Detection of Parkinson's Disease from speech recordings data

Authors: Lluis Grifols, Marcel Romaní

Abstract: Traditional diagnosis of Parkinson's Disease (PD) involves the observation of motor skills of the patient in various situations and the monitoring of the progression of the disease over time, which requires repeated clinic visits. The goal of this project is to train various kernel-based algorithms to diagnose PD by using data from speech recordings and compare their performance against traditional non-kernelized methods.

I. INTRODUCTION

Parkinson's Disease (PD) is a degenerative neurological disorder marked by decreased dopamine levels in the brain. It manifests itself through a deterioration of movement, including the presence of tremors and stiffness. There is commonly a marked effect on speech, including dysarthria (difficulty articulating sounds), hypophonia (lowered volume), and monotone (reduced pitch range). Additionally, cognitive impairments and changes in mood can occur, and risk of dementia is increased.

Traditional diagnosis of Parkinson's Disease involves a clinician taking a neurological history of the patient and observing motor skills in various situations. Since there is no definitive laboratory test to diagnose PD, diagnosis is often difficult, particularly in the early stages when motor effects are not yet severe. Monitoring progression of the disease over time requires repeated clinic visits by the patient. An effective screening process, particularly one that doesn't require a clinic visit, would be beneficial. Since PD patients exhibit characteristic vocal features, voice recordings are a useful and non-invasive tool for diagnosis. If machine learning algorithms could be applied to a voice recording dataset to accurately diagnosis PD, this would be an effective screening step prior to an appointment with a clinician.

II. RELATED WORK

In this project we have used the dataset provided by the UCI Machine Learning Repository [1] which has been previously used in [2]. The classification algorithms applied there include Naive Bayes, Logistic Regression, k-Nearest Neighbors, Random Forest and SVC with Linear and RBF kernels. The best results were obtained with the SVC and RBF kernel.

III. DATA EXPLORATION AND PRE-PROCESSING

The data used in this study were gathered from 188 patients with PD (107 men and 81 women) with ages ranging from 33 to 87 (65.1 $\pm$ 10.9) at the Department of Neurology in Cerrahpasa Faculty of Medicine, Istanbul University. The control group consists of 64 healthy individuals (23 men and 41 women) with ages varying between 41 and 82 (61.1 $\pm$ 8.9). During the data collection process, the microphone was set to 44.1 KHz and following the physician's examination, the sustained phonation of the vowel /a/ was collected from each subject with three repetitions.

In total, we have 756 samples (564 positive and 192 negative for PD) and 753 features of clinically useful information extracted from speech recordings using various speech signal processing algorithms including Time Frequency Features, Mel Frequency Cepstral Coefficients (MFCCs), Wavelet Transform based Features, Vocal Fold Features and TWQT features, as well as the gender.

TABLE I: Overview of the features

Feature Set	Measure	# of features
Baseline Features	Jitter variants	5
	Shimmer variants	6
	Fundamental frequency parameters	5
	Harmonicity parameters	2
	Recurrence Period Density Entropy	1
	Detrended Fluctuation Analysis	1
	Pitch Period Entropy	1
Time frequency features	Intensity Parameters	3
	Formant Frequencies	4
	Bandwidth	4
Mel Frequency Cepstral Coef.	MFCCs	84
Wavelet Transform based Features	WT	182
	TWQT	432
Vocal fold features	Glottis Quotient	3
	Glottal to Noise Excitation	6
	Vocal Fold Excitation Ratio	7
	Empirical Mode Decomposition	6

A. Pre-processing

The data was loaded into a pre-processing pipeline, consisting of missing values detection and outlier handling. For the first step we found no missing values in the data. For the outliers we followed one of the methodologies described by Silva et al. [3], where the mean μ_i and standard deviation σ_i is computed for each attribute $x_1, x_2, ..., x_n$, and an upper and lower thresholds are set at $\mu_i \pm 3\sigma_i$. Any value larger than the upper threshold will be replaced by the upper threshold value and any value lower than the lower threshold will be replaced by the lower threshold value.

We also implemented a function that splits the data into training and testing sets, while keeping the ratio of the target variable the same for both sets. Both attribute data sets are then scaled into a [-1,1] range by training a scaling function on the training set and applying it on both sets.

IV. KERNEL PCA

In order to reduce the number of dimensions of the data, we implemented dimensionality reduction via kernel PCA. In regular PCA, we perform a linear transformation of the data into a new base, where each component maximizes the explained variance, such that the explained variance of component 1 is greater than that of component 2, and so on. PCA is limited by the fact that it can only capture linear correlations between the features, but this can be overcome by implementing kernels. Kernel PCA allows us to find non-linear correlations between features by performing the PCA in a Hilbert Space created by the Kernel function. In this space, relationships that originally did not seem linear, now may appear linear and therefore can be identified by the algorithm.

V. KERNEL FISHER DISCRIMINANT ANALYSIS

Linear Discriminant Analysis is closely related to Principal Component Analysis (PCA) in that they both look for linear combinations of variables which best explain the data, with the difference that the former works in a supervised way. In our case, we implemented the kernelized version, Kernel Fisher Discriminant Analysis (KFDA), in order to find non-linear combinations of variables that maximally separate the two categories of our data. In particular, KFDA applies Fisher's linear discriminant rule, which tries to maximize the ratio between $SS_{between}$ and SS_{within}, where euclidean distances have been replaced by a similarity measure given by a kernel function.

The goal is to compare its performance with the non-kernelized version LDA.

VI. SUPPORT VECTOR CLASSIFIER

For the prediction of the target variable we also used an implementation of a Support Vector Classifier, which is the benchmark for classification techniques.

VII. METHODOLOGY

A. Kernels

The Kernel families proposed for this work are five: Polynomial (1), RBF (2), Laplacian (3), Sigmoid (4) and Chi-squared (5).

$$k(x, x') = (a\langle x, x'\rangle + c)^q \tag{1}$$

$$k(x, x') = \exp -\gamma \|x - x'\|_2^2 \tag{2}$$

$$k(x, x') = \exp -\gamma \|x - x'\|_1 \tag{3}$$

$$k(x, x') = \tanh(a\langle x, x'\rangle + c) \tag{4}$$

$$k(x, x') = \exp -\gamma \sum (x - x')^2/(x + y) \tag{5}$$

Polynomial kernels are described using three parameters: a,c and q. From eq.(1) we can see that a acts as a multiplier to the inner product $\langle x, x'\rangle$ that together with c form the elements of a binomial of degree q.

RBF, Laplacian and Chi-squared kernels are characterized by only one parameter: γ. This parameter is proportional to the root mean square width, as it is defined as $\gamma \equiv \frac{1}{2\sigma^2}$

B. Kernel PCA

The main challenge we faced with the KernelPCA was to identify the kernel that would perform best with our data, given that in an unsupervised problem we don't have a ground truth, so the decision criteria for choosing the best kernel used in this work was to minimize the mean squared error of the reconstruction error. We defined a search space for the hyperparameters by randomly sampling 50 values for each parameter following a log-uniform distribution from 10^{-4} to 10^2. The search for the best parametes was conducted using Bayesian optimization for hyperparameters alg.(1) as the search space is too large to implement a regular grid search algorithm. To mitigate the impact of overfitting, we used a 5-fold cross-validation for the hyperparameter tuning.

This pseudocode summarizes the strategy that the Bayesian optimizer uses to search for the best parameters. The set P contains all the possible values for the

Algorithm 1 Model Hyperparameter Tuning Strategy

$P \leftarrow p$
for $n = 1 : N$ **do**
  $p \leftarrow posterior$
  $model_i \leftarrow p$
  $y_i \leftarrow f(x_i)$
  $posterior \leftarrow update\_for(y_i)$
end for

parameters. Then for a specified number of iterations, it draws a set of parameter values using the posterior distribution that aims for an expected improvement. A model is generated with said parameters and is then evaluated using the specified metric. Lastly, the posterior distribution is updated with the result of the iteration.

C. Kernel FDA

In order to find the best hyperparameters to perform the discriminant analysis we used a Grid Search over the five kernels exposed in section VII A and the coefficients γ from 10^{-4} to 1, q from 2 to 10 and $robustness\_offset$ from 10^{-12} to 10^{-6}. The metrics to assess the optimality of the parameters included the recall and the f1-score in order to minimize the amount of False Negatives. This is because it is preferred to diagnose a person with something they don't have than to miss it, which could have very serious consequences.

D. Support Vector Classifier

We implemented dimensional reduction to the data before using it to train the SVM, this was done using the Kernel PCA trained in previous steps. For this work we evaluated the performance of a *rbf* kernel, a *polynomial* kernel and a *sigmoid* kernel. The hyperparameter tuning strategy for each of the kernels was done by using Bayesian grid search, and the possible values for each parameters were drawn from a log-uniform distribution from 10^{-4} to 10^{2}. A 5-fold cross-validation was implemented in order to reduce overfitting of the model.

Once the best parameters are obtained for each Kernel, we evaluate their performance using the validation set. Finally, using the kernel that scored best on the validation set, we performed a LOOCV to obtain an accurate estimation of the classification error.

VIII. RESULTS

A. KPCA

These are the results obtained for the Kernel PCA. In table (II) we can see the MSE for each of the tested Kernels. They return similar values, but ultimately the polynomial kernel's result is slightly better than the rbf.

TABLE II: Kernel PCA results

	1 - MSE	SD
RBF	-5.34 $\times 10^{-3}$	1.8 $\times 10^{-4}$
Polynomial	-4.12 $\times 10^{-3}$	1.26 $\times 10^{-4}$

The following figures show the data reorganized into the new principal component space. Fig.(1) shows a 2D representation where the data appears to have a spherical behaviour, with points belonging to the control group (0) to the lower right. In fig.(2) we can see a 3D representation of the first three principal components, where we can confirm a spherical behaviour of the data previously hinted in fig.(1)

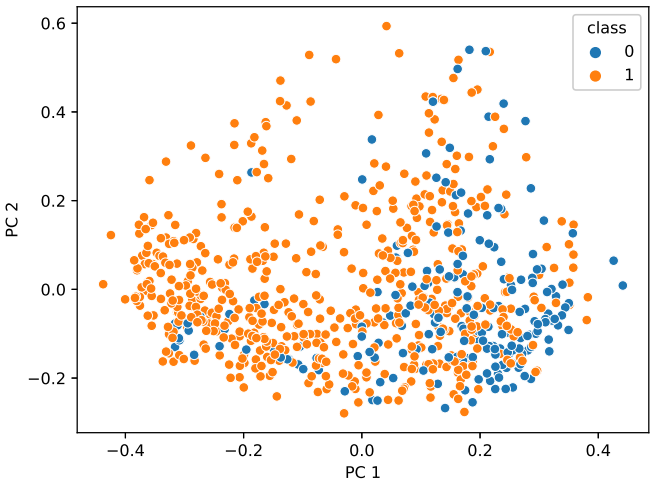

FIG. 1: 2D projection of the Principal Components.

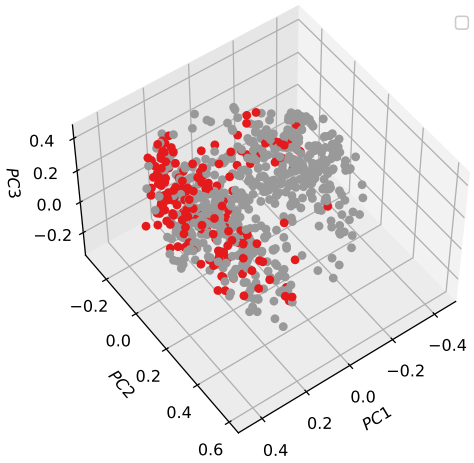

FIG. 2: 3D projection of the Principal Components.

Another interesting analysis of the Principal Component Analysis are the study of the obtained eigenvalues. In fig(3) we can see the cumulative sum of the eigenvalues for each of the studied kernels. The polynomial kernel seems to be much better at summarizing the information, as it achieves nearly 100% explained variance with only 200 components.

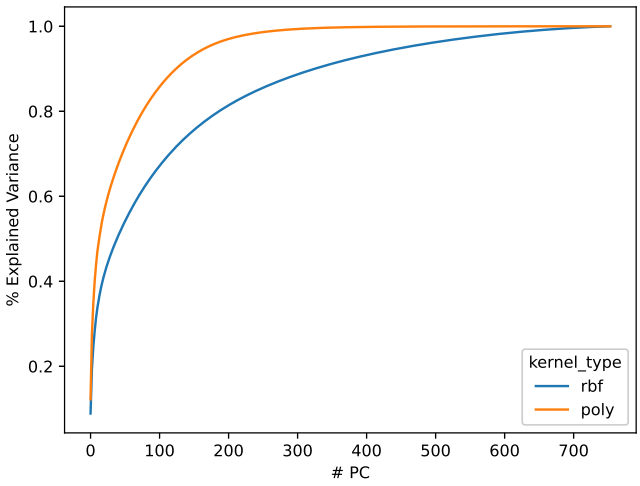

FIG. 3: Cumulative sum of eigenvalues.

B. KFDA

These are the classification results we obtained applying the Kernel Fisher Discriminant Analysis on the initial data, that is, without using the kernel-PCA transformation.

Best parameters:

Kernel	γ	**Robustness offset**
Laplacian	0.01	10^{-12}

TABLE III: Kernel FDA results

	precision	**recall**	**f1-score**	**support**
Negative	0.92	0.75	0.83	48
Positive	0.92	0.98	0.95	141
Accuracy			0.92	189
Macro Avg	0.92	0.86	0.89	189
Weighted Avg	0.92	0.92	0.92	189

TABLE IV: Confusion matrix

	Negative	**Positive**
Negative	36	12
Positive	3	138

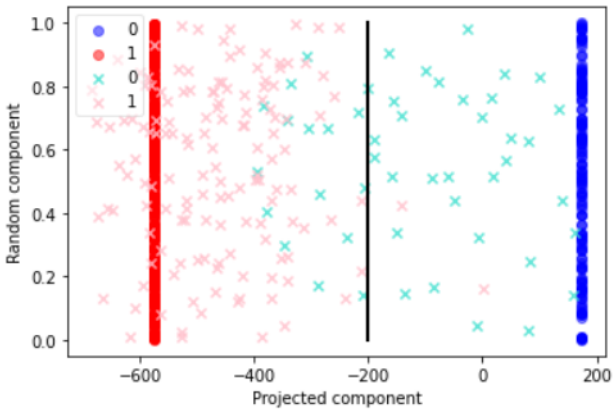

FIG. 4: Projection of the data onto the one dimensional transformed space (the second dimension is added for clarity). Blue and red dots represent the training data, while pink and turquoise crosses represent the projected test data. The black line is the decision boundary.

And these are the results using LDA:

TABLE V: LDA results

	precision	**recall**	**f1-score**	**support**
Negative	0.43	0.62	0.51	48
Positive	0.85	0.72	0.78	141
Accuracy			0.70	189
Macro Avg	0.64	0.65	0.89	189
Weighted Avg	0.74	0.71	0.92	189

TABLE VI: LDA Confusion matrix

	Negative	**Positive**
Negative	30	18
Positive	39	102

As we can see, the overall results of KFDA are much better than the non-kernelized version. Thanks to the kernel function, we are able to map our data to a higher dimensional space where data are linearly (more) separable.

The actual results show that KFDA yields a recall of 98% of the positive cases, which means that in practice almost all positive (sensitive) cases will be detected, while 1/4 of the negative ones will be False Positives. As we said earlier, this is preferable to having an overall good accuracy which treats Type I and Type II error the same way.

The scatter plot of the transformed data also shows that the two classes are pretty well separated and the decision boundary (which in reality is a 1-dimensional value) favors the recall of positive cases.

C. SVC

To evaluate the performance of the Support Vector Classifier we make use of four different metrics: Receiver Operating Characteristic (ROC) score, Accuracy, f1_score and Diagnostic Odds Ratio (DOR). We then selected the best model and evaluated how it performed for as we changed the number of Principal Components that where used as attributes for the model. Finally, with the optimal kernel and number of attributes, we performed a Leave-One-Out Cross-Validation to estimate a reliable validation error for our model.

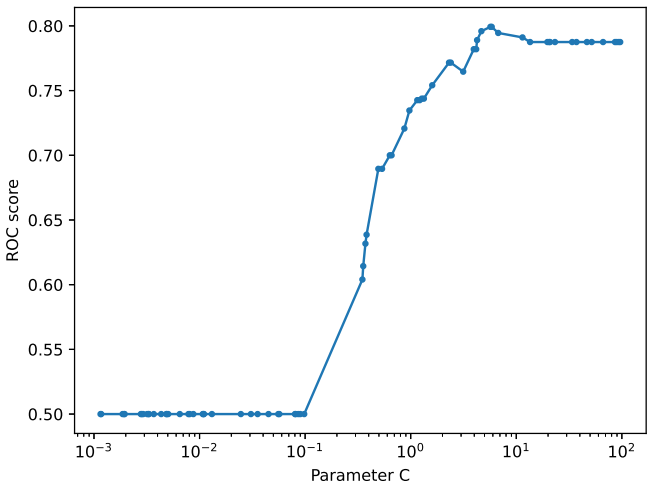

FIG. 5: ROC score obtained for different values of the parameter C for the RBF kernel.

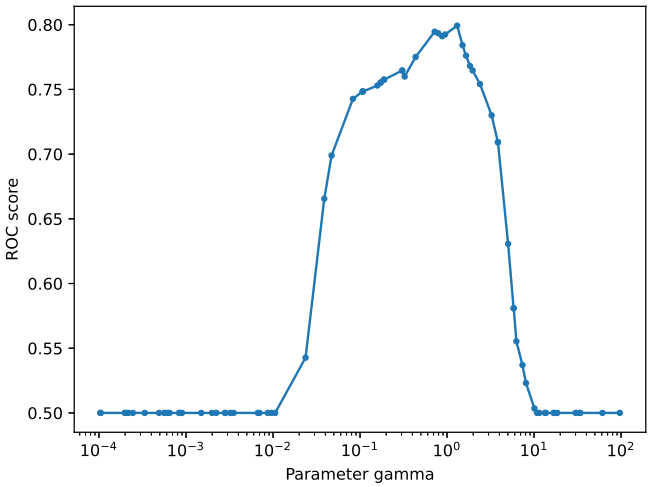

FIG. 6: ROC score obtained for different values of the parameter gamma for the RBF kernel.

In fig.(5) and fig.(6) we can see how the ROC score of the model changes as we modify the parameters (here for a *rbf* kernel). We can see how for Parameter C the accuracy takes off when $C > 10^{-1}$ and stabilizes for $C > 10$. For parameter gamma we see that the accuracy starts to increase at $gamma > 10^{-2}$, peaks at $gamma = 1$ and floors for $gamma > 10$.

Results can be seen in the following table.

TABLE VII: SVC Results

	ROC	**Accuracy**	**F_1 score**	**DOR**
RBF	0.80	0.87	0.91	29.3
Polynomial	0.77	0.85	0.91	25.4
Sigmoid	0.72	0.84	0.90	25.0

The next step we performed was to study how the amount of principal components affected the predictions. In fig.(7) we can see that the ROC score increases up until 100 features and then decreases with some perturbations around 400. This tells us that the information up to the 100th PC is relevant for this classification problem, but for the PC 200-300 they act as noise and confuse the algorithm, PC 300-400 contain relevant information, as it increases the score, and from PC 400 onward the score decreases

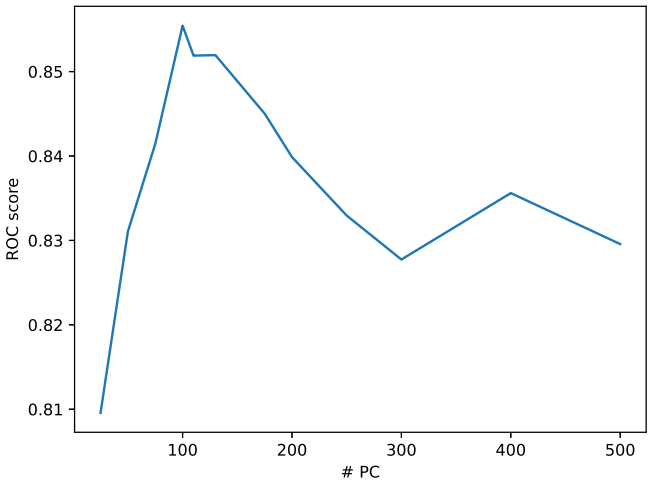

FIG. 7: ROC score obtained for the best model as we change the number of Principal Components we use for the classification.

Here we present the results for the Leave-One-Out Cross-Validation. We can see that the ROC, Accuracy and F_1 score for the Leave-One-Out Cross-Validation slightly improve from those in table VII, and DOR increases in great measure due to the decrease in false positives/negatives.

TABLE VIII: Leave-One-Out Cross-Validation

ROC	**Accuracy**	**F_1 score**	**DOR**
0.85	0.90	0.94	69.6

TABLE IX: LOOCV Confusion matrix

	Negative	**Positive**
Negative	140	52
Positive	21	543

IX. CONCLUSIONS

We have studied many different kernel applications for machine learning and how the choices of the kernel used affect the result. Linear analysis methods like PCA or LDA show a great improvement in performance when kernelized. Standard kernel methods, like the SVM show very good performance for our dataset, as the relationship between features is not linear. The highest accuracy for the validation data for the identification of PD was 0.92, using Kernel FDA, and 0.87 using an SVC. We then also used an optimal number of PC to train an SVC and obtained an Accuracy of 0.90 for the LOOCV.

In a future work on this dataset it would be interesting to try to model each type of feature with its own kernel method. For example, Baseline Features and Time Frequency features may require the projection to different spaces in order to elicit hidden structures of the data. Finally, the various outputs would be put together in order to make the final decision.

[1] *UCI Machine Learning Repository.* https://archive.ics.uci.edu/ml/datasets/Parkinson%27s+Disease+Classification.

[2] C. Okan Sakar, Gorkem Serbes, Aysegul Gunduz, Hunkar C. Tunc, Hatice Nizam, Betul Erdogdu Sakar, Melih Tutuncu, Tarkan Aydin, M. Erdem Isenkul, Hulya Apaydin, *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, Volume 74, 2019, Pages 255-263, ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2018.10.022.

[3] Letícia Silva, Juliana Hermsdorf, Victor Guedes, Felipe Teixeira, Joana Fernandes, Bruno Bispo, João Paulo Teixeira, *Outliers Treatment to Improve the Recognition of Voice Pathologies*, Procedia Computer Science, Volume 164, 2019, Pages 678-685, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2019.12.235.