



FACULTY OF SCIENCE, TECHNOLOGY AND MEDICINE

---

# Identification of Parkinson's Disease from Speech Using Deep Learning

---

Thesis Submitted in Partial Fulfillment of the  
Requirements for the Degree of Master in Information  
and Computer Sciences

*Author:*  
Ze WANG

*Supervisor:*  
Prof. Christoph SCHOMMER

*Reviewer:*  
Prof. Rejko KRÜGER

*Advisor:*  
Dr. Vladimir DESPOTOVIC

December 2020

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

## Abstract

Parkinson’s disease (PD) is the second most common neurodegenerative disease caused by loss of dopamine producing cells. Most PD patients exhibit voice disorders, manifested by soft, monotone, excessively breathy and hoarse voice. This study uses sustained vowel phonations for identification of Parkinson’s disease with convolutional neural networks (CNNs). Mel-spectrograms are used as low-level feature representation of sustained vowel phonations in time and frequency domain as the input of CNN, which further learns higher level features that are more suitable for discrimination between PD patients and healthy control subjects. Three frameworks are proposed: end-to-end self-customised CNN architecture, combination of CNN for feature extraction and Support Vector Machines (SVM) for classification and fully connected neural network architecture with mel-spectrograms as features. Five subsets from mPower data are used to evaluate the ability of the models under different conditions (early detection vs. detection of progressed cases, detection with imperfect recordings corrupted by background noise etc.). Two data splitting schemes are implemented: traditional sample-based data splitting and donor-based data splitting, which removes bias towards particular voice donor that may exist when different recordings of the same donor are present in both the training and validation dataset. The experimental results show that end-to-end CNN architecture outperforms the other two frameworks, and furthermore proves the automatic feature extraction ability of CNNs. The best performance of CNN architecture with sample-based data splitting reaches validation accuracy of 94%, whereas with donor-based data splitting the validation accuracy drops to only 68%, showing substantial overfitting. Since adding standard regularization techniques to deal with overfitting did not improve the results, additional measures, such as data augmentation and expanding the dataset with additional voice samples should be undertaken.

## **Acknowledgements**

Firstly, I would like to show my deepest gratitude to my supervisor, Prof. SCHOMMER, who guided my thesis, for his teaching, concern, and help. Thank him for his kindly leading in the process of completing this thesis.

Secondly, I would like to thank Prof. KRUGER. It is my honor to invite him to be my reviewer.

At the same time, I am very grateful to my advisor, Dr. DESPOTOVIC, whose careful instruction and support keeps me in track and helps me step by step go through this project.

Last but not least, thank my mum, my husband and my son for the support and understanding, especially during the time when I was not present or lost strength.

# Table of Contents

<b>List of Tables</b>	<b>8</b>
<b>List of Figures</b>	<b>10</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Previous work</b>	<b>4</b>
<b>3 Methodology</b>	<b>9</b>
3.1 Signal Preprocessing . . . . .	9
3.1.1 Spectrogram and Mel-spectrogram . . . . .	9
3.1.2 Window length . . . . .	11
3.2 Architectures . . . . .	11
3.2.1 Convolutional Neural Networks . . . . .	11
3.2.2 Support Vector Machines . . . . .	19
3.2.3 Nonlinear classification . . . . .	21
3.3 Dataset . . . . .	21
3.4 Dataset Splitting . . . . .	25
3.4.1 Simple Random Sampling (SRS) . . . . .	25
3.4.2 K-fold Stratified Cross Validation . . . . .	25
3.4.3 Donor-based Data Splitting . . . . .	26
3.5 Evaluation and Performance Measures . . . . .	26

<b>4</b>	<b>Experiments and Results</b>	<b>28</b>
4.1	Data Preparation . . . . .	28
4.1.1	Demographics Survey . . . . .	29
4.1.2	Voice Activity . . . . .	29
4.1.3	Phonation audio files . . . . .	29
4.1.4	Countdown audio files . . . . .	30
4.2	Training datasets . . . . .	31
4.2.1	Data availability and feasibility . . . . .	31
4.2.2	Training datasets . . . . .	33
4.2.3	Data splitting . . . . .	35
4.3	Data Preprocessing . . . . .	36
4.4	Model Training . . . . .	37
4.4.1	CNN model . . . . .	37
4.4.2	CNN feature extraction + SVM . . . . .	44
4.4.3	Dense only network . . . . .	46
4.5	Results . . . . .	47
4.5.1	Three Frameworks . . . . .	47
4.5.2	Extended Experiment . . . . .	49
<b>5</b>	<b>Conclusions and future work</b>	<b>52</b>
5.1	Conclusions . . . . .	52
5.2	Future work . . . . .	54
	<b>References</b>	<b>55</b>
	<b>APPENDICES</b>	<b>62</b>
<b>A</b>	<b>Demographics Survey</b>	<b>63</b>

<b>B Data distribution of all the available data for this study in mPower dataset</b>	<b>65</b>
<b>C Data Citations</b>	<b>67</b>
<b>D Dataset Download</b>	<b>68</b>
D.1 Steps for Access . . . . .	68
D.2 Install synapseclient for Python User . . . . .	69
D.3 Python Code for Audio Files Download . . . . .	69
D.4 Python Code for Audio Files Reorganisation . . . . .	69
<b>E Activation Function</b>	<b>71</b>

# List of Tables

3.1	Batch Normalizing Transform, applied to activation x over a mini-batch. [1]	16
3.2	Popular kernels in SVM, from [2]	21
3.3	Data available for each survey and activity completed by study participants[3]	23
3.4	Data distribution	24
4.1	Used variables from Demographics Survey	29
4.2	Used variables from Voice Activity	30
4.3	Data distribution of participants and sample cases used regarding the Diagnosed Year	32
4.4	Data distribution of All500 dataset	33
4.5	Data distribution of EarlyCase dataset	34
4.6	Data distribution of ProgressedCase dataset	34
4.7	Data distribution of All2000 dataset	35
4.8	Architecture of CNN baseline model	38
4.9	Model training performance	38
4.10	Model training performance with Dropout Layers (performance of baseline model is also added for comparison)	40
4.11	Model training performance with Dropout Layers (performance of baseline model is also added for comparison)	41
4.12	Model training performance with kernel_constraint (performance of experiment No.12 is also added for comparison)	42



4.13 Model training performance with L2 regularization (performance of baseline model is also added in for comparison) . . . . .	42
4.14 Model training performance with L2 regularization (performance of baseline model is also added for comparison) . . . . .	43
4.15 Model training performance with L2 regularization and Dropout (performance of baseline model is also added for comparison) . . . . .	43
4.16 Model training performance with batch normalization (performance of baseline model is also added for comparison) . . . . .	44
4.17 Architecture of the proposed CNN model . . . . .	44
4.18 Architecture of Dense Only model . . . . .	47
4.19 Overall performance using accuracy, sensitivity and specificity as performance measures . . . . .	47
4.20 CNN+SVM model without feature selection . . . . .	49
4.21 Performance using sample-based data splitting without denoising preselection (All2000 dataset) . . . . .	50
4.22 Performance using donor-based data splitting without denoising preselection (ReAll2000 dataset) . . . . .	50
A.1 Demographics Survey . . . . .	64
B.1 Data distribution of participants and audio records regarding the Diagnosed Year . . . . .	66

# List of Figures

3.1	Mel-spectrogram(right) and spectrogram(left) from one audio record . . .	10
3.2	Spectrograms for 3 different window lengths. Window length increases from left to right. X axis is time. Y axis is frequency. Images are from [4]	11
3.3	A $2 \times 2$ filter applied to a $6 \times 6$ two-dimensional input to create a $5 \times 5$ feature map (stride = 1) . . . . .	13
3.4	Pooling process for max pooling and average pooling . . . . .	14
3.5	Dropout process. <b>Left:</b> Standard network without dropout. <b>Right:</b> Network with dropout applied to 3 first layers, including one input layer and two hidden layers. The crossed nodes are dropped . . . . .	15
3.6	Flattening . . . . .	17
3.7	The black line well fits the data well, while the green line over fits the noises in the dataset. . . . .	18
3.8	Two separating lines: the one with the maximum margin is the preferred separating hyperplane(on the right) to the one with a smaller margin(on the left). Image is taken from [2] with minor modification. . . . .	19
4.1	Downloaded audio files' folder structure . . . . .	30
4.2	SVM model accuracy with feature selection. . . . .	46
4.3	SVM model accuracy with feature selection. . . . .	46
4.4	Confusion matrix for Dense only CNN model with ProgressedCase dataset	49
4.5	Training Curve with ReAll2000 dataset . . . . .	51
E.1	Activation Functions . . . . .	71



# Chapter 1

## Introduction

Parkinson's disease (PD) is the second most common neurodegenerative disease after Alzheimer's disease [5]. Age is one of the main factors of Parkinson's disease. People start to develop this disease usually after 60. Due to the rise of the life expectancy, the prevalence of PD is estimated to increase.

It is a progressive disorder with multiple motor and non-motor symptoms, which are related to loss of dopamine producing cells. Motor symptoms are manifested by involuntary tremulous motions, tremor, muscle rigidity, muscle cramps, slowness of movement and, lessened muscular power which makes it difficult for performing simple activities [6]. Further symptoms are falls and dizziness, freezing, but also speech disorders and writing changes. Non-motor symptoms include pain, fatigue, low blood pressure, thinking difficulties, depression and emotional changes, swallowing problems, sleeping problems and sleeping disorder, etc. Results from a wide range of researches on patients with PD indicate that both motor and non-motor symptoms significantly affect quality of life of the patients and their families [7].

Even though it has high prevalence and severe impacts on the quality of life, the disease is so far considered as not being able to be cured. Medications or surgical intervention can often dramatically help to control the symptoms, slow down the progress and improve the quality of life.

Meanwhile PD symptoms develop progressively over time, and are unnoticeable in early stages of disease. A diagnosis is usually performed after the patient experiences visible or noticeable motor symptoms. The loss of dopamine-producing neurons increases rapidly and reaches 50% mostly at point of clinical diagnosis[8].

There is no single test which can conclusively diagnose Parkinson’s disease. The standard diagnosis is based mainly on the clinical examination. Due to the complicity of the diagnostic procedure, it is either not available on time, or not available at all in some developing areas. The diagnosis of PD is currently mostly based on motor symptoms assessment during clinical consultation. It is then followed by L-Dopa trials to see after medication L-Dopa (dopamine supplementation) if the symptoms of the patient get improved. The improvement with dopamine supplementation is a sign of PD. Further laboratory tests are needed, when there are still doubts. Therefore, new effective and timely available methods for PD diagnosis are needed to tackle the disease and introduce treatment earlier to improve the quality of life and extend the life spans of patients. Especially, during the global pandemic, a new diagnosis method, which can avoid or reduce the mandatory time to access the clinic or hospital, can help to reduce the spread of infection and save public resources.

Most people with Parkinson’s disease will experience changes in speech and voice at some point in their disease. The muscles disabilities, which cause the body muscle symptoms of PD, e.g. tremor, stiffness and slow movement, can as well occur to the muscles used for speaking and swallowing, which causes the voice disorder and speech disorder symptoms, e.g. a soft voice, humbled or fast speech, problems in communication and swallowing.

PD diagnosis with speech or voice samples is not invasive and it is simple to apply. Voice samples can be collected at patient own residence with a mobile device, and transmitted via telemedicine applications to hospital, or directly provide estimation of their current state locally at home. Sustained vowel phonations are most often used vocal tasks to detect PD, where patients are asked to hold their voice as long as possible and as steady as possible while phonating a certain vowel (e.g. /a/).

The main purpose of this study is identification of Parkinson’s disease from sustained vowel phonations with convolutional neural networks (CNNs). Mel-spectrograms are used as low-level feature representation of sustained vowel phonations in time and frequency domain as the input of CNN, which further learns higher level feature that are more suitable for discrimination between PD patients and healthy control subjects. Three frameworks are proposed. In the first framework, mel-spectrograms of voice records are fed into multiple layer CNN model for training, tuning and classification. In the second proposed framework, the CNN model trained on mel-spectrograms was used only for feature extraction, in order to learn higher-level feature representations from speech, which

were further fed into Support Vector Machines (SVM) for classification. In the third framework, the mel-spectrograms are input directly into a fully connected network which is used as the final, dense classification layers of the CNN model in the first framework, i.e. without using convolutional layers for feature learning. The idea was not only to identify PD with CNN, but also to assess what is the overall contribution of features learned by CNN in identification of PD.

Dataset collected within the mPower study is used for training and testing the models. It contains recordings of sustained vowel phonations recorded in non-controlled environment, using the mobile phone application; hence, a substantial number of samples is corrupted by noise, or has low quality of recording. Therefore, the models are trained on the dataset without preselection of voice samples, as well as on datasets with only clean, noise free, voice samples, in order to estimate the effect of quality of recording on the overall performance.

Last, but not least, though most of the previous state-of-art results did not stress the importance of data splitting of the training and testing dataset with respect to voice donors, the correlation of the voice records that belong to the same voice donor is noticed in this research, which can make models biased toward the recognition of the particular donor. To prevent this dataset is divided in the way that not only the same voice samples, but also the same voice donors do not appear in training and test subsets at the same time, in order to evaluate the effect of this data splitting method on model performance.

The rest of the paper is organized as follows. Chapter 2 gives an overview of previous work in detection of PD from voice. Chapter 3 introduces the terminologies and technologies which will be used in this study. Chapter 4 describes the three proposed experimental frameworks, the training datasets and the step by step implementation, and discusses obtained results. Chapter 5 provides concluding remarks and gives recommendation of future work.

## Chapter 2

# Previous work

Many researches have been done to try to apply the available technologies to diagnose, improve, and treat PD. According to [9][10], approximately 90% of PD patients exhibit speech impairment, and speech impairment is one of the earliest symptoms of the disease. Meanwhile, voice samples are simple to collect from patients and the process is not invasive. Accordingly, a huge number of researches were focused on voice samples processing and analyzing.

In the previous work, researchers focused mainly on two topics. These two parts, both need solid knowledge of two totally disparate domains.

1. Find the best feature or the best combination of the extracted features from sustained phonation and speech signals for better performance on PD detection, which requires expertise of acoustic processing.
2. Apply traditional machine learning algorithms for classification, which requires sound knowledge of machine learning and data processing.

Meanwhile there were only limited relatively small datasets available with less than 100 participants, including PD patients and healthy controls (HC), due to the complex data collection method. Datasets were mostly recorded in controlled and soundproof environment with professional devices. The fundamental concept of Machine Learning is to accurately estimate the behavior of unknown domains by learning the pattern from sufficient training data. Model trained with a small dataset tends to have high bias or variance and is not able to produce accurate results.

The performances of models using the sustained vowel phonations and running speeches

were compared in [11] by Vaiciukinas et al, as well as the different recording tools, e.g. professional microphones and smartphone. The research is based on recorded data from 35 HC participants and 64 PD<sup>1</sup> patients. Random forests were used as classifier. Extracted single feature sets were compared to find the best detector. Minimum cost of log-likelihood-ratio  $C_{llr}$  reached 0.529 and Equal Error Rate reached 20.30% as the best performance. Results using signals recorded with professional microphones outperformed the ones recorded using smart phones, whereas the results with running speech outperformed sustained vowel phonations.

The same dataset was used by Almeida et al in [12]. Sustained vowel phonations performed better with accuracy using the professional microphones reaching 94.5% , and smart phones reaching 93%. The best performance using Equal Error Rate as the performance measure was 14.15%. Individual feature sets were analysed with Multilayer Perceptron (MLP), k Nearest Neighbours (kNN) and Support Vector Machines (SVM) classifiers. The performance of telephone and professional microphone as recording devices is compared in [18] for Parkinson’s Disease early detection. 129 male french speaker participated, 75 early PD patients under pharmacological treatment and 54 HC participants. A classical speaker recognition method, based on Mel-Frequency Cepstral Coefficients (MFCC) extraction and Gaussian Mixture Model (GMM) is adapted, which reached accuracy of 83% with professional microphone and 75% with telephone recordings.

Despotovic et al [13] applied Gaussian processes with Automatic Relevance Determination to determine the best combination of dysphonia features from sustained vowel phonations for the task of PD detection and PD progress estimation. The dataset contains 195 vowel phonations from 23 PD patients and 8 HC participants. The best accuracy of Gaussian Process Classification for identifying PD reached 94.84% with 86% sensitivity and 97.24% specificity. It outperformed baseline SVM, Decision Tree (Boosting, Bagging and Random Forests) models. Models were validated with 10-fold cross validation.

With 188 PD patients and 64 HC participants, each of which donated three sustained vowel phonations, Gunduz compared two frameworks based on Convolutional Neural Networks[14]: feature-level combination and model-level combination. Model-level combination reached the best accuracy of 86.9% with F-Measure rate of 0.917 and Matthews Correlation Coefficient of 0.632. Models were validated with Leave-One-Person-Out

---

<sup>1</sup>Parkinson’s Disease



Cross Validation.

In [15], Mcsharry et al introduced feature measurement Pitch Period Entropy as a more robust feature of uncontrollable effects, including noise and normal frequency variations, for PD identification. The dataset used in this study contains 195 sustained vowel phonations from 23 PD patients and 8 HC participants, which were recorded in IAC sound-treated booth with head-mounted microphone. The combination of Harmonics-to-Noise Ratio, Recurrence Period Density Entropy, Detrended Fluctuation Analysis and Pitch Period Entropy was proved to have the best accuracy of 91.4% with SVM.

With more positive outputs for PD identification, researchers were encouraged to start to put effort on early detection of PD. In [16], the performance of vocal features in Parkinson’s Disease early detection is estimated. The vocal features were extracted from 5875 sustained phonations of 42 PD patients. The data were collected at patient’s home and transmitted over internet. The concept of telediagnosis is as well mentioned. The severity of speech impairments are valued with Unified Parkinson’s Disease Rating Scale(UPDRS) scores to identify the samples of PD patients at early stage. The healthy control group contained 48 samples from 8 healthy participants. Support Vector Machines with third-degree polynomial kernel was used and reached best accuracy of 96.4% and Matthews Correlation Coefficient of 0.77.

The automatic assessment of severity of PD from speech also drawn the attention of the researchers. in [17], Bayestehtashk et al extracted 1582 features from 168 subjects with PD. The level of severity of PD patient’s condition was measured by clinician using UPDRS. Three speech tasks are compared, including sustained vowel phonation task, diadochokinetic task and reading task, with results with the reading task outperforming the other ones [17]. For the effectiveness of learning regularized regression, ridge regression outperformed lasso and support vector regression. A portable device was used for recording for the purpose of building a home-based assessment platform.

In recent years, with the increased availability of the computational power and the collected data, Deep Learning, which enables detail-rich feature modeling, starts to take an important part in Machine Learning and Artificial Intelligence. It is broadly used in computer vision and speech processing. State-of-art algorithms of image classification on ImageNet [18] dataset outperform human’s accuracy. The improvements of image understanding have been widely applied in many domains, including autonomous driving and intelligent healthcare, etc.

State-of-art performance is not the only advantage of deep learning. Another advantage in comparison to the traditional machine learning algorithms is the ability to perform feature engineering. With traditional machine learning algorithms, feature engineering is a key pre-step. It extracts features which best describe the underlying problem from raw data and help to improve the prediction performance. On the other hand, after sufficient data training, deep learning algorithm can find the most correlated features or feature combinations without manual feature engineering, which requires solid knowledge of given specific domain. It is worth to mention that there are cases that neural networks discovered more complex features which are not recognized by human.

Acoustic Event Detection, based on spectrogram-inputs using Deep Neural Networks (DNN) was assessed in [19][20]. High resolution spectrograms, which enable the modeling with high-dimensional data, replaced predefined acoustic features, which saved the effort for feature engineering. DNNs with spectrogram inputs were also applied for sound event recognition in [21] under noise-corrupted environment, speech recognition in [22], speech activity detection in [23] and speech emotion recognition(SER) in [24].

To my best knowledge, there are still not many researches which apply spectrogram-inputs with DNNs for PD detection. In [25], Laiba Zahid et al fed Convolutional Neural Network models with spectrograms of speech recordings from PC-GITA dataset [26] with 50 PD patients and 50 HC participants. Three methods were applied and results were compared. Transfer learning approach was employed in the first two methods. The first method uses full spectrogram-inputs with pre-trained AlexNet model. The result was not ideal due to the fact that transfer learning only worked well when the initial and target dataset were similar enough. The second method extracted features with pre-trained AlexNet model where the last layer was replaced with traditional machine learning classifiers. This method reached the best accuracy of 99,3% with multilayer perceptron. Third method used traditional, so called handcrafted feature extraction approach and traditional machine learning classifier. It reached the best accuracy of 84.60%. This research proved that spectrogram-inputs and pre-trained AlexNet model can be a good automatic features extraction tool for PD detection.

The dataset from mPower study [3], aiming to allow researchers to understand and analyze PD severity and features, provides relatively large dataset with 65022 sustained vowels phonations recorded from 5826 subjects. The dataset was recorded with a smartphone under non-controlled conditions. In Hanbin Zhang et al [27] simple neural network with

two convolutional layers, was fed with both spectrograms and time-domain features extracted from the mPower dataset. Spectrograms outperformed the time-domain features by 28%. Five seconds long voice recording were proved to contain enough information under the tested framework.

Mel-spectrograms were used to feed DNNs for music genre classification in [26]. Mel-spectrograms differentiate from the regular spectrograms on the frequency axis. Mel-spectrograms emphasize more on the human hearing scale and compress the higher frequencies parts.

With the early increase of the internet availability and the telecommunication bandwidth, researchers started to evaluate the possibility to reduce the physical access to clinic for control and treatment for the purpose of lowering the costs and simplifying the procedures. Voice as an easy-to-use and non-invasive biomarker can be easily exploited for development of telemonitoring applications for estimating and tracking the Parkinson's disease progress, avoiding the need for frequent visits of patients to their hospitals.

## Chapter 3

# Methodology

### 3.1 Signal Preprocessing

#### 3.1.1 Spectrogram and Mel-spectrogram

Converting the voice records into corresponding mel-spectrograms is the first step for all the proposed frameworks.

Spectrogram is a visual representation of signal frequency spectrum when observed over a period of time. It preserves three-dimensional information, i.e. time on horizontal axis, frequency on vertical axis and amplitude represented by the brightness of the colors of the pixels in the image.

In order to transform the sampled discrete speech signal from time-domain, the discrete-time STFT (Short-Time Fourier Transform) [28] is applied. During discrete-time STFT, signal, which in our case is audio file, is broken down into equal size overlapping frames or windows in time domain. Discrete Fourier Transformed (DFT) is applied on each frame, and then all the processed frames are added back together to record the magnitude and phase of each point in time and frequency domain. Mathematically, the transform is described as:

$$\mathbf{STFT}\{x[n]\}(m, \omega) \equiv X(m, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n} \quad (3.1)$$

where  $\mathbf{x}[\mathbf{n}]$  is the signal to transform and  $\mathbf{w}[\mathbf{n}]$  is the window function,  $\mathbf{m}$  and  $\omega$  are the time and frequency discrete domain indices [29]. Fast Fourier Transform(FFT) is a digital fast implementation of DFT.

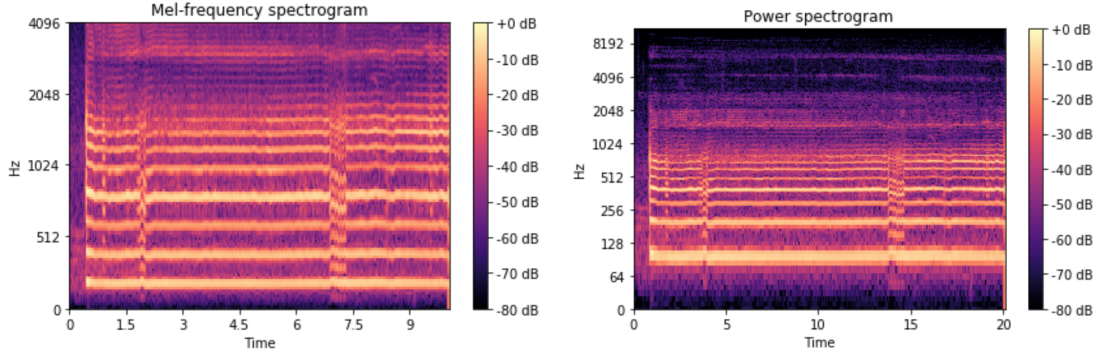


Figure 3.1: Mel-spectrogram(right) and spectrogram(left) from one audio record

The magnitude squared of the STFT yielding the spectrogram is described as:

$$\mathbf{spectrogram}\{x(t)\}(\tau, \omega) \equiv |X(\tau, \omega)|^2 \quad (3.2)$$

The time index  $\tau$  is normally considered as "slow" time.

Humans do not perceive frequencies on a linear scale. The normal range of human hearing is between 20 and 20,000Hz. A more restricted range of low-frequency sounds is important in normal listening conditions [30]. For example, we can easily differentiate a sound of 1000 Hz from a sound of 1,500 Hz, but we will hardly be able to do the same with a sound of 10,000 Hz and a sound of 10,500 Hz, even though the distances between these two pairs are the same. Mel scale, created by Stevens, Volkmann, and Newman in 1937 [31], was proposed as a perceptual scale of pitches such that equal distances in pitch sound are perceived as equal distances for listeners.

Mel-spectrograms are different from the regular spectrograms on the frequency axis, the range of which can be customised according to needs. Mel-spectrograms can emphasise more on the human hearing scale and compress the higher frequency parts. It is considered to be more efficient and relevant representation than spectrograms. Mel-spectrograms are proven to performs well in deep learning approaches without demanding larger datasets [32].

In Figure 3.1, spectrogram on the left and mel-spectrogram on the right created from the same audio file are shown, showing equidistant frequency components in the case of the mel-spectrogram.

### 3.1.2 Window length

The resolution of generated spectrogram is fixed with its window length. A signal is represented differently with different values of window length [4][33]. Window length determines either there is good frequency resolution or good time resolution. Good frequency resolution means that frequency components have clear edges in the frequency axis direction and can be easily identified. Good time resolution means that there is clear separation at frequency change in the time axis direction. With wide window, the spectrogram gets better frequency resolution but poor time resolution. On the contrary, with narrower window, the spectrogram gets better time resolution but poor frequency resolution. Figure 3.2 shows three spectrograms created from the same audio file with different window lengths.

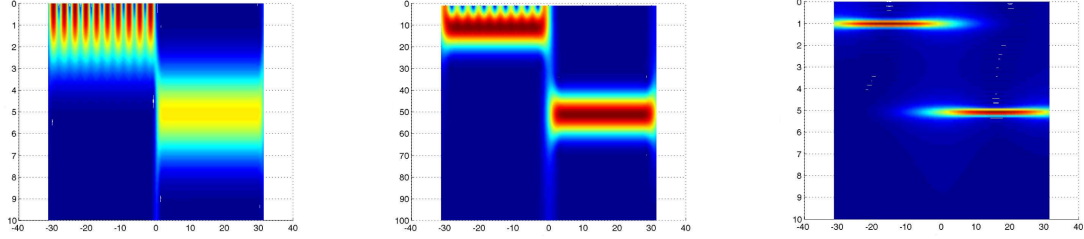


Figure 3.2: Spectrograms for 3 different window lengths. Window length increases from left to right. X axis is time. Y axis is frequency. Images are from [4]

## 3.2 Architectures

### 3.2.1 Convolutional Neural Networks

Convolutional Neural Networks(CNNs) are Deep Neural Networks mostly used for image processing. CNNs are built up with one input layer, one output layer and multiple convolutional and pooling layers. The deep architecture enables learning high-level abstractions from complex features [34]. The output of each layer represents a different level of feature extraction [35]. There are two main functional parts in the whole CNN structure:

- **Features extraction:** This part differs CNNs from traditional machine learning algorithms. With the traditional machine learning algorithms, feature extraction

is done manually. This process demands profound domain knowledge, and involves experimental work to find out the features which meet the needs. In CNNs, without the necessity of manual feature extraction, raw data or low-level feature representations can be fed directly into the networks for further processing. Feature extraction is automatized using a series of convolutional layers and pooling layers.

- **Classification:** After feature extraction, the next and final task is classification according to the extracted high-level features. The functionality of this part of CNNs is similar to the traditional machine learning algorithm. Classification part is mostly composed of one or more fully connected layers.

### Input layer

The input layer is the very first layer of CNN. It is composed of input neurons, which bring the initial data into the network for further processing of following layers.

In general, neurons in one layer receive a set of weighted inputs from the neurons of the previous layer. As input layer is located at the input of the network, the neurons of the input layer do not receive any information from the previous layer, which distinguish the input layer from the other layers.

### Convolutional layers

Convolutional layers are the most important layers in CNNs. A convolution is an activation during which a filter is applied on the input vector repeatedly and generates a feature map as the result. Feature map, a vector of weights, is the indicator of detected feature.

A dot product is an element-wise multiplication between the filter sized part of input vector and filter, and then summed up into one value. Dot product is applied systematically across an image, left to right, top to bottom during the convolution. The summed-up numbers build a feature map. When the filter is well trained or adjusted during the model training to detect specific features, a convolution is a process to discover these features in the image. The process is illustrated in Figure 3.3.

Each feature in the feature map is transformed using the nonlinear activation function. The most common choice of activation function in modern CNNs is the Rectified Linear

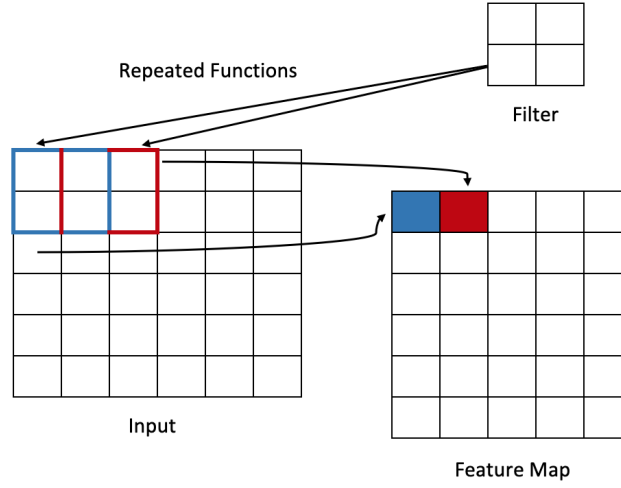


Figure 3.3: A  $2 \times 2$  filter applied to a  $6 \times 6$  two-dimensional input to create a  $5 \times 5$  feature map (stride = 1)

Unit (ReLU), defined as:

$$f(x) = \max(0, x) \quad (3.3)$$

From Equation 3.3, it follows that only the neurons with positive values are kept, while the neurons with negative values are deactivated by converting their values into zero. ReLU activation function increases the computational efficiency substantially by reducing the number of active neurons in the network.

Details of activation function can be found in Appendix E, including:

- Linear activation function
- Binary step function
- Nonlinear activation functions

### Pooling Layer

Pooling layer is a vital part during CNN model training [36][37][38]. The function of pooling layer is to reduce the spatial size of the previous feature map and gain more meaningful information, which results in:

- Reducing the computing time,



- Removing some noise,
- Increasing the robustness of learned features,
- Reducing overfitting.

There are mainly two types of pooling layers: max pooling and average pooling. With max pooling, the maximum value of the neighbourhood, which is covered by the filter at one time, is picked as the value of output neuron. For average pooling, the average value of the neighbourhood is selected. The processes of max pooling and average pooling, with  $2 \times 2$  filter size and stride 2 on both vertical and horizontal directions are demonstrated in Figure 3.4.

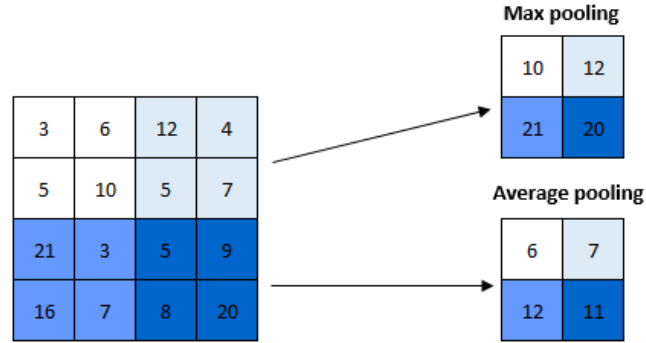


Figure 3.4: Pooling process for max pooling and average pooling

### Dropout layer

Co-adaptation is one of the the major issues during module training, especially for large networks [39]. During module training all the weights are learned at the same time. Some connections gain more predictive capability than the others. In such case, after iterative training, the powerful connections are learned more while the less powerful connections become weaker and ignored. This is called co-adaptation. Co-adaptation causes the network works good for training data, but not generalize to unseen data, so called overfitting.

Dropout is one of the regularization methods which can be used to reduce overfitting and improve generalization [40]. It is computationally cheap, but very effective. Dropout

randomly drops or ignores a number of neurons and its incoming and outgoing connections temporarily during model training, as demonstrated in Figure 3.5. Each neuron is assigned with a fixed dropout probability which is independent of any other neurons. Dropout is applied only for network training, not for validation or prediction.

During model training, the weights and biases are tuned for extracting the expected fea-

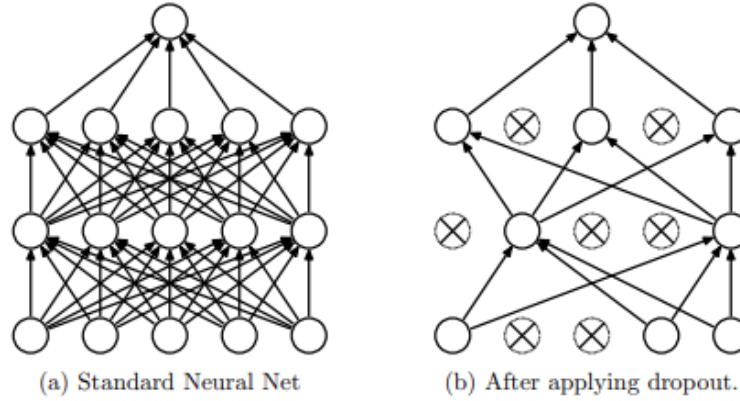


Figure 3.5: Dropout process. **Left:** Standard network without dropout. **Right:** Network with dropout applied to 3 first layers, including one input layer and two hidden layers. The crossed nodes are dropped

tures from the samples. If the model relies on all neurons and connections, it is prone to overfitting, and the model is specialized to only the training data, not being able to generalize well for new, previously unseen data. When some neurons are randomly dropped out of the network, the connections of other neurons have to be readjusted, resulting in lower sensitivity to weights of specific neurons for the purpose of better generalization and less overfitting.

### Batch Normalization

Training DNNs requires lots of time and resources; therefore, techniques such as batch normalization [1][41] provide an efficient way for accelerating the convergence. During model training, batch normalization calculates the mean and the standard deviation of the layer inputs for each mini-batch. The input of the activation layer then subtracts the batch mean and it is divided by the standard deviation estimated in the current minibatch. Batch normalization (BM) transform algorithm is shown in Figure 3.5.

Deep neural networks (DNNs) with more layers can benefit more from batch normaliza-

tion. With standardizing the inputs, batch normalization:

- Stabilizes the learning process and enables the model training with higher learning rate,
- Increases the robustness of the neuron network against different initialization schemes and learning rates,
- Accelerates the model training and reduces the number of epochs,
- Similar to dropout, has a slight regularization effect to control overfitting by adding some noise within the mini-batch.

Batch normalization(BM) transform algorithm is shown in Table 3.1.

<b>Input:</b> Values of x over a mini-batch: $\beta = \{x_1 \dots x_m\}$ ;	
Parameters to be learned: $\gamma, \beta$	
<b>Output:</b> $\{y_i = \mathbf{BN}_{\gamma, \beta}(x_i)\}$	
$\mu_\beta \leftarrow \frac{1}{m} \sum_{i=1}^m x_i$	//mini-batch mean
$\sigma_\beta^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_\beta)^2$	//mini-batch variance
$\hat{x}_i \leftarrow \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}$	//normalize
$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \mathbf{BN}_{\gamma, \beta}(x_i)$	//scale and shift

Table 3.1: Batch Normalizing Transform, applied to activation x over a mini-batch. [1]

## Flatten layer

In general, CNNs have two functional parts. The first part takes the responsibility of feature extraction, whereas the second part takes the features extracted from the first part for classification. Flatten layer is located between the feature extraction part and the classification part, and it converts the multi-dimensional data, which is the output of the feature extraction part, into one-dimensional data, which is the desired format for the classification part. The process is demonstrated in Figure 3.6.

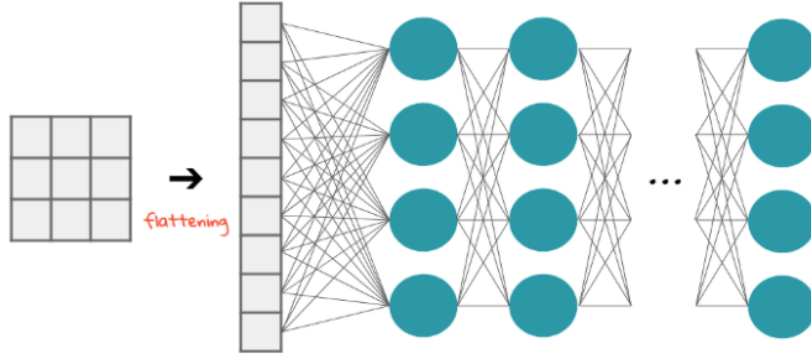


Figure 3.6: Flattening

### Fully Connected Layer

Fully connected(FC) layer (sometimes also called dense layer) connects every neuron in the current layer with every activation in the previous layer. After the feature extraction and flattening, the one-dimensional vector goes through one or more fully connected layers, to perform the task of classification. FC layers, with numerous parameters and nonlinear activation functions map the input data to its expected output label during the training. After the convolutional layer, it is the second most computationally consuming layer in CNNs [42].

The processes in FC layer are similar with the processes in convolutional layer. A list of the differences and similarities of convolutional layers and fully connected layers is given below:

- Neurons in the fully connected layer are fully connected with all the activations of the previous layer. Neurons in convolutional layer are only connected to the activations in the local region of the previous layer.
- Convolutional layer shares the parameters using the filter, while fully connected layer does not share the parameters.
- Both of them use activation functions, which can be found in [Appendix E](#)
- Both of them use dot products.

### Backpropagation

The fully connected layers learn a (possibly non-linear) mapping between the high-level features given as an output of the convolutional layers and the labels. The fully con-

nected part of the CNN network uses backpropagation process to determine the most optimal network parameters(weights and biases). Backpropagation algorithm calculates the gradients of the cost function with respect to all network weights backwards through the network, starting from the output layer and ending in the input layer. Partial computations of the gradient in one layer are reused for the gradient computation in the previous layer, allowing more efficient network training.

## Overfitting

Deep neural networks with multiple non-linear hidden layers and a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks. Overfitting occurs when the model performs very well on the training dataset, but it is not able to generalize well for new, previously unseen data, leading to poor prediction. This happens usually when the model is too complex for the given task. Douglas M. Hawkins gives further explanation in [43]. Figure 3.7 illustrates overfitting.

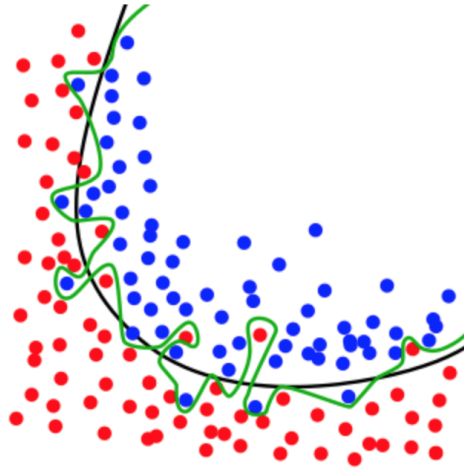


Figure 3.7: The black line well fits the data well, while the green line over fits the noises in the dataset.

There are several methods to control overfitting:

- Training the model with bigger dataset,
- Removing noise and outliers from dataset,

- Early stopping, before validation loss increases,
- Adding regularization. Regularization refers to the techniques which can be used to artificially force the simplification of the model (for example, dropout, batch normalization and weight regularization).

All the above methods are adapted to decrease overfitting in this study at different stages.

### 3.2.2 Support Vector Machines

Support Vector Machines(SVM), introduced by Vapnik et al. [44], is a widely used traditional machine learning algorithm. It is the most popular binary classifier, which provides both linear and non-linear solution, and can be also adapted to deal with multi-class classification problems.

Given a training dataset with binary labelling, each sample represents a data point in a  $n$ -dimensional space. The objective of SVM training algorithm is to construct a hyperplane (decision boundary) which can separate the data points of the two categories and achieve the maximum distance or margin between the data points from two categories. Margin refers to the distance between decision boundary and data points closest to it from both categories. Figure 3.8 demonstrates two possible separators in a two-dimensional space linear classification.

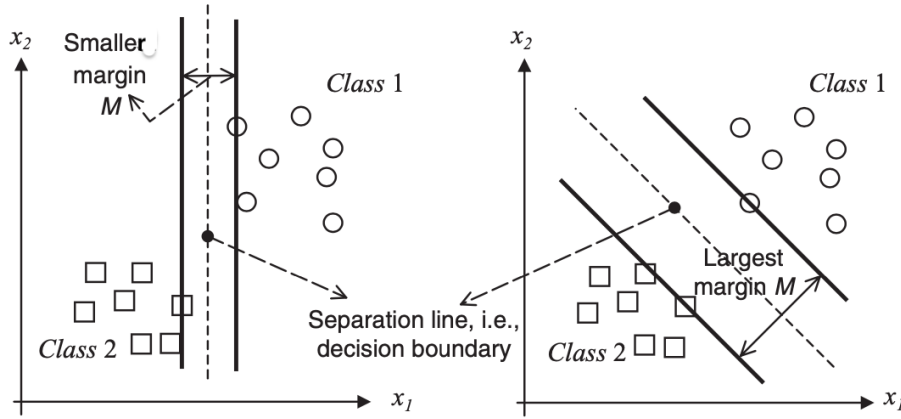


Figure 3.8: Two separating lines: the one with the maximum margin is the preferred separating hyperplane(on the right) to the one with a smaller margin(on the left). Image is taken from [2] with minor modification.

**Linear SVM.** Given a training dataset with  $n$  sample points, such as:

$$(\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n), \quad (3.4)$$

where  $\vec{x}_i$  is  $p$ -dimensional real vector, and  $y_i \in \{1, -1\}$ , indicating the class of  $\vec{x}_i$ , any hyperplane can be written as:

$$\vec{\omega} \cdot \vec{x} - b = 0, \quad (3.5)$$

where  $\vec{\omega}$  is the normal vector of the hyperplane. Therefore  $\frac{b}{\|\vec{\omega}\|}$  is the hyperplane offset from the origin along  $\vec{\omega}$ .

**Hard-margin.** Two parallel hyperplanes with the possible maximum distance are separating the two classes of the dataset. Margin is the region between these two hyperplanes, and the maximum-margin hyperplane is the third parallel hyperplane which equally separates the maximum region. After dataset normalization, the hyperplanes can be described as:

$$\vec{\omega} \cdot \vec{x} - b = 1, \quad (3.6)$$

and

$$\vec{\omega} \cdot \vec{x} - b = -1. \quad (3.7)$$

Accordingly, the width of the maximum margin is  $\frac{2}{\|\vec{\omega}\|}$ , which means when the minimum  $\|\vec{\omega}\|$  is obtained, we reach the maximum margin. The following constraints are used to make data points fall into the correct side of the margin. For each data point  $i$ :

$$\vec{\omega} \cdot \vec{x}_i - b \geq 1, \text{ if } y_i = 1, \quad (3.8)$$

or

$$\vec{\omega} \cdot \vec{x}_i - b \leq -1, \text{ if } y_i = -1, \quad (3.9)$$

the data points which are nearest to the maximum-margin hyperplane are called support vectors.

**Soft-margin.** When the data points are not linearly separable, hinge loss function is used:

$$\max(0, 1 - y_i(\vec{\omega} \cdot \vec{x}_i - b)), \quad (3.10)$$

where  $y_i$  is the  $i$ -th target, and  $\vec{\omega} \cdot \vec{x}_i - b$  is the  $i$ -th output. When 3.8 and 3.9 are satisfied, this function equals to zero. The following function needs to be minimized for

the purpose of optimization:

$$\left[ \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i(\vec{\omega} \cdot \vec{x}_i - b)) \right] + \lambda \|\vec{\omega}\|^2, \quad (3.11)$$

where the trade-off between margin maximization and the constraints of 3.8 and 3.9 is controlled by parameter  $\lambda$ .

### 3.2.3 Nonlinear classification

Linear separation is mostly computationally cheap. However, in real life, data points are often not linearly separable in original input space [45], meaning that the data points belonging to different classes are distributed in the input space in such way that it is not possible to separate them with a linear hyperplane. SVM transforms the data points into a higher-dimensional feature space, in which a linear separation might be possible.

The most popular kernels are listed in Table 3.2. Radial Basis Function (RBF), also

Table 3.2: Popular kernels in SVM, from [2]

Kernel Functions	Type of Classifier
$K(x, x_i) = (x^T x_i)$	Linear, dot product, kernel, CPD
$K(x, x_i) = [(x^T x_i) + 1]^d$	Complete polynomial of degree d, PD
$K(x, x_i) = e^{\frac{(x-x_i)^T}{2\sum(x-x_i)}}$	Gaussian RBF, PD
$K(x, x_i) = \tanh[(x^T x_i) + b]^*$	Multilayer perceptron, CPD
$K(x, x_i) = \frac{1}{\sqrt{\ x - x_i\ ^2 + \beta}}$	Inverse multiquadric function, PD

\*only for certain values of b, (C)PD = (conditionally) positive definite.

called Gaussian Kernel, is used in this study.

## 3.3 Dataset

The dataset used in this research was collected in the frame work of the mPower study (Mobile Parkinson Disease Study) [3]. It was an observational smartphone-based study for the purpose of evaluating the feasibility of diagnosing and estimating the stage of Parkinson's disease of the participants with remotely collected information. Current released dataset is the first six months donation of participants between March 9 and



September 9, 2015. The mPower app was launched through Apple App Store and it is available only in United States. It was open for enrollment to individuals with PD professional diagnosis, as well as to non-PD participants used as control cases.

A dashboard of tasks, including 3 surveys and 4 activity tasks, were presented to participants after enrollment. It was possible to skip any questions or activity tasks at any time. A baseline survey, Demographics, was the first task. It was one time task for the collection of participants' general information. The other two surveys were expected to be filled monthly. They were standard survey of Parkinson Disease Questionnaire 8 (PDQ-8) and subset of questions from the Movement Disorder Society Universal Parkinson's disease Rating Scale (MDS-UPDRS). The information collected with Demographics survey is used in this work to identify PD patients and non-PD participants. Diagnosed year is used to classify the patients into groups: early-stage cases or developed cases.

The four separate activity tasks include memory, tapping, voice and walking task. These tasks were expected to be practiced 3 times a day per participant. The participants identified as PD patients with professional diagnosis were asked to perform these activity tasks, and as well indicate under which following circumstances they take the tasks:

- Immediately before taking their medication.
- Just after taking Parkinson medication (when they are feeling at their best).
- Another time (at some other time).
- I don't take Parkinson medications.

The data collected with voice activity task is used for this research. Participants were instructed to wait for 5s countdown time before start to say /a/ into the smart phone embedded microphone at a steady volume for 10s. The data of this activity contains audio files of 10s phonation, as well as the 5s countdown period recorded with iPhone microphone. The voice signals were recorded at 44.1Kbps sample rate and saved in .m4a format.

From Table 3.3 , we can see 6,805 participants finished Demographics survey, and 5,826 participants took voice activity task for 65,022 times in total.

Audio records only from participants who claim themselves as 'I don't take Parkinson medications' are considered for model training and testing. The rationale behind this is that medications for PD can reduce the voice disorder symptoms and the degree of

Task Name	Type of Task and Schedule	Unique Participants	Unique Tasks
Demographics	Survey-once	6,805	6,805
PDQ8	Survey—monthly	1,334	1,641
UPDRS	Survey—monthly	2,024	2,305
Memory	Activity—t.i.d.	968	8,569
Tapping	Activity—t.i.d.	8,003	78,887
Voice	Activity—t.i.d.	5,826	65,022
Walking	Activity—t.i.d.	3,101	35,41

Table 3.3: Data available for each survey and activity completed by study participants[3]

deduction cannot be qualified.

There are some audio files with incomplete or inconsistent information, which includes:

- Professional diagnosis is FALSE and diagnosis-year is filled.
- Professional diagnosis is TRUE and diagnosis-year is empty.
- The circumstance under which the task is taken is not indicated.

With all the 65,022 audio files, the data distribution regarding the professional diagnosis and the circumstance, under which the task is taken, is given in Table 3.4. The audio files, taken from both non-PD participants (professional diagnosis is FALSE) and PD patients (professional diagnosis is TRUE), are marked with bold characters in the table.

Data distribution of audio files indicated with "I don't take Parkinson medications" is given in Appendix B.

Due to the fact that the audio files are collected using a mobile phone in uncontrolled conditions, audio files that are severely corrupted by noise are excluded from the model training in this study. Audio files were excluded in the following cases:

- No sound was recorded in the audio file.
- Instruction of using steady volume was not followed.
- Background noise is dominant, for example: sound from TV, other people talking, sound from pets.

Professional Diagnosis	medTimepoint	No. of Audio Files
FALSE		175
FALSE	Another time	114
<b>FALSE</b>	<b>I don't take Parkinson medications</b>	<b>22810</b>
FALSE	Immediately before Parkinson medication	63
FALSE	Just after Parkinson medication (at your best)	43
TRUE		52
TRUE	Another time	17541
<b>TRUE</b>	<b>I don't take Parkinson medications</b>	<b>2849</b>
TRUE	Immediately before Parkinson medication	8957
TRUE	Just after Parkinson medication (at your best)	9727

Table 3.4: Data distribution

One 5-second countdown audio file in the dataset corresponds to one 10-second phonation audio file. The countdown audio files can theoretically be used as the noise pattern of corresponding phonation file for denoising. However, after checking the countdown files individually, they were not used for denoising, since the countdown recordings include:

- Coughing to clean throat.
- Sound of deep breathing.
- Sound of finding a good position to record the phonation, etc.

The dataset for this research is available on Synapse<sup>1</sup>, which is a general-purpose data and analysis sharing server. Enrollment to become a certified user are mandatory for the purpose of downloading and using the data<sup>2</sup>. Certain constraints must be followed with respect to the privacy of the participants.

Links of the dataset can be found in Appendix C. Demographics survey (information given in Appendix A) and Voice Activity general information can be downloaded as two .csv files directly from the data citation links in Appendix C. Audio files can only be downloaded separately. Each audio file was downloaded into one folder named with the serial number of the audio file in the Voice Activity .csv file. After downloading the audio

<sup>1</sup>Link: <https://www.synapse.org>

<sup>2</sup>Instructions link: [https://docs.synapse.org/articles/getting\\_started.html](https://docs.synapse.org/articles/getting_started.html)

files were rename and reorganised accordingly. Related information of environment setup and codes can is given in AppendixD.

### 3.4 Dataset Splitting

To build a computational model with high prediction accuracy and good generalization is the main task of machine learning [46]. At the end of the model training, the model is expected to learn from the training data and build an effective mapping from the input data to the output data. To be able to achieve this goal, the model is initially fit on a training dataset to learn the optimal model parameters (e.g. weights and biases in a neural network). The performance of the trained model is further evaluated on the validation dataset during training in order to select the optimal hyperparameters of a model (e.g. the number of hidden layers or the number of hidden units in each layer of a neural network). Finally, the independent test dataset previously not seen by the model during training is used to provide a final unbiased evaluation of a model performance. A well distributed training, validation and test dataset is vital for CNN model training. The following techniques are used for training and testing dataset splitting.

#### 3.4.1 Simple Random Sampling (SRS)

Simple random sampling is the most used method. During dataset splitting, each sample is assigned an equal probability of selection and is selected randomly with uniform distribution [47]. This method is effective to reduce bias of the model [48].

#### 3.4.2 K-fold Stratified Cross Validation

Cross validation is used when the number of available samples are relatively small. It is a resampling procedure to estimate the model performance on unseen data with limited samples. It is usually called k-fold cross validation, where k is a chosen specific value.

In stratified cross validation each fold has the same proportion of observations classified into each class, i.e. each class is approximately equally distributed across all folds. This ensures that each fold is a good representative of the dataset.

The general procedure of dataset splitting in K-fold cross validation is as follows:

1. Randomly split the whole dataset into  $k$  parts of the same size. Random splitting is the same concept as in SRS.
2. Take one part as validation dataset and all the other parts as training dataset, so that there are  $k$  times different splittings into training dataset and validation dataset.
3. Train models training separately with different combinations of training and validation subsets (different folds).

### 3.4.3 Donor-based Data Splitting

So far in most of the previous studies of Parkinson’s disease detection, because of the limited number of available donors and samples, during data splitting, the donors of samples are not taken into account, only the sample are randomly split into training, validation and test subsets.

Leave-one-subject-out [49], or leave-one-individual-out [50] scheme used in [11] takes care additionally that the donors of samples that appear in the validation and/or test datasets are not previously seen in the training dataset. When there are multiple samples from one donor, all samples of one donor should belong to either training dataset or validation dataset. Two data splitting schemes, donor-based data splitting and sample-based data splitting, will be applied with the dataset in this study to analyse the effects of these two data splitting schemes.

## 3.5 Evaluation and Performance Measures

The performance of the model for discrimination of Parkinson’s disease patients and Healthy Control participants is evaluated with validation accuracy, sensitivity and specificity.

During machine learning model training, one major task is to decrease overfitting. This means to train the model not only fitting well the training data, but also generalizing well and being able to give accurate prediction for testing or unseen data. Therefore, testing or validation accuracy is a valuable measure for the prediction ability of machine learning model. Accuracy can be defined as:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (3.12)$$

Sensitivity and specificity are statistical measures[51] used to evaluate clinical diagnostics and screening tests.

Sensitivity is a measure of the ability to correctly identify an individual with the disease.

It can be defined as:

$$Sensitivity = \frac{TP}{TP + FN} \quad (3.13)$$

Specificity is a measure of the ability to correctly identify an individual without the disease. It can be defined as:

$$Specificity = \frac{TN}{TN + FP} \quad (3.14)$$

True Positive(TP) refers to PD patients correctly identified as ill. False Positive(FP) refers to healthy participants incorrectly identified as ill. True Negative(TN) refers to Healthy participants correctly identified as healthy. False Negative(FN) refers to PD patients incorrectly identified as healthy.

## Chapter 4

# Experiments and Results

In this study, three frameworks are proposed for identification of Parkinson’s disease using only speech signal as a biomarker. In the first framework, mel-spectrograms of voice records are fed into multiple layer Convolutional Neural Network model for training, tuning and classification. In the second proposed framework, the CNN model trained on mel-spectrograms was used only for feature extraction, in order to learn higher-level feature representations from speech, which were further fed into SVM for classification. In the third framework, the mel-spectrograms are input directly into a fully connected network which is used as the final, dense classification layers of the CNN model from the first framework, i.e. without using convolutional layers for feature learning. The idea was to assess what is the overall contribution of features learned by CNN in identification of PD.

### 4.1 Data Preparation

The data used in this study from mPower includes four parts:

- Demographics Survey data in .csv file
- Voice Activity data in .csv file
- 10-second phonation .m4a audio files
- 5-second count-down .m4a audio files

#### 4.1.1 Demographics Survey

The information of Demographics Survey can be downloaded via Synapse website and stored as the comma-separated value (.csv) file. Information for downloading the survey is available in Appendix C. Detailed information about the Demographics Survey is given in Appendix A. The variables from the downloaded .csv file, which are used in this study, are listed in Table 4.1. The data in the .csv file is imported into MySQL database for further processing and analysis. There are 6805 records in total imported into database, such that one record refers to one donor.

MySQL workbench 8.0 version 8.0.19 is used for implementation.

Variable name	Description
healthCode	Technical identification number for each donor.
professional-diagnosis	Parkinson's disease diagnosed for the participant in the study by medical professional.
diagnosis-year	The year in which the participant is diagnosed Parkinson's disease by medical professional, if applicable.

Table 4.1: Used variables from Demographics Survey

#### 4.1.2 Voice Activity

The data of Voice Activity can be download via Synapse website stored as the comma-separated value (.csv) file. Information for downloading the Voice Activity data is available in Appendix C. The file contains mostly technical information related to identification of the donor and the audio files. The list of variables used in this study from is given in Table 4.2. The data in the .csv file is imported into MySQL database into one table named voice\_activity for further processing. 65022 records are imported in total, such that one record contains the information for one voice activity task.

#### 4.1.3 Phonation audio files

The 10-second phonation /a/ audio files were downloaded with Python code in Appendix D.3. Each file is approximately 1.3 MB in size. In total, the audio files occupy 87.16 GB



Variable name	Description
healthCode	Identification code for each donor.
audio_audio.m4a	Sequence number of 10s-phonation audio file.
audio_countdown.m4a	Sequence number of 5s-countdown-time audio file.
medTimepoint	Circumstances when the task is taken: <ul style="list-style-type: none"> <li>• Immediately before taking their medication.</li> <li>• Just after taking Parkinson medication (when they are feeling best).</li> <li>• Another time (at some other time).</li> <li>• I don't take Parkinson medications</li> </ul>

Table 4.2: Used variables from Voice Activity

of storage space. After downloading, the audio files are named with random numbers, and are separately saved in a multi-level folder structure, which is demonstrated in Figure 4.1. The name of the last level folder is the sequence number of the phonation audio file.

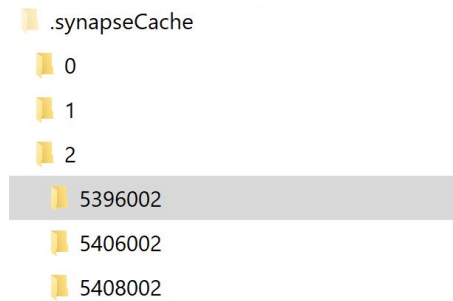


Figure 4.1: Downloaded audio files' folder structure

#### 4.1.4 Countdown audio files

The 5-second countdown audio files have size of approximately 600 KB each, or 39.14 GB in total. The sequence numbers of countdown audio files are saved under variable named `audio_countdown.m4a` in table `voice_activity`.

## 4.2 Training datasets

### 4.2.1 Data availability and feasibility

As we discussed in the previous chapter, only audio files which are indicated as "I don't take Parkinson medications" are considered in this study. These are:

- 2849 sustained vowel phonation audio files taken from 133 PD patients, which satisfy the following conditions:
  - Professional-diagnosis is TRUE.
  - Diagnosis-year is indicated.
  - medTimepoint is equal to: I don't take Parkinson medications.
- 22810 sustained vowel phonation audio files taken from 3844 non-PD participants, which satisfy the following conditions:
  - Professional-diagnosis is FALSE.
  - Diagnosis-year is kept NULL.
  - medTimepoint is equal to I don't take Parkinson medications.

Due to the fact that these audio files are recorded in uncontrolled conditions, they are more or less corrupted by different patterns of noise. In order to reduce the noise and control the complexity of the model architecture, the phonation audio files are listened individually to select the relatively clean ones for training. The information from two tables (demographics\_survey and voice\_activity) is joined with healthCode to identify the scope of the selection process and extract the sequence number of the audio files, which is the name of the audio file after data preparation.

507 phonation audio files with less noise from each class, PD patients and non-PD participants, are selected for the clean dataset, 1014 audio files in total.

During the selection, when it was difficult to assess the level of background noise, the corresponding 5-second countdown audio file is used for verification.

One extra variable, named cat, is added in table voice\_activity to indicate the category of the phonation audio files. Relatively clean audio files are indicated with "c" or "1c".

In the preselected clean audio dataset, there are 507 audio files from 362 non-PD participants, 390 audio files from 41 early stage PD patients<sup>1</sup>, and 117 records from 25 PD patients<sup>2</sup> with progressed disease. Details about the data distribution can be found in Table 4.3.

Professional Diagnosis	Diagnosis Year	No. of Participants	No. of Sample Cases
FALSE		362	507
TRUE	1976	1	8
TRUE	1982	1	1
TRUE	1998	1	3
TRUE	2000	2	10
TRUE	2004	1	1
TRUE	2005	2	3
TRUE	2006	1	2
TRUE	2009	1	1
TRUE	2010	2	4
TRUE	2011	5	19
TRUE	2012	8	65
TRUE	2013	11	131
TRUE	2014	23	186
TRUE	2015	7	73
TRUE	In total	67	507

Table 4.3: Data distribution of participants and sample cases used regarding the Diagnosed Year

---

<sup>1</sup>‘Early stage PD patients’ used in this paper refers to participants who self-reported as being professionally diagnosed in 2013, 2014, 2015 or 2016. The data was collected in 2015 and 2016, which means these PD patients are diagnosed within 3 years.

<sup>2</sup>‘Developed PD patients’ used in this paper refers to participants who self-reported as being professionally diagnosed in 2012 or before.

### 4.2.2 Training datasets

In general, four datasets are proposed for model training and testing in this study. They are named:

- All500 dataset
- EarlyCase dataset
- ProgressedCase dataset
- All2000 dataset

The first three datasets are used for the model training in each of the first three frameworks. All2000 dataset is used only for model training in the first framework.

The number of available audio files from non-PD participants is much higher than the number of available audio files from PD patients. In order to make the best use of available data and at the same time keep the balance of two classes in the training dataset, all the available audio files from PD patients are used, and the number of audio files from non-PD patients are selected to match the number of the audio files from PD patients for All500 dataset and All2000 dataset.

#### All500 dataset

This dataset contains all the preselected clean audio files in two classes:

- 507 audio files from PD patients
- 507 audio files from non-PD participants

The number of donors and the number of audio files in both classes are given in Table 4.4.

Professional Diagnosis	No. of Donors	No. of Sample Cases
FALSE	362	507
TRUE	66	507

Table 4.4: Data distribution of All500 dataset

### EarlyCase dataset

This dataset contains a part of the preselected clean audio files in two classes:

- 390 audio files from early stage PD patients.
- 507 audio files from non-PD participants

The number of donors and the number of audio files in both classes are given in Table 4.5.

Professional Diagnosis	No. of Donors	No. of Sample Cases
FALSE	362	507
TRUE	41	390

Table 4.5: Data distribution of EarlyCase dataset

### ProgressedCase dataset

This dataset contains a part of the preselected clean audio files in two classes:

- 117 audio files from developed PD patients.
- 507 audio files from non-PD participants

The number of donors and the number of audio files in both classes are given in Table 4.6.

Professional Diagnosis	No. of Donors	No. of Sample Cases
FALSE	362	507
TRUE	25	117

Table 4.6: Data distribution of ProgressedCase dataset

### All2000 & ReAll2000 dataset

These two datasets contain the same samples, while split with different data splitting method. All2000 is split with sample-based data splitting method, and ReAll2000 is split

with donor-based data splitting method. All the available samples from PD patients are adopted for model training without preselection. The same number of samples from Non-PD participants are adopted to keep a balanced dataset. The dataset consists of:

- 2824 audio files from PD patients
- 2824 audio files from non-PD participants

The number of donors and the number of audio files in both classes are given in Table 4.7. Audio files with length less than 7 seconds are discarded during data preprocessing

Professional Diagnosis	No. of Donors	No. of Sample Cases
FALSE	758	2849
TRUE	133	2849

Table 4.7: Data distribution of All2000 dataset

which will be explained in the next section. That is the reason in this dataset the exact number of audio files from PD patients, which is used for model training, is 2824, instead of 2849.

### 4.2.3 Data splitting

Scikit-learn [52] is used for data splitting. During model training to achieve the best performance architecture, Simple Random Sampling is applied for data splitting with ratio of 4:1 for training and validation dataset. Taking into account the limited number of data samples, 5-fold stratified cross validation is applied on the first three datasets to evaluate the performance of previously defined architecture.

Due to a large number of samples (more than 2000) in each class in All2000 dataset, cross validation is not applied for this dataset. Two schemes of data splitting, donor-based data splitting and sample-based data splitting, are applied to All2000 dataset. Both follow the principle of Simple Random Sampling. Sample-based data splitting uses SRS based on samples, while donor-based data splitting uses SRS based on donors.

During donor-based data splitting, firstly, donors are randomly distributed to either training or validation datasets. In the donor distribution algorithm, the number of samples of to-be distributed donor and the total number of samples representing the already-distributed donors need to be considered to keep the splitting ratio of training

and validation dataset. The samples are distributed into training or validation dataset according to the distribution of their representing donors.

### 4.3 Data Preprocessing

Converting the audio files of our dataset into corresponding mel-spectrograms is a mutual step for all the proposed frameworks. Python library Librosa [53] for music and audio analysis is used to transform audio files into mel-spectrograms.

#### Sample rate

The original sample rate of the audio files is 44100bps. The sample rate is kept to avoid losing any useful information, when audio files are loaded with Librosa library.

#### Window size

The time-frequency resolution of the generated Mel-Spectrogram is dependent upon the chosen window size. A larger window size will result in higher frequency resolution, but lower temporal resolution, whereas the opposite will result in a lower frequency resolution, but higher temporal resolution[29].

Librosa recommends window size of 2048 for music signals for better frequency resolution, and window size of 512 for speech signals for better temporal resolution. Our samples are sustained vowel phonations /a/ with steady volume and no separations of words; therefore, the frequency is expected to be continuous. Accordingly, the exact edges of the frequency changes are not critical to be observed for analysis, and high temporal resolution is not necessary. With the reason given above, window size of 2048 is adopted in our experiments.

#### Frequency scale

In this study, we adopt 4096 Hz as the upper bound of the frequency range with mel-spectrogram. The maximum frequency never exceeds half the sample rate, but lower frequency is chosen to hide the unimportant higher frequencies and focus on the more important lower frequencies.

## Image Size

The inputs of the CNN model are the mel-spectrograms as discussed previously in this study. The resolution of the generated mel-spectrograms is one of the factors which can decide if the CNN model has enough information for training to get the best performance. Contrarily, high resolution and high dimension will need more complex CNN model structure to deal with, and both increases dramatically the training time. RGB mel-spectrograms of size  $144 \times 144$  are adopted in this study as the trade-off of above two factors.

## 4.4 Model Training

Model training in this study is implemented with Keras [54] on Google Colab [55].

### 4.4.1 CNN model

A full Convolutional Neural Networks architecture is employed in the first framework for feature extraction and classification for PD detection. Different from transfer learning method, it is a self customised architecture.

#### CNN baseline model

The architecture of the layers for feature extraction and classification is motivated by the research work in [27]. The CNN model, in [27], shares the same task, Parkinson's disease detection, and was trained on spectrograms transferred from mPower sub-dataset. The sub-dataset contained 500 samples, though the sample extraction criterion was not specified. The architecture in [27] referred to ILSVRC<sup>3</sup> 2012 winner AlexNet network[18]. AlexNet has five convolutional layers, with 3 max pooling layers in between, followed by three fully connected layers. AlexNet was trained on ImageNet dataset [56]. Applying AlexNet with the sub-dataset of mPower dataset resulted in very high training accuracy and low validation accuracy in [27]. It was explained by the fact that the deep neural network architecture of AlexNet is trained with much larger dataset comparison to the one used in [27], as well as the ones in this study, and AlexNet is created for more

---

<sup>3</sup>ImageNet Large Scale Visual Recognition Challenge



complex feature extraction tasks, therefore the overfitting occurs. Accordingly, in [27], the number of the layers was reduced and the hyperparameters were adjusted to make a simpler model and reduce overfitting. The architecture in [27] is given in Table 4.8.

Input: input_shape (200×200×3)
Conv2D(64, kernel_size=(10,10),strides=(2,2),activation='relu')
Conv2D(64, kernel_size=(10,10),strides=(2,2),activation='relu')
MaxPooling2D(pool_size=(2,2),strides=(2,2))
Flatten()
Dense(1000, activation='relu')
Output layer: Dense(2, activation='softmax')

Table 4.8: Architecture of CNN baseline model

To check the performance of the architecture with the first dataset All500 from [27], input\_size is adjusted to (144×144×3):

- The exact architecture from [27] is applied to check the performance of the architecture with the dataset prepared in this study.
- The same architecture, with kernel\_size of both convolutional layers adjusted from (10×10) to (9×9), is applied to check if there is still margin for improving.

The best performance of 70 epoch is given in Table 4.9. From the results presented in

kernel_size	Training Accuracy(%)	Validation Accuracy(%)
(9×9)	62.35	56.86
(10×10)	99.38	68.63

Table 4.9: Model training performance

Table 4.9, we can conclude that:

- The architecture from [27] is deep enough to develop complex mappings from input to output, reaching 99.38% training accuracy.
- The validation accuracy is much lower than the training accuracy, showing that overfitting is high. The model does not generalize well to validation dataset.
- Small adjustment of kernel\_size results in a large drop of both training accuracy and validation accuracy.

According to the analysis above, the baseline model can be used only as a starting point, but the steps deal with overfitting need to be undertaken.

## Reducing Overfitting

Different methods can be applied to reduce overfitting. Let us first analyze how AlexNet controls or prevents overfitting. In the architecture of AlexNet, there are no regularization layers, which are obviously responsible to reduce overfitting. However with abundance of training data and image augmentation, AlexNet managed the state\_of\_art performance for image recognition task. In this study, the following procedures are considered and proceeded:

- Training dataset. The size of the training data can not be enlarged, and furthermore large number of audio sample are corrupted by noise. Therefore, we decided to preselect only the relatively clean samples, leading to the All500 dataset for model training.
- Keras.callbacks.ModelCheckpoint is applied to save the model with the best validation accuracy during model training.
- Regularization layers are added. The experiment results are given below.

### – Dropout layers

The probability of a node being dropped is adjusted with hyperparameter. In Keras, the hyperparameter of dropout function is the percentage of the nodes being dropped out in the input features map.

From the performance given in Table 4.10, we can conclude:

- \* The dropout layers should not be added after the first convolutional layer according to experiment No.5, since the performance drops substantially both for training and validation dataset.
- \* Dropout(0.4) or dropout(0.5) after maxpooling layer, and dropout(0.2) after dense(1000) layer help to reduce the overfitting and improve performance.

When multiple dropout layers are added into the baseline model, from the performance given in Table 4.11, we have:

<b>N o.</b>	<b>Layer Added</b>	<b>Added Position Behind</b>	<b>Training Accuracy (%)</b>	<b>Validation Accuracy(%)</b>
	<b>Baseline</b>		<b>99.38</b>	<b>68.63</b>
1	Dropout(0.1)	Maxpooling	61.36	62.75
2	Dropout(0.2)	Maxpooling	83.09	62.75
3	Dropout(0.4)	Maxpooling	100	78.43
4	Dropout(0.5)	Maxpooling	92.84	77.94
5	Dropout(0.2)	1st Conv	51.11	50.98
6	Dropout(0.2)	Dense(1000)	98.27	77.45

Table 4.10: Model training performance with Dropout Layers (performance of baseline model is also added for comparison)

- \* Dropout should not be added after the first convolutional layer, according to experiment No.7 and No.8, since the performance drops substantially both for training and validation dataset.
- \* Adding dropout(0.2) after the second convolutional layer dramatically improves the performance, according to experiment No.2 and No.8.
- \* Adding 3 dropout layers in experiment No.9 after the second convolutional layer, max pooling layer and dense layer improves the performance.
- \* Dropout rate percentage should not exceed 0.2 after the dense(1000) layer, according to experiment No.10, No.12 and No.13.
- \* Dropout rate should not exceed 0.2 after the second convolutional layer, according to experiment No.11 and No.12.
- \* Experiment No.14 with four dropout layers added after the second convolutional layer, two pooling layers and dense layer reaches the best performance.

#### – Layer Weight Constraints

Layer weight constraints are added to the architecture of experiment No.12. Hyperparameter maxnorm is set to be 3, kernel\_constraint=maxnorm(3). Performance is given in Table 4.12.

With even the best result in this group of experiments, the performance is not improved. Therefore, kernel\_constraint is not adopted.

<b>N o.</b>	<b>Layer Added</b>	<b>Added Position Behind</b>	<b>Training Accuracy (%)</b>	<b>Validation Accuracy(%)</b>
	<b>Baseline</b>		<b>99.38</b>	<b>68.63</b>
7	Dropout(0.2)	1st Conv	50.74	51.96
	Dropout(0.2)	2nd Conv		
8	Dropout(0.2)	2nd Conv	100	77.94
	Dropout(0.2)	Maxpooling		
9	Dropout(0.2)	2nd Conv	98.40	83.33
	Dropout(0.4)	Maxpooling		
	Dropout(0.2)	Dense(1000)		
10	Dropout(0.2)	2nd Conv	50	50
	Dropout(0.2)	Maxpooling		
	Dropout(0.4)	Dense(1000)		
11	Dropout(0.4)	2nd Conv	97.16	77.45
	Dropout(0.4)	Maxpooling		
	Dropout(0.2)	Dense(1000)		
12	Dropout(0.4)	2nd Conv	80.62	65.20
	Dropout(0.4)	Maxpooling		
	Dropout(0.4)	Dense(1000)		
13	Dropout(0.4)	Maxpooling	65.80	50
	Dropout(0.4)	Dense(1000)		
14	Dropout(0.2)	2nd Conv	98.75	87.25
	Dropout(0.4)	Maxpooling		
	Dropout(0.2)	Maxpooling		
	Dropout(0.2)	Dense(1000)		

Table 4.11: Model training performance with Dropout Layers (performance of baseline model is also added for comparison)

#### – Layer Weight Regularizers

Experiments with layer weight regularizers based on baseline model are performed. Results are given in Table 4.13.

With L2 regularization with regularization parameter set to 0.01 added on the first Convolutional Layer, the performance is improved.

N o.	Number of Kernel Constraint Added	Added Position	Training Accuracy (%)	Validation Accuracy(%)
12	Dropout(0.4) Dropout(0.4) Dropout(0.4)	2nd Conv Maxpooling Dense(1000)	80.62	65.20
15	$3 \times$ maxnorm(3)	on each Conv & Maxpooling	65.80	50
16	$2 \times$ maxnorm(3)	on each Conv	50.86	50.98
17	$1 \times$ maxnorm(3)	on 2nd Conv	85.68	62.75

Table 4.12: Model training performance with kernel\_constraint (performance of experiment No.12 is also added for comparison)

N o.	Layer Added	Added Position on	Training Accuracy (%)	Validation Accuracy(%)
	Baseline		99.38	68.63
18	l2(0.01)	1st Conv	100	75.98
19	l2(0.01)	2nd Conv	92.84	65.20
20	l2(0.04)	2nd Conv	50	50

Table 4.13: Model training performance with L2 regularization (performance of baseline model is also added in for comparison)

In further experiments, L2 regularization is added on the second best performance architecture with dropout layers, experiment No.9. The results given in Table 4.14 show decreased overall performance, indicating that data are over constrained, and there is not enough information left for classification.

Other supplemental combinations using weight regularizers and Dropout Layers are as well tested. The number of Dropout layers are reduced to lower the constraints in comparison to the last group of experiments. The results are given in Table 4.15 show that the model is still over constrained.

#### – Batch Normalization

Batch Normalization layer is added to the baseline model. The results are given in Table 4.16. Though with batch normalization layer the accuracy is

N o.	Layer Added	Added Position on	Training Accuracy (%)	Validation Accuracy(%)
9	Dropout(0.2) Dropout(0.4) Dropout(0.2)	2nd Conv Maxpooling Dense(1000)	98.40	83.33
21	$2 \times l2(0.01)$	each Conv	51.48	50
22	$2 \times l2(0.001)$	each Conv	67.41	65.20
23	$1 \times l2(0.02)$	1st Conv	62.35	61.76
24	$1 \times l2(0.001)$	2nd Conv	49.88	50

Table 4.14: Model training performance with L2 regularization (performance of baseline model is also added for comparison)

N o.	Layer Added	Added Position	Training Accuracy (%)	Validation Accuracy(%)
	<b>Baseline</b>		<b>99.38</b>	<b>68.63</b>
25	$l2(0.01)$ Dropout(0.4) Dropout(0.2)	2nd Conv Maxpooling Dense(1000)	94.94	63.73
26	$l2(0.04)$ Dropout(0.4) Dropout(0.2)	2nd Conv Maxpooling Dense(1000)	77.65	64.22
27	$2 \times l2(0.02)$ Dropout(0.4) Dropout(0.2)	each Conv Maxpooling Dense(1000)	91.48	64.71

Table 4.15: Model training performance with L2 regularization and Dropout (performance of baseline model is also added for comparison)

not increased, considering the advantages mentioned in the previous chapter, batch normalization layer is adopted in the applied model.

From the results of the experiments, the best performance is achieved with model from experiment No.14. The architecture of experiment No.14 with BatchNormalization() is adopted as the CNNs model in this study.

<b>N o.</b>	<b>Layer Added</b>	<b>Added Position Behind</b>	<b>Training Accuracy (%)</b>	<b>Validation Accuracy(%)</b>
	<b>Baseline</b>		<b>99.38</b>	<b>68.63</b>
28	BatchNormalization()	1st Conv	70.62	67.16

Table 4.16: Model training performance with batch normalization (performance of baseline model is also added for comparison)

According to the experimental results, the final CNN architecture of the first framework contains one input layer, two convolutional layers, one batch normalization layer, one pooling layer, four dropout layers, one flatten layer, one fully-connected dense layers and one output layer. The architecture in detail with the hyperparameters is demonstrated in table 4.17.

Input: input_shape (144×144×3)
Conv2D(64, kernel_size=(10,10),strides=(2,2),activation='relu')
BatchNormalization()
Conv2D(64, kernel_size=(10,10),strides=(2,2),activation='relu')
Dropout(0.2)
MaxPooling2D(pool_size=(2,2),strides=(2,2))
Dropout(0.2)
Dropout(0.4)
Flatten()
Dense(1000, activation='relu')
Dropout(0.2)
Output layer: Dense(2, activation='softmax')

Table 4.17: Architecture of the proposed CNN model

#### 4.4.2 CNN feature extraction + SVM

The second proposed method is composed of CNN used feature extraction combined with Support Vector Machines (SVM) classification. Scikit-learn [52] is used for implementation of SVM. The purpose of this model is:

- to compare the classification ability of CNNs classification part and SVM.
- to check if the features extracted by CNNs can be used also by other classifier, and how it can be used.

To build this architecture, the following steps are applied:

- The best performing CNN model, trained with All500 dataset from the first method, is applied for feature extraction.
- For the purpose of comparison, the same datasets for training and validation are used in this framework. Each data sample is input into the well trained CNNs model for prediction. The outputs of the flatten layer of all the above sample predictions, which represent the extracted features, are fed into SVM model for model training and validation.
- The RBF kernel function is applied for SVM.
- Hyperparameters of the SVM: the soft margin constant  $C$  and the inverse of the standard deviation of the RBF kernel  $\gamma$ , are optimally determined using the grid search in the range  $2^i$  ( $i \in [-5, 5]$ ,  $i$  is integer). For each pair of hyperparameters, SVM model training is performed twice with 2 of 5-fold cross validation datasets. The average is calculated as the performance of this pair of hyperparameters.
- Feature selection is added before SVM model training to reduce the size of data fed to SVM model, since the initial experiments without feature selection did not provide satisfying results. Only 10% of the most informative features are kept.

Achieved training data accuracy (with blue line) and validation data accuracy (with orange line) are shown in Figure 4.2. As well, Figure 4.2 shows that when  $\gamma$  is equal to  $2^{-5}$  and  $C$  is between  $2^0$  to  $2^5$ , SVM model has the best performance. The smallest value we test for  $\gamma$  is  $2^{-5}$ , therefore further tests are taken with  $\gamma$  equal to  $2^i$  ( $i \in [-7, -5]$ ) and  $C$  equal to  $2^j$  ( $j \in [-1, 10]$ ). The results given in Figure 4.3 shows that decreasing the value of  $\gamma$ , and at same time increasing the value of  $C$  does not make too much difference. According to the value of the validation accuracy achieved,  $\gamma$  is set to be  $2^{-5}$  and  $C$  is set to be  $2^2$  for further processing.



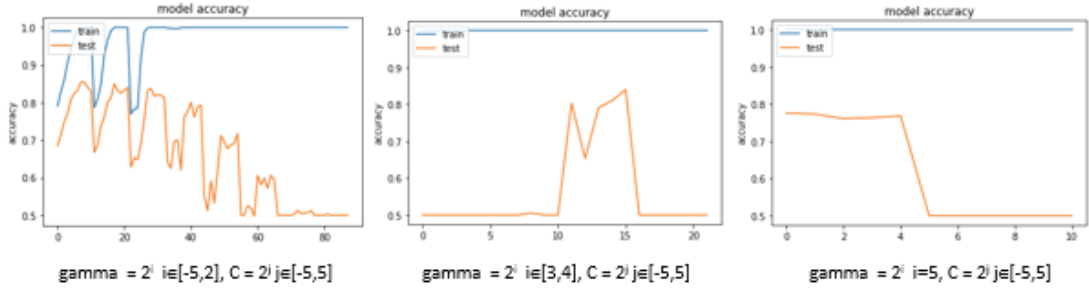


Figure 4.2: SVM model accuracy with feature selection.

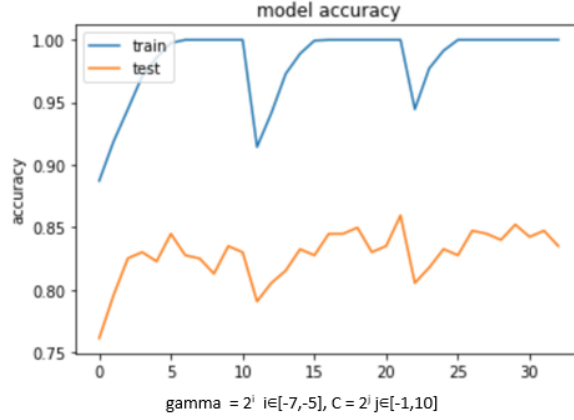


Figure 4.3: SVM model accuracy with feature selection.

#### 4.4.3 Dense only network

The third proposed method assumes using a fully connected(dense) only network for classification, omitting the feature extraction part with CNN. The purpose of this method is to demonstrate how crucial the feature extraction part is in the whole CNN architecture. To compare the results with the first model:

- The same datasets are adopted, which are used for training and validation in the first two frameworks.
- The same architecture for classification as in the first model is adopted. The architecture of this model is given in Table 4.18:

Input: input_shape (144×144×3)
Flatten()
Dense(1000, activation='relu')
Dropout(0.2)
Output layer: Dense(2, activation='softmax')

Table 4.18: Architecture of Dense Only model

## 4.5 Results

### 4.5.1 Three Frameworks

After the optimal architecture of CNN model is selected according to the experimental results above, the experiments for the proposed frameworks are performed. The results are given in Table 4.19. The following conclusions can be drawn:

		All500	ProgressedCase	EarlyCase
<b>CNN</b>	Accuracy (%)	86.29	94.07	90.19
	Sensitivity	0.8560	0.7514	0.8590
	Specificity	0.8698	0.9843	0.9349
<b>CNN+SVM</b>	Accuracy (%)	85.51	91.69	88.07
	Sensitivity	0.8560	0.6927	0.8512
	Specificity	0.8541	0.9705	0.9033
<b>Dense CNN</b>	Accuracy (%)	54.45	84.61	62.61
	Sensitivity	0.4256	0.2819	0.3972
	Specificity	0.6632	0.9763	0.8028

Table 4.19: Overall performance using accuracy, sensitivity and specificity as performance measures

- For the first framework with the CNN model, the performance changes with different datasets. With All500 dataset that contains clean audio files of both early and progressed PD patients, the experiment gives reasonable results. With ProgressedCase dataset that contains clean audio files of only progressed PD patients, the accuracy and specificity increase, while the sensitivity drops. The cases in ProgressedCase dataset are supposed to show more clear symptoms of voice disorder

than the cases in EarlyCase dataset (that contains clean audio files only early-stage PD patients), as well as cases in All500 dataset, which makes more sense if the accuracy and the sensitivity increase at same time. The reason for the sensitivity decrease and specificity increase might be the unbalanced training dataset demonstrated in Table: 4.6. There are 117 progressed PD patients combined with 507 Non-PD participants in this dataset, which affects the model to be biased towards Non-PD participant. This as well explains the increased specificity of EarlyCase, which is between the specificity of All500 and ProgressedCase.

- Comparison of the results of CNN model and CNN+SVM model.
  - The performance of the second framework that uses the combination of CNN and SVM (CNN+SVM) follows the same pattern as in case of the first framework (CNN); therefore, the same conclusions can be drawn.
  - In general, the performance of CNN+SVM drops in comparison to CNN, but not substantially. The feature extraction performed by CNN model can also be used as an input to SVM classifier, though CNN original classification part performs better.
- Comparison of the results of CNN model and Dense only model.
  - The performance of the third framework (Dense only model) drops dramatically in comparison with the first framework (CNN), suggesting the key role of feature extraction part of CNNs.
  - With ProgressedCase dataset, the validation accuracy drops from 94.07% to 84.6%, which is still acceptable, while it is almost not possible for the model to identify an individual with the disease, according to the low sensitivity of 0.2819. At the same time, with very high specificity, the model is most likely to identify each individual without the disease. This can be also easily demonstrated by analyzing confusion matrix in Figure 4.4. The validation dataset contains 102 Non-PD participants who are correctly identified as healthy subjects, and 24 PD patients who are incorrectly identified as healthy subjects (false positives). No voice recording is classified into PD patient class, showing the inability of the model to learn correctly. The relatively high accuracy misleads and it is the impact of unbalanced training dataset; therefore, the model is highly biased towards healthy subjects.

		Confusion matrix	
True label	Non PD	102	0
	PD	24	0
		Non PD	PD
		Predicted label	

Figure 4.4: Confusion matrix for Dense only CNN model with ProgressedCase dataset

#### 4.5.2 Extended Experiment

##### CNN+SVM without Feature Selection

An additional experiment to assess the influence of feature selection in SVM classifier is done with the 5-fold cross validation. The results are given in Table 4.20. Comparing with the results of CNN+SVM with feature extraction in Table 4.19, the performance of all the three measure obviously decreases, indicating the necessity of using feature selection with SVM.

		All500
<b>CNN+SVM</b>	Accuracy(%)	74.86
<b>Without</b>	Sensitivity	0.6724
<b>Feature Selection</b>	Specificity	0.8245

Table 4.20: CNN+SVM model without feature selection

##### Donors-based data splitting

Previous experiments, as well as most of the existing research on identification of Parkinson’s disease from voice signal, do not use leave-one-patient-out method for training dataset and validation dataset splitting. The main reason is the limited number of available data recordings produced by different participants in the study. However, in order to keep the full disjointedness of training dataset and validation dataset with respect to

both the audio samples and the voice donors, we further apply the leave-one-patient-out splitting to dataset to exclude the potential bias towards particular voice donor that may exist when different recordings of the same donor are present in both the training and validation dataset.

To compare the results, datasets, All2000 and ReAll2000, with the same samples, are prepared. Results in Table 4.21 come from All2000 dataset using normal data splitting which we used for all the previous experiments. Results in Table 4.22 come from the ReAll2000 dataset which is split with donor-based splitting.

		<b>All2000</b>
<b>CNN</b>	Validation Accuracy(%)	91.42
<b>Without</b>	Sensitivity	0.8867
<b>Feature Selection</b>	Specificity	0.9416

Table 4.21: Performance using sample-based data splitting without denoising preselection (All2000 dataset)

		<b>ReAll2000</b>
<b>CNN</b>	Training Accuracy(%)	96.23
<b>CNN</b>	Validation Accuracy(%)	67.99
<b>Without</b>	Sensitivity	0.3358
<b>Feature Selection</b>	Specificity	0.6685

Table 4.22: Performance using donor-based data splitting without denoising preselection (ReAll2000 dataset)

We can see that the performance dropped dramatically with donor-based data splitting. The reason might be the fact that with the sample-based data splitting method, there is high possibility that the audio files from one donor are distributed into both training and validation dataset, making a model biased towards this donor. It means the model is trained to recognize the donor related features, instead of disease related features.

This can further explain why in Table 4.19 the performance of model trained with EarlyCase dataset is better than the model trained with All500 dataset. If the PD related features are used for classification, the performance of model trained with All500 dataset

should be better than the model trained with EarlyCase dataset, due to the fact that All500 dataset contains more samples with more severe PD symptoms. In sample-based data splitting, there is higher possibility to split the sample from one donor into both training and validation dataset, which results in higher performance as the model is biased toward a particular donor. The average number of samples per donor in EarlyCase dataset is 9.51 (390/41), whereas in All500 dataset it is 7.68 (507/66).

In donor-based data splitting, the samples from one donor can not exist in both training and validation dataset, removing the bias towards a particular donor, which results in lower performance.

While the training accuracy is high and amounts 96.23%, the validation accuracy is substantially lower in Table 4.22 and training curve in Figure 4.5, suggesting overfitting. Since adding standard regularization techniques to deal with overfitting did not improve the results, additional measures, such as data augmentation and expanding the dataset with "Immediately before taking their medication" samples should be undertaken.

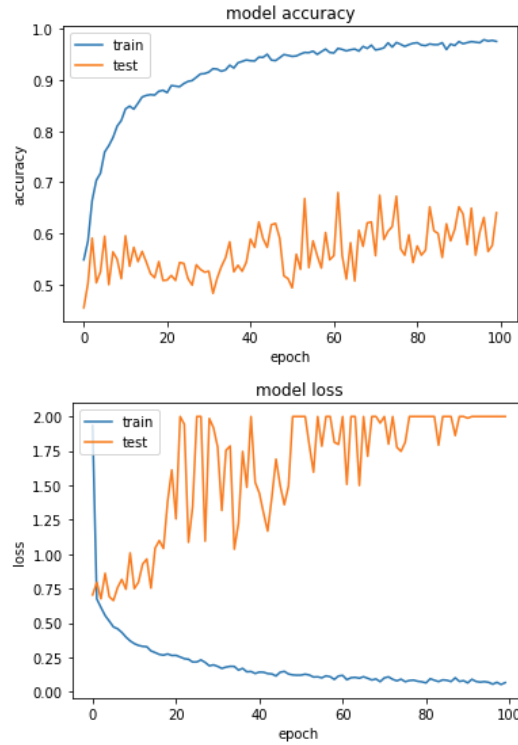


Figure 4.5: Training Curve with ReAll2000 dataset

## Chapter 5

# Conclusions and future work

### 5.1 Conclusions

Speech and voice disorders are common in Parkinson's disease, and can be detected early in disease development. The main aim of this thesis is to develop a deep neural network model for identification of Parkinson's disease from sustained vowel phonation recordings. Three frameworks are proposed based on Convolutional Neural Networks and Support Vector Machines. In the first framework, mel-spectrograms of voice records are used as low-level voice representations and fed into multiple layer Convolutional Neural Network model for training, tuning and classification. In the second proposed framework, Convolutional Neural Network trained on mel-spectrograms was used only for feature extraction, in order to learn higher-level feature representations from speech, which were further fed into Support Vector Machines for classification. In the third framework, the mel-spectrograms are input directly into a fully connected network which is used as the final classification layer of Convolutional Neural Network from the first framework, i.e. without using convolutional layers for feature learning. The idea was not only to identify Parkinson's disease with Convolutional Neural Network, but also to assess what is the overall contribution of features learned by Convolutional Neural Network in identification of Parkinson's disease.

Five subsets taken from the mPower dataset are created for training and testing the models:

- All500 - dataset that contains 507 clean sustained vowel phonations taken from PD patients and 507 clean sustained vowel phonations taken from healthy control

subjects,

- EarlyCase - subset of All500 dataset with 390 samples from early stage PD patients and 507 samples from healthy control subjects,
- ProgressedCase - subset of All500 dataset with 117 samples from progressed PD patients and 507 samples from healthy control subjects,
- All2000 – 2824 noisy sustained vowel phonations taken from PD patients and 2824 noisy sustained vowel phonations taken from healthy control subjects, using the sample-based data splitting,
- ReAll2000 - 2824 noisy sustained vowel phonations taken from PD patients and 2824 noisy sustained vowel phonations taken from healthy control subjects, using the donor-based data splitting scheme.

The motivation was to evaluate the ability of the models to identify Parkinson’s disease under different conditions (early detection vs. detection of progressed cases, detection with imperfect recordings corrupted by background noise etc.).

The CNN architecture is self-customized. Different methods are applied in the experimental phases to improve generalization and reduce overfitting, including dropout, L2 regularization and batch normalization. Convolutional neural networks have proven to be a good choice for feature extraction and disease identification leading to over 90% of accuracy for detection of early cases of PD (patients diagnosed within 3 years from data collection) and over 94% of accuracy for progressed cases (patients diagnosed more than 3 years before data collection). Moreover, the high level feature representations have shown to be very useful with standard machine learning classifiers, such as Support Vector Machines, leading to only a minor drop of performance. Finally, the feature extraction using a series of convolutional and pooling layers has proven to have a key role, leading to dramatic drop in performance when mel-spectrograms are directly fed into the fully connected neural network used for classification, i.e. without using the high level feature representations learned by CNN. Accuracy drops to 84% for progressed PD cases and only 62% for early cases.

Experiments with voice samples that contain background noise led to satisfying results of over 91% of accuracy, 88% of sensitivity and 94% of specificity, suggesting that using much larger dataset for model training compensates for the imperfections of data corrupted by noise.



Finally, applying donor-based data splitting (leave-one-patient-out) to exclude the potential bias towards particular voice donor that may exist when different recordings of the same donor are present in both the training and validation dataset, shows that the proposed Convolutional Neural Network heavily overfits dropping the accuracy for the validation dataset to only 68%.

## 5.2 Future work

Further investigation is necessary to reduce the overfitting in case of donor-based data splitting and improve the performance. The following steps can be undertaken:

- Data augmentation can be adopted to artificially increase the size of dataset by including slightly modified copies of already existing data. It has a regularization effect for training a machine learning model.
- Only the voice samples of PD patients that don't take Parkinson medications were used in model training, since medications can reduce the voice disorder symptoms to some extent. Expanding the dataset with voice samples recorded immediately before taking the medication might be beneficial, since the symptoms in these case should be affected the least.
- Using modalities other than speech, such as handwriting or gait, that are also affected by Parkinson's disease can further help in improving the model performance.

# References

- [1] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. volume 37 of *Proceedings of Machine Learning Research*, pages 448–456. PMLR, 07–09 Jul 2015. doi: [10.1007/978-3-642-15825-4\\_10](https://doi.org/10.1007/978-3-642-15825-4_10).
- [2] Vojislav Kecman. Support vector machines – an introduction. *Studies in Fuzziness and Soft Computing*, May 2005. doi: [10.1007/10984697\\_1](https://doi.org/10.1007/10984697_1).
- [3] Bot BM, Suver C, and Neto EC et al. *The mPower study, Parkinson Disease Mobile Data Collected Using ResearchKit*. Sci Data 3:160011, 2016 Mar 3. doi: [10.1038/sdata.2016.11](https://doi.org/10.1038/sdata.2016.11).
- [4] D. Ozog. Signal analysis. Whitman College, 2007.
- [5] Elbaz A, Carcaillon L, Kab S, and Moisan F. *Epidemiology of Parkinson’s disease*, volume 172(1), pages 14–26. Rev Neurol (Paris), 2016. doi: [10.1016/j.neurol.2015.09.012](https://doi.org/10.1016/j.neurol.2015.09.012).
- [6] Jefferson Almeida, Pedro Pedrosa Rebouças Filho, Tiago Carneiro, Wei Wei, and Robertas Damaševičius et al. Detecting parkinson’s disease with sustained phonation and speech signals using machine learning techniques. In *Pattern Recognition Letters, Elsevier*, volume 125, pages 55–62. hal-02380596, 2019. doi: [10.1016/j.patrec.2019.04.005](https://doi.org/10.1016/j.patrec.2019.04.005).
- [7] Fabrizio Stocchi, Pablo Martínez-Martin, and Heinz Reichmann. Quality of life in parkinson’s disease patient, clinical and research perspectives. *European Neurological Review*, 9(1):12–8, 2014. doi: [10.17925/ENR.2014.09.01.12](https://doi.org/10.17925/ENR.2014.09.01.12).

- [8] Julian M. Fearnley and Andrew J. Lees. Ageing and parkinson’s disease: substantia nigra regional selectivity. *Brain*, 114(Pt 5):2283–2301, 1991. doi: [10.1093/brain/114.5.2283](https://doi.org/10.1093/brain/114.5.2283).
- [9] A. K. Ho, R. Iansek, C. Marigliani, J. L. Bradshaw, and S. Gates. Speech impairment in a large sample of patients with parkinson’s disease. *Behavioural Neurology*, 11:131–137, 1998.
- [10] J. A. Logemann, H. B. Fisher, B. Boshes, and E. R. Blonsky. Frequency and co-occurrence of vocal-tract dysfunctions in speech of a large sample of parkinson patients. *Journal of Speech and Hearing Disorders*, 43:47–57, 1978.
- [11] Evaldas Vaiciukynas, Antanas Verikas, Adas Gelzinis, and Marija Bacauskiene. Detecting parkinson’s disease from sustained phonation and speech signals. *PLOS ONE*, 12(10):1–16, 10 2017. doi: [10.1371/journal.pone.0185613](https://doi.org/10.1371/journal.pone.0185613).
- [12] Jefferson Almeida, Pedro Pedrosa Rebouças Filho, Tiago Carneiro, Wei Wei, and et al Robertas Damaševičius. Detecting parkinson’s disease with sustained phonation and speech signals using machine learning techniques. *Pattern Recognition Letters*, 125:55–62, 2019. doi: [10.1016/j.patrec.2019.04.005](https://doi.org/10.1016/j.patrec.2019.04.005).hal-02380596f.
- [13] V. Despotovic, T. Skovranek, and C. Schommera. Speech based estimation of parkinson’s disease using gaussian processes and automatic relevance determination. *Neurocomputing*, 401:173–181, 8 2020. doi: [10.1016/j.neucom.2020.03.058](https://doi.org/10.1016/j.neucom.2020.03.058).
- [14] H. Gunduz. Deep learning-based parkinson’s disease classification using vocal feature sets. *IEEE Access*, 7:115540–115551, 2019.
- [15] M. A. Little, P. E. McSharry, E. J. Hunter, J. Spielman, and L. O. Ramig. Suitability of dysphonia measurements for telemonitoring of parkinson’s disease. *IEEE transactions on bio-medical engineering*, 56(4):1015, 2009. doi: [10.1109/TBME.2008.2005954](https://doi.org/10.1109/TBME.2008.2005954).
- [16] Betul Erdogdu Sakar, Gorkem Serbes, and C. Okan Sakar. Analyzing the effectiveness of vocal features in early telediagnosis of parkinson’s disease. 9 2017. doi: [10.1371/journal.pone.0182428](https://doi.org/10.1371/journal.pone.0182428).

- [17] Alireza Bayestehtashk, Meysam Asgari, Izhak Shafran, and James McNames. Fully automated assessment of the severity of parkinson’s disease from speech. *Computer Speech Language*, 29(1), 1 2015.
- [18] A. Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *CACM*, 2017.
- [19] Masakiyo Fujimoto, Keisuke Kinoshita, and Tomohiro Nakatani. Exploiting spectro-temporal locality in deep learning based acoustic event detection. *EURASIP Journal on Audio, Speech, and Music Processing*, 26:1687–4722, 2015. doi: [10.1186/s13636-015-0069-2](https://doi.org/10.1186/s13636-015-0069-2).
- [20] X Lu, Y Tsao, and S Matsuda. Sparse representation based on a bag of spectral exemplars for acoustic event detection. *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 6255–6259, 2014. doi: [10.1109/ICASSP.2014.6854807](https://doi.org/10.1109/ICASSP.2014.6854807).
- [21] H. Zhang, I. McLoughlin, and Y. Song. Robust sound event recognition using convolutional neural networks. pages 559–563, 2015. doi: [10.1109/ICASSP.2015.7178031](https://doi.org/10.1109/ICASSP.2015.7178031).
- [22] S-Y Chang and N Morgan. Robust cnn-based speech recognition with gabor filter kernels. pages 905–909, 2014.
- [23] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau. Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions. pages 2519–2523, May 2014. doi: [10.1109/ICASSP.2014.6854054](https://doi.org/10.1109/ICASSP.2014.6854054).
- [24] M. Stolar, M. Lech, R. S. Bolia, and M. Skinner. Acoustic characteristics of emotional speech using spectrogram image classification. In *2018 12th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–5, 2018. doi: [10.1109/ICSPCS.2018.8631752](https://doi.org/10.1109/ICSPCS.2018.8631752).
- [25] L. Zahid, M. Maqsood, M. Y. Durrani, M. Bakhtyar, J. Baber, H. Jamal, I. Mehmood, and O. Song. A spectrogram-based deep feature assisted computer-aided diagnostic system for parkinson’s disease. *IEEE Access*, 8:35482–35495, 2020. doi: [10.1109/ACCESS.2020.2974008](https://doi.org/10.1109/ACCESS.2020.2974008).

- [26] Juan Rafael Orozco-Arroyave, Julián David Arias-Londoño, Jesús Francisco Vargas-Bonilla, María Claudia González-Rátiva, and Elmar Nöth. New Spanish speech corpus database for the analysis of people suffering from Parkinson’s disease. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC’14)*, pages 342–347, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).
- [27] H. Zhang, A. Wang, D. Li, and W. Xu. Deepvoice: A voiceprint-based mobile health framework for parkinson’s disease identification. In *2018 IEEE EMBS International Conference on Biomedical Health Informatics (BHI)*, pages 214–217, 2018. doi: [10.1109/BHI.2018.8333407](https://doi.org/10.1109/BHI.2018.8333407).
- [28] J. B. Allen. Applications of the short time fourier transform to speech processing and spectral analysis. *Proceedings - ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*, 1982-May:1012–1015, January 1982. 1982 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1982 ; Conference date: 03-05-1982 Through 05-05-1982.
- [29] Nisar S, Khan OU, and Tariq M. An efficient adaptive window size selection method for improving spectrogram visualization. *Computational intelligence and neuroscience*, 2016. doi: [10.1155/2016/6172453](https://doi.org/10.1155/2016/6172453) Epub 2016 Aug 24. PMID: 27642291; PMCID: PMC5013242.
- [30] Lincoln Gray. Properties of sound. In *Journal of perinatology*, volume 20, pages 5–10, 2000.
- [31] Stevens Stanley Smith, Volkmann John, and Edwin B. Newman. A scale for the measurement of the psychological magnitude pitch. In *Journal of the Acoustical Society of America*, volume 8, pages 185–190, 1937. doi: [10.1121/1.1915893](https://doi.org/10.1121/1.1915893).
- [32] Choi K, Fazekas G, Sandler MB, and Cho K. A comparison of audio signal preprocessing methods for deep neural networks on music tagging. In *26th European signal processing conference*, page 1870–1874. EUSIPCO IEEE Roma Italy, 2018.
- [33] Shibli Nisar, Omar Usman Khan, and Muhammad Tariq. An efficient adaptive window size selection method for improving spectrogram visualization. In *Computational Intelligence and Neuroscience*, volume 2016, pages 1–13. Hindawi Publishing Corporation, 2016. doi: [10.1155/2016/6172453](https://doi.org/10.1155/2016/6172453).

- [34] Ivars Namatevs. Deep convolutional neural networks: Structure, feature extraction and training. In *Information Technology and Management Science*, volume 20, December 2017. doi: [10.1515/itms-2017-0007](https://doi.org/10.1515/itms-2017-0007).
- [35] Y. Bengio. Learning deep architectures for ai. In *Foundations and Trends® in Machine Learning*, volume 2, pages 1–127, 2009. doi: [10.1561/22000000006](https://doi.org/10.1561/22000000006).
- [36] Putra Wanda and Huang Jin Jie. Runpool: A dynamic pooling layer for convolution neural network. In *International Journal of Computational Intelligence Systems*, volume 13, January 2020. doi: [10.2991/ijcis.d.200120.002](https://doi.org/10.2991/ijcis.d.200120.002).
- [37] Dingjun Yu, Hanli Wang, Peiqiu Chen, and Zhihua Wei. Mixed pooling for convolutional neural networks. In *The 9th International Conference on Rough Sets and Knowledge Technology*, October 2014. doi: [10.1007/978-3-319-11740-9\\_34](https://doi.org/10.1007/978-3-319-11740-9_34).
- [38] Dominik Scherer, Andreas C Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *Artificial Neural Networks - ICANN 2010 - 20th International Conference*, volume Proceedings, Part III. Thessaloniki, Greece, January 2010. doi: [10.1007/978-3-642-15825-4\\_10](https://doi.org/10.1007/978-3-642-15825-4_10).
- [39] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012.
- [40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. In *J. Mach. Learn. Res.*, volume 15, page 1929–1958. JMLR.org, 2014. doi: [10.5555/2627435.2670313](https://doi.org/10.5555/2627435.2670313).
- [41] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization?, 2019.
- [42] Sh Shabbeer Basha, Shiv Ram Dubey, Viswanath Pulabaigari, and Snehasis Mukherjee. Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing* 378, October 2019. doi: [10.1016/j.neucom.2019.10.008](https://doi.org/10.1016/j.neucom.2019.10.008).
- [43] Douglas M. Hawkins. The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, 44(1):1–12, 2004. doi: [10.1021/ci0342472](https://doi.org/10.1021/ci0342472).

- [44] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, November 1995. doi: [10.1007/BF00994018](https://doi.org/10.1007/BF00994018).
- [45] Mariette Awad and Rahul Khanna. Support vector machines for classification. *Efficient Learning Machines*, 3:39–66, January 2015. doi: [10.1007/978-1-4302-5990-9\\_3](https://doi.org/10.1007/978-1-4302-5990-9_3).
- [46] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.
- [47] Z. Reitermanová. Data splitting. 2010.
- [48] S.L. Lohr. *Sampling: Design and Analysis*. Advanced (Cengage Learning). Cengage Learning, 2009.
- [49] J. R. Orozco-Arroyave, F. Hönig, J. D. Arias-Londoño, J. F. Vargas-Bonilla, K. Daqrouq, S. Skodda, J. Ruzs, and E. Nöth. Automatic detection of Parkinson’s disease in running speech spoken in three different languages. *Acoustical Society of America Journal*, 139(1):481–500, January 2016. doi: [10.1121/1.4939739](https://doi.org/10.1121/1.4939739).
- [50] Sakar CO and Kursun O. Telediagnosis of parkinson’s disease using measurements of dysphonia. *J Med Syst*, 34(4):591–599, Aug 2010. doi: [10.1007/s10916-009-9272-y](https://doi.org/10.1007/s10916-009-9272-y).
- [51] Rajul Parikh, Annie Mathai, Shefali Parikh, G Chandra Sekhar, and Ravi Thomas. Understanding and using sensitivity, specificity and predictive values. *Indian Journal of Ophthalmology*, 56(1):45–50, Jan-Feb 2008. doi: [10.4103/0301-4738.37595](https://doi.org/10.4103/0301-4738.37595).
- [52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [53] Brian McFee; Vincent Lostanlen; Alexandros Metsai; Matt McVicar; Stefan Balke; Carl Thomé; Colin Raffel; Frank Zalkow; Ayoub Malek; Dana; Kyungyun Lee; Oriol Nieto; Jack Mason; Dan Ellis; Eric Battenberg; Scott Seyfarth; Ryuichi Yamamoto; Keunwoo Choi; viktorandreevichmorozov; Josh Moore; Rachel Bittner; Shunsuke Hidaka; Ziyao Wei; nullmightybofo; Darío Hereñú; Fabian-Robert Stöter; Pius Friesch; Adam Weiss; Matt Vollrath; Taewoon Kim. Librosa version 0.7.2. Jan 2020. doi: [10.5281/zenodo.3606573](https://doi.org/10.5281/zenodo.3606573).

- [54] Francois Chollet et al. Keras, 2015.
- [55] Bisong E. *Google Colaboratory*. In: *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Apress, Berkeley, CA, 28 September 2019. doi: [10.1007/978-1-4842-4470-8\\_7](https://doi.org/10.1007/978-1-4842-4470-8_7).
- [56] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [57] Clayton R. Pereira, Danilo R. Pereira, Silke A.T. Weber, Christian Hook, Victor Hugo C. de Albuquerque, and João P. Papa. A survey on computer-assisted parkinson’s disease diagnosis. *Artificial Intelligence in Medicine*, 95:48–63, 4 2019. doi: [10.1016/j.artmed.2018.08.007](https://doi.org/10.1016/j.artmed.2018.08.007).
- [58] Laetitia Jeancolas, Mangone Graziella, Jean-Christophe Corvol, Marie Vidailhet, Stéphane Lehericy, Badr-Eddine Benkelfat, Habib Benali, and Dijana Petrovska-Delacrétaz. Comparison of telephone recordings and professional microphone recordings for early detection of parkinson’s disease, using mel-frequency cepstral coefficients with gaussian mixture models. pages 3033–3037, 09 2019. doi: [10.21437/Interspeech.2019-2825](https://doi.org/10.21437/Interspeech.2019-2825).
- [59] Sungheon Park and Nojun Kwak. Analysis on the dropout effect in convolutional neural networks. In Shang-Hong Lai, Vincent Lepetit, Ko Nishino, and Yoichi Sato, editors, *Computer Vision – ACCV 2016*, pages 189–204, Cham, 2017. Springer International Publishing.



# APPENDICES

## Appendix A

# Demographics Survey

Table A.1: Demographics Survey

Question	Variable name
How old are you?	age
What is your sex?	gender
Which race do you identify with?	race
What is the highest level of education that you have completed?	education
What is your current employment status?	employment
What is your current marital status?	maritalStatus
Are you a spouse, partner or care-partner of someone who has Parkinson disease?	are-caretaker
Have you ever participated in a research study or clinical trial on Parkinson disease before?	past-participation
How easy is it for you to use your smartphone?	smartphone
Do you ever use your smartphone to look for health or medical information online?	phone-usage
Do you use the Internet or email at home?	home-usage
Do you ever use the Internet to look for health or medical information online?	medical-usage
Did you happen to do this yesterday, or not?	medical-usage-yesterday
Do you ever use your smartphone to participate in a video call or video chat?	video-usage
Have you been diagnosed by a medical professional with Parkinson disease?	professional-diagnosis
In what year did your movement symptoms begin?	onset-year
In what year were you diagnosed with Parkinson disease?	diagnosis-year
In what year did you begin taking Parkinson disease medication? Type in 0 if you have not started to take Parkinson medication.	medication-start-year
What kind of health care provider currently cares for your Parkinson disease?	healthcare-provider
Have you ever had Deep Brain Stimulation?	deep-brain-stimulation
Have any surgery for Parkinson disease, other than DBS?	surgery
Have you ever smoked?	smoked
How many years have you smoked?	years-smoking
On average, how many packs did you smoke each day?	packs-per-day
When is the last time you smoked (put today's date if you are still smoking)?	last-smoked
Has a doctor ever told you that you have any of the following conditions? Please check all that apply.	health-history

## Appendix B

Data distribution of all the available  
data for this study in mPower  
dataset

Professional Diagnosis	Diagnosis Year	No. of Participants	No. of Audio Records
FALSE		3844	22810
TRUE	1976	1	21
TRUE	1982	1	3
TRUE	1995	1	4
TRUE	1997	1	2
TRUE	1998	1	9
TRUE	1999	1	4
TRUE	2000	2	108
TRUE	2002	1	1
TRUE	2003	1	1
TRUE	2004	3	7
TRUE	2005	2	15
TRUE	2006	2	6
TRUE	2007	2	6
TRUE	2008	3	8
TRUE	2009	8	15
TRUE	2010	8	29
TRUE	2011	11	105
TRUE	2012	12	724
TRUE	2013	27	705
TRUE	2014	32	787
TRUE	2015	13	289
TRUE	In total	133	2849

Table B.1: Data distribution of participants and audio records regarding the Diagnosed Year

## Appendix C

### Data Citations

- Demographics Survey:

Bot, B. M., et al. Synapse <http://dx.doi.org/10.7303/syn5511429.1>

---

- Voice Task:

Bot, B. M., et al. Synapse <http://dx.doi.org/10.7303/syn5511444.1>

---

## Appendix D

# Dataset Download

### D.1 Steps for Access

To be qualified for access to mPower data, the instructions below need to be followed.

1. Register for a Synapse account (if you do not have one already). During registering, The Synapse governance policies and procedures must be reviewed.
2. Become a Certified Synapse user(if you are not already). A short quiz of 15 questions, regarding the Synapse governance policies and procedures, need to be passed.
3. Have your user profile validated. In order to valid your profile,
  - you must confirm the completion of your user profile.
  - Link to your ORCID profile and make sure it is set to public.
  - Sign a Oath.
  - Submit supplementary validation documents.
4. Request Access to mPower Data and Submit your Intended Data Use statement.
5. Agree to the data-specific Conditions for Use.
6. Download the data. Data stored in Synapse data repository can be accessed either with Synapse website based download or programmatically with Synapse R or Python clients. It depending on what and how much data is required.

## D.2 Install synapseclient for Python User

The synapseclient package can be installed or update with pip:

```
(sudo) pip3 install (--upgrade) synapseclient[pandas, pysftp]
```

## D.3 Python Code for Audio Files Download

```
import synapseclient
syn = synapseclient.Synapse()
syn.login(Username, Password)

for offset in range(0,65000,100):
    results = syn.tableQuery(
        'SELECT_*_FROM_syn5511444_LIMIT_100_OFFSET_' + str(offset))
    file_map = syn.downloadTableColumns(results,['audio_audio.m4a'])
print('done\n')
```

## D.4 Python Code for Audio Files Reorganisation

Change names of downloaded audio files from .tmp to .m4a for data preprocessing.

```
import os, shutil, csv
names = []
#Destination folder for reorgnised audio files
dest = 'Destination_folder'
# Source folder of downloaded audio files
d = 'Source_folder'
A = [os.path.join(d, o) for o in os.listdir(d)
      if os.path.isdir(os.path.join(d,o))]
print('====',A)
for i in range(len(A)):
    A1 = [os.path.join(A[i], o) for o in os.listdir(A[i])
          if os.path.isdir(os.path.join(A[i],o))]
```



```

A1_names = [o for o in os.listdir(A[i])
             if os.path.isdir(os.path.join(A[i],o))]

for i1 in range(len(A1)):
    A2 = [os.path.join(A1[i1], f) for f in os.listdir(A1[i1])
          if f.endswith('.tmp')]

    if len(A2) != 0:
        shutil.copy(A2[0], dest+'/' + A1_names[i1] + '.m4a')
        names.append(A2)

# Code to save the folder structure for future reference
# An empty csv file 'tmp.csv' has to be created before running this code
csvfile = 'Self_defined_folder/tmp.csv'

#Assuming res is a flat list
with open(csvfile, "w") as output:
    writer = csv.writer(output, lineterminator='\n')
    for val in names:
        writer.writerow([val])

```

## Appendix E

# Activation Function





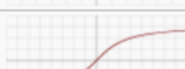

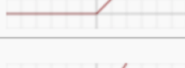


Name	Plot	Equation
Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) <sup>[2]</sup>		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) <sup>[3]</sup>		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$

Figure E.1: Activation Functions