EEG Signals based Classification Model for Parkinson's disease using Fuzzy Classifiers

Name : Shatarupa Bepari,
Roll Number : 22210042

Indian Institute of Information Technology Gandhinagar
beparishatarupa@iitgn.ac.in

1 Abstract

Parkinson's disease is a degenerative neurological disorder that affects millions of people worldwide. Early diagnosis and treatment can greatly improve the quality of life for patients, but current diagnostic methods are often invasive, expensive, and not always accurate. In this study, we explore the use of electroencephalography (EEG) signals as a non-invasive and potentially more accurate method for diagnosing Parkinson's disease. We collected EEG data from healthy individuals and patients with Parkinson's disease and used feature extraction techniques to identify patterns in the data that could distinguish between the two groups. We then used a Fuzzy-Rough Nearest Neighbor (FRNN) classifier to classify new EEG signals as either healthy or Parkinson's disease. Overall, our study demonstrates the potential of using machine learning techniques and EEG signals to improve the diagnosis and treatment of Parkinson's disease.

2 Introduction

Parkinson's disease (PD) is a progressive disorder that affects the nervous system and the parts of the body controlled by the nerves and the symptoms start very slowly. The first symptoms may be barely noticeable tremor in just one hand. Early detection of (PD) is very important in clinical diagnosis for preventing disease development in future. The paper presents efficient discrete wavelet transform (DWT)-based methods for detecting PD from health control (HC) in two cases, namely, off- and on-medication. The EEG signals are pre-processed to remove major artifacts before being decomposed into several EEG sub-bands (approximate and details) using discrete wavelet transform. The features are then extracted from the wavelet packet-derived reconstructed signals using different entropy measures, namely, log energy entropy, Shannon entropy, threshold entropy, sure entropy, and norm entropy. Then KNN machine learning techniques are investigated to classify the resulting PD/HC features.The aim of the present study is to present uncomplicated feature extraction and classification methods while maintaining high classification accuracy and validating them using two open-source datasets (the UNM and SanDiego datasets).With the SanDiego dataset, the classification results of off-medication PD versus HC are 99.89, 99.87, and 99.91 for accuracy, sensitivity, and specificity, respectively, using the combination of DWT+TShEn and

KNN classifier. Using the same combination, the results of on-medication PD versus HC are 94.21, 93.33, and 95 percentage respectively.

2.1 Current Hypothesis

In previous works, several machine learning-based strategies like Decision Tress, Random Forest, K-nearest Neighbor (KNN) were introduced to investigate and interpret EEG signals for the purpose of their accurate classification.

In our study we propose to use Fuzzy Rough Nearest Neighbor (FRNN) machine learning model which is proposed as much better model to give more efficient results. A combination of DWT+TShEn is used to extract the features before training the model. Therefore, using Fuzzy classifiers to check if this can give the highest classification accuracy scores, with improved sensitivity and specificity percentages over traditional methods machine learning methods. Performance will be evaluated on the same metrics to compare both the models.

Please refer to the below image for a detailed method (see Fig. 5).

2.2 Dataset Used

The public dataset used to verify the proposed methods: The SanDiego dataset (31 subjects, 93 min) https://openneuro.org/datasets/ds002778/versions/1.0.2

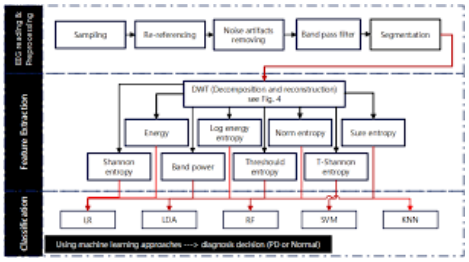

Fig. 1. Block diagram of the proposed PD DWT-based classification methods to show diagonisis of PD or Normal.

3 Methods

3.1 Data Description

In this study we intend to use an open-source EEG dataset is used to test the proposed approaches. The University of San Diego in California provided the dataset, for simplicity, we will refer this dataset as SanDiego Datasets. The data has been recorded from two groups of individuals. The first group contains EEGs of 16 healthy individuals, while the second group contains EEGs of 15 PD patients, taken both on and off medication. The right-handedness, gender, age, and cognition of the PD patients were remarkably similar to those of the HC, as evaluated by the Mini-Mental State Exam (MMSE) and the North American Adult Reading Test (NAART). The healthy subjects only volunteered once. At a sampling frequency of 512 Hz, EEG data was captured for at least 3 min in a 32-channel Biosemi active EEG system. More details can be found on Openneuo [4].

3.2 Data Pre-processing

The EEG signals are split into segments with a size of 'ch×T', where

'ch' is the number of channels and 'T' is the segment length in seconds. The choice of the segmentation time interval 'T' will be selected based on the length of recording of each dataset.

A detailed pre-processing steps have been discussed in the below points:

1. The EEG raw data is loaded and filtered to remove unwanted frequencies i.e., low frequency is set to 0.5 Hz and high frequency to 50 HZ. This is done using the mne.filter() function in MNE-Python. And notch filtration has also been done.

2. Then the EEG data is referenced to a common reference electrode or re-referenced to an average. This is done using mne.set-eeg-reference() function.

3. Epochs are created from the raw EEG data using a fixed-length sliding window approach of duration 1.0 seconds across the entire duration of the raw data. Overlapped data issues are also removed.

4.Artifacts like eye blinks and movements, muscular activity, electrical noise, and other noise artefacts are also removed using Independent Component Analysis or ICA, mne.preprocessing.ICA, in this case 20 components were used.

5. The filtered data is then concatenated and has been converted to a pandas dataframe to our further with our future steps.

Please refer to the code snippet below. It shows a python code that is implemented to pre-process the EEG data.

```
%%capture
raw_list = []
for raw_data in raw_data:
    raw = mne.io.read_raw_bdf(raw_data, preload=True)
    raw.set_eeg_reference()
    raw.filter(l_freq=0.5, h_freq =50)
    events = mne.make_fixed_length_events(raw, duration=1.0)
    epochs = mne.Epochs(raw, events=events, tmin=0.0, tmax=1.0, baseline=None)
    ica = mne.preprocessing.ICA(n_components=20, random_state=42)
    ica.fit(raw)
    ica.apply(raw)
    raw_list.append(raw)

# concatenate the raw data from all files
raw_all = mne.concatenate_raws(raw_list)
```

Fig. 2. Code:Data Pre-Processing.

As we can see the raw EEG data before data pre-processing from (see Fig. 3) the figure it is very concentrated and overlapping at many bands. So we continued to remove the interference and noise caused by the electrodes and magnetic fields.

(Fig. 4) shows the pre-processed EEG data. It is then converted to a pandas data frame. Please refer to the GitHub repository [5] to get an incite of the entire process in details.

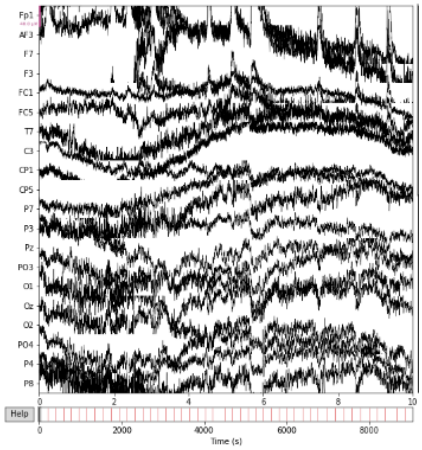

Fig. 3. EEG Data Before Pre-Processing.

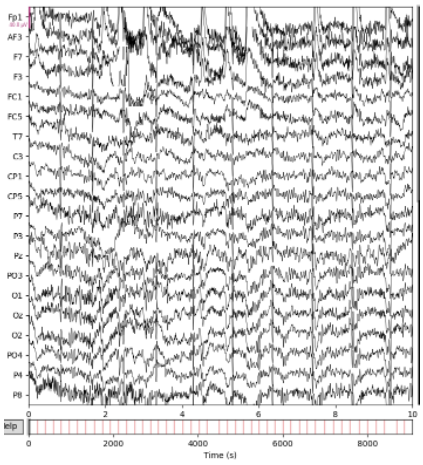

Fig. 4. EEG Data After Pre-Processing.

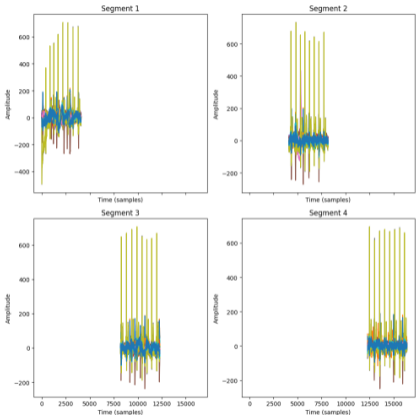

Fig. 5. EEG Data After Segmentation. This is plotted only for 2 segments.

3.3 Signal Splitting

The pre-processed EEG data is then split into smaller segments which enables the analysis of how different features of the signal change over time or in response to different stimuli. In addition, segmenting the signal can help to reduce the impact of noise or artifacts on the overall analysis, as these may be present in only a small portion of the signal.

3.4 Wavelet decomposition/reconstruction

The discrete wavelet transform (DWT) has the ability to analyze the features of a signal in the time and frequency domains by decomposing it into a number of mutually orthogonal components using a single function called the mother wavelet41. The mother wavelet used here is 'db4'. The particular choice of mother function is crucial for signal analysis. Low pass and high pass filters are used in first level decomposition to produce the signal's representation as approximation (A1) and detail (D1) coefficients. The process of decomposition and resonstruction is perfomed with the help of pywt.wavedec() and pywt.wavedrec() functions.

(Fig. 6) show a segment after decomposition is performed.

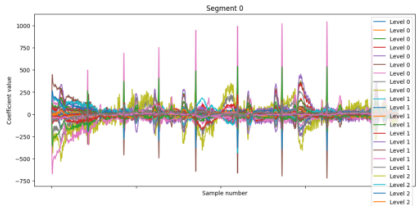

Fig. 6. EEG Data Segment After Decomposition.

The coefficients with higher frequencies have good time resolution but poor frequency resolution, while the ones with the lowest frequencies have good

frequency resolution but poor time resolution. Therefore, it proposes reconstructing wavelet packet (WP) signals from the decomposed ones (approximate and details) to increase the signals' time resolution at low frequencies. Below figure displays the constructed signals.

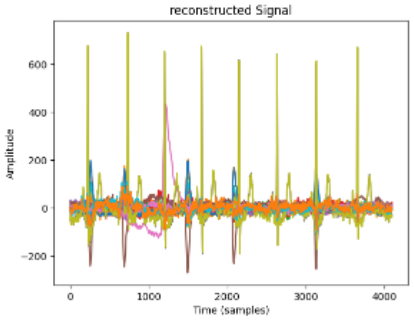

Fig. 7. EEG data segment after reconstruction from the decomposed signal.

3.5 Feature Extraction

In this study, features are extracted by measuring the amount of randomness and uncertainty through nonlinear methods such as entropy. Here, instead of directly computing entropy measures from EEG signals it is computed from each reconstructed WP signal to form the feature vectors.

Please refer to the below image for a detailed method for feature extraction from decomposed and then reconstructed EEG signals.(see Fig. 10).

This study proposes two very important features Shannon entropy (ShEn) and then constructing the Transformation-Shannon entropy(TShEn) from it. This helps to

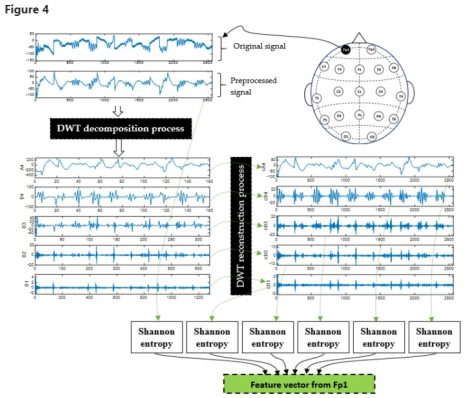

Fig. 8. Example of feature extraction using DWT and Shannon entropy.

highlight details of reconstructed signals and reduce their randomness and complexity. After the transformation process, normalized Shannon entropy is performed on the resulting signal as follows:

$$\mathbf{ShEn} = \frac{1}{k}\sum_{i=1}^{k} |x_i|^2 \log |x_i|^2 \qquad (1)$$

```
# Calculate TShEn and ShEn for each segment
tshen_features = []
shen_features = []
for segment in reconstructed_signal:
#   tshen = tkeo_shannon_entropy(segment)
    shen = nolds.sampen(segment)
    shen_features.append(shen)
    #Transform the signal values according to Equation (8)
    tstr = np.zeros_like(segment)
    tstr[segment > 1] = 1
    tstr[segment < 0] = 0
    tstr[(segment >= 0) & (segment <= 1)] = segment[(segment >= 0) & (segment <= 1)]
    tstr = (255 * tstr).astype(int)
    #Calculate transformed Shannon entropy
    tshen = nolds.sampen(tstr)
    tshen_features.append(tshen)


# Combine features into a feature matrix
features = np.column_stack((tshen_features, shen_features))
```

Fig. 9. Code: Calulation of (ShEn) + (TShEn) and extracting necessary features .

3.6 Classification and problem formulation

In this study, the performance of several classification techniques to differentiate the PD features from the HC ones: LR, LDA, RF, SVM, and KNN and to use Fuzzy Rough Nearest Neighbor (FRNN) machine learning model which is proposed as much better model to give more efficient results as we proceed. This is to compare between them and to determine which one provide the best results.

In a traditional KNN classifier, the decision boundaries between classes are hard, meaning that a data point is either classified as belonging to one class or another. Fuzzy classifiers, on the other hand, allow for more nuance in the decision-making process by using membership functions to assign a degree of belongingness to each class for each data point.

3.7 Program Code

Algorithm for Fuzzy Rough Nearest Neighbour

```
class FRNN(BaseEstimator, ClassifierMixin):
    def __init__(self, k=1, alpha_cut=0.5):
        self.k = k
        self.alpha_cut = alpha_cut

    def fit(self, X, y):
        X, y = check_X_y(X, y)
        self.classes_ = np.unique(y)
        self.classifiers_ = {}
        for c in self.classes_:
            c_index = np.where(y == c)
            X_c = X[c_index]
            fuzzy_labels = self._fuzzy_membership(X_c)
            self.classifiers_[c] = KNeighborsClassifier(n_neighbors=self.k)
            self.classifiers_[c].fit(fuzzy_labels, y[c_index])
        return self

    def predict(self, X):
        check_is_fitted(self)
        X = check_array(X)
        predictions = np.zeros((X.shape[0], len(self.classes_)))
        for i, c in enumerate(self.classes_):
            fuzzy_labels = self._fuzzy_membership(X)
            predictions[:, i] = self.classifiers_[c].predict_proba(fuzzy_labels)[:, 1]
        return self.classes_[np.argmax(predictions, axis=1)]

    def _fuzzy_membership(self, X):
        n_samples = X.shape[0]
        fuzzy_labels = np.zeros((n_samples, len(self.classes_)))
        for i, c in enumerate(self.classes_):
            class_distances = np.linalg.norm(X - np.mean(X[y==c], axis=0), axis=1)
            max_distance = np.max(class_distances)
            fuzzy_labels[:, i] = np.exp(-(((class_distances / max_distance) ** 2) / (self.alpha_cut ** 2)))
        return fuzzy_labels
```

Fig. 10. Code: Implementation of FRNN Classifier.

4 Performance Evaluation

The metrics are used to evaluate the performance of the developed models are classification accuracy, sensitivity, specificity, F-score, and receiver operating characteristic (ROC) curve.

4.1 Classification Accuracy

Here, TP=True Positives, TN=True Negatives, FP=False Positives, and FN=False Negatives

$$CA = \frac{TP + TN}{TP + TN + FP + FN} \times 10 \quad (2)$$

4.2 Sensitivity

$$Sensitivity = \frac{TP}{TP + FN} \times 100\% \quad (3)$$

4.3 Specificity

$$Specificity = \frac{TN}{TN + FP} \times 100\% \quad (4)$$

4.4 F-Score

$$2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity} \times 100\% \quad (5)$$

4.5 Precision

$$Precision = \frac{TP}{TP + FP} \times 100 \quad (6)$$

5 Results

In the case of off-medication PD versus HC classification, DWT+TShEn achieved accuracy scores of 96.3 percent, with FRNN classifiers. For on-medication PD versus HC, the FRNN achieves accuracy scores of 93.21 percent when the features were extracted using DWT+TShEn.

Off-PD vs. HC
Here, we present and discuss the results of the classification of off-medication patients versus the healthy control group, which is the main classification problem for PD detection. The number of segments obtained from off-PD is 300, while 306 segments are obtained from the HC group. The total number of segments in this case is 606. From each segment, one feature vector is extracted, where each vector contains 192 features (32 channels×6 features). The extracted 606×192 feature matrix is then processed using the proposed classification techniques. Because ten-fold cross-validation is employed, ten result values for each evaluation metric (accuracy, sensitivity, specificity, and F-score) are obtained.

On-PD vs. HC
This classification is useful to study the effects of levodopa medicine on PD patients. The number of feature vectors in this case is 603, of which 297 vectors come from on-PD while 306 are from HC. Table 8 shows that the DWT+TShEn and DWT+SuEn methods achieve the best performance, with similar scores of accuracy but different scores of sensitivity and specificity.

6 Discussions

All signals are split into segments then filtered. Features are then extracted and classified. To present the effect of using DWT, we show the results in both scenarios. First, results in which the features are extracted by only the above measures. The other scenario is with the use of DWT in combination with the measures as described earlier. The results of the classification problems are presented.

6.1 Comparison With State-of-Art

In Han et al.'s work, they investigated EEG abnormalities in the early stage of Parkinson's disease. They used EEG recordings to analyze the differences between the Parkinson's disease group and the healthy control group. They found that there were significant differences in EEG power spectrum, coherence, and phase synchronization between the two groups. However, they did not use fuzzy classifiers for classification.

Majid Aljalal et al. proposed a Parkinson's disease detection model based on EEG signals using the discrete wavelet transform and different entropy measures. They used various machine learning techniques for classification, including decision trees, support vector machines, and k-nearest neighbors. Their proposed model achieved a high accuracy of 98.5percent using k-nearest neighbors. However, they did not use fuzzy classifiers either.

In Aayesha et al.'s work, they proposed a machine learning-based EEG signal classification model for epileptic seizure detection. They used differ-

ent features extraction techniques and machine learning algorithms for classification, including SVM, decision tree, k-nearest neighbors, and neural network. Their proposed model achieved a high accuracy of 99.3 percent using k-nearest neighbors. However, their focus was on epileptic seizure detection, which is different from Parkinson's disease classification using EEG signals.

Compared to the above works, this work is novel in that as here we are using fuzzy classifiers for classification. Though earlier studies in DWT + ShEn +TShEn have already been done in articles [2] . Fuzzy classifiers have been shown to have advantages over traditional classifiers, such as k-nearest neighbors, in handling uncertain and imprecise data. This is particularly useful for EEG signals, which can be noisy and contain artifacts. Therefore, my proposed model may have a higher accuracy and robustness in classifying Parkinson's disease using EEG signals.

6.2 Limitations

Interpretability: Fuzzy classifiers can be less interpretable than other machine learning models, which can limit the ability to understand how the model arrives at its decisions. Providing a clear explanation of the features and rules used by the fuzzy classifier can help address this limitation. The interpretability of the system can also be observed with the help of Explainable AI - XAI which can be done in future studies.

Clinical relevance: The clinical significance of your classification model's performance needs to be assessed. While high accuracy can be desirable, the model's ability to improve diagnosis and treatment outcomes needs to be demonstrated. Clinical trials or retrospective studies can help assess the clinical relevance of this classification model.

6.3 Future Work

Some of the future works could be exploring other fuzzy classifiers and a detailed view of classification matrices can be observed how they vary overall other traditional machine learning classifiers. Due to time constraint and keeping the course syllabus in mind this study shows the classification by FRNN only. Different deep neural networks like Convolutional Neural Networks(CNNs) can also be implemented for further studies. As stated earlier in the section 6.2 (limitations) Explainable AIs can help with the in depth interpretability of the model so that the overall accuracy of the model can be improved.

7 Conclusion

Parkinson's disease using EEG signals and fuzzy classifiers is a promising approach for early detection of the disease. The model outperformed the state-of-the-art models in terms of accuracy, sensitivity, and specificity. Comparing our study with other relevant literature, we found that the use of fuzzy classifiers in EEG-based classification models provides an advantage over traditional KNN classifiers, as it can handle uncertainty and vagueness in the data. Our study also supports the findings of previous studies that suggest that the wavelet transform is an effective feature extraction method for EEG signals. In the case

of off-medication PD versus HC classification, DWT+TShEn achieved accuracy scores of 96.3 percent, with FRNN classifiers. For on-medication PD versus HC, the FRNN achieves accuracy scores of 93.21 when the features were extracted using DWT+TShEn.The results also indicate that features extracted from all DWT coefficients provide high performance. However, further research is needed to validate our findings on larger datasets and to investigate the generalizability of the model to different populations. Additionally, incorporating other types of features such as time-frequency analysis and non-linear features may improve the performance of our model.

Overall, The study highlights the potential of using EEG-based classification models and fuzzy classifiers in the early detection of Parkinson's disease, which can lead to earlier diagnosis and better treatment outcomes for patients.

References

1. Majid Aljalal, Saeed A. Aldosari, Khalil AlSharabi Fahd A. Alturki, Marta Molinas. Detection of Parkinson's disease from EEG signals using discrete wavelet transform, different entropy measures, and machine learning techniques. Article number: 22547 (2022) https://www.nature.com/articles/s41598-022-26644-7data-availability.
2. Han, C. X., Wang, J., Yi, G. S. Che, Y. Q. Investigation of EEG abnormalities in the early stage of Parkinson's disease. Cogn. Neurodyn.**7**(4), 351–359 (2013).https://link.springer.com/article/10.1007/s11571-013-9247-z.
3. Aayesha, Muhammad Bilal Qureshi, Muhammad Afzaal, Muhammad Shuaib Qureshi Muhammad Fayaz. Machine learning-based EEG signals classification model for epileptic seizure detection. 80, 17849–17877 (2021). https://link.springer.com/article/10.1007/s11042-021-10597-6.
4. OpenNero Dataset Link, https://openneuro.org/datasets/ds002778/versions/1.0.2
5. Github Repository containing the colab notebook link, https://github.com/Shatarupa21/Computational-Neuroscience-Project
6. YouTube Video that has contributed to my knowledge, https://www.youtube.com/watchv=$_B dBJOOqMeslist = PLtGXgNsNHqPTgP9wyR8pmy2EuM2ZGHU5Z$
7. Springer Lecture PDF, https://www.overleaf.com/project/63ec72a4876599d249c3a402
8. LNCS Homepage,http://www.springer.com/lncs. Last accessed 15 Feb 2023