CONSTRUCTOR UNIVERSITY

Clinically useful early detection of Parkinson's Disease using uncertainty-aware classification with CNNs

by

Anurag Sundar

Master Thesis in Data Engineering

Supervisor(s):

Prof Dr.-Ing Markus Wenzel, Prof. Dr. Stefan Kettemann

Submission: August 29, 2023

Statutory Declaration

Family Name, Given/First Name	Sundar, Anurag
Matriculation number	20332594
Kind of thesis submitted	Master Thesis

English: Declaration of Authorship

I hereby declare that the thesis submitted was created and written solely by myself without any external support. Any sources, direct or indirect, are marked as such. I am aware of the fact that the contents of the thesis in digital form may be revised with regard to usage of unauthorized aid as well as whether the whole or parts of it may be identified as plagiarism. I do agree my work to be entered into a database for it to be compared with existing sources, where it will remain in order to enable further comparisons with future theses. This does not grant any rights of reproduction and usage, however.

This document was neither presented to any other examination board nor has it been published.

German: Erklärung der Autorenschaft (Urheberschaft)

Ich erkläre hiermit, dass die vorliegende Arbeit ohne fremde Hilfe ausschließlich von mir erstellt und geschrieben worden ist. Jedwede verwendeten Quellen, direkter oder indirekter Art, sind als solche kenntlich gemacht worden. Mir ist die Tatsache bewusst, dass der Inhalt der Thesis in digitaler Form geprüft werden kann im Hinblick darauf, ob es sich ganz oder in Teilen um ein Plagiat handelt. Ich bin damit einverstanden, dass meine Arbeit in einer Datenbank eingegeben werden kann, um mit bereits bestehenden Quellen verglichen zu werden und dort auch verbleibt, um mit zukünftigen Arbeiten verglichen werden zu können. Dies berechtigt jedoch nicht zur Verwendung oder Vervielfältigung.

Diese Arbeit wurde noch keiner anderen Prüfungsbehörde vorgelegt noch wurde sie bisher veröffentlicht.

29/08/2023

. . . . . .

Date, Signature

Abstract

Parkinson's disease is a neurodegenerative disorder that affects a part of the brain called the Substantia Nigra. Parkinson's disease can be detected using a SPECT (single photon emission computerised tomography) Brain DaTscan, which is a type of nuclear imaging test that shows 2D and 3D images of the brain. This study aims to investigate the uncertainty in the classification of early Parkinson's disease (PD) and a Healthy Control (HC) using these brain images. The goal is to explore and compare various image processing and predictive deep learning methods that can help quantify uncertainty in each prediction. The presence of uncertainty metrics for each prediction helps in estimating/quantifying confidence in that prediction, thereby, helping the healthcare providers make more informed decisions about the appropriate course of action. This study aims to perpetuate and build on the existing studies on Parkinson's disease classification by adding an element of predictive uncertainty quantification.

Contents

List of Figures

List of Tables

List of Abbreviations

1. **FP-CIT:** N-ω-fluoropropyl-2β-carbomethoxy-3β-(4-iodophenyl) nortropane
2. **SPECT:** Single-photon emission computed tomography
3. **CNN:** Convolutional Neural Network
4. **HC:** Healthy Control
5. **PD:** Parkinson's Disease
6. **SNGP:** Spectral-normalised Neural Gaussian Process
7. **DE:** Deep Ensemble
8. **TS:** Temperature Scaling
9. **MC:** Monte Carlo
10. **TP:** True Positive
11. **FP:** False Positive
12. **TN:** True Negative
13. **FN:** False Negative
14. **NN:** Neural Network
15. **CE:** Cross Entropy
16. **BN:** Batch Normalisation
17. **UMAP:** Uniform Manifold Approximation and Projection

1 Introduction

Parkinson's disease is a neurodegenerative disorder that affects the brain. There are physiological and anatomical symptoms that can help detect Parkinson's disease. For this study, we only need to concern ourselves with the anatomical symptom detection. Using the FP-CIT-SPECT method, Parkinson's disease is detected by capturing and analysing an image of the brain post scanning. This scanned image of the brain highlights the areas of the brain that show dopamine activity. By analysing these features in the scan, medical professionals detect whether a patient has Parkinson's disease. Therefore, the brain scan images, at least in most cases, have discerning features that decide whether a patient has Parkinson's or not. Studies already exist where neural networks have been used to predict Parkinson's disease from the brain scan images. Specifically, Convolutional Neural Networks (CNNs) have been used to extract discernible features from the brain scans in order to train and predict between a Parkinson's patient and a Healthy Case. Studies such as using an ensemble of CNN models to predict Parkinson's disease already exist and have given high accuracy in predictions(almost 85 percent)[9]. However, none of these models have complete accuracy in their predictions. Since we are dealing with vital clinical issues, we do not want to misdiagnose even a single Parkinson's patient as a healthy one. This could possibly cause fatal damage to the patient and propel the worsening of his/her condition. Therefore, for the cases that have been misdiagnosed by neural network models, there needs to be some indication or measure to reassess such predictions. One such method is to show the degree of uncertainty in such predictions. For each prediction, uncertainty measures helps to understand the confidence behind each prediction. This can help medical professionals to look more carefully at certain cases with high uncertainty so that they can confirm the diagnosis. This method allows for semi-automation of Parkinson's disease prediction while still having provision for maintaining complete accuracy in classification.

2 Background

This study is concerned with quantifying uncertainty in the classification of brain scan image data between a HC and a PD case. In this study, we are also attempting to predict the SBR values, which may act as a supporting step for better classification. This research builds upon the existing PPMI study.[21].

2.1 What is Parkinson's Disease?

Parkinson's disease affects the central nervous system, causing degeneration of mental and motor functions. Parkinson's Disease is caused by the malfunction of an area of the brain called Substantia Nigra. Dopamine is a neuroendocrine transmitter produced in the Substantia Nigra that affects movement, motivation, and pleasure in the human brain. Malfunction of the Substantia Nigra causes loss of dopaminergic cells which leads to reduced dopamine production and activity, subsequently leading to Parkinson's disease.[20]

2.2 FP-CIT-SPECT

FP-CIT: N-ω-fluoropropyl-2β-carbomethoxy-3β-(4-iodophenyl) nortropane is a radiopharmaceutical that binds reversibly to striatal presynaptic dopamine transporters. It is used in nuclear medicine for the diagnosis of Parkinson's disease and the differential diagnosis of Parkinson's disease over other disorders presenting similar symptoms.[7] FP-CIT is injected into the blood, where it circulates around the body and makes its way into the brain. It attaches itself to the dopamine transporter, a molecule found on dopamine neurons. Several hours after the tracer has been injected, special imaging equipment scans the head to detect the presence of the dopamine transporter via SPECT.[2] SPECT (Single photon emission computed tomography) is a non-invasive nuclear medicine test that shows brain function. In the data for this study, SPECT scans were used to detect the presence and activity of the dopamine transporter in the Striatum. A SPECT scan specifically for detecting Parkinson's Syndrome is called a Dopamine Transporter scan (DaTscan).[12][23] SPECT is considered a semi-quantitative imaging modality because it allows for the counting of emitted photons, providing information on the actual amount of tracer present in the body. Through calibration, SPECT images can be standardized, ensuring comparability across different imaging sites and scanners. This semi-quantitative nature sets it apart from MRI, which does not directly measure tracer concentration, and makes it somewhat similar to CT, where Hounsfield units offer a quantitative measure of X-ray attenuation. By knowing the tracer's actual concentration, SPECT becomes a valuable tool for functional imaging in various medical applications.[1]

2.3 Parkinson's Disease detection

The substantia nigra sends dopaminergic projections to another part of the brain called the striatum, which is involved in controlling movement and coordination. Due to the malfunction of the substantia nigra in Parkinson's disease, there is considerable reduction

in the dopaminergic projections sent to the striatum. As a result, visually, the detection of Parkinson's disease is done in the striatum. The striatum is a bean-shaped part of the brain. More specifically, the dorsal striatum, which is the exact area where the dopaminergic response is checked, is made up of the putamen and the caudate nucleus. On performing FP-CIT SPECT, the striatum shows a highlighted area which indicates the activity of the dopamine transporter in the brain. The presence of considerable unilateral or bilateral reduction in dopaminergic binding regarding the two halves of the striatum helps medical professionals detect Parkinson's disease. Even for deep learning models, it is these distinct features of the brain scan images that is learnt by the model layers to make the best predictions.

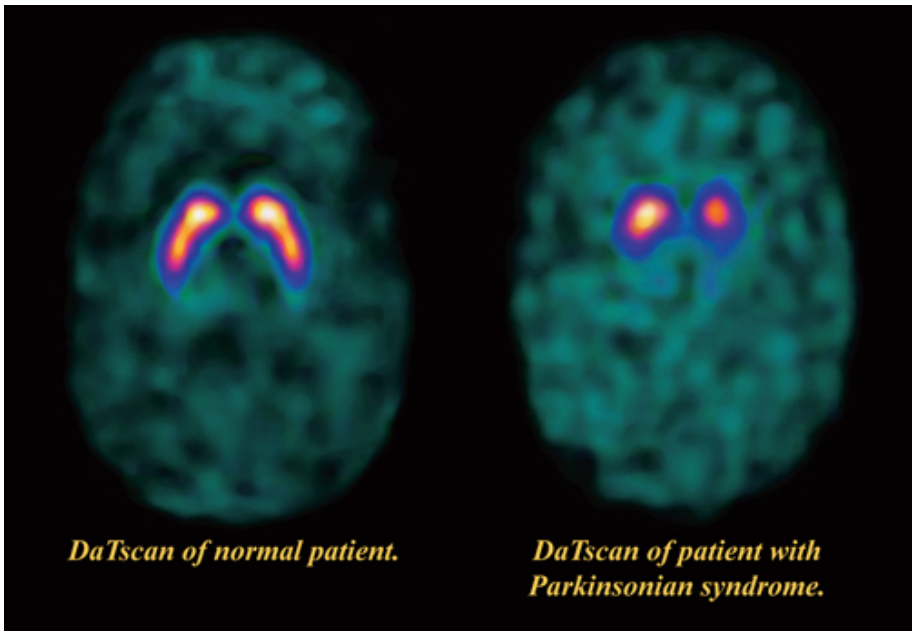


Figure 1: Parkinson's disease detection

The image on the left shows a normal patient's DatScan while the one on the right shows the DatScan of a patient with Parkinsonian Syndrome. This images shows the difference in appearance between the two cases.[2]

In the example Figure 1, we can see one of the differences between a healthy patient and an abnormal/Parkinsonian case. The healthy patient has a more "Comma" shaped highlighted region, whereas the Parkinsonian case has a "Period" shaped highlighted region. The difference in shape between the two cases is due to the different dopamine uptakes due to the scarcity of dopaminergic cells in the Parkinsonian case as compared to the Healthy Case.[13] Therefore, by observing such features, medical professionals are able to make a diagnosis. And this is exactly what the deep learning model needs to do as well.

[16]

2.4 Uncertainty-aware model

Uncertainty, in terms of deep and machine learning, refers to the lack of confidence or certainty in the output of the model. It is highly unlikely to create a model that can correctly predict each target with absolute certainty. Therefore, in some situations, uncertainty estimations are quite useful to better evaluate and draw inferences from the results of a model. Such uncertainty estimations are of paramount importance in the healthcare sector where the results of a model holds life and death of a human in balance. Neural networks are heavily inclined towards overconfident predictions. Therefore, for safe and informed decisions, a model should not just provide an output but also the certainty/uncertainty metrics related to each outcome. Such a model is referred to as an Uncertainty-aware model.[8]

One of the most crucial parts in a binary classification NN model is to understand the balance between *Type I* and *Type II* errors. Lets first understand *Type I* and *Type II* errors:

Type I Error (False Positive): This occurs when the model incorrectly predicts a positive outcome when the true outcome is negative. In the context of this study, it means classifying a patient's brain scan image as having Parkinson's disease(PD) when it does not.

Type II Error (False Negative): This occurs when the model incorrectly predicts a negative outcome when the true outcome is positive. In the context of this study, it means classifying a patient's brain scan image as being a Healthy Control(HC) when in reality the patient has Parkinson's disease(PD).

Ethically, for most medical applications, it is more important to focus on reducing *Type II* errors. Therefore, it is more important to make sure that a patient with Parkinson's disease is not wrongly classified as being a HC because that could be really fatal. But at the same time, it is also important to have a balance between the two errors to have a trade-off between over-inclusion and under-inclusion. This ensures that our model delivers accurate predictions while maintaining a degree of flexibility that prevents undue skew towards one type of error. Therefore, by balancing the model based on the specific problem, we can enhance the overall reliability and utility of our model for real-world applications.

Apart from specifically uncertainty-aware methods, the study also involves model calibration methods that could potentially help to scale the model probabilities for more accurate uncertainty quantification. One such method is Temperature Scaling, which is explained in the subsection 4.4.

3 Statement and Motivation of Research

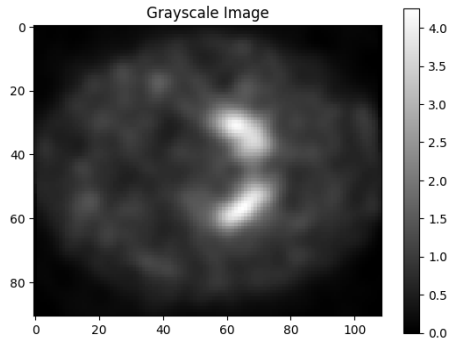
Uncertainty in medical diagnosis is a big problem. A wrongful diagnosis could potentially be the difference between the life and death of a patient. Automation techniques like deep learning could really help in making disease detection very quick and efficient, but they are also prone to making mistakes that could be fatal. Alas, when it comes to a person's life, the only area that is important is the minimisation of fatal mistakes, especially during diagnosis. So, the main goal is to be able to harness the power of these advanced detection methods while also minimising mistakes. This can be done by introducing the quantification of uncertainty for each prediction. Uncertainty metrics help attach a confidence level to a given prediction. Looking at this confidence level, a medical professional can decide whether this diagnosis needs to be confirmed further. A prediction with a low level of uncertainty would not require a second look whereas a prediction with a high level of uncertainty would definitely require to be looked at again by a healthcare professional. Uncertainty could also possibly help in detecting early and ambiguous cases of Parkinson's disease, if tuned correctly. The whole idea is to make a system where there is no possibility for a wrongful diagnosis of Parkinson's disease, yet is efficient and automated to reduce detection duration and effort. Beyond this, it is also important to see how a model deals with unfamiliar scenarios. There can be situations when a model is uncertain about certain predictions because the input data is dissimilar from what they were trained on. By making use of uncertainty, we can possibly detect data distributions which are quite distant from the training data. And because of such out of distribution(OOD) data, models can often make erroneous predictions. Therefore, it is crucial to detect such data as it helps to understand the limitations of the model. Understanding the failure modes of the model allows us to focus on improving its performance in these critical areas. In summary, embracing uncertainty in deep learning helps us gain insights into how the model handles different scenarios and identify potential weaknesses. This understanding can lead to guiding further development and research for making the most reliable Parkinson's disease detection system.

4 Methodology

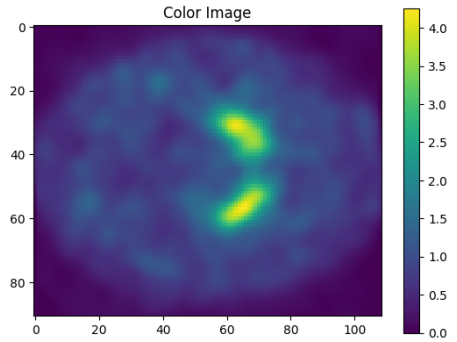
Predictive uncertainty has been explored in various applications in machine and deep learning. However, there is not yet a standard approach or research done for exploring the predictive uncertainty of Parkinson's disease using Convolutional Neural Network on Brain Scan images. Therefore, this study deals with a comprehensive step-wise approach on how to estimate predictive uncertainty for Parkinson's disease detection. The methodology was to systematically try different classification CNN models with specific techniques that could potentially predict uncertainty and to compare these different techniques to see on which aspects they performed well. This study involves the use of advanced uncertainty estimation techniques like Monte Carlo Dropout, Deep Ensemble, Temperature Scaling and Spectral-normalised Neural Gaussian Process (SNGP). The basic concept of all of these techniques have been explained in detail in the coming subsections. Before we get into the different models, we will also describe what the data looks like. In the coming subsections we will also explore the concepts of different setups, techniques, models and metrics that were used in this study.

4.1 FP-CIT SPECT Image Data

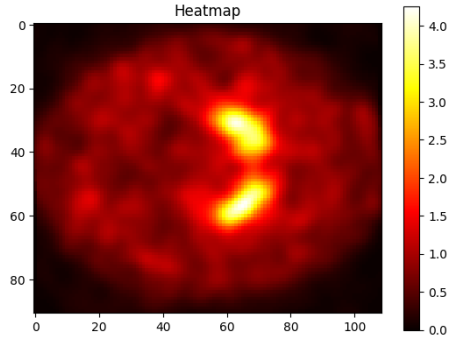
The original source data is around 645 2D images of the brain scans of healthy and Parkinson's patients. The images are in *.nii* format. The *.nii* or Neuroimaging Informatics Technology Initiative (NIfTI) is an open file format commonly used to store brain imaging data obtained using Magnetic Resonance Imaging methods.[22] There is another set of 645 2D images which are just the smoothed versions of the original. In image processing, smoothing is a method employed to decrease image noise and intricate details by utilizing a low-pass filter. This filter functions by substituting every pixel value with the neighboring pixels' average value. Smoothing enhances image appearance, facilitating analysis by diminishing the influence of minor fluctuations in pixel values.[17] Therefore, sometimes, by removing the noise, it becomes easier for CNNs to better extract the important features from the image. Consequently, it may also lead to loss of important information from the image. For this study, both smoothed and non-smoothed data were used to make sure to cover both cases. Therefore, in total, there are about 1290 images to train, evaluate and predict upon.



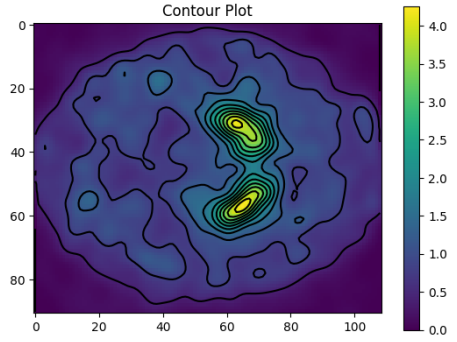
(a) Greyscale image



(b) Colorized Image



(c) Heatmap Image



(d) Contour Image

Figure 2: Different representations of a single FP-CIT SPECT brain scan image

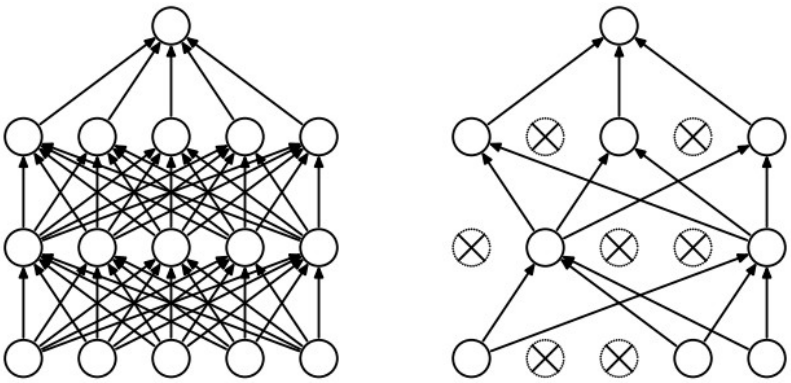
The above images are different versions of the original SPECT Image that highlight the main feature areas for our classification task. The highlighted areas are the areas of dopamine activity which are used to classify between a Parkinson's and a healthy case. The CNNs not only have the ability to extract these features for classification but also to accurately quantify uncertainty for each prediction. In fact, CNNs could also potentially learn features from other regions of an image to improve their classification and/or probabilistic accuracy. Using different methods and approaches along with CNNs, we have tried to accurately quantify predictive uncertainty.

4.2 Monte Carlo(MC) Dropout

Dropout is a technique in neural networks where a few neurons are dropped during training. This helps in reducing model complexity while also avoiding overfitting of a model. MC Dropout works initially by randomly switching off neurons in a neural network during training, which regularizes the network. Each dropout configuration corresponds to a different sample from the approximate parametric posterior distribution. Then, during the inference or test prediction step, multiple forward passes are made while dropping different neurons each time randomly. At the end, the statistics of the predictions are summarized and the output is considered an approximation of probabilistic Bayesian models in deep Gaussian processes.[4] With different sets of neurons dropping each time,

different distributions of outputs are created. This distribution of output helps in estimating the predictive uncertainty of a model.

In the Figure 3 below, you can see how MC Dropout works in a Neural Network. The example in the figure is for a single output, but MC Dropout works under the same principles for multi-output models as well. Different neurons dropping out could potentially focus on different parts of the model. Some sets of neurons might focus on accurate prediction of one of the outputs or, in the case of a single output, one of the subsets of data. While other neurons might focus on other outputs or different subsets of the data in case of a single output model. Therefore, by combining the strengths and weaknesses of all of these different neuron sets, there is a strong possibility of getting a probabilistic distribution that could more or less encapsulate the predictive uncertainty of a model.



(a) Standard Neural Net (b) After applying dropout.

Figure 3: Monte Carlo Dropout Concept

The image on the left shows a standard neural network with no dropout. The Image to the right shows the same network after applying dropout[18]

The number of forward passes through the data should be evaluated quantitatively, but 30-100 is an appropriate range to consider.[4] For this study, we have made 100 such evaluations during test time to get a big range of probabilities and aggregate over them.

4.3 Deep Ensemble

Deep Ensemble is a deep learning method that involves training multiple instances of the same neural network architecture with different random initialization or data subsets (*Bootstrapping*). It is widely used for classification, regression and clustering tasks. More specifically, for this study, it is used to quantify uncertainty of our classification problem. The idea behind Deep Ensemble is to capture epistemic uncertainty, which arises due to the model's lack of knowledge about the true underlying data distribution. By combining multiple models, the output predictions that are resulted are supposed to be better at generalisation and at quantifying the certainty or uncertainty of individual predictions. Deep ensembles have shown the potential to improve the accuracy and out-of-distribution robustness, as well as reduce the uncertainty of deep learning models.[10] Deep ensemble

learning models combine the advantages of both the deep learning models as well as the ensemble learning, such that the final model has a better generalized performance. The averaged predictions work best when each individual model has different parameters and are possibly also trained on different data samples. Therefore, when combined together, each model's strengths could mask the other model's weaknesses.[14] So, it is very important to carefully select the configurations of individual NN models such that each model can focus on confidently predicting different cases and/or areas of the dataset. In doing so, the resultant ensemble model would have the highest possibility to cover all the cases. Hence, giving the best performing model in terms of predictive uncertainty.

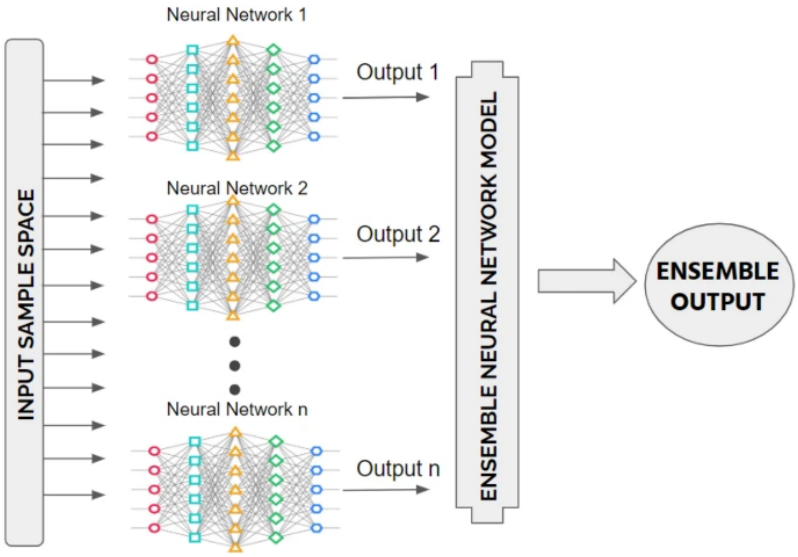


Figure 4: Deep Ensemble concept

The diagram shows a Deep Ensemble model is created by combining multiple NN models[5]

4.3.1 Mathematical Explanation of Deep Ensemble for Classification

Let's consider a classification problem with a dataset of N input-output pairs, denoted as $\{x_i, y_i\}$, where x_i is the input feature vector, y_i is the corresponding target (label), and $i = 1, 2, ..., N$.

In the context of Deep Ensemble, we train K separate neural network models (ensemble members) denoted as $f_1(x), f_2(x), ..., f_K(x)$, where $k = 1, 2, ..., K$. Each model is initialized with different random parameters or trained on different random subsets of the data.

Given a new input sample x_i, each ensemble member produces a predictive distribution over the possible classes. This is often represented as a probability vector $p_{i,k}$, where $p_{i,k,j}$ is the probability that the i-th sample belongs to class j according to the k-th ensemble member.

The final predictive distribution is obtained by aggregating the individual predictions from each ensemble member. One common approach is to average the predicted probabilities:

$$p_{i,j} = \frac{1}{K} \sum_{k=1}^{K} p_{i,k,j} \quad \text{for } j = 1, 2, ..., \text{number of classes.}$$

In this ensemble-based approach, the uncertainty associated with a specific prediction is quantified by aggregating the predictions of the individual NN models in the ensemble. The intuition behind this is that if all the models agree on a prediction, the uncertainty is low, but if they differ significantly, the uncertainty is high.

One way to represent this uncertainty is by calculating the entropy of the predictive distribution:

$$H(p_i) = -\sum_{j} p_{i,j} \log(p_{i,j}) \quad \text{for } i = 1, 2, ..., N.$$

Higher entropy values indicate higher uncertainty, whereas lower entropy values indicate higher confidence in the prediction. In this study, we will be calculating the entropy of the predictions in order to quantify uncertainty. Therefore, in summary, Deep Ensembles quantify epistemic uncertainty by considering the variation in predictions across the ensemble members, often measured using the entropy of the predictive distribution. Using this method, a model can not only make a point estimate for the predicted class but also an estimate of its uncertainty in the prediction, which can be useful in applications where confidence or reliability of the model's output is essential.

4.4 Temperature Scaling(TS)

Temperature Scaling is a post-processing step in deep learning, more specifically for probabilistic classification models. The basic idea is to calibrate the model's confidence or uncertainty for each prediction. A probabilistic model outputs probabilities for a sample belonging to each class, representing how likely it thinks the input belongs to that class. TS involves finding the optimal temperature value T that calibrates the model's confidence for more accurate probabilistic distribution of predictions. A model typically uses activation functions that can convert raw model outputs(called *logits*) into probabilities. [15] For example: If we use an activation function called *softmax*, then this is how we can calculate the new calibrated probabilities using the optimal temperature value:

$$P(y_i) = \frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}}$$

Where:

- $P(y_i)$ is the probability of class i,
- z_i is the raw output (logit) for class i,
- T is a parameter called the temperature.

So, technically, the raw logits are calibrated and softened using the optimal temperature and from these calibrated logits we get the calibrated probabilities by using the required activation function. Calibrated probabilities are extremely essential for applications involving uncertainty estimations. By applying temperature scaling, the model's uncertainty estimates are more accurate and meaningful, leading to better decision-making and more reliable performance in various applications. Therefore, for this study, TS is an important method to try to estimate the predictive uncertainty.

4.5 Spectral-normalised Neural Gaussian Process(SNGP)

The core idea of SNGP is to improve a deep classifier's distance awareness by applying simple modifications to the network. A model's distance awareness is a measure of how its predictive probability reflects the distance between the test example and the training data. It can also be defined as the model's ability to properly quantify the distance of a testing example from the training data manifold, as a necessary condition for a DNN to achieve high-quality (i.e., minimax optimal) uncertainty estimation. This is a desirable property that is common for gold-standard probabilistic models (for example, the Gaussian process.external with RBF kernels) but is lacking in models with deep neural networks. SNGP provides a simple way to inject this Gaussian-process behavior into a deep classifier while maintaining its predictive accuracy.[19][11]

SNGP is a simple approach to improve a deep classifier's uncertainty quality while maintaining a similar level of accuracy and latency. Given a deep residual network, SNGP makes two simple changes to the model:

1. It applies spectral normalization to the hidden residual layers.
2. It replaces the Dense output layer with a Gaussian process layer.

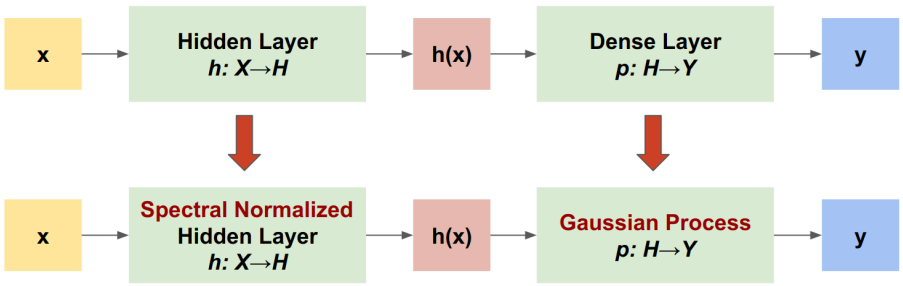


Figure 5: SNGP model concept

The diagram shows how to change a general Neural Network model to apply SNGP to it[19]

4.5.1 Spectral Normalisation

Spectral Normalisation is a normalization technique that is used to control the behavior of certain mathematical functions. This method basically puts a limit on how fast these functions can change their output compared to their input. Spectral Normalisation helps

to smooth the learning process by adding these speed limits. Suppose there is a function that takes some input and gives an output. The Lipschitz constant of this function will be the measure of how much the output can change concerning changes in the input. This change should be limited so that the function does not generate extreme and unpredictable outputs for small changes in the input.We will focus on Lipschitz continuous functions for our purpose. This basically means that if you take two different inputs and look at the difference between the outputs, it won't be too much larger than the difference between the inputs. Now, let's consider a specific kind of function called a linear map. It takes an input and multiplies it by a matrix (let's call it W). For this linear map to be well-behaved, we want to ensure that the output doesn't grow too fast compared to the input. In other words, we want to limit the amplification of the input. By normalizing the spectral radius of the matrix W, which is essentially the largest value of its singular values, we can enforce this desirable behavior. It's like scaling the matrix in such a way that its largest singular value becomes equal to 1. This ensures that the function is 1-Lipschitz, meaning the output doesn't grow more than the input. This concept is also applicable to convolutions, which we will be primarily using in all our neural network models for this study. By putting a cap on and controlling the maximum value of convolution operations, spectral normalisation can keep a check on their behavior. In summary, Spectral Normalisation is a technique used to restrict the behavior of certain mathematical functions, like linear maps and convolutions, so that they don't amplify their input too much. This helps in stabilizing and improving the performance of various functions, algorithms and models.[6]

Spectral Normalisation also acts as a regularization technique that restricts the model's capacity to over-fit the training data. By controlling the Lipschitz constant, it constrains the expressiveness of the neural network, preventing it from memorizing noise or irrelevant patterns in the data. This regularization effect improves the model's ability to generalize to unseen data, leading to better performance on validation and test sets. It is also very applicable and adaptable to any deep neural network architecture, which makes it quite easy to implement. Overall, it is a good approach that could possibly help in stabilising the predictive probability and uncertainty so that it gives more realistic values that could help understand the uncertainty associated with the predictions.

4.5.2 Random Feature Gaussian Process

A Gaussian process in probability and statistics, represents a distribution over functions by specifying a multivariate normal (Gaussian) distribution over all possible function values. It is possible to easily manipulate Gaussian distributions to find the distribution of one function value based on the values of any set of other values. A Gaussian Process in machine learning is a supervised machine learning method that is designed to solve regression and probabilistic classification models. For example: You have the price of 5 houses in your area for the last 10 years. Now you want to predict the price of those 5 houses after 5 years. You can do this using a Gaussian process. The Gaussian process will learn from past house prices and form mathematical relations to make predictions for future house prices, including after 5 years. It's called *"Gaussian"* because it uses something called a Gaussian distribution (a bell-shaped curve) to describe uncertainty and probabilities.

Traditionally, a Gaussian Process calculates the similarity between all the data points and then uses that information to make predictions. However, when the data is massive, this approach could turn out to be quite slow and computationally expensive. To circumvent this issue, we can use something called a *'Random Feature'*. Instead of directly using all data points to calculate the similarity, a Random Feature can be used to transform the data into higher dimensions in such a way that it becomes easier to analyze and identify patterns. This transformation to higher dimensions is done in a completely random manner, hence, the name *"Random"*. Therefore, by combining a Gaussian Process with a Random Feature expansion, we can make it easier to find patterns and make predictions without getting bogged down by complicated calculations. This is called a Random Feature Gaussian Process.[3]

4.6 Uncertainty Quantification Metrics

In order to quantify uncertainty, we need certain metrics on which we can compare the different models and approaches that we have mentioned in the above sections. The following metrics have been used in this study that help in comparing the uncertainty of different models and approaches:

4.6.1 Entropy

Entropy is a measure of uncertainty. More specifically for this study, Entropy is the measure of how uncertain or certain a model can get for each prediction. It will basically tell us how certain or uncertain a model can be regarding a patient either having Parkinson's disease(PD) or being a Healthy Control (HC).

To get more specific, the Entropy that is related to predictive uncertainty in machine and deep learning studies such as this one is called Shannon's Entropy.[24]

Shannon's entropy formula is given by:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) \tag{1}$$

where:

- $H(X)$ is the entropy of the random variable X,
- n is the number of possible outcomes of X,
- $p(x_i)$ is the probability of outcome x_i.

As shown above, entropy is calculated from probability. All the NN models used are probabilistic models and, therefore, can provide probabilities. Entropy will be calculated using these probabilities and will be used to measure and compare the uncertainties among all NN model approaches. Therefore, entropy will be the prime differentiating factor between the performance of various approaches.

5 Implementation

In this section, we will have a look at the details of how the methods mentioned in the previous section were actually used to train neural network (NN) models to classify and quantify predictive uncertainty in the Parkinson's classification task. This section provides a detailed walk-through starting from the data preparation, data augmentation, model architecture, model training and finally, model testing.

5.1 Parkinson's image data preparation

The data to be trained upon in this study is 2D brain scan gray-scale image data of dimension (109, 91). Originally, the data was in the *.nii* 3D image format. For this study, however, we are specifically focusing on quantifying predictive uncertainty for classification of 2D brain scan images. Therefore, we already had access to 2D brain scan images in *png* format. Having the images in *png* format, it was quite fast, and convenient to load, transport, save and use the data. As compared to *.nii* image files, *png* image data formats are smaller in size, compatible, easy to handle and fast to load and process. Initially, there were about 645 images, but for the possibility of further feature extraction, these 645 images were smoothed and these smoothed images were also added to the dataset. As a result, in total, we had access to 1290 images to work with. The next step was to process the image data and prepare it so that it is compatible to be put through a Deep Learning Convolutional Neural Network. It would be unwise to feed all 1290 images through a Deep CNN model because it can lead to memory overflow, especially when dealing with a large number of high-resolution images. Another crucial reason to use a data generator is to make the implementation scalable and adaptable for future endeavours. The data preparation system should be versatile enough to be able to tackle not only different sizes, but also different modalities of data for further work. The use of a data generator allows for such functionalities and more. It can be used for loading the image data in batches so as to prevent any memory overflow while also supporting parallelization. As the first batch of data will be fed to the deep NN, the next batch will simultaneously be prepared for delivery.

5.2 Data Processing and Augmentation

For this study, the main reason for using a data generator was data processing and augmentation. Data augmentation is a common technique used to enhance the diversity of the training data and improve model generalization. Data generators can apply data augmentation techniques in real-time during training, generating different variations of the same data on the fly for each batch. The data generator used for this purpose was *ImageDataGenerator*. The *ImageDataGenerator* class is part of the Keras (now part of TensorFlow) library and is used for real-time data augmentation and pre-processing of image data during model training. Now let's have a look at the specifications of image processing and augmentation.

Parameter	Value	Type and/or Range
rescale	1/255	N/A
shear_range	0.1	0-1
zoom_range	0.1	0-1
width_shift_range	0.1	0-1
height_shift_range	0.1	0-1
rotation_range	10	0-90
horizontal_flip	True	boolean(True, false)
vertical_flip	True	boolean(True, false)

Table 1: *ImageDataGenerator* Augmentation Parameters

The above table shows the data augmentation parameter settings based on which the images were augmented during training. Following are the description of what each parameter does:

rescale=1./255: Scales the pixel values of images to the range [0, 1] by dividing each pixel value by 255. This is a common normalization technique for image data.

shear range, zoom range, width shift range, height shift range, rotation range, horizontal flip, vertical flip:

These are parameters that control the data augmentation techniques applied to the images during training.

shear range: Applies shear transformations (slanting) to the images.

zoom range: Applies random zooming in or out to the images.

width shift range and *height shift range*: Apply random horizontal or vertical shifts to the images.

rotation range: Applies random rotations to the images.

If *horizontal flip*=True, random horizontal flipping of the image will be applied during training.

If *vertical flip*=True, random vertical flipping of the image will be applied during training.

Now, let us see how the image data will look like upon randomly applying the above augmentations.

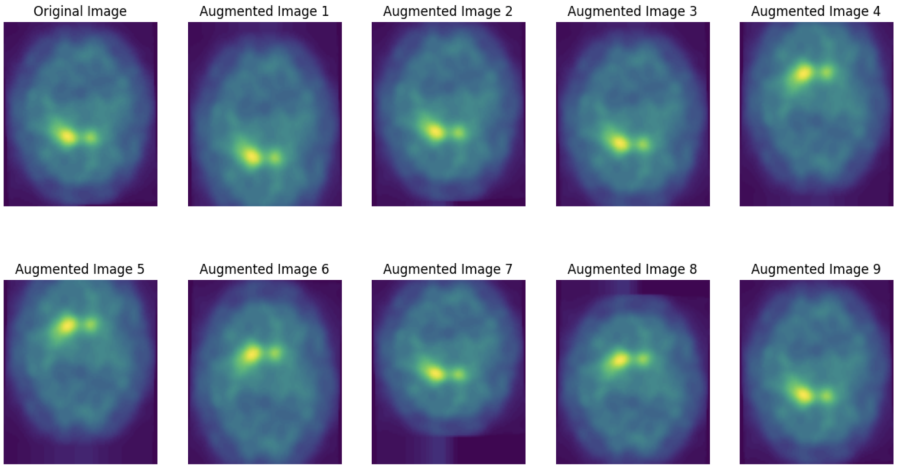


Figure 6: Data Augementation by the *ImageDataGenerator*

The diagram shows the original image(top left) and the various random image augmentations done by the *ImageDataGenerator* for the mentioned settings.

In the above Figure 6, the original image can be seen on the top left. The rest of the 9 images are generated by applying some or all of the data augmentations randomly to the original image by the *ImageDataGenerator*. The data generator generates these random augmentations each time it is run. This random nature can be controlled by setting a seed that ensures that the augmentations are generated in the same manner each time. The augmented images are only slightly different to the original image because the degree of augmentation is quite subtle for all parameters. Subtle augmentations aim to mimic real-world variations that images might encounter, such as minor changes in lighting, rotation, or perspective. This can lead to more realistic and natural-looking augmented images. These subtle changes also help preserve the context of the image while still making some alterations and therefore are quite essential for the problem statement. Drastic augmentations could possibly cause the loss of feature rich regions, thus causing poor feature extraction and loss of context.

The purpose of using an image data generator with data augmentation is to enhance the diversity and size of the training dataset without manually creating additional images. By applying random transformations to the images, the model gets more diverse and specific data to train on. Hence, the model becomes more robust and generalizes better to unseen data. During model training, the generator will take the input images, apply the specified data augmentation techniques, and feed the augmented images to the model in batches. This process is done on-the-fly during training, which is memory-efficient and allows for the training of large datasets.

5.3 Data Splitting

From a total of 1290 images, this is how the data was split using the Train-Validation-Test strategy:

Dataset	Percentage	Number of Images	Purpose
Train	roughly 70%	904	Model Training
Validation	roughly 15%	200	Model Validation
Test	roughly 15%	186	Performance Evaluation

Table 2: Train-Validation-Test Split

The training set, as the name suggests, is strictly used during the training process. The validation set was also used in parallel with the model training to validate the performance of the model at every epoch and help in converging the loss of the model to the minimal possible value. The test set was to be kept untouched until making predictions and evaluating the predictive power of the model. It is always a best practice to not involve the test set in another prior process so as to avoid any data leakage that could possibly cause bias while evaluating the predictions. Another way data leakage was avoided by carefully splitting the original images and their smoothed counterparts. If an original image is in the training set, then its smoothed version must also be in the training set itself and not in the validation or test set. This is because they are essentially two versions of the same image which will definitely cause data leakage if present in different sets. The reason is that the model has already trained on a version of the same image and so on encountering another version of the same image in a validation or a test set would introduce bias into the whole process. Therefore, to prevent such bias, care was taken to split both versions of the images across all the sets.

The same data splitting strategy is used for all the future methods throughout the duration of this study, unless specified otherwise.

5.4 Model Architectures

This section deals with a detailed description of the NN model architectures for the various methods being implemented for uncertainty quantification.

5.4.1 Base CNN Classification Model Architecture

The base CNN Classification model was constructed by using a hit and trial method. After comparing the classification results of various CNN architectures, the following architecture was settled upon. This is what the model architecture looks like:

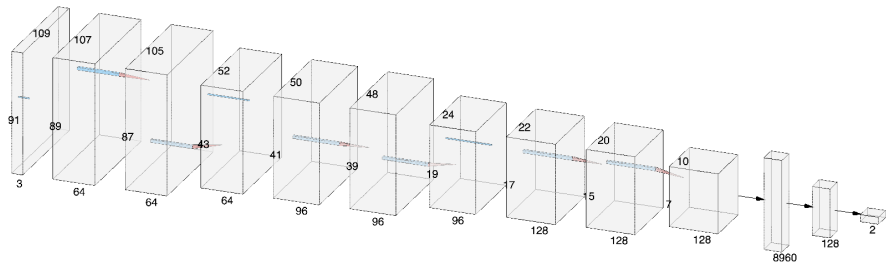


Figure 7: Base CNN Model Architecture

The plot shows the architecture for the Base CNN model

The architecture for the base CNN model starts with two consecutive convolutional Layers followed by a max pooling layer. As visible from the model architecture, both the convolutional layers have 64 filters with a kernel size of 3x3. Then we have the max pooling layer. This above set of 2 convolutional layers and a max pooling layer is repeated two more times. The only difference being that the number of filters for the convolution layers changes in each repeated set from 64 to 96 and then to 128 for the final set. Therefore, in total we have 6 convolutional layers and 3 max pooling layers. The progressive increase in the number of filters in each set of convolutional layers is helpful in learning large scale patterns and features in an image. Larger filters also allow the network to learn more hierarchical representations of the input data. With an ascending number of filters in subsequent convolution layers, the early layers capture repetitive low-level features, whereas the later layers with a higher number of filters are able to capture more high-level and a variety of features, which are composed of elementary features with a larger range. Deeper layers with increased filters can help improve the network's ability to recognize objects regardless of their position, rotation, or scale. The network learns to extract features that are invariant to these transformations, leading to more robust predictions. Since we already know that Parkinson's detection is done by checking the bilateral reduction in dopamine activity in the Striatum (See Figure 2.3), a model which could capture these spatial relationships could be quite beneficial for relevant feature extraction. The max pooling layers, on the other hand, are used to reduce feature map dimensions, increase translation invariance, and capture hierarchical features within the images. It selects the most prominent feature within a region, making the network robust to variations, memory-efficient, and better at recognizing complex patterns. Then comes the flatten layer that helps to flatten the image into a 1-D vector. It does so so as to properly relay the information between the convolutional layers and the upcoming dense layers. The flattened information is then fed to a dense layer of 128 units with spectral normalisation (See subsubsection 4.5.1) applied to it. After the spectral normalisation layer, the final output layer is added. This output layer consists of 2 outputs with a softmax activation function. The softmax activation function helps to give a normalized probability score for each of the 2 classes. It is a common and crucial part of a NN model that helps to interpret the raw outputs of a model into meaningful probability values.

This architecture is treated as a baseline for experimentation for this study. And the upcom-

ing architectures will build upon this base architecture with some additional components. Therefore, the base structure of the base CNN model will be continuously referenced to further explain other models.

5.4.2 Monte carlo (MC) Dropout model architecture

For Monte Carlo Dropout, the model architecture is quite identical to the Base CNN model explained in Figure 5.4.1. The only difference is that the MC Dropout model has 2 dropout layers and an extra dense layer. Both the dropout layers have a dropout probability of 0.25. This means that at each epoch, 25% of neurons in the layers will randomly switch off. The dense layer is added after the first dropout layer and is made up of 256 dense units with a ReLU activation. The first dropout layer is applied to the output of the spectral normalisation layer. Then we have the above mentioned dense layer. On the output of this dense layer, another dropout layer with the same configuration is applied. After this final dropout layer, there is the output layer with 2 units and softmax activation.

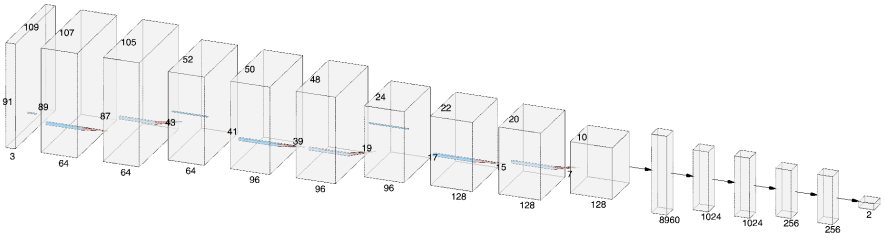


Figure 8: Monte Carlo Dropout Model Architecture

The plot shows the architecture for the Monte Carlo Dropout Model

5.4.3 Deep Ensemble Model Architecture

The Deep Ensemble model is an aggregate combination of multiple models. As mentioned in Figure 4.3, this model combines the predictions of different multiple models to give a good probability distribution of predictions. In this study, the deep ensemble model is an ensemble of 3 different CNN models. These 3 CNN models are very similar in their basic architecture except for a couple of layer changes. Let's start with the architecture of the first model.

The first model used in the Deep Ensemble is the Monte Carlo Dropout model as explained in Figure 5.4.2 and depicted by Figure 8. All the remaining configurations and processes also remain the same as that of the MC dropout model.

The second model used in the Deep ensemble is the base CNN model as explained in Figure 5.4.1 and depicted by Figure 7. All the remaining configurations and processes also remain the same as that of the base CNN model.

The third model of the deep ensemble is very similar to the MC dropout model architecture except for the addition of 3 strategically placed batch normalisation layers.

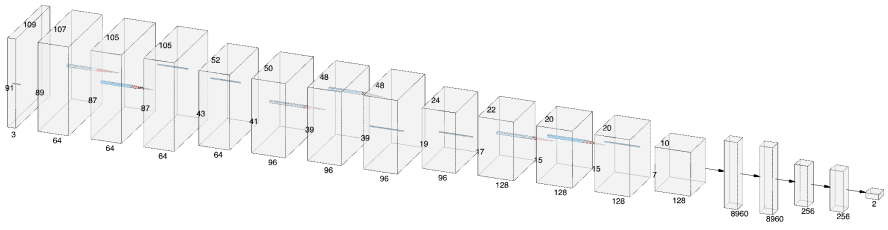


Figure 9: Deep Ensemble 3rd Model(or BN model) Architecture

The plot shows the architecture for the 3rd Model(or BN model) in the Deep Ensemble

The additional layers in the above model architecture are the Batch Normalisation(BN) layer after every set of 2 consecutive convolutional layers followed by a 1-max pooling layer. Since there are 3 such sets, we have 3 batch normalisation layers, each placed after each set. A batch normalisation layer helps to stabilize and control the learning of a NN. This helps stabilize training by preventing extreme activations that can lead to vanishing or exploding gradients. In totality, it helps to improve model stability, faster convergence and makes the model more robust.

An aggregate of the probabilities of the 3 models was taken and the resulting probabilities were considered as the final deep ensemble probabilities. These probabilities were then used to calculate the entropy for each prediction.

5.4.4 Temperature Scaling Model Architecture

The Temperature Scaling(TS) model has all the same top layers as the MC dropout model as explained in Figure 5.4.2. The only difference being the output layer having one output unit with no activation. With no activation layer, the model is also setup to output raw logits. These raw logits are required for TS. The new temperature value is calibrated upon these raw logits and not the probabilities. The initial default temperature value is set to 1.0.

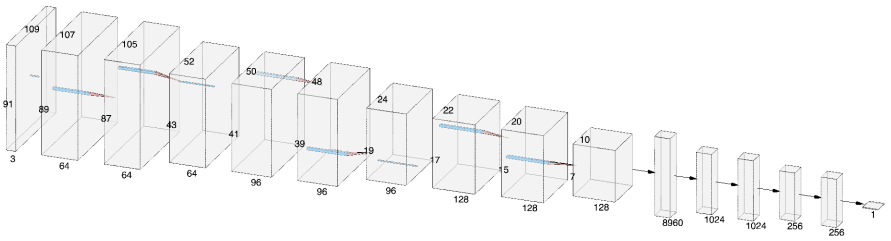


Figure 10: Temperature Scaling Model Architecture

The plot shows the architecture for the Temperature Scaling Model

The first step is to divide the predicted logits with the initial temperature value. Then,

the sigmoid cross-entropy loss between the scaled logits and the ground truth labels is computed. The mean of this sigmoid cross-entropy loss computed over the whole dataset is considered as the final loss that makes up the loss function for finding the new temperature value.

An Adam optimizer with a learning rate of 0.01 is initialized. The training loop for finding the new temperature performs 300 optimization steps (iterations). In each iteration, the optimizer minimizes the loss function with respect to the temperature value. This means it adjusts the value of temperature to minimize the loss and better calibrate the model's predictions.

After all the iterations, the final optimum value of the temperature is achieved. Using this new temperature value, the raw logits are scaled and calibrated, which gives the calibrated logits. These calibrated logits, when fed to a sigmoid function, give the scaled and calibrated probabilities of class 1 (PD). The probabilities of class 0 (HC) can be calculated by simply subtracting the probabilities of class 1 from 1.0 (total probability).

Therefore, to summarize, the goal is to adjust the temperature parameter to achieve better-calibrated probabilities for the model's predictions. This approach helps align the model's confidence with its accuracy and can improve the reliability of the predicted probabilities.

5.4.5 SNGP Model Architecture

The SNGP model has all the same layers as the Base CNN model except for the following changes: The hidden dense layer is replaced by a Spectral Normalised hidden layer. The output dense layer is replaced by a Gaussian Process Output layer. As mentioned in Figure 4.5, by training a model with the combination of spectral normalisation and the Random Feature Gaussian process output layer, the distance awareness of a model gets significantly better[19].

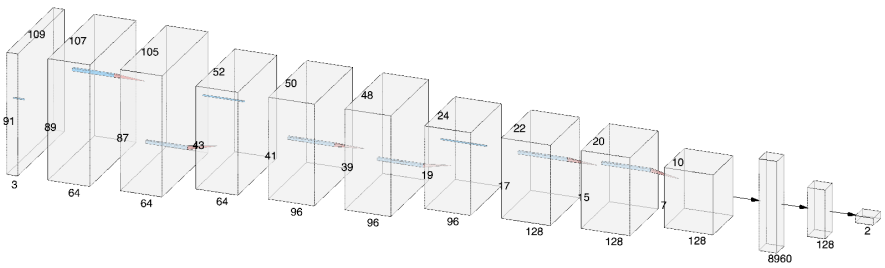


Figure 11: SNGP Model Architecture

The plot shows the architecture for the SNGP Model

Another important process involved with using the SNGP method is resetting the covariance matrix at the beginning of each epoch. Resetting the covariance matrix introduces a form of regularization. By doing so, the model is prevented from becoming overly

confident in its predictions based on the covariance estimates from previous epochs. This encourages the model to explore different predictions and can help to prevent it from getting stuck in local minima. Therefore, specifically for uncertainty quantification, resetting the covariance matrix could be quite helpful for accurate and realistic probabilistic prediction for each case.

After training, the SNGP model is predicted on the 193 test images. The output of the SNGP model for these predictions are the raw logits and the covariance matrix. Using the raw logits and the covariance matrix, the posterior mean probabilities for the predictions are computed. This is the final probability for the class prediction for each image. Using these probabilities, the entropy can be calculated for each prediction to get the measure of uncertainty.

5.5 Model Training and Validation

All the NN model methods were trained and validated using similar model configuration settings. Therefore, the model training specifics for all the models are explained altogether in this section. The model is trained for 100 epochs in batches, with each batch having 128 images. During the model training, the 219 validation set images were also used simultaneously for validating the performance of the model training. A validation set helps to see in real time how well the model is learning and generalizing on new data in parallel. For training the model, the following specific loss functions, metrics and optimizer were used:

Setting	**Value**
Loss Function	Sparse Categorical Cross Entropy
Metrics	Sparse Categorical Accuracy
Optimizer	Adam(learning rate: 1×10^{-4})

Table 3: Model Configuration

Sparse Categorical Cross Entropy (CE) is a loss function that is usually used for classification tasks with multiple classes. In our case, even though we have 2 classes, we could use this loss function. The main reason behind using Sparse Categorical CE is because the target labels/classes have already been integer-encoded by the data generator. In this case, you don't need to one-hot encode the target labels. The loss function computes the cross entropy between the predicted class probabilities and the true integer class labels. Instead of taking a vector of classes, Sparse Categorical CE takes a single integer for each class, which makes it quite memory efficient. Therefore, to summarize, Sparse Categorical CE is quite suitable for this task as it reduces overhead on one-hot encoding while also preserving memory.

Since we used Sparse Categorical CE as the loss function, for the same reasons, Sparse Categorical Accuracy was used as the metrics for checking the training and validation performance of the model.

For optimizing the model, the Adam optimizer was used with a learning rate of 1×10^{-4}.

The above model configurations are exactly the same for all the implemented methods.

5.6 Model testing

After training and validation, another additional method of finally evaluating the performance of the model is by testing the model predictions on an unseen set of data, which in this case, is the test set with 193 images. The model predicts probabilities for both the classes for the test images. To check the classification performance of the model, the probabilities are converted to classes by picking the class with the bigger probability value for each image. The classification performance of the model is judged by using metrics such as confusion matrix, accuracy, precision, recall, f1-score, ROC curve and AUC. As for the uncertainty quantification, the probability values are converted to entropy. This entropy is plotted and visualized in multiple different ways to assess the predictive uncertainty of the model. The above mentioned plots and inference metrics for classification and uncertainty quantification are discussed in the results, in Figure 6.4.

For all the methods except MC dropout and SNGP, the testing/prediction process is the same. For testing the classification performance of the MC Dropout model, predictions were made on the unseen test dataset with 193 images. Although the prediction method for the MC Dropout model as explained in Figure 4.2 is different than a normal model prediction. In order to get a good predictive probability distribution, during testing, 100 forward passes are made and at each pass, different sets of neurons are dropped off randomly. As specified in the model architecture, 25% of the neurons are randomly dropped at each pass. The probabilities of a 100 forward passes are averaged and that gives the final probability values for both classes. These probabilities are values between 0 and 1. Therefore, class predictions are made by comparing the probability values of the two classes for the same image. The class having the bigger probability value is considered the predicted class for the image.

As for the SNGP model, the predictions are also made on the 193 test images. But the output of the SNGP model for these predictions are multidimensional. This model gives out the raw logits and the covariance matrix. Even though the model architecture(See Figure 11) shows 2 units for the output layer, the actual output rather gives 2D model logits and 2D model covariance matrix. Using the raw logits and the covariance matrix, the posterior mean probability for the predictions are computed. This gives the final probability for the class prediction for each image. Using these probabilities, the entropy can be calculated for each prediction to get the measure of uncertainty.

Once the class predictions are computed, the classification accuracy is measured by using classification metrics that we will see in the Results(Figure 6.1) section.

5.7 Uncertainty Quantification Evaluation

The predictive probabilities that are outputted from all the different models is used to measure the uncertainty of the model. For measuring this uncertainty, the entropy is calculated. The entropy is calculated using the formula explained in Equation 4.6.1. This entropy is calculated for each prediction and it helps to understand the level of uncertainty associated with each prediction. The higher the entropy value, the higher will be the uncertainty and vice versa. In the results in Figure 6.4, various visualizations of the entropy of each method are displayed that help better understand and compare the predictive uncertainty quantification of each method.

6 Results

This section deals with analysing and comparing the performance of each of the 6 experimental deep learning methods discussed previously. The main areas of interest for this study is to look at how correctly can the model predict between a Parkinson's patient and a healthy control and with how much certainty/uncertainty. This would help understand how each model performs in terms of classification and uncertainty quantification specifically for the purpose of this study. For classification, confusion matrices have been plotted. The uncertainty has been visualised with the aid of UMAP entropy projections (Figure 6.2) and entropy categorical plots (Figure 6.3) to present a properly interpretable visualization. The mechanism for the above mentioned plots are explained in detail in the coming subsections.

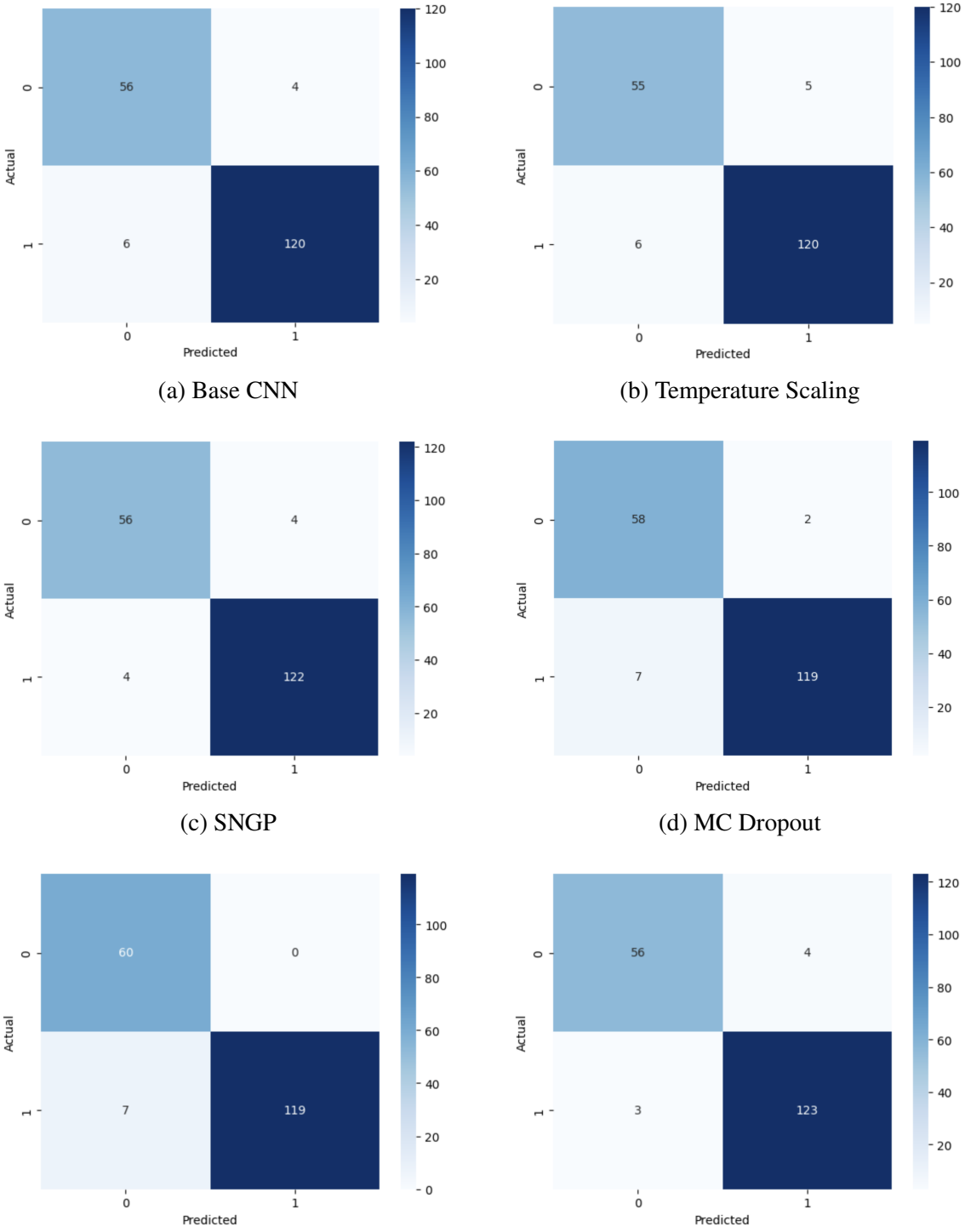
6.1 Confusion Matrix Analysis

A confusion matrix is a very common and crucial tool to help analyse how well a model classifies between the different classes. This metric plots the relationship between the predicted and true labels. Thus, it gives a comprehensive breakdown of correct and wrong predictions into further different categories to better understand the performance. The following table shows the different categories/breakdowns in a confusion matrix:

Term	Description
False Positives	Incorrectly predicted positive instances
False Negatives	Incorrectly predicted negative instances
True Positives	Correctly predicted positive instances
True Negatives	Correctly predicted negative instances

Table 4: Confusion Matrix classification terminologies

A positive instance in the context of this study is the presence of Parkinson's disease in a patient and a negative instance is a healthy control. This breakdown is essential for diagnosing and addressing specific shortcomings of the model. Specifically for this study, which relates to a medical issue, it is important to keep false negatives in check so that a Parkinson's patient is not wrongly diagnosed as a healthy patient as it could potentially be fatal.



(a) Base CNN

(b) Temperature Scaling

(c) SNGP

(d) MC Dropout

(e) BN (3rd DE)

(f) Deep Ensemble

Figure 12: Confusion matrix plots for all the NN model methods

The above confusion matrices give a robust idea of the classification performance of each model. The Base CNN model has 6 False Negatives (FNs) and 4 False Positives (FPs) with a total of 10 misclassified points. Compared to it, the Temperature Scaling (TS), SNGP and the Monte Carlo Dropout(MC Dropout) perform only marginally better with 9

misclassified points. For the SNGP model, the FPs and FNs are equal with 4 misclassified samples for each class. Thus, for SNGP, there does not seem to be a bias towards a particular class in terms of classication prediction. The MC Dropout model, on the other hand, has 7 FNs and 2 FPs. This indicates that the MC Dropout model is relatively more proficient in correctly predicting Healthy Control (HC) cases as compared to Parkinson's disease (PD) cases. Thus, it can be said that the classification predictions for the MC dropout model has a high Negative Predicted Value (NPV). NPV is the ratio of the True Negatives(TNs) to all the negatively predicted results.

The BN (or 3rd DE) model and the Deep Ensemble(DE) model perform the best in classification with only 7 misclassified points for each model. All the 7 misclassified points are FNs, which indicates that the predictions are biased towards the HC cases and there is scope of improvement for the prediction of PD cases. The Deep Ensemble(DE) model is very balanced in its misclassification of HC and PD cases with 4 FPs and 3 FNs. Therefore, just like the SNGP model, the DE model confusion matrix does not reveal any major biases towards one of the 2 classes.

6.2 UMAP Projection Entropy Plots Analysis

The predictive uncertainty of a model will be measured by calculating the entropy of each prediction as mentioned in subsection 5.7.This entropy has been visualized in multiple ways in this study.

Since the data points are images, it is challenging to plot them in a 2D representation, as is. Therefore, in order to visualize the predictive uncertainty for each image, the images need to be converted to a low dimension data point so that they can be visualized in a 2D space. One of the ways to achieve this is using UMAP (Uniform Manifold Approximation and Projection). UMAP is a dimensionality reduction technique used to convert higher dimension objects into lower dimensions for better visualisation. For this study, UMAP is used to create a 2D representation of the high-dimensional feature space that the image's pixel values were transformed into. UMAP uses the concept of distance between data points in the original high-dimensional space. It creates a neighborhood graph where each data point is connected to its nearest neighbors. This graph captures the local relationships among data points. It also builds a fuzzy topological representation of the data by modeling how likely points are to be connected based on their distances. This is done through a process that balances the desire to preserve the neighborhood structure with the need to simplify the representation. The goal is to create a projection that preserves the intrinsic structure of the data while providing a simplified representation for visualization, clustering, or other data analysis tasks. Once the images are reduced successfully to be represented in a 2D space, they are plotted along with the respective entropy values that will help understand the uncertainty performance of the respective models. The following are the UMAP settings used for the study for converting the test images into a 2D representation:

Settings	Value	Description
n_neighbors	15	Determines the number of neighbors
min_dist	0.6	Controls the minimum distance between points in the low-dimensional space
n_components	2	Specifies the number of components (dimensions) in the low-dimensional representation
random_state	42	Sets the random seed for reproducibility

Table 5: UMAP Projection Settings

After attempting multiple different UMAP setting combinations, the above one was settled upon as it provided the clearest visual representation compared to others.

In these UMAP projection entropy plots, the data points are categorised by different representations, such as: red solid circle: correctly predicted PD (TP), green solid circle: correctly predicted HC (TN), green solid triangle: misclassified HC points (FP), red solid triangle: misclassified PD points (FN). The entropy is displayed as a translucent marker on top of the data points to indicate the level of uncertainty. The red translucent markers are for the PD cases and the green translucent markers are for the HC cases. The bigger the size of the marker, more is the entropy and therefore, more is the uncertainty for the respective predictions and vice versa. This plot gives a qualitative idea of how certain/uncertain the model is with all the predictions. There are 6 UMAP plots, one for each model, that might give a rough idea about the uncertainty quantification prowess of each of them.

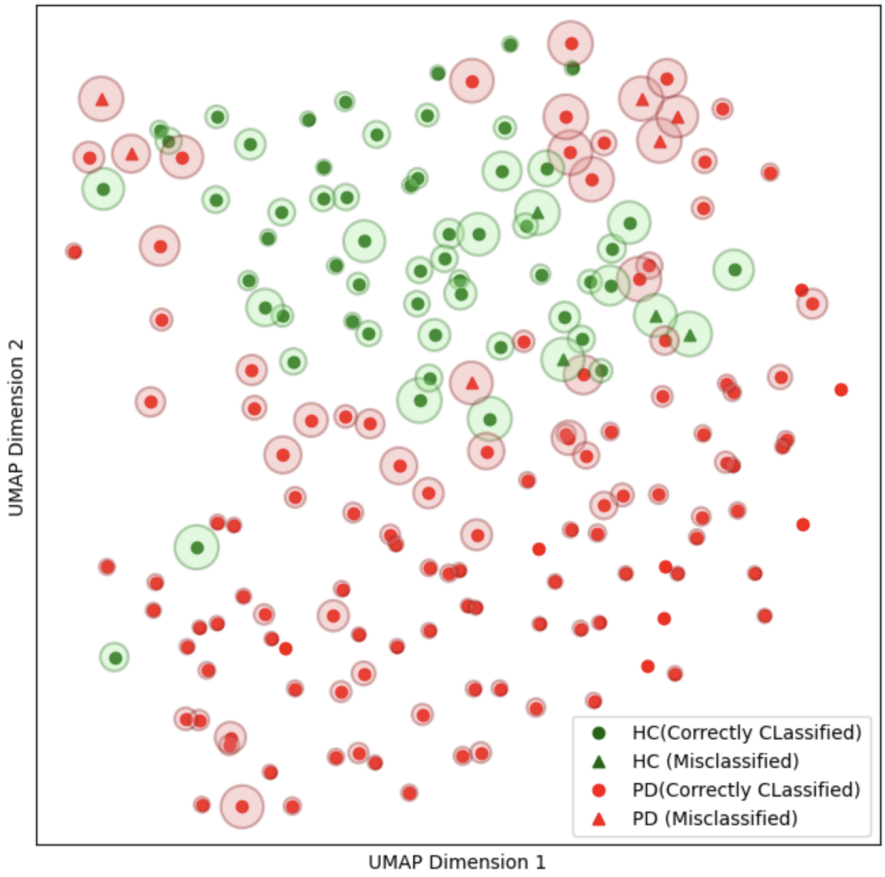


Figure 13: Base CNN UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the baseline model(Base CNN model).

In the above Figure 13, upon noticing the misclassified points(red and green triangles), most of them have a relatively large entropy marker above them. This means that for most of the misclassified points, the Base CNN model at least has the ability to provide some uncertainty quantification to provide a degree of confidence in the prediction. A high entropy value indicates high uncertainty and therefore, it symbolizes a low confidence in the given prediction. So, we can at least say that the Base CNN model is not completely lopsided in the uncertainty quantification for most of the misclassified points. On further analysis, some of the correctly classified HC and PD points also have a relatively large entropy marker. This could either be because these images are genuinely ambiguous or that the model is unable to correctly quantify the uncertainty of prediction for these samples. Another majorly noticeable information that we could get from this plot is that most of the correctly classified PD points have a relatively small or in some cases even negligible entropy markers around them, as compared to most of the correctly classified HC cases. Therefore, the Base CNN model is comparatively more confident in the prediction of PD cases than HC cases. It should be noted that a high confidence/certainty does not equate to an accurate uncertainty quantification. This model could very well be overconfident in its predictions for the PD cases. Therefore, in an attempt to asses and improve the uncertainty quantification, multiple advanced methods have been applied for which we will see the

results in the coming paragraphs.

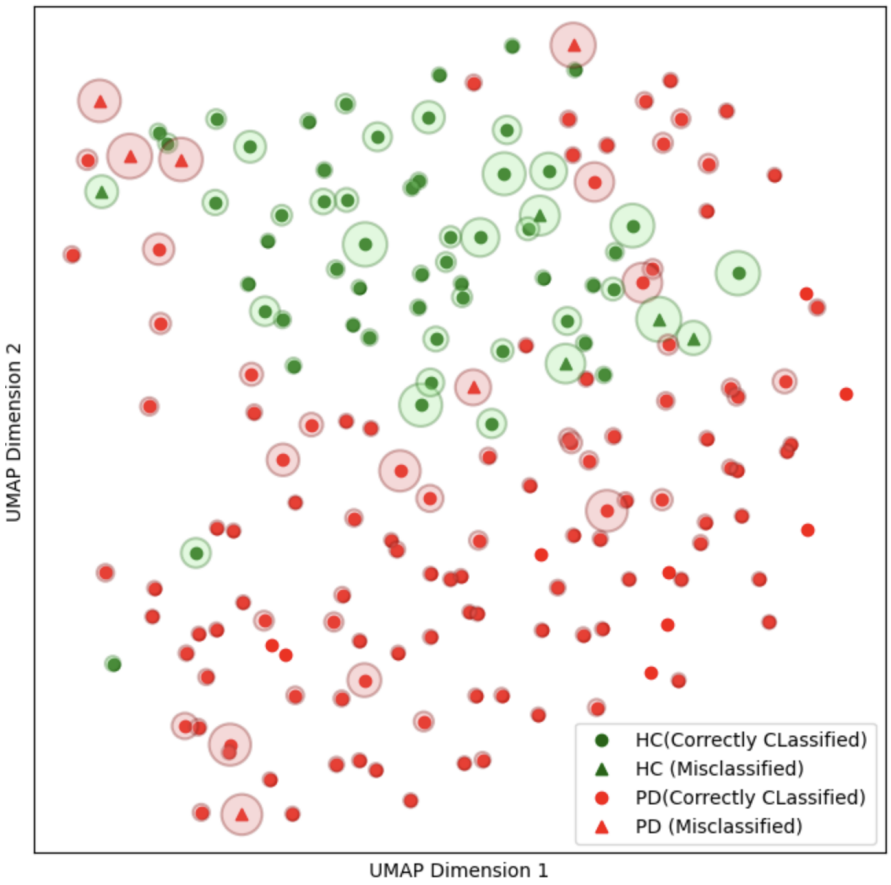


Figure 14: Temperature Scaling UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the Temperature Scaling model.

The Temperature Scaling(TS) UMAP Entropy plot looks quite similar to the Base CNN model(see Figure 13). The TS model essentially has the same architecture as the Base CNN model except for the final output layer(for details see Figure 5.4.4). So, it is essentially a calibrated version of the Base CNN model. The entropies of the misclassified points(red and green triangles) are relatively large, which is a good sign regarding the uncertainty quantification of misclassified points. Upon a very close look at the correctly classified HC points, it can be observed that the entropy markers for most of these samples are slightly smaller than in the Base CNN model. Therefore, Temperature Scaling has helped the model become less uncertain for the HC cases by adequately scaling the model outputs. Even after applying TS, the model is still relatively more uncertain for correctly classified HC cases than the correctly classified PD cases, although with relatively more moderate values.

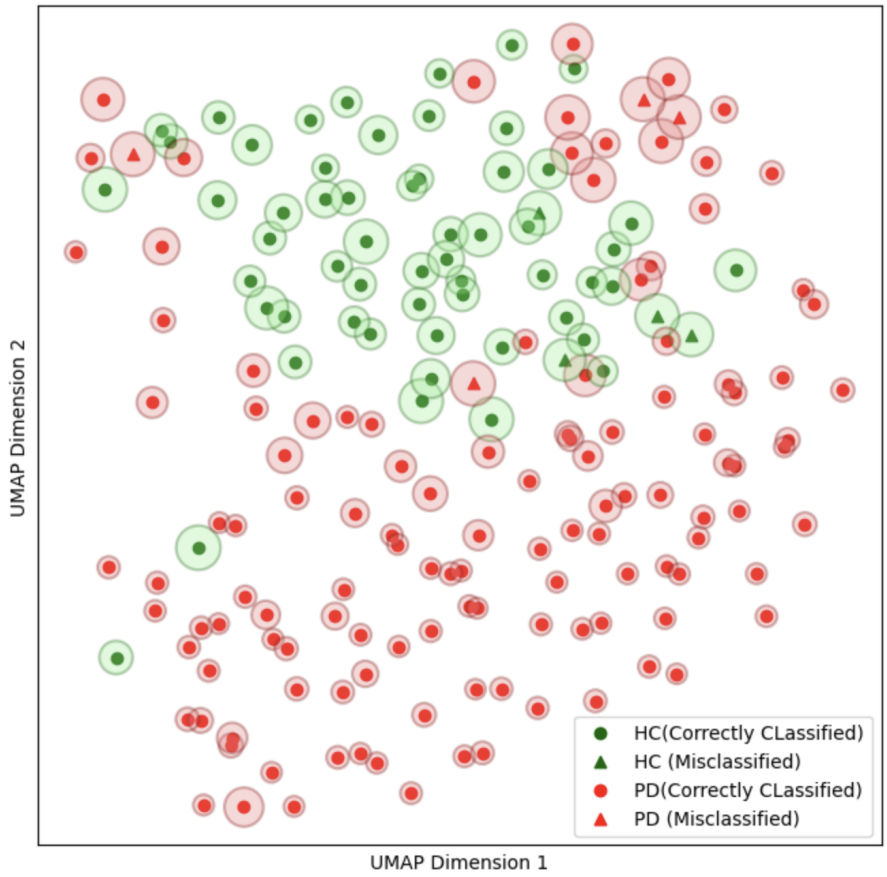


Figure 15: SNGP UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the Temperature Scaling model.

In the above plot(Figure 15), most of the points(for both HC and PD) have a visible entropy marker. Nonetheless, we can still see a difference in the size of the entropy markers between the correctly classified HC and PD cases, with the PD cases showing relatively more certainty. However, the PD cases in the above plot still have a larger entropy if compared to those of Base CNN and TS models. This shows a marginally lower confidence in correct PD predictions. The majority of the misclassified points have a relatively large entropy marker, which is a beneficial feature as it is atleast indicative of a good uncertainty quantification, despite a misclassification.

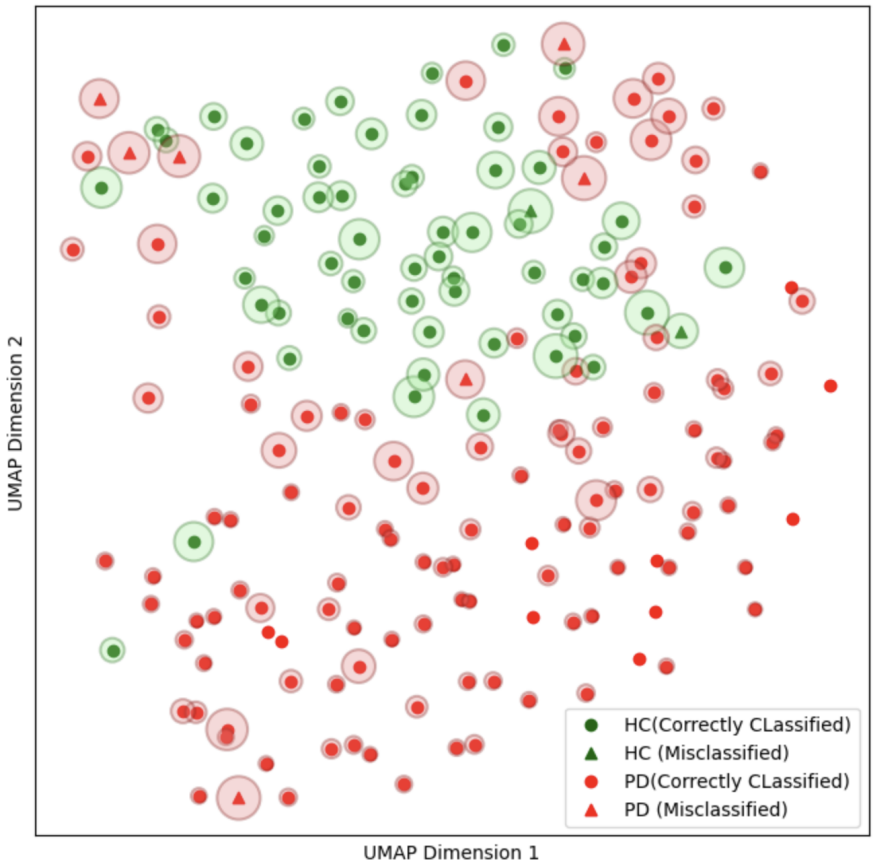


Figure 16: MC Dropout UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the Temperature Scaling model.

For the MC Dropout Model, the UMAP entropy plot shows a relatively large entropy marker for most of the misclassified points. A major chunk of the correctly classified PD cases are very certain due to the lack or relatively smaller size of the entropy markers around them. The correctly classified HC cases have a relatively larger entropy marker when compared to the PD cases. Therefore, the above MC Dropout UMAP plot also indicates a bias in the certainty of predictions towards the PD cases.

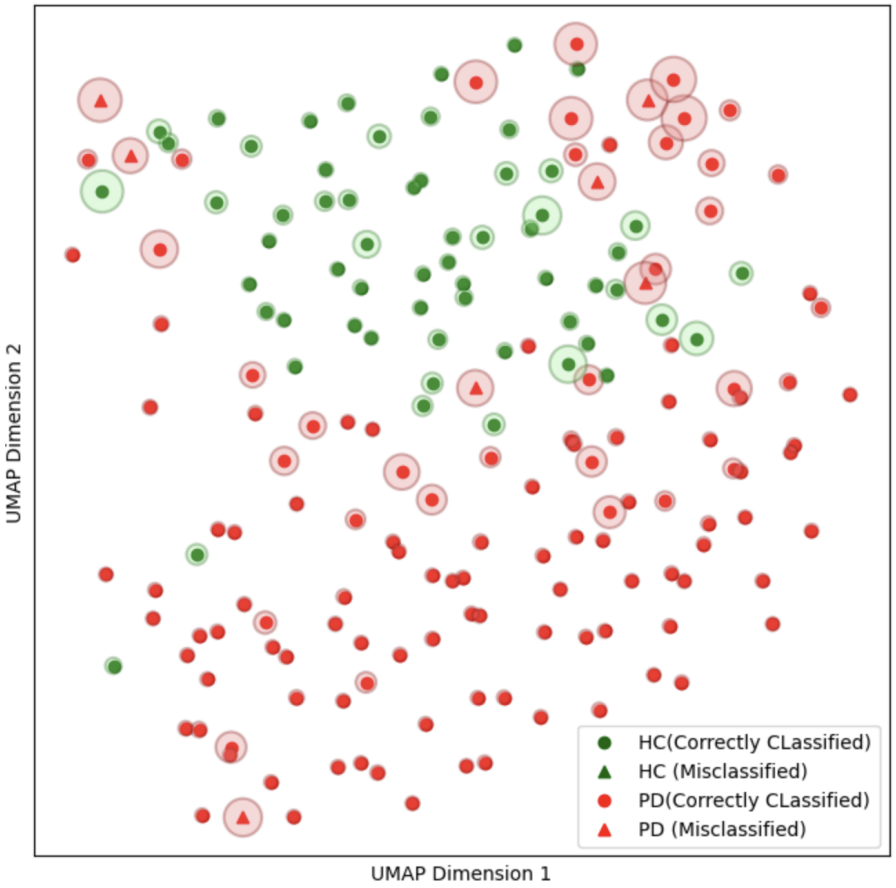


Figure 17: BN UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the Temperature Scaling model.

The BN model UMAP Entropy plot above displays some interesting results. There does not seem to be a major visible bias in certainty of predictions towards the PD class samples. From the plot, the model seems relatively certain for most of the correctly classified PD as well as most of the correctly classified HC cases. This could either be indicative of a very good model or a very over confident one. As for the misclassified points, they have a relatively larger entropy marker which indicates high uncertainty and thus indicates atleast a good uncertainty quantification for the misclassified points.

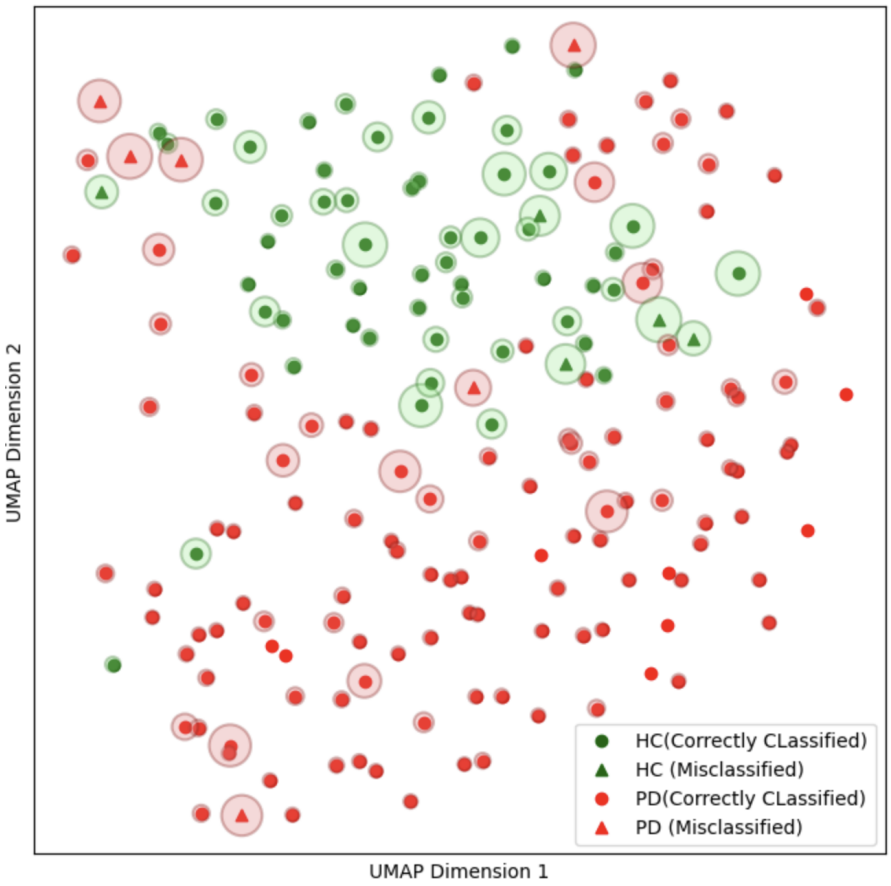


Figure 18: Deep Ensemble UMAP Entropy Plot

The plot shows the Entropy associated with each prediction made of the test images for the Temperature Scaling model.

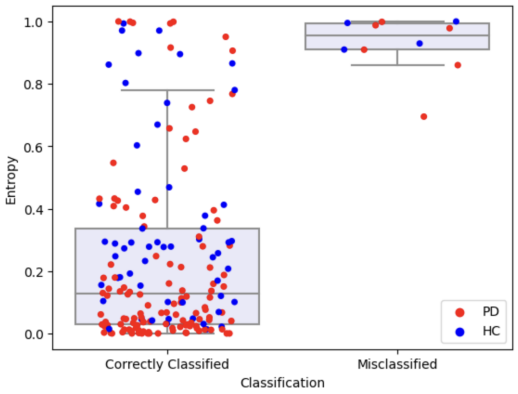
The UMAP Entropy plot for the Deep Ensemble(DE) model looks like a hybrid between that of the BN model(Figure 17) and the MC Dropout model(Figure 16). Most of the correctly classified PD cases are extremely certain due to the lack of large entropy markers around them. For the correctly classified HC cases, some points have a negligible entropy marker while some other points have a visible entropy marker. Overall, we can still call the model slightly biased towards the PC cases in terms of having high predictive certainty. As for the few misclassified samples, the model is able to show high uncertainty for most of them.

Overall, all the above models seem to be more certain about the classification of PD cases, although to varying degrees. The plots show how each model performs in a rough and qualitative manner. These plots, however, do not give a quantitative analysis of the uncertainty. They also do not provide hard evidence about the uncertainty quantification performance of the respective methods. In order to do that, it is important to see the uncertainty/entropy values for the respective predictions. This can be helpful in getting a more detailed idea of the uncertainty assessment of each model and actually determining which method performs better in uncertainty quantification. The Categorical Entropy Plot as explained in the next Figure 6.3, does a considerably better job at this quantitative

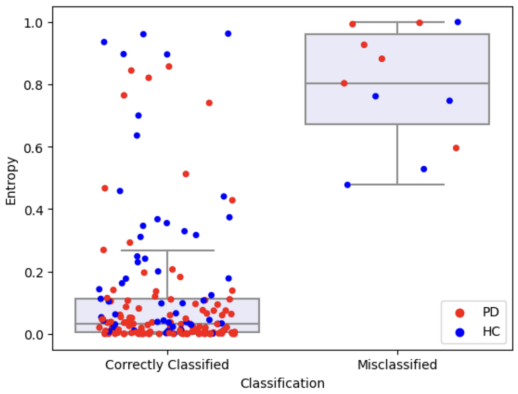
comparison.

6.3 Categorical Entropy Plots Analysis

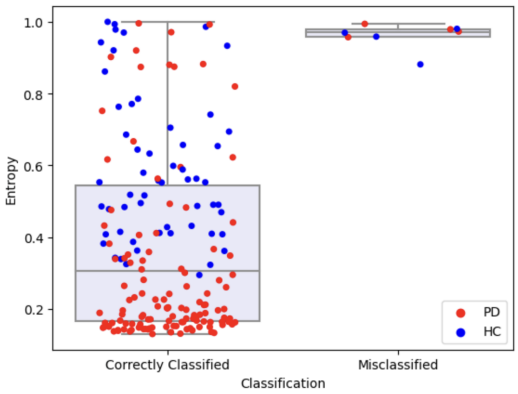
The Categorical Entropy plot gives a holistic view of how well each model quantifies uncertainty as a whole. To have a closer look at the overall uncertainty quantification, box plots and scatter plots were generated to see the difference in entropy values for the correctly classified and misclassified points. The scatter plot has been overlayed on top of the box plot to combine and represent the information of both the plots together. The X-axis has 2 categorised columns: correctly classified points and misclassified points, respectively. The Y-axis represents the entropy values from 0 to 1. For the box plot, there is no distinction between PD and HC points. It just shows the range of distribution between the correctly classified and the misclassified points based on their entropy values. The scatter plot not only shows the range of distribution, but also the class of each point. This gives more detailed information on how well uncertainty is quantified at a more granular level, such as for each class.



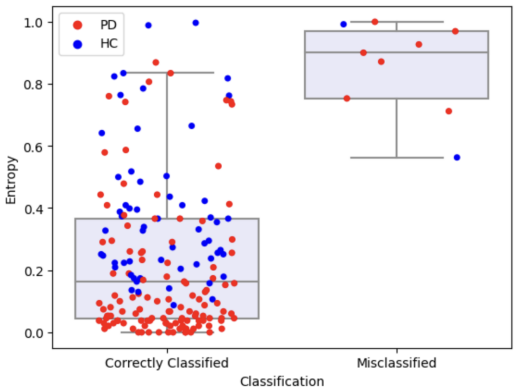
(a) Base CNN



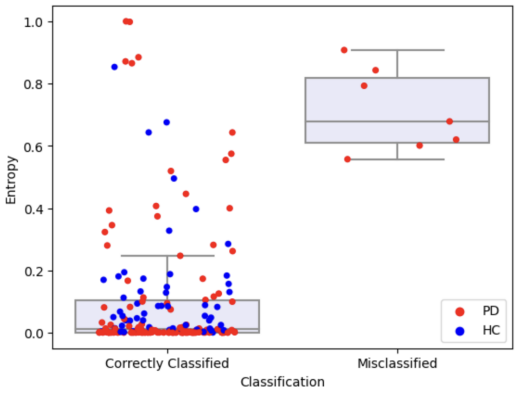
(b) Temperature Scaling



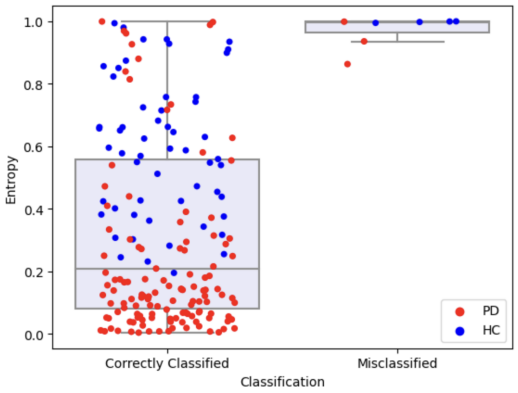
(c) SNGP



(d) MC Dropout



(e) BN (3rd DE)



(f) Deep Ensemble

Figure 19: Categorical entropy plots for the NN model methods

For the Categorical Entropy plot of the Base CNN, the distributions of the correctly classified and misclassified points can be observed. The box plot provides a quantitative insight into the entropy range where the highest percentage of points reside, encompassing both correctly classified and misclassified points. For the Base CNN box plot, first, for the correctly classified points, we can observe that around 75% of the points lie below the 0.35 entropy range. The rest 25% of the data are scattered between the range of 0.35

- 0.8. The rest of the data above 0.8 and above the box plot whiskers are considered as outliers. By looking at the scatter plot, it seems that the points for both the classes are very well distributed across all ranges for the correctly classified points. Coming to the box and scatter plot for the misclassified points for Base CNN, the majority of the data is concentrated above 0.85(rough), except for one outlier at the entropy value of around 0.7.

For the categorical plot of the Temperature Scaled model, first for the correctly classified, we see 75% of the data distribution below the entropy value of around 0.14. Rest of the points are scattered above this point till the maximum entropy value of 1. For the misclassified points, the majority of the points (around 75%) lie above the entropy value of 0.7. The scatter plot shows that a few points (roughly 25%) lie between 0.5 and 0.7.

For the correctly classified points in the SNGP categorical plot, the majority (around 75%) of them are between the entropy range from 0.1 to 0.55. The rest of the points are scattered above 0.55 till 1. Also, from the scatter plot it is noticeable that a majority of the chunk of PD cases are bundled at very low entropy values. Whereas, the HC cases are distributed more around relatively higher entropy value ranges as compared to the PD cases. For the misclassified points, all the points are above the 0.95 entropy range, with the exception of one HC case at a entropy value of around 0.90. It is treated as an outlier in terms of the total distribution. Overall, by the plot, the SNGP model seemed to have performed extremely well in quantifying the uncertainty in the misclassified points.

For the MC Dropout plot Figure 19d for the correctly classified column, around 75% of the points are below the entropy value of 0.38. The rest of the points lie between the entropy range of 0.38 and 0.84. The plot also displays a greater number of PD points in the extremely low certainty range and as the entropy increases, we see more HC cases instead. Therefore, it further solidifies the theory from the UMAP plots that the model in general is relatively more confident for PD cases as compared to HC cases. For the misclassified points, the majority of the points lie between 0.76 and 1. The rest of the points lie between the entropy range of 0.58 and 0.76. And from the scatter plot, it can be observed that the model fails to predict a greater number of PD points than the HC ones. This is even though the model is relatively more certain for most PD cases in general.

For the BN model, the categorical entropy plot for the correctly classified column shows roughly 75% points between the entropy range of 0 and 0.1. This indicates a very good performance as it shows that the model can predict most of the points correctly with close to absolute certainty. The rest of the points are very sparsely distributed across the entire entropy range. From the scatter points it is also evident that the model is certain for both HC and PD cases for the correctly classified column. Therefore, it does not have any confidence bias towards one of the classes. On looking at the plot for the misclassified points, it is observed that 75% of the points lie between the entropy range of 0.61 and 0.90. The rest of the points are in between the range of 0.57-0.61.

For the Deep Ensemble categorical entropy plot, the majority(75%) of the points in the correctly classified column lie between 0 and 0.58. And same as for the SNGP and MC Dropout model, the DE model categorical entropy plot has also predicted more PD points with more certainty as compared to the HC points. For the misclassified points, all the points, but one, lie in the range of 0.95 and 1.0. The one misclassified point has an entropy

of around 0.83 and is classified as an outlier in the data distribution.

6.4 Threshold-based uncertainty analysis

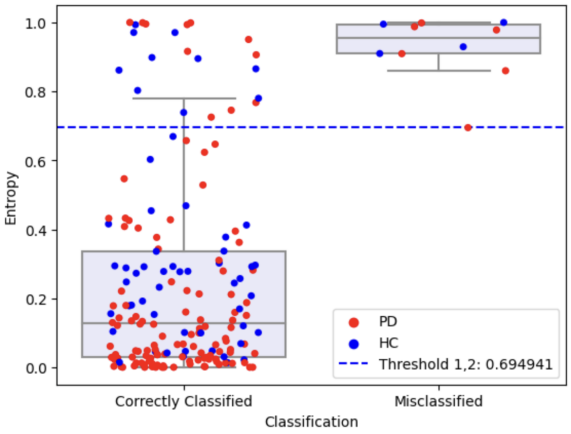
For classification problems, the best case scenario is to correctly classify all samples of the data. Although, it is not always possible to achieve 100% efficiency in real life applications. For our Parkinson's classification problem, the main goal in terms of medical ethics is to make sure that no patient with Parkinson's is wrongly classified as a healthy individual. This is because the effect of this wrong diagnosis could be potentially fatal. Based on the above scenarios, appropriate entropy thresholds could be applied to differentiate the truly uncertain predicted samples. With the aid of such thresholds, a medical professional would only have to review the cases above a specific entropy level. Assuming that the medical professional reviews only these uncertain points to classify them correctly, then that not only creates an accurate but also an efficient system. Therefore, in order to compare the different models in order to create such an accurate and efficient system, two types of thresholds have been applied to the entropy plots.

1. Threshold 1(denoted by a green dotted line): This threshold is set to the lowest entropy value out of all the misclassified points. Therefore, this threshold ensures that no sample is misclassified for the respective model predictions.

2. Threshold 2(denoted by a red dotted line): This threshold is set to the lowest entropy value out of all the misclassified points which belong to class PD(Parkinson's disease). Therefore, this threshold ensures that no Parkinson's patient is wrongly classified by the respective model predictions. This threshold does have the possibility of misclassifying an HC case as a PD case but never vice versa.

3. Threshold 1,2(denoted by a blue dotted line): This threshold will be visible for some of the entropy plots when *Threshold 1* and *Threshold 2* have the same entropy value. Therefore, in order to differentiate between the cases when both thresholds are the same, *Threshold 1,2* will come into play.

Using the above thresholds, the number of points above the threshold for each model will be analysed and compared to find the most efficient one. The model offering the least number of points above the thresholds will be considered as the most efficient model out of all six. This efficiency refers to the least amount of effort a medical professional might have to take while also ensuring complete accuracy. In some of the upcoming plots it will be observed that there is no green line(*Threshold 1*) visible, but only a red line(*Threshold 2*). For such plots, *Threshold 1* and *Threshold 2* are the same because the misclassified points with the lowest entropy also happen to belong to the PD class.



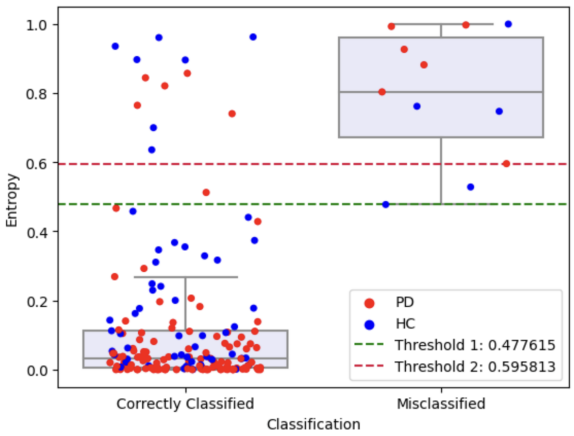
(a)

Category(Threshold 1,2)	HC	PD
Correctly Classified	10	11
Misclassified	4	5
Samples per class	14	16
Total samples to observe	**30**	

(b)

Figure 20: Threshold analysis for Base CNN:(a) Thresholds on entropy plot, (b) Count of samples above Threshold 2

For the Base CNN entropy plot, *Threshold 1* and *Threshold 2* have the same value of 0.694941 as the sample with the lowest entropy also happens to belong to the PD class. From the Figure 20b, there are around 10 correctly classified and HC cases and 11 correctly classified PD cases, which makes a total of 21 total correctly classified cases above the threshold. The table also shows around 14 HC and 16 PD samples above this threshold. Therefore, the number of samples above the threshold for each class is quite similar, which indicates a lack of major bias at higher entropy values. Under the assumption that the model makes the same predictions with the mentioned threshold(*Threshold 2*), then a medical professional would have to look at 30 brain scan images to check and re-classify them into their correct classes. Thus, this threshold would act as an indicator above which all the points are predicted with high uncertainty and therefore signify that the prediction for these points cannot be trusted.



(a)

Category(Threshold 1)	HC	PD
Correctly Classified	7	6
Misclassified	4	6
Samples per class	11	12
Total samples to observe	**23**	

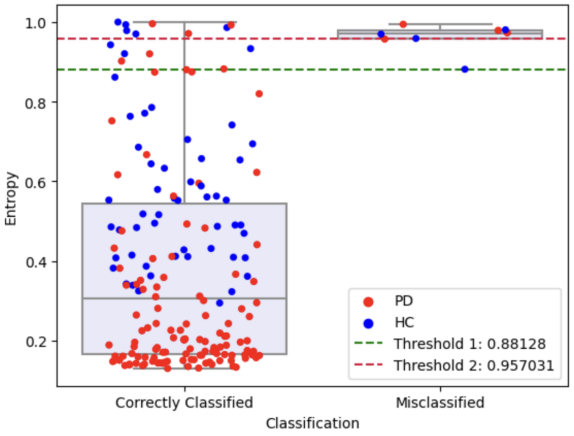
(b)

Category(Threshold 2)	HC	PD
Correctly Classified	7	5
Misclassified	3	5
Samples per class	10	10
Total samples to observe	**20**	

(c)

Figure 21: Threshold analysis for Temperature Scaling:(a) Thresholds on entropy plot, (b) Count of points above Threshold 1, (b) Count of samples above Threshold 2

The Temperature Scaling entropy plot has both *Threshold 1* and *Threshold 2* applied to it. The lowest entropy point in the misclassified samples belongs to the HC class with a value of 0.477615(Threshold 1), whereas Threshold 2 is 0.595813. For *Threshold 1*, the samples per class is almost the same, with 11 and 12 for HC and PD cases respectively. The total number of samples to observe for *Threshold 1* is 23. Therefore, around 23 brain scan images will need to be manually reviewed for reclassification to ensure 0 wrong diagnoses. As for *Threshold 2*, there are 20 images above the threshold that will need to be manually reviewed for reclassification to ensure 0 wrong diagnoses for specifically cases with Parkinson's disease.



(a)

Category(Threshold 1)	HC	PD
Correctly Classified	8	6
Misclassified	3	4
Samples per class	11	10
Total samples to observe	**21**	

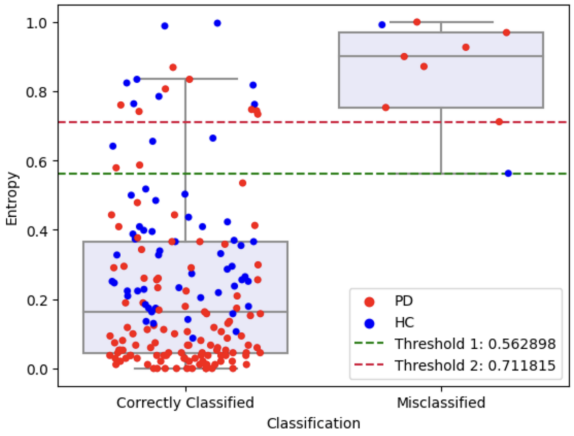
(b)

Category(Threshold 2)	HC	PD
Correctly Classified	5	3
Misclassified	3	3
Samples per class	8	6
Total samples to observe	**14**	

(c)

Figure 22: Threshold analysis for SNGP:(a) Thresholds on entropy plot, (b) Count of points above Threshold 1, (b) Count of samples above Threshold 2

The SNGP entropy plot also has two different values for *Threshold 1* and *Threshold 2*. From Figure 22b, we observe that for *Threshold 1*, there are 21 samples/images that are extremely uncertain and need to be given a second look and ensure their correct classification. And from Figure 22c, *Threshold 2* indicates only 14 images as having a sufficiently high entropy/uncertainty, such that their classification prediction needs to be reviewed again.



(a)

Category(Threshold 1)	HC	PD
Correctly Classified	11	10
Misclassified	1	7
Samples per class	12	17
Total samples to observe	**29**	

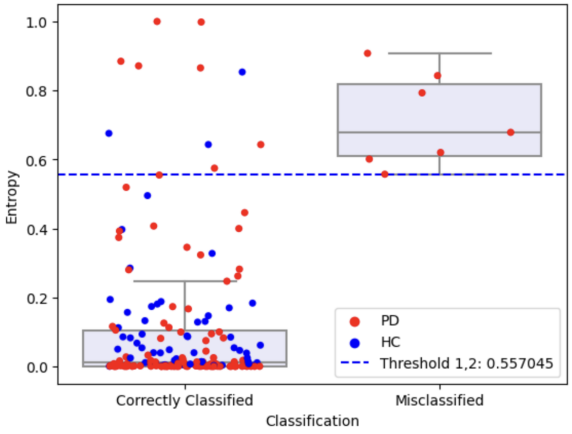
(b)

Category(Threshold 2)	HC	PD
Correctly Classified	8	8
Misclassified	1	6
Samples per class	9	14
Total samples to observe	**23**	

(c)

Figure 23: Threshold analysis for Monte Carlo Dropout:(a) Thresholds on entropy plot, (b) Count of points above Threshold 1, (b) Count of samples above Threshold 2

The MC Dropout model has 2 different values for *Threshold 1* and *Threshold 2* as well. From Figure 23b *Threshold 1* dictates 29 images for reviewing due to sufficiently high entropy values. It also shows just 1 HC case misclassified point above the threshold, whereas there are 7 misclassified PD cases above the threshold. MC Dropout model prediction is therefore, more efficient in classifying and quantifying HC cases as compared to the PD cases for *Threshold 1*. For Threshold 2, as is evident from Figure 23c, there are 23 brain scan images to verify the correct diagnosis. For *Threshold 2*, just like *Threshold 1*, there is a strong bias in favor of the prediction of HC cases as compared to the prediction of PD cases. This is theorised based on the HC class having only 1 misclassified point as compared to the PD class having 6 misclassified points above Threshold 2.



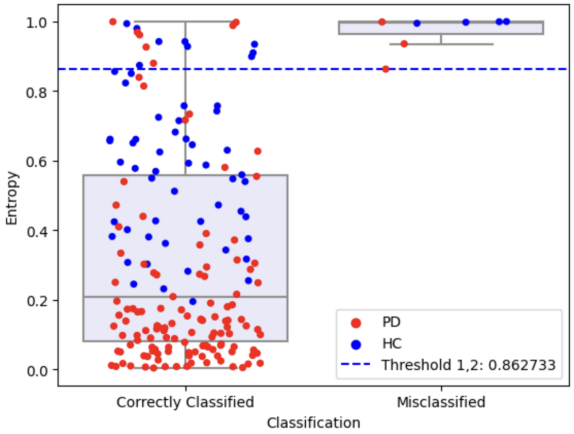
(a)

Category(Threshold 1,2)	HC	PD
Correctly Classified	3	7
Misclassified	0	6
Samples per class	3	13
Total samples to observe	**16**	

(b)

Figure 24: Threshold analysis for BN model:(a) Thresholds on entropy plot, (b) Count of samples above Threshold 2

The BN model has the same entropy value for both *Threshold 1* and *Threshold 2* because the lowest entropy sample out of the misclassified samples also happened to be a PD case. This common threshold(called as *Threshold 1,2*) here is displayed by a blue dotted line. Based on *Threshold 1,2*, as can be seen from Figure 24b, there are a total of 16 samples that are uncertain and need to be reviewed again. Out of these 16, only 3 images belong to class HC and the rest 13 belong to class PD. Therefore, the BN model predictions with *Threshold 1,2* does better in uncertainty quantification and prediction for HC cases as compared to PD cases. Even if we look at the count of misclassified samples above the threshold, there are 0 samples for class HC that have been misclassified. That shows a perfect classification for HC cases with the applied threshold.



(a)

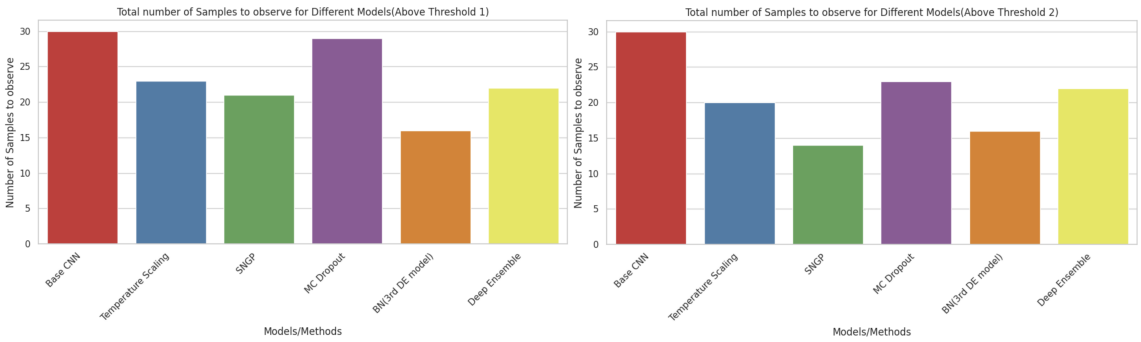
Category(Threshold 1,2)	HC	PD
Correctly Classified	9	7
Misclassified	4	2
Samples per class	13	9
Total samples to observe	**22**	

(b)

Figure 25: Threshold analysis for Deep Ensemble:(a) Thresholds on entropy plot, (b) Count of samples above Threshold 2

The Deep Ensemble Entropy plot, just like the BN model, has a common threshold (*Threshold 1,2*) value of around 0.86 for both *Threshold 1* and *Threshold 2* applied to it(see Figure 25a). Based on *Threshold 1,2*, Figure 25b shows that around 22 images are above this threshold and need to be reviewed for re-diagnosis. Thus, with *Threshold 1,2*, if the 22 cases are correctly re-diagnosed, then these predictions with the mentioned threshold would ensure a 100% diagnosis for all the potential test cases.

The criteria for comparing the different methods for the entropy-based threshold analysis, the efficiency in application of the results need to be observed. This efficiency here is given by the least amount of effort a medical professional might have to put in to re-diagnose uncertain cases. Less is the effort, the more will be the efficiency. And this minimum effort depends directly on the number of samples to be observed. Therefore, the less is the number of samples, the easier it is for the medical professional to re-diagnose. Based on this idea, to compare the results of the Entropy based threshold analysis, the total number of samples to be observed will be compared among the different models. This comparison of the number of samples will be done for both the threshold scenarios.



(a) Number of Samples for Threshold 1

(b) Number of Samples for Threshold 2

From the above histogram plot Figure 26a, it is evident that for *Threshold 1*, the BN(3rd DE) model has the least number of samples to be observed. Therefore, the BN model predictions are the most efficient across all the 6 models when *Threshold 1* is applied. Also, for the BN model, as we observed in Figure 24, *Threshold 1* and *Threshold 2* are the same. Thus, it ensures that any potentially misclassified point(regardless of whether it is a HC or a PD case) is marked as highly uncertain and needs to be reviewed and reclassified to guarantee 100% accuracy.

For *Threshold 2*, the Figure 26b shows the SNGP model to have the least number of samples to observe above the threshold. Thus, if we want to ensure 100% accuracy in a correct diagnosis for all PD cases, the SNGP model predictions provide the least number of samples to be reviewed, consequently cutting down the workload. This makes it more efficient than all the other model methods.

Depending on the application and need, either the BN model predictions or the SNGP model predictions could potentially quantify predictive uncertainty the best. This claim is solely based on the current state of results and is subject to change due to the stochastic nature of deep learning models and techniques.

7 Conclusions

Upon testing different Neural Network models and techniques, we have seen how each of them quantify predictive uncertainty. From the UMAP projection Entropy plots(Figure 6.2), it was gleaned that all of the models, except the BN model showed significant bias for the uncertainty quantification towards the PD class. All the models showed high certainty in the prediction of the correctly classified PD class. Using this, one could speculate at the possibility of the existence of certain features in the Parkinson's disease scans that are quite evident and which might cause the models to predict it with high uncertainty. Consequently, another interpretation could be that the models were not capable enough to capture the features of majority of the HC cases to predict it with high certainty.

The BN model predictions performed the best in terms of almost all criteria. For the classification, it had the least amount of misclassifications along with the Deep Ensemble model, with 7 misclassified points for each. The UMAP projection entropy plots and the categorical entropy plots displayed how well the BN model managed to cluster the correctly classified and misclassified points into relatively distinct entropy ranges. For threshold analysis, specifically for *Threshold 1*, the BN model gave the most efficient model because it offered the least number of points to be observed for reclassification. Therefore, for *Threshold 1*, the BN model correctly classified the most number of samples and with solid certainty. The BN model, as discussed in Figure 5.4.3, is the only model which has batch normalisation layers in the architecture. It can be speculated that having strategically placed batch normalisation layers in a model can significantly enhance the classification as well as predictive uncertainty quantification for our problem statement. The BN model even outperformed more elaborate methods like Deep Ensemble, Monte Carlo Dropout and Temperature Scaling.

The SNGP model classification predictions were second best to the Deep Ensemble and the BN model, but only by 1 more misclassified sample. For the uncertainty quantification, SNGP model predictions were able have a very small range of high entropy values for the misclassified points. Therefore, the SNGP model did a remarkable job of quantifying uncertainty for the misclassified points. With such clustering and separation between the correctly classified and misclassified samples, it becomes easier to set a threshold to differentiate between high uncertainty and low uncertainty. On setting *Threshold 2*, the SNGP model predictions only had 14 samples in the high uncertainty zone which needed to be observed for reassessment. This was the lowest number of samples to observe for *Threshold 2* as compared to any other model. Therefore, based on the results, SNGP model with *Threshold 2* would be the most efficient model to ensure that none of the PD cases would be misdiagnosed. As has been discussed in Figure 4.5, SNGP introduces distance awareness to the predictions of a neural network model which helps in calibrating the predictive uncertainty for each prediction. This property of SNGP worked really well using the Parkinson's image data for Parkinson's disease detection as is evident from the results.

This whole study is a step towards making an uncertainty-aware automated system that can detect Parkinson's disease with complete accuracy so that no patient is ever misdiagnosed. By introducing thresholds to the entropy values, we make the models more aware and vocal about the information they display. Apart from the class predictions, an entropy threshold

adds a new dimension to the predictions by displaying the degree of faith or confidence in that prediction. This is so that a user is aware of the risks associated with trusting the predictions. It provides a scope for reclassification in order to prevent a misdiagnosis. Specifically for crucial medical applications such as detecting Parkinson's disease, it is essential to ensure an accurate diagnosis. The studied methods and their results showcase the ability of these different neural network models and techniques that could eventually help in making such a perfect uncertainty-aware model for detecting early Parkinson's disease.

8 Future Work

In the future work of this thesis, a promising avenue for advancing uncertainty quantification involves the integration of multiple models. By combining complementary models that capture different aspects of uncertainty, we can potentially enhance the accuracy and reliability of uncertainty estimates. The objective is to leverage the strengths of various models mentioned in this study to create a more comprehensive framework for uncertainty quantification. This integration could result in a better representation of predictive uncertainties, leading to more informed decision-making in fields where uncertainty assessment is critical.

Another aspect of future work revolves around the implementation of a generalized entropy threshold tailored for image data. By establishing a generalized entropy threshold specific to images, we aim to identify regions of interest within images that exhibit significant uncertainty. This threshold could help in identifying anomalous or critical areas within images, which could be especially valuable in medical imaging, security, and anomaly detection applications.

A very crucial and obvious direction from this thesis would involve delving into the analysis of brain scan images with high entropy values. By investigating the underlying features that contribute to high uncertainty in brain scan images, one could potentially uncover patterns or characteristics associated with neurological disorders or abnormalities. This analysis could provide insights into the causes of uncertainty and contribute to a deeper understanding of the conditions that lead to uncertain predictions in medical imaging. Such findings could have implications for refining imaging techniques and improving diagnostic accuracy.

A pivotal and a more direct step involves the practical implementation of an uncertainty-aware model in a real-world application. By integrating the insights gained from combining models and exploring entropy thresholds, one could build a practically usable uncertainty-aware model that could be used on real data where the true labels are unknown. This model could be applied to various domains, such as autonomous systems, medical diagnostics, or financial risk assessment. The implementation of a reliable uncertainty-aware model as a real application underscores the practical significance of our research and its potential to positively impact decision-making processes in complex scenarios.

In summary, this research aimed to contribute to a more nuanced understanding of uncertainty and its implications in a specific domain. The outcomes of these endeavors could pave the way for improved models, better-informed decision-making, and enhanced reliability in predictions.

9 Acknowledgements

I would like to express my deepest gratitude to all the people who supported me directly and indirectly throughout the process of implementing and writing my Master thesis. I would like to start by whole-heartily thanking my primary supervisor, Dr. Markus Wenzel who gave me this opportunity to work on such an engrossing and relevant topic. He was instrumental in guiding and helping me to give a proper direction to my thesis to completion. I would also like to thank Kai Gießler who always had intriguing and extremely useful insights for all my queries. I am thankful for my fellow classmate, Anand Gajaria with whom I have had endless discussions on various topics regarding my thesis. I would like to also extend my gratitude to Fraunhofer MEVIS as a whole, which provided me with a nurturing and knowledge-rich environment where I could grow and develop my understanding further. I received valuable feedback on my study from multiple people at MEVIS for which I would like to forward my immense appreciation and respect.

Last but not the least, I am indebted to my parents and my elder sister who have always been a pillar of support throughout my entire life, let alone the duration of this thesis. Overall, all the people mentioned above have been absolutely crucial for me being able to efficiently implement and write my thesis, and for that I will always be grateful to them. It was a long journey during which I gathered invaluable learning and unwavering support.

References

[1] Fawaz F Alqahtani. "SPECT/CT and PET/CT, related radiopharmaceuticals, and areas of application and comparison". In: *Saudi Pharmaceutical Journal* (2022).

[2] apda. "What role does DaTscan have in the diagnosis of Parkinson's disease?" In: (2019). URL: https://www.apdaparkinson.org/article/what-is-a-datscan-and-should-i-get-one/.

[3] Kurt Cutajar et al. "Random feature expansions for deep Gaussian processes". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 884–893.

[4] Yarin Gal and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059.

[5] Neha Gianchandani et al. "Rapid COVID-19 diagnosis using ensemble deep transfer learning models from chest radiographic images". In: *Journal of ambient intelligence and humanized computing* (2020), pp. 1–13.

[6] Florin Gogianu et al. "Spectral normalisation for deep reinforcement learning: an optimisation perspective". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 3734–3744.

[7] Robert A Hauser and Donald G Grosset. "[123I] FP-CIT (DaTscan) SPECT brain imaging in patients with suspected parkinsonian syndromes". In: *Journal of Neuroimaging* 22.3 (2012), pp. 225–230.

[8] imerit. "A Comprehensive Introduction to Uncertainty in Machine Learning". In: (2022). URL: https://imerit.net/blog/a-comprehensive-introduction-to-uncertainty-in-machine-learning-all-una/#:~:text=Uncertainty%20refers%20to%20the%20lack,and%20how%20to%20reduce%20it..

[9] Ankit Kurmi et al. "An Ensemble of CNN Models for Parkinson's Disease Detection Using DaTscan Images". In: *Diagnostics* 12.5 (2022), p. 1173.

[10] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Advances in neural information processing systems* 30 (2017).

[11] Jeremiah Liu et al. "Simple and principled uncertainty estimation with deterministic deep learning via distance awareness". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 7498–7512.

[12] Mordechai Lorberboym et al. "[123I]-FP/CIT SPECT imaging for distinguishing drug-induced parkinsonism from Parkinson's disease". In: *Movement disorders: official journal of the Movement Disorder Society* 21.4 (2006), pp. 510–514.

[13] Kenneth Marek et al. "The Parkinson Progression Marker Initiative (PPMI)". In: *Progress in Neurobiology* 95.4 (2011). Biological Markers for Neurodegenerative Diseases, pp. 629–635. ISSN: 0301-0082. DOI: https://doi.org/10.1016/j.pneurobio.2011.09.005. URL: https://www.sciencedirect.com/science/article/pii/S0301008211001651.

[14] Daily Milanés-Hermosilla et al. "Monte carlo dropout for uncertainty estimation and motor imagery classification". In: *Sensors* 21.21 (2021), p. 7241.

[15] Azadeh Sadat Mozafari et al. “Attended temperature scaling: a practical approach for calibrating deep neural networks”. In: *arXiv preprint arXiv:1810.11586* (2018).

[16] Mahmudur GM Rahman et al. “Count-based method for specific binding ratio calculation in [I-123] FP-CIT SPECT analysis”. In: *Annals of Nuclear Medicine* 33 (2019), pp. 14–21.

[17] scaler. “Smoothing in Image Processing”. In: (2023). URL: https://www.scaler.com/topics/smoothing-in-image-processing/.

[18] Tobias Sterbak. “Model uncertainty in deep learning with Monte Carlo dropout in keras”. In: (2022-06-24). URL: https://www.depends-on-the-definition.com/model-uncertainty-in-deep-learning-with-monte-carlo-dropout/.

[19] TensorFlow. “Uncertainty-aware Deep Learning with SNGP”. In: (2023-07-27). URL: https://www.tensorflow.org/tutorials/understanding/sngp#about_sngp.

[20] Lazaros C Triarhou. “Dopamine and Parkinson’s disease”. In: *Madame Curie Bioscience Database [Internet]*. Landes Bioscience, 2013.

[21] Markus Wenzel et al. “Automatic classification of dopamine transporter SPECT: deep convolutional neural networks can be trained to be robust with respect to variable image characteristics”. In: *European journal of nuclear medicine and molecular imaging* 46 (2019), pp. 2800–2811.

[22] Wikipedia. “Neuroimaging Informatics Technology Initiative”. In: (2003). URL: https://en.wikipedia.org/wiki/Neuroimaging_Informatics_Technology_Initiative#:~:text=The%20Neuroimaging%20Informatics%20Technology%20Initiative,gz%20%2C%20.hdr%20%2F%20.img.

[23] Srikanth Yandrapalli and Yana Puckett. “SPECT imaging”. In: (2020).

[24] Yu Zhang et al. “A New Total Uncertainty Measure from A Perspective of Maximum Entropy Requirement”. In: *Entropy* 23.8 (2021), p. 1061.