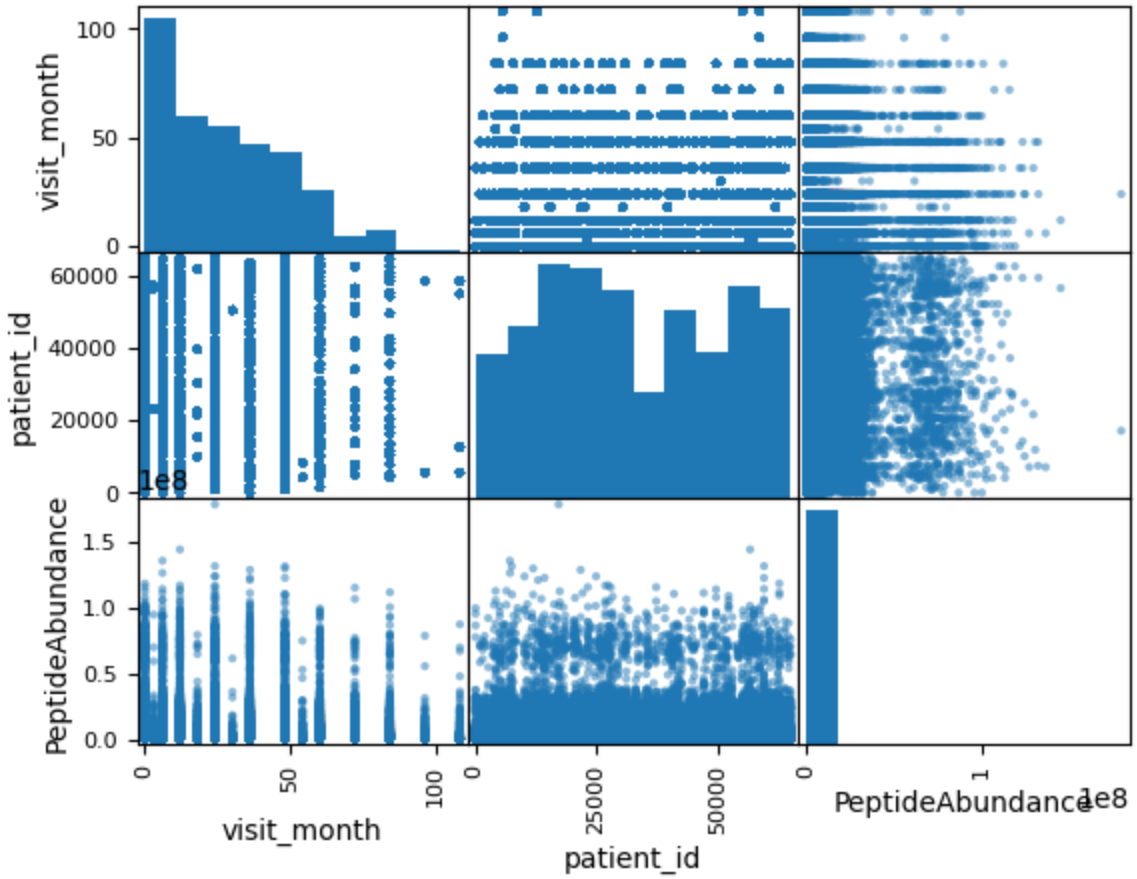


Figure 15: Scatter plot of the peptides_train data

Here we have created a scatter plot to understand the visual representation of the data for the peptides_train dataset, to observe the relationship between the variables is shown in Figure 15.

In the next plot, we have created a scatter plot to understand the visual representation of the data for the clinical_train dataset, to observe the relationship between the variables is shown in Figure 16.

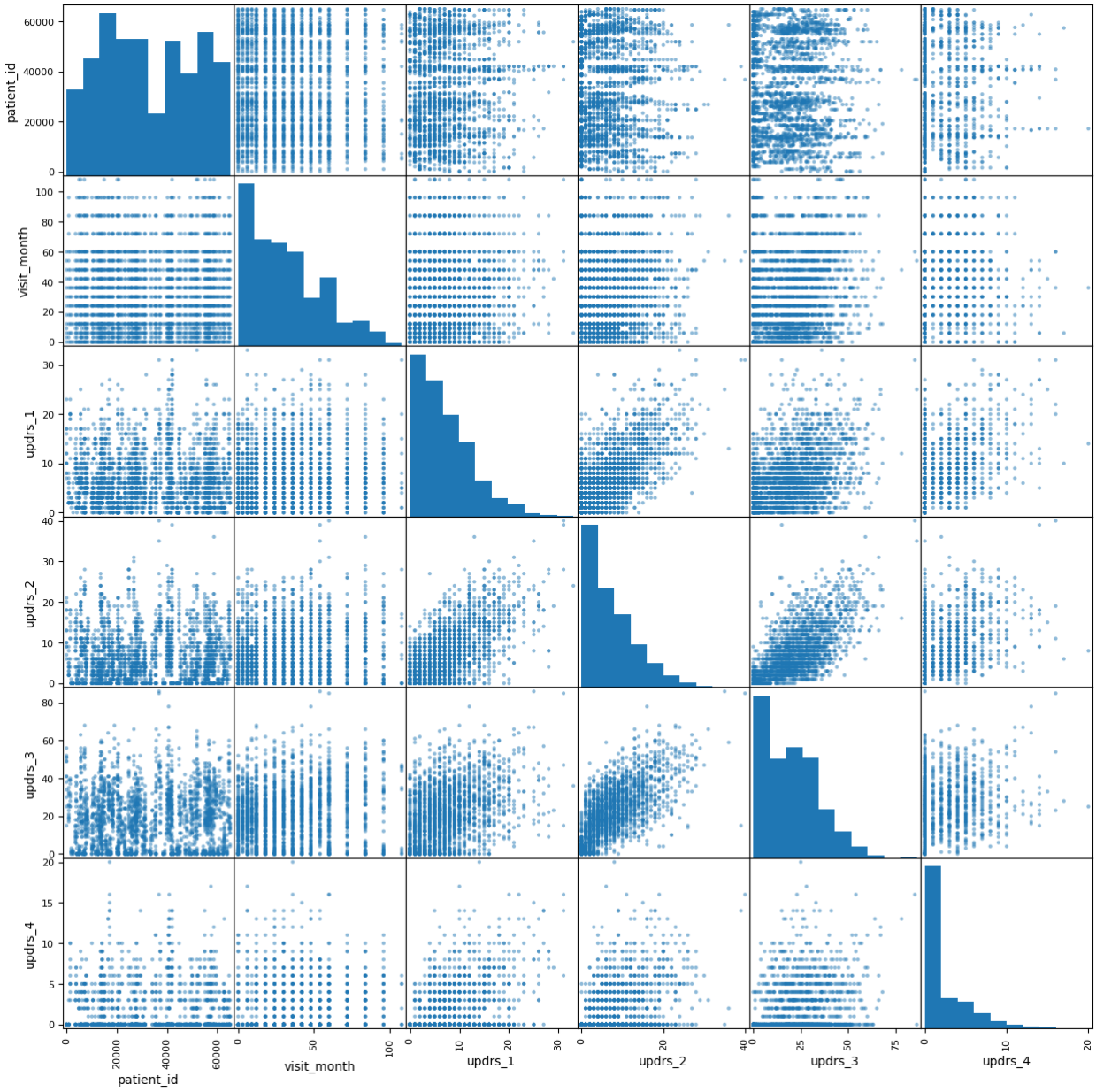


Figure 16: Scatter plot of the clinical_train data

7. Problem Formulation

It is critical to predict a patient's MDS-UPDRS score for the forthcoming hospital visit using the protein abundance data from several earlier visits. The issue is modeled in such a way that, when a patient's Baseline year (the first year) protein data is sent to the predictive model, it predicts the patient's MDS-UPDRS score for the next year, in order to understand the patient's

status. When the baseline year and the second year's protein data are used as input, the model predicts the third year's MDS-UPDRS score, and when the baseline year, second year, and third year's data are used as input, the model predicts the fourth year's MDS-UPDRS score. The model makes use of the temporal correlation to estimate the disease condition in the near future using the historical protein abundance sequence data as we advance through time.

Given that a subject receives N yearly visits, the input at time t_n is a feature vector indicated by X_n. The feature vector X_n contains a Q-number of characteristics that mix both motor features and non-motor features that were present during the patient's nth visit. For instance, at the patient's first visit, X_1 is a feature vector with Q features. In addition, if the "N^{th}" visit denotes the subject's most recent visit, the "$(N+1)^{th}$" visit denotes the subject's visit as soon as in the coming 6 months. The suggested prediction model can be described in its simplest form if the MDS-UPDRS final score at the Nth is represented by y_n and the future value of MDS-UPDRS is represented by y_{n+1}, then the suggested model for the prediction can be expressed as follows in its most straightforward form:

$$\hat{y}_{<n+1>} = f\left(X_{<1>}, X_{<2>}, X_{<3>}, \ldots X_{<n>}\right) \quad (1)$$

where the output $\hat{y}_{<n+1>}$ is the anticipated MDS-UPDRS score for the patient's subsequent visit and the function f() is the central component of our proposed model. The ground-truth value of the MDS-UPDRS scores from the most recent and previous visits is added to The Eq. (1). To forecast the MDS-UPDRS score of the (N+1)th visit, for instance, we input the model with the theMDS-UPDRS score of the Nth visit as well as the scores of all the prior visits (N-1)th, (N-2)th,..., 1st. The period of time between two consecutive visits was also included, where the time

interval between the first visit and the following visit can be represented as $\Delta t_{<1>}$ and the gap between the Nth visit and the (N+1)th visit can be represented as Δt<n>.

The finished version of our suggested model is as follows:

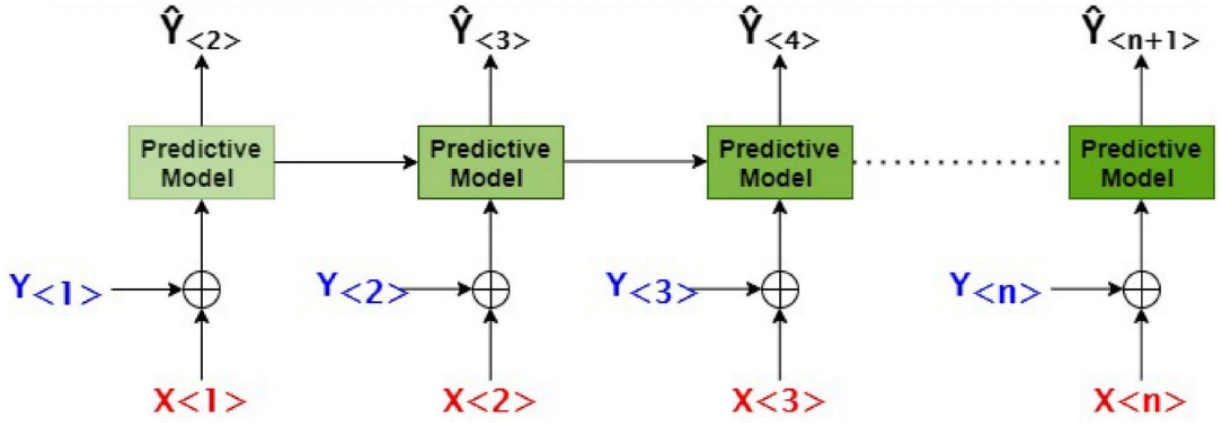


Figure 17: Block Diagram of Our Problem Formulation

8. Project Architecture

The project comprises several Deep Learning Models and the goal is to estimate the UPDRS scores for the current visit and predict the scores for any potential visits that could occur after 6, 12, and 24 months. The complete architecture of the code can be divided into several stages, which can be described as follows:

8.1 Data Loading and Preprocessing: The training data has three different .csv files. These files include clinical data, peptide data, and protein data for the patients. The clinical data is preprocessed by creating new columns representing the UPDRS scores at different time intervals (0, 6, 12, and 24 months). This is done by iterating through the unique patient IDs and combining the UPDRS scores for each event at the specified time intervals.

8.2 Feature Extraction: The model uses two different feature extraction methods from the input data, which is then used separately for the final deep learning models. One method uses peptides

as features with peptide abundance values from train_peptides.csv. The peptide abundance shows the frequency of amino acids in the sample. Another method uses Unit Protein from train_proteins.csv as features and uses its NPX values. NPX Normalized protein expression and shows the frequency of protein's occurrence in the sample.

8.3 Data Merging: The preprocessed clinical data is then merged with the protein data and peptide data separately using the 'visit_id' column as the index. The merged DataFrame is then used to create the feature matrix 'X' and target matrix 'y' for training the models. Since there is no overlap in the protein and peptide data, these features are used separately.

8.4 Data Transformation: The feature matrix 'X' is transformed using a column transformer that applies KNN imputation and standard scaling to the numerical features. This results in a transformed feature matrix 'X_transformed', which is used to train the model.

8.5 Model Architecture: The deep learning model based on Gated Recurrent Units (GRUs), Sequential Model with multiple layers, and Simple RNN are used is defined using the Keras library, which is explained in the next section of the report. The transformed feature matrix 'X_transformed' and target matrix 'y' are used to train the model with different hyperparameters to choose the best model.

8.6 Model Evaluation: The training history is plotted to visualize the model's training and validation loss, as well as the training and validation SMAPE+1 metric, which is required for evaluation in the competition. The model is evaluated using hidden test data in the Kaggle competition.

The complete architecture of the code focuses on loading and preprocessing the data, defining and training a deep learning model based on various Deep learning models, and using

the trained model to make predictions on test data. The following Figure represents the entire Project architecture.

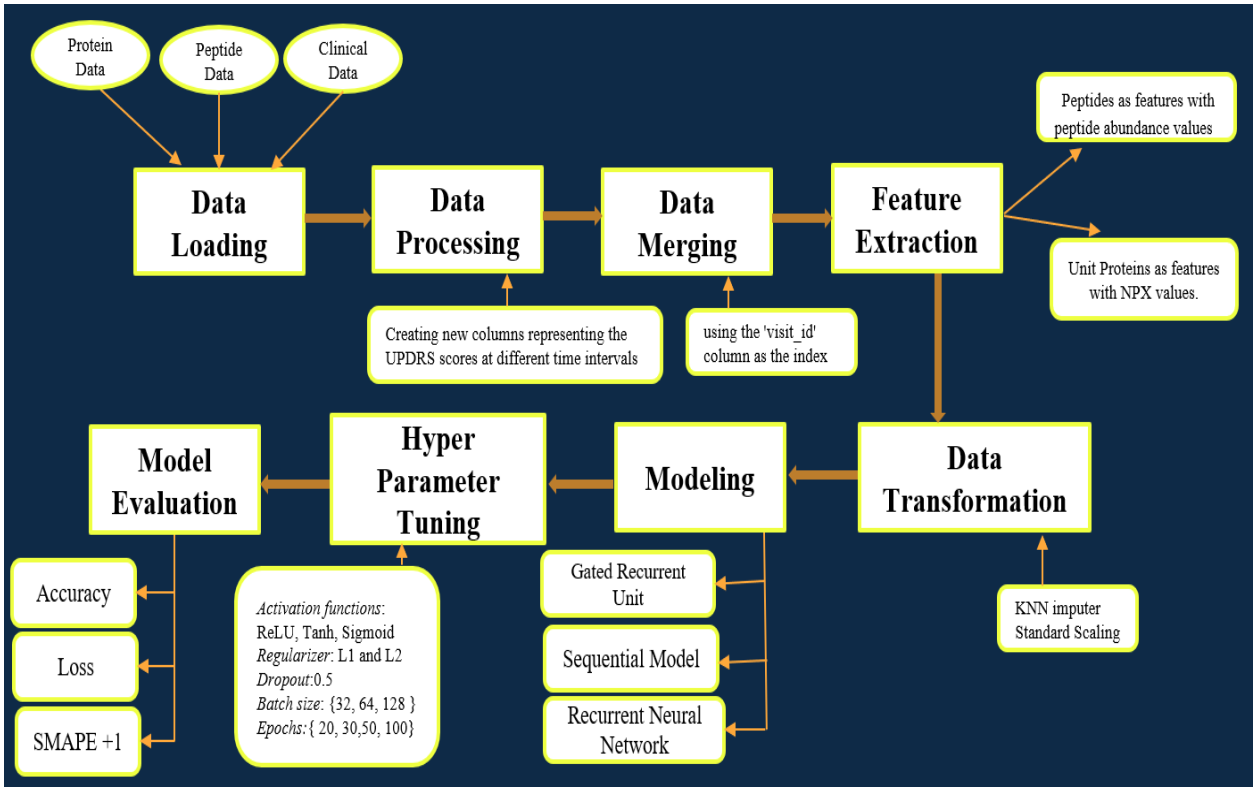


Figure 18: Project Architecture

9. Models Used and Results

This project focuses on exploring different deep learning models for predicting clinical scores in Parkinson's disease patients using two different methods of feature selection. The first method involves using peptides as features, which are derived from the train_peptides.csv file. Peptides are short chains of amino acids that are the building blocks of proteins, and the abundance of each peptide reflects the frequency of amino acids in the sample. Therefore, the peptide abundance values can be used as features to train a NN model.

The second method involves using the Unit Protein as features, which are taken from the train_proteins.csv file. Unit Protein refers to a group of proteins that are identified by a unique

identifier, and their NPX values are used as features. NPX stands for normalized protein expression which is a measure of the relative abundance of a protein in a sample. Therefore, the NPX values of Unit Proteins can be used as features to train a deep learning model.

By using these two different methods of feature selection, the project aims to compare the performance of various deep learning models and identify the most effective approach for predicting clinical scores in Parkinson's disease patients.

9.1 GRU (Gated Recurrent Unit)

The GRU model is a type of recurrent neural network (RNN) architecture that was introduced in 2014 by Kyunghyun Cho. It was designed to address some of the issues with the LSTM model, such as the vanishing gradient problem and the computational cost. Similar to the LSTM model, the GRU model has a memory cell, which stores information over time, but it uses a simpler architecture to achieve this.

The GRU model has two gates: the update gate and the reset gate. The update gate controls how much of the previous memory cell state is passed on to the next time step, while the reset gate controls how much of the new input is used to update the memory cell state. These gates allow the GRU model to selectively forget or remember information from previous time steps, allowing it to better capture long-term dependencies in sequential data. The below figure shows the architecture of GRU.

The sequential model with Gated Recurrent Units (GRUs) demonstrates its potential for predicting the progression of Parkinson's disease. The model's ability to capture temporal dependencies in the data can provide valuable insights for healthcare professionals and patients in managing the condition and developing personalized treatment plans.

The GRU model is trained on a preprocessed dataset of clinical, peptide, and protein data from Parkinson's disease patients using KNN imputation and standard scaling. A custom loss function, SMAPE+1, is used to measure the symmetric mean absolute percentage error. The Adam optimizer is employed during the compilation phase. The model is trained with a batch size of 64 and 50 epochs, using a validation split of 0.2 to monitor its performance on unseen data. The training history is plotted to visualize the training and validation loss and SMAPE+1 metric.

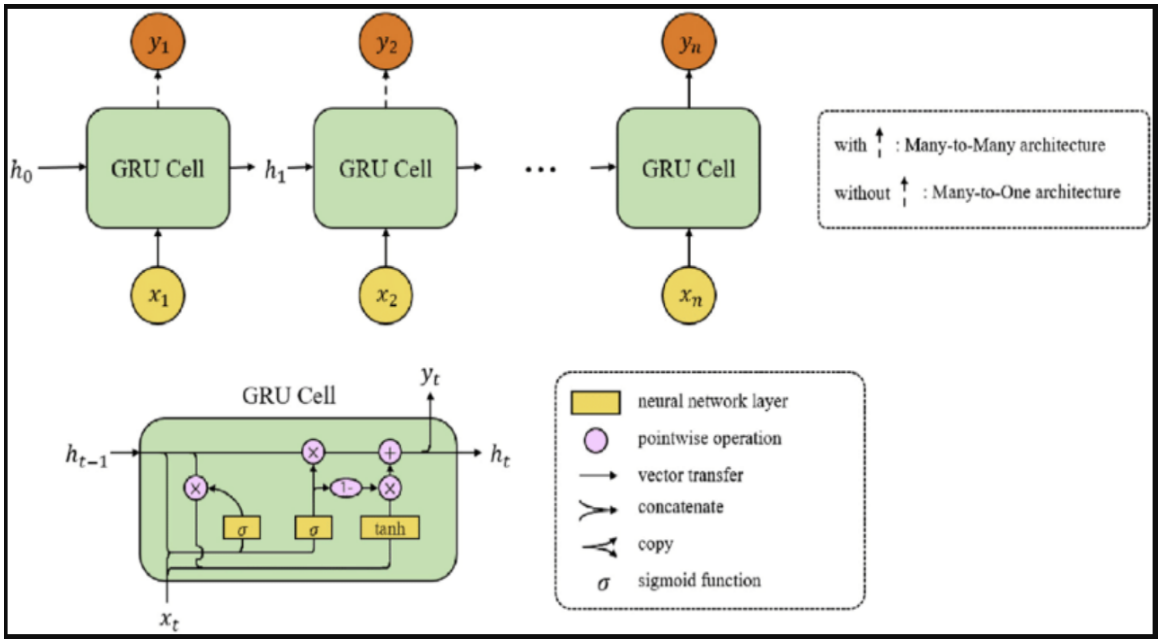


Figure 19: Architecture of GRU (Adapted from researchgate.net/publication/346346921/figure/fig3/AS:962138164170756@1606403022483/Architecture-of-GRU-based-neural-network.png)

The sequential deep learning model architecture used in this project comprises three Gated Recurrent Unit (GRU) layers, a type of recurrent neural network (RNN) that captures temporal dependencies in the sequential data. The first GRU layer has 128 units, a ReLU activation function, and returns sequences for the next layer to process. Regularization is added

using a dropout layer with a rate of 0.15 to prevent overfitting. The second GRU layer has 64 units, ReLU activation function, and applies L2 regularization on kernel weights. Another dropout layer is added using the same rate. The third GRU layer has 32 units, a ReLU activation function, and applies L2 regularization but does not return sequences. A final dropout layer is included, and the output layer has a number of units equal to the target variables, using a linear activation function to generate the final predictions.

9.2 RNN (Recurrent Neural Network)

RNN, which stands for Recurrent Neural Network, is a type of neural network architecture that is commonly used for sequential or time-series data processing. In this model, the input data is fed into the network in a sequential order, and the network processes it one step at a time, maintaining a memory of previous steps. The memory is updated with each new input, allowing the network to capture the dependencies between the different elements in the sequence.

The Sequential DL Model with RNN used in the project consists of multiple layers of RNN cells, which are connected to each other in a sequence. Each cell has a set of weights that are updated during the training process to learn the patterns in the data. Dropout layers are often added for regularization to prevent overfitting. The model architecture can be customized by adjusting the number of layers, the number of cells in each layer, the activation functions, the dropout rates, and other hyperparameters. The goal of the model is to learn a mapping between the input sequence and the output sequence, which could be, for example, predicting future values in a time series or classifying text data. The below figure shows the architecture of traditional RNN that allows the output in the previous step to be used as inputs along with the hidden states.

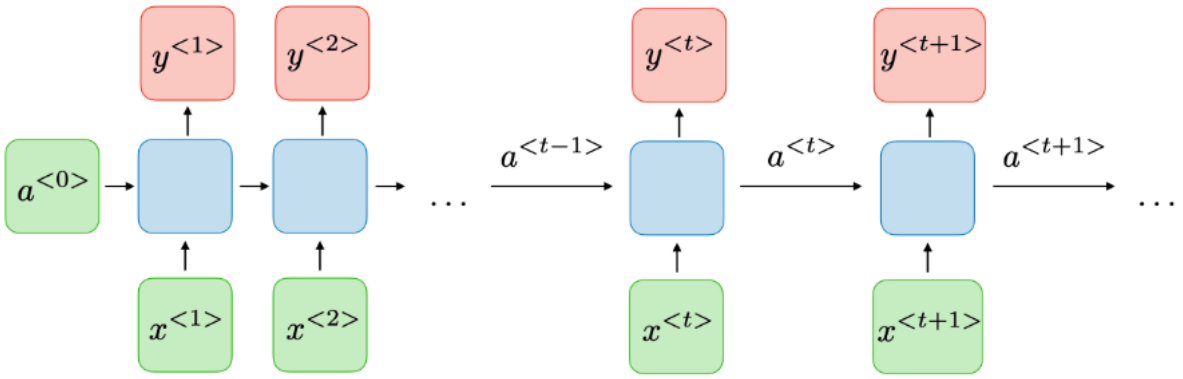


Figure 20. Architecture of Traditional RNN (Adapted from https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks)

Here, for this project sequential architecture with SimpleRNN layers has also been used for predicting Parkinson's disease progression. The model is built using the Sequential API, consisting of three SimpleRNN layers with 128, 64, and 32 units, respectively, and ReLU activation functions. The first two RNN layers return sequences, allowing the following layers to process their output, and use L2 regularization for preventing overfitting. Dropout layers with a 0.15 dropout rate are added after each RNN layer for further regularization. The final Dense layer with a linear activation function produces the output predictions. The model is compiled using the Adam optimizer and a custom SMAPE+1 loss function as the objective function and evaluation metric. The input data is reshaped to fit the expected input shape for the SimpleRNN layers, and the model is trained for 50 epochs with a batch size of 64 and a 20% validation split.

9.3 Sequential Dense Neural Network

A sequential Dense NN is a type of artificial neural network that is commonly used for classification or regression tasks. Here, the layers are stacked one after the other, forming a sequence that maps the input to the output. The input is passed through the first layer, and then the output is passed to the next layer, and so on until the final layer produces the output.

The fully connected layers, also known as dense layers, are responsible for learning the complex relationships between the input features and the target variable. Each neuron in a dense layer is connected to every neuron in the previous layer, allowing the model to learn complex, non-linear relationships between the input and the output.

The architecture of a sequential DL model with fully connected layers typically consists of an input layer, one or more hidden layers, and an output layer. The number of neurons in each layer, the activation function used, and the number of layers can be varied depending on the specific task and the complexity of the data. Training a sequential DL model with fully connected layers involves iteratively updating the weights of the neurons to minimize the difference between the predicted output and the true output. This is typically done using an optimization algorithm such as stochastic gradient descent (SGD), which adjusts the weights in the direction that minimizes the loss function. The below figure shows the basic architecture of a dense neural network.

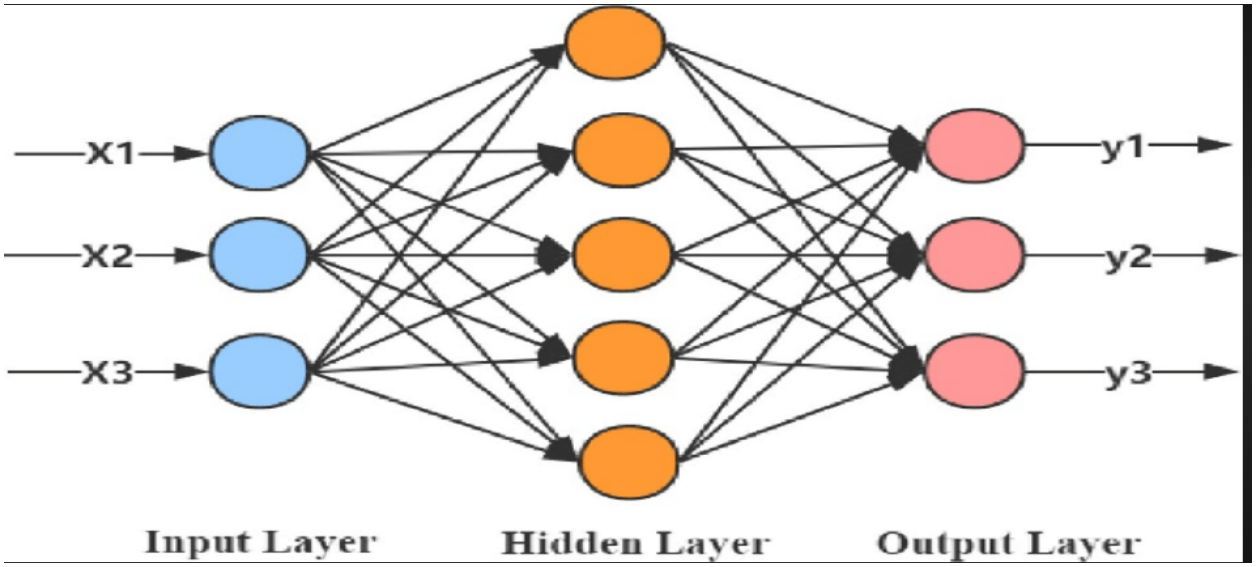


Figure 21. Basic architecture of Deep NN (Adapted from https://www.researchgate.net/publication/335592380_Mechanical_Fault_Diagnosis_and_Prediction_in_IoT_Based_on_Multi-source_Sensing_Data_Fusion/figures?lo=1)

A Neural Network model using a sequential architecture with fully connected Dense layers has also been used for predicting Parkinson's disease progression. The model is constructed using the Sequential API and is composed of four Dense layers with 256, 128, 64, and 32 units, respectively, and ReLU activation functions. The first layer specifies the input shape based on the input data's features. L2 regularization is applied to the Dense layers with 128, 64, and 32 units to prevent overfitting. Dropout layers with a 0.20 dropout rate are included after each Dense layer for further regularization. The final Dense layer with a linear activation function generates the output predictions.

The model is compiled using the Stochastic Gradient Descent (SGD) optimizer and the custom SMAPE loss function as the objective function and evaluation metric. The model is trained for 500 epochs with a batch size of 32 and a 20% validation split.

10. Hyperparameter Tuning

In the process of training a neural network, it is important to experiment with different parameters in order to optimize the performance of the model. This is often done through a trial-and-error process, where different combinations of parameters are tested and evaluated to determine which ones produce the best results.

In this project, various parameters were explored in order to optimize the different neural networks used. These parameters included different activation functions such as ReLu, sigmoid, and tanh, which are used to introduce non-linearity to the model; regularizers such as l1 and l2, which help to prevent overfitting by adding a penalty term to the loss function; different optimizers, which are algorithms used to update the weights of the model during training; different layers and dropouts in the model, which can help to improve the generalization of the

model; and different batch sizes and epochs, which are used to control the amount of data used to update the weights of the model during each iteration of training.

By experimenting with these different parameters, the goal was to find the best combination that would optimize the performance of the model, as measured by the SMAPE evaluation. This iterative process of testing and evaluating different parameters is a crucial step in the development of any neural network model, as it allows for the identification of the optimal configuration of parameters that will produce the best results.

The results of all the models have been discussed in the results section.

11. Evaluation

The goal of the competition is to predict UPDRS scores for patients with Parkinson's disease at different time points based on protein/peptide samples. The accuracy of the predictions will be evaluated using the Symmetric Mean Absolute Percentage Error (SMAPE) metric.

The SMAPE metric measures the percentage difference between the actual and predicted values, and is calculated as follows:

$$SMAPE = (1/n) * \sum(|Ft - At| / ((|Ft| + |At|)/2)) * 100 \quad (2)$$

where Ft is the forecast/predicted value, At is the actual value, and n is the number of forecast/predicted values. If both the actual and predicted values are 0, then SMAPE is defined to be 0.

For each patient visit where a protein/peptide sample was taken, the goal is to estimate the patient's UPDRS scores for that visit, as well as predict their scores for potential visits 6, 12, and 24 months later. The predictions for any visits that didn't ultimately take place will be ignored.

To ensure fairness and prevent time leakage, Kaggle requires to use the provided Python time-series API for submitting the solutions. The API ensures that models do not access any future data during the training or prediction phases. Below is the snippet of the code which is intended to be used as a starting point for making predictions on the AMP-PD Peptide dataset and submitting them to the competition. The actual prediction method will depend on the specific model being used, but the overall structure of the code remains similar.

```
import amp_pd_peptide
env = amp_pd_peptide.make_env()   # initialize the environment
iter_test = env.iter_test()    # an iterator which loops over the test files
for (test, test_peptides, test_proteins, sample_submission) in iter_test:
    sample_prediction_df['rating'] = np.arange(len(sample_prediction))  # make your
predictions here
    env.predict(sample_prediction_df)   # register your predictions
```

Figure 22. Submission API

12. Result analysis and Visualization

After training the deep learning model on the preprocessed data, two plots are generated after training: one for the model loss (training and validation) and another for the SMAPE+1 metric (training and validation). These plots help visualize the model's performance and convergence during training. The x-axis represents the number of epochs, and the y-axis represents the loss or SMAPE+1 values. The model loss plot shows the training and validation losses over the number of epochs (i.e., training iterations). The training loss represents the error between the model's predicted values and the actual values of the training data, while the validation loss represents the error between the predicted values and actual values of the validation data, which is a subset of the training data.

The SMAPE+1 metric plot also shows the training and validation performance of the model. SMAPE+1 is a custom evaluation metric used in the Parkinson's disease prediction

competition on Kaggle. It is used to measure the accuracy of the predictions made by the model. The SMAPE+1 metric takes into account the relative error between the predicted and actual UPDRS scores and the clinical relevance of the predictions by weighing the error differently based on the magnitude of the UPDRS score. SMAPE+1 is a symmetric mean absolute percentage error that gives more weight to larger errors. The training SMAPE+1 represents the average error of the model on the training data, while the validation SMAPE+1 represents the average error of the model on the validation data. The goal is to minimize the SMAPE+1 score, which indicates higher prediction accuracy. The plot helps visualize how well the model is performing during training and if there is overfitting or underfitting. The SMAPE+1 metric plot shows the training and validation SMAPE+1 values over the number of epochs.

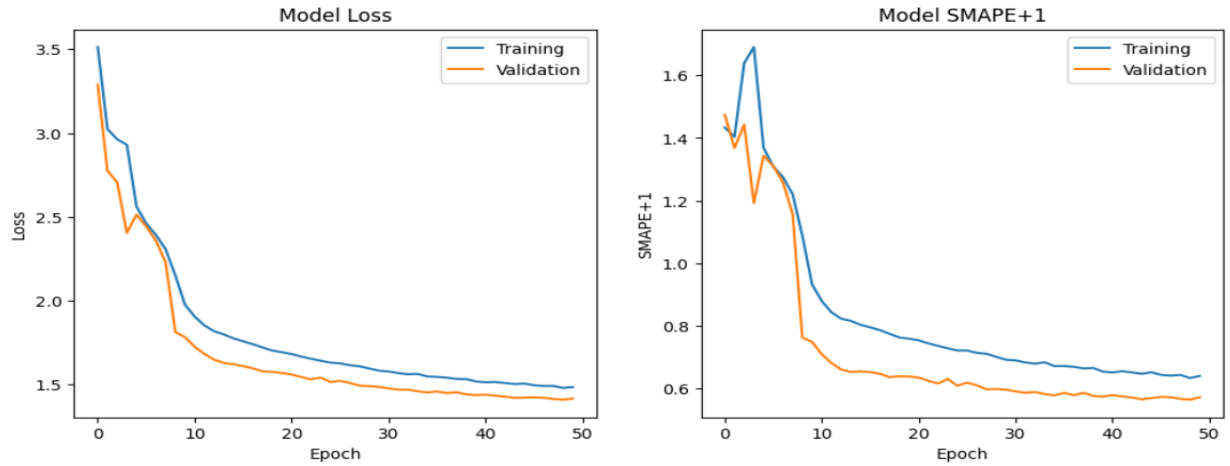
By observing these plots, we can determine if the model is overfitting or underfitting. If the training loss is significantly lower than the validation loss, the model may be overfitting, which means it is not generalizing well to new data. If the training loss and validation loss are both high, the model may be underfitting, which means it is not capturing the patterns in the data. Similarly, if the training SMAPE+1 is much lower than the validation SMAPE+1, the model may be overfitting, and if both the training and validation SMAPE+1 values are high, the model may be underfitting.

Overall, these plots help to visualize the model's performance and convergence during training, which can be used to adjust the model's hyperparameters, such as the learning rate or number of layers, to improve its performance. The following figures represent the result visualisations for our models using Peptides and Unitprot as individual features for different Deep Learning Models. Among all the models, using Unit Protein with GRU as feature gave us the best score on kaggle with training loss of .66 and validation loss of .55. The model was

trained GRU layers with 128, 64, and 32 units, respectively, and dropout layers with a rate of 0.15 after each GRU layer to reduce overfitting. An L2 regularization with a coefficient of 0.01 is applied to the second and third GRU layers. The output layer is a dense layer with linear activation. The model is compiled using the 'adam' optimizer, with a custom SMAPE+1 loss function and metric required to be used in the kaggle competition. The rest of the models and features used have been shown in below figures.

12.1 Using Peptide as features:

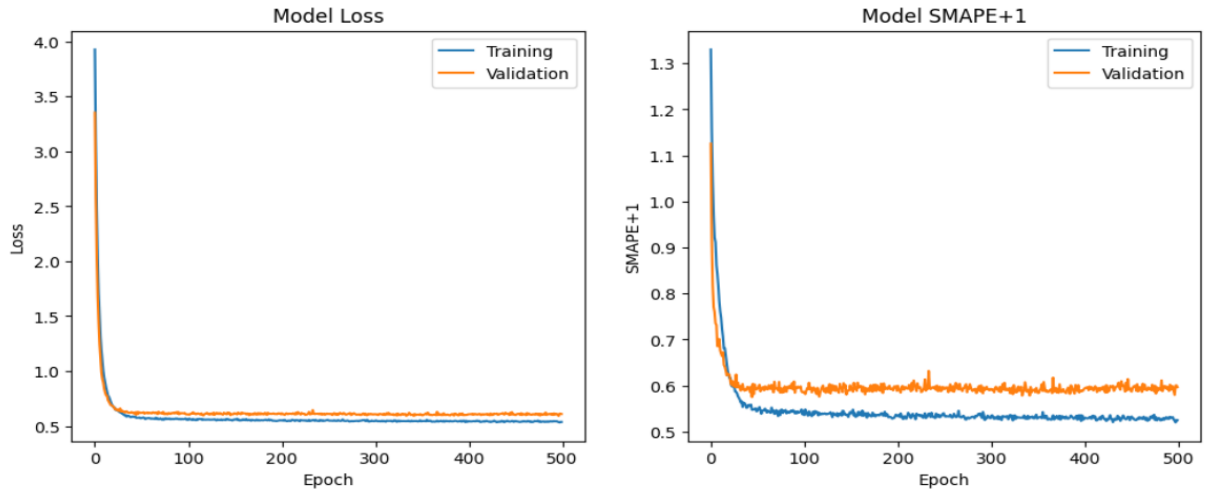
1. **GRU**




```
Final training loss: 1.4853540658950806
```

Figure 22

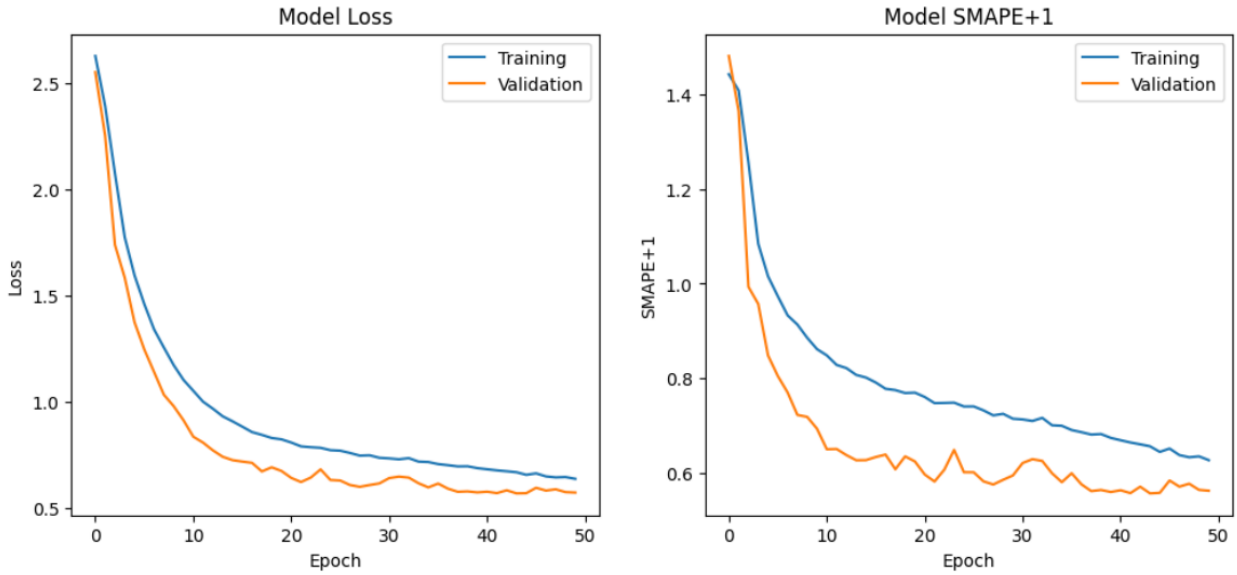
2. **Deep NN Model**




```
Final training loss: 0.5371091961860657
Final validation loss: 0.609063982963562
```

Figure 23

3. RNN

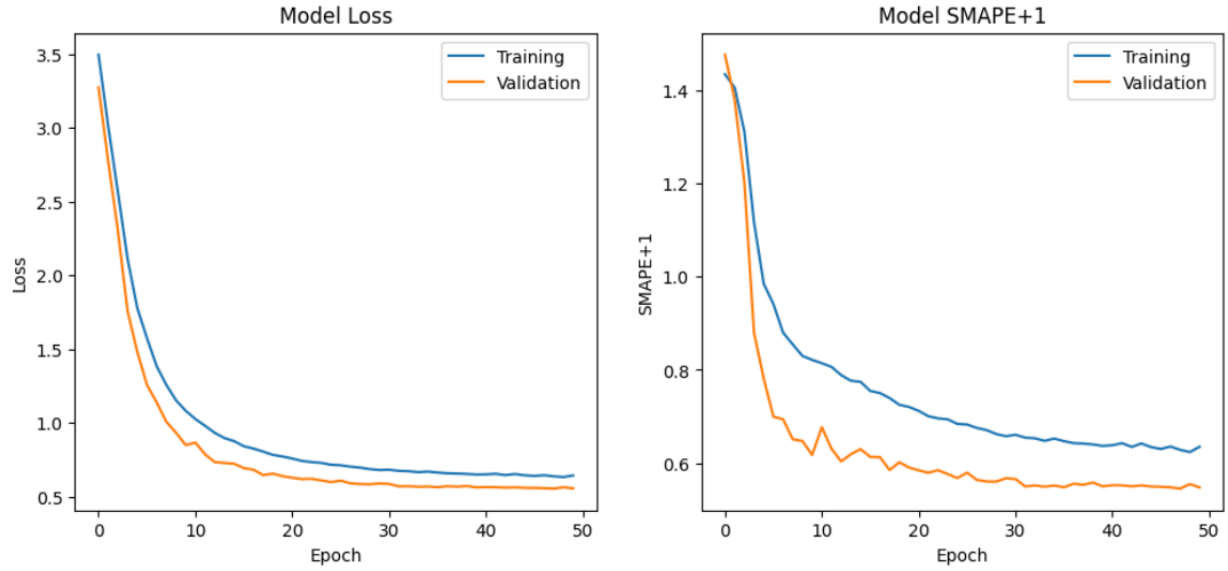



```
Final training loss: 0.6354522109031677
Final validation loss: 0.5708218812942505
```

Figure 24

12.2 Using UnitProt as features

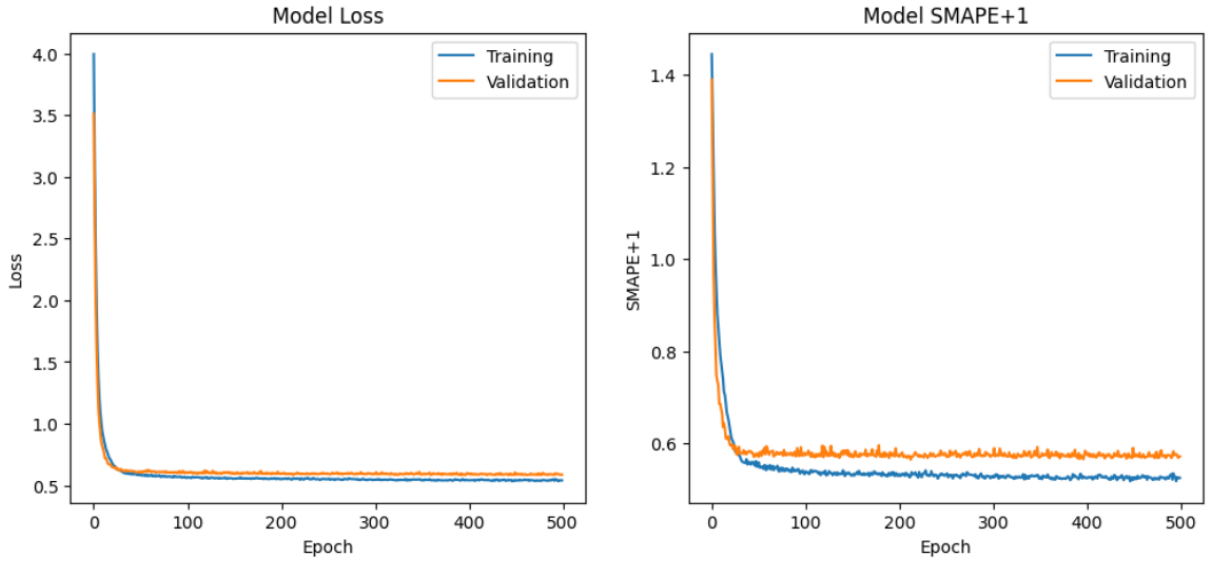
1. GRU




```
Final training loss: 0.6427332162857056
Final validation loss: 0.5552383065223694
```

Figure 25

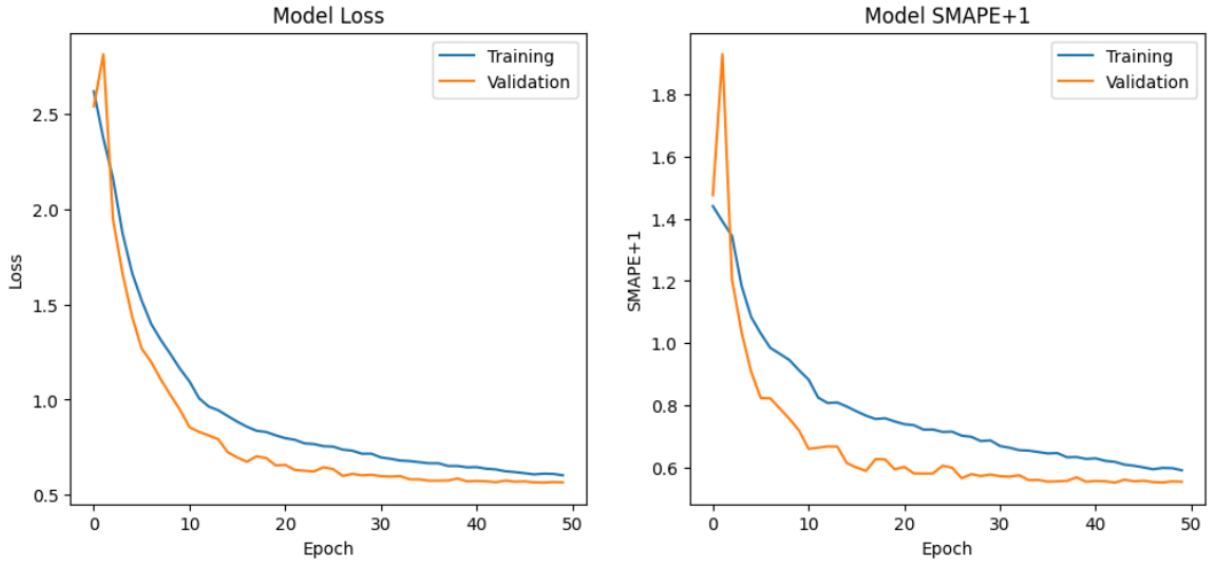
2. Deep NN model




```
Final training loss: 0.539899468421936
Final validation loss: 0.5870407223701477
```

Figure 26

3. RNN




```
Final training loss: 0.6029387712478638
Final validation loss: 0.565978467464447
```

Figure 27

13. Ranking

As a participant in the Kaggle competition, our team is currently ranked at 258 out of 1677 teams, which puts us in the top 16 percent of all teams. While we are proud of our current

rank, we know that there is still room for improvement. We are continuously working to refine our models and explore new approaches to improve our performance in the competition. Being in the top 16 percent is a great achievement, but we are determined to push ourselves further and aim for even higher rankings.

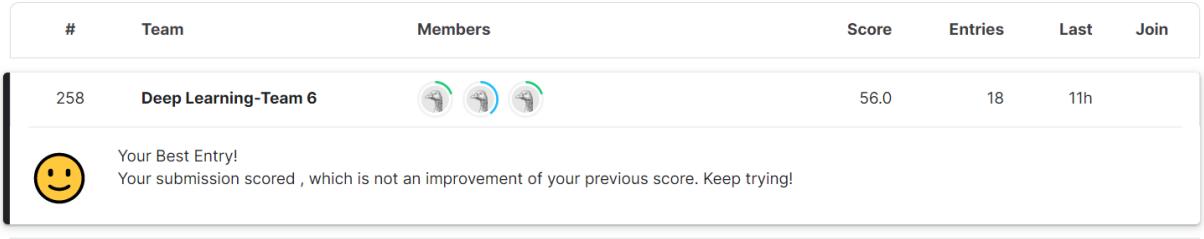


Figure 28: Screenshot of the Kaggle competition ranking

14. Conclusion

In conclusion, the project aims to predict the MDS-UPDRS score for the next hospital visit of a patient using protein abundance data from several earlier visits. The suggested model uses temporal correlation to estimate the disease condition in the near future by analyzing the historical protein abundance sequence data. The proposed deep learning models, based on Gated Recurrent Units (GRUs), Sequential Model, and Simple RNN, are used to predict UPDRS scores for the current visit and future visits that could occur after 6, 12, and 24 months. The project architecture involves several stages, including data loading and preprocessing, feature extraction, data merging, data transformation, model architecture, and model evaluation. The final model is evaluated using hidden test data in the Kaggle competition. Overall, the project has the potential to help physicians make more informed decisions about patient care by predicting the disease progression of Parkinson's patients. This project has significant implications in the field of Parkinson's disease research and clinical care. Parkinson's disease is a chronic and progressive

neurological disorder that affects millions of people worldwide. Accurate and timely prediction of a patient's disease progression can help clinicians optimize treatment plans and improve patient outcomes. The proposed model can potentially aid in the development of personalized treatment plans for Parkinson's disease patients.

Furthermore, the proposed model can also be adapted and applied to other diseases where protein abundance data can be used for disease progression prediction. The project's architecture can serve as a framework for similar studies that involve the use of deep learning models for disease progression prediction. In summary, this project provides a promising approach to predicting Parkinson's disease progression using protein abundance data and deep learning models. The proposed model can potentially have a significant impact on improving patient care and developing personalized treatment plans for Parkinson's disease patients.

15. Future Scope

This project can be expanded in a number of different ways in the future. First, the accuracy of the UPDRS score predictions can be enhanced by including more types of data in the current model, such as genetic or imaging data. Second, the model can be altered to forecast the likelihood that a patient's disease would worsen based on their present UPDRS scores and other pertinent details like age and drug usage. Thirdly, a bigger patient sample can be used to test the model's performance in clinical situations and increase its generalizability. To enhance the performance of the model, the usage of transfer learning and ensembling approaches can be investigated. The model can also be included into clinical decision support systems to help doctors decide what treatments to provide Parkinson's disease patients. The creation of a better understandable model that might shed light on the underlying biological pathways involved in Parkinson's disease is another area that may benefit from improvement. This might entail the

application of explainable AI approaches or the creation of a hybrid model that fuses deep learning with conventional statistical techniques. In order to create a more thorough strategy for anticipating disease development and patient outcomes, the project can be expanded to include other neurodegenerative illnesses like Alzheimer's disease or Huntington's disease.

16. Project Management

Our project's work breakdown structure has been defined using the agile project management methodology. We utilized the Trello project management tool to manage our project.

Trello is a versatile project management tool that utilizes a kanban approach and is suitable for various teams and projects. It enables effective collaboration and task management. Trello provides a user-friendly interface with boards, lists, and cards to help users visually manage and arrange their tasks and workflows. It is flexible and adaptable to meet the specific needs of teams or individuals. The Trello tool enables the creation of cards to represent particular tasks or concepts, which can be shifted across various lists to show advancement or state. The lists can be tailored to align with the project's workflow, and can incorporate categories like "To Do", "Doing", and "Done". Apart from its fundamental features, Trello also provides several beneficial functions like due dates, labels, checklists, comments, and attachments. The Figure 29 displayed below illustrates a screenshot of Trello.

16. 1 Gantt chart

The Gantt chart for our project was created using TeamGantt. TeamGantt is a popular project management solution that emphasizes visual project planning and scheduling. Its main feature is an interactive Gantt chart that enables users to map out their project timeline, assign tasks to team members, and monitor progress in real-time. TeamGantt also offers a variety of

additional features such as task dependencies, milestone setting, and timeline customization options. Its user-friendly interface and comprehensive toolset make it a top choice for teams seeking to simplify their project management processes. The Gantt chart is depicted in Figure 30 below.

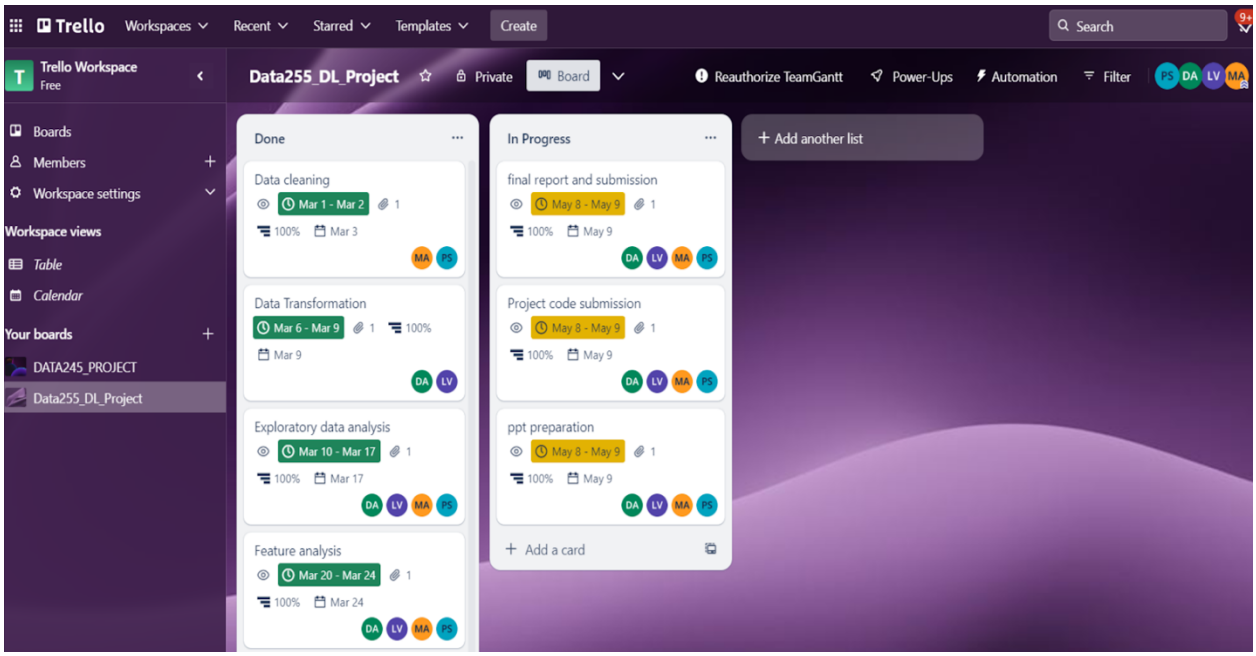


Figure 29: Screenshot of Trello workspace

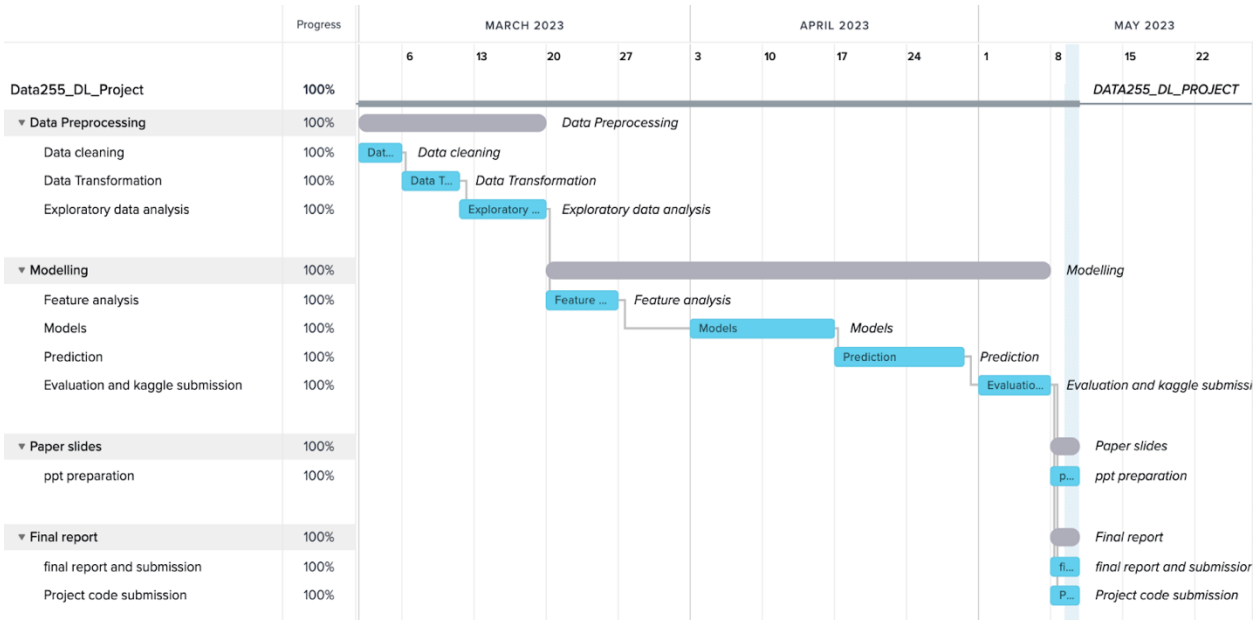


Figure 30: Screenshot of Gantt chart

References

Ahmed, S., Komeili, M., & Park, J. (2022). Predictive modelling of Parkinson's disease progression based on RNA-Sequence with densely connected deep recurrent neural networks. *Scientific Reports*, *12*(1), 21469. https://doi.org/10.1038/s41598-022-25454-1

Aşuroğlu, T., & Oğul, H. (2022). A deep learning approach for parkinson's disease severity assessment. *Health and Technology, 12(5),* 943–953. https://doi.org/10.1007/s12553-022-00698-z

Berg D, Lang AE, Postuma RB, Maetzler W, Deuschl G, Gasser T, Siderowf A, Schapira AH, Oertel W, Obeso JA, Olanow CW, Poewe W, Stern M. Changing the research criteria for the diagnosis of Parkinson's disease: obstacles and opportunities. Lancet Neurol. 2013 May;12(5):514-24. doi: 10.1016/S1474-4422(13)70047-4. Epub 2013 Apr 11. PMID: 23582175.

Dragana Miljkovic et al, "Machine Learning and Data Mining Methods for Managing Parkinson's Disease" LNAI 9605, pp. 209-220, 2016.

Grover, S., Bhartia, S., Akshama, Yadav, A., & K.R., S. (2018). Predicting Severity Of Parkinson's Disease Using Deep Learning. *Procedia Computer Science*, *132*, 1788–1794. https://doi.org/10.1016/j.procs.2018.05.154

Jankovic, J. (2008). Parkinson's disease: clinical features and diagnosis. *J. Neurol. Neurosurg. Psychiatry* 79, 368–376. doi: 10.1136/jnnp.2007.131045

Litvan I, Goldman JG, Tröster AI, Schmand BA, Weintraub D, Petersen RC, Mollenhauer B, Adler CH, Marder K, Williams-Gray CH, Aarsland D, Kulisevsky J, Rodriguez-Oroz MC, Burn DJ, Barker RA, Emre M. Diagnostic criteria for mild cognitive impairment in

Parkinson's disease: Movement Disorder Society Task Force guidelines. Mov Disord. 2012 Mar;27(3):349-56. doi: 10.1002/mds.24893. Epub 2012 Jan 24. PMID: 22275317; PMCID: PMC3641655.

Lones, M. A., Alty, J. E., Cosgrove, J., Duggan-Carter, P., Jamieson, S., Naylor, R. F., Turner, A. J., & Smith, S. L. (2017). A New Evolutionary Algorithm-Based Home Monitoring Device for Parkinson's Dyskinesia. *Journal of medical systems*, *41*(11), 176. [176]. https://doi.org/10.1007/s10916-017-0811-7

Meireles J, Massano J. Cognitive impairment and dementia in Parkinson's disease: clinical features, diagnosis, and management. Front Neurol. 2012 May 25;3:88. doi: 10.3389/fneur.2012.00088. PMID: 22654785; PMCID: PMC3360424.

Nissar, Iqra & Rizvi, Danish & Masood, Sarfaraz & Mir, Aqib. (2018). Voice-Based Detection of Parkinson's Disease through Ensemble Machine Learning Approach: A Performance Study. EAI Endorsed Transactions on Pervasive Health and Technology. 5. 162806. 10.4108/eai.13-7-2018.162806.

Parkinson's Disease Foundation, 2015. Available at: http://parkinson.org/UnderstandingParkinsons/Causes-and-Statistics/Statistics [Accessed on 11th April 2018].

Postuma, R. B., Berg, D., Stern, M., Poewe, W., Olanow, C. W., Oertel, W., et al. (2015). MDS clinical diagnostic criteria for Parkinson's disease. *Move. Disord.* 30, 1591–1601. doi: 10.1002/mds.26424

Rustempasic, Indira & Can, Mehmet. (2013). Diagnosis of Parkinson's Disease using Fuzzy C-Means Clustering and Pattern Recognition. SOUTHEAST EUROPE JOURNAL OF SOFT COMPUTING. 2. 10.21533/scjournal.v2i1.44.

Saad A, Zaarour I, Zeinedine A, Ayache M, Bejjani P, Guerin F, Havre-France L. A preliminary study of the causality of freezing of gait for Parkinson's disease patients: Bayesian belief network approach. International Journal of Computer Science Issues 2013; 10(3): 88-95.

Sakar, C. O., Serbes, G., Gunduz, A., Tunc, H. C., Nizam, H., Sakar, B. E., Tutuncu, M., Aydin, T., Isenkul, M. E., & Apaydin, H. (2019). A comparative analysis of speech signal processing algorithms for Parkinson's disease classification and the use of the tunable Q-factor wavelet transform. *Applied Soft Computing Journal*, *74*, 255-263. https://doi.org/10.1016/j.asoc.2018.10.022

Samii A, Nutt JG, Ransom BR. Parkinson's disease. Lancet. 2004 May 29;363(9423):1783-93. doi: 10.1016/S0140-6736(04)16305-8 PMID: 15172778.

Tsanas, M. A. Little, P. E. McSharry, J. Spielman and L. O. Ramig, "Novel Speech Signal Processing Algorithms for High-Accuracy Classification of Parkinson's Disease," in IEEE Transactions on Biomedical Engineering, vol. 59, no. 5, pp. 1264-1271, May 2012, doi: 10.1109/TBME.2012.2183367.

T. Swapna, Y. Sravani Devi, "Performance Analysis of Classification algorithms on Parkinson's Dataset with Voice Attributes". International Journal of Applied Engineering Research ISSN 0973-4562 Volume 14, Number 2 pp. 452-458, 2019

Vallejo, Marta & Jamieson, Stuart & Cosgrove, Jeremy & Smith, Stephen & Lones, Michael & Alty, Jane & Corne, David. (2016). Exploring Diagnostic Models of Parkinson's Disease with Multi-Objective Regression. 10.1109/SSCI.2016.7849884.

Zesiewicz, T. A., Sullivan, K. L., and Hauser, R. A. (2006). Nonmotor symptoms of Parkinson's disease. *Expert Rev. Neurother.* 6, 1811–1822. doi: 10.1586/14737175.6.12.1811

Zineddine, M. (1 C.E.). A Novel Approach to Parkinson's Disease Progression Evaluation Using Convolutional Neural Networks. *Https://Services.igi-Global.com/Resolvedoi/Resolve.aspx?Doi=10.4018/IJSI.315655*. https://www.igi-global.com/gateway/article/full-text-html/315655&riu=true