UNIVERSITY OF CALIFORNIA
RIVERSIDE

Using Random Forest to Classify Raman Spectra of Brain Tissues

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Mechanical Engineering

by

Weiyi Zhang

December 2022

Thesis Committee:
Dr. Chen Li, Chairperson
Dr. Jun Sheng
Dr. Peter Greaney

The Thesis of Weiyi Zhang is approved:

_____

_____

_____

Committee Chairperson

University of California, Riverside

Acknowledgements

I would like to express my deep and sincere gratitude to my advisor, Prof. Chen Li, for his professional and kind guidance on my studies and research, giving me this precious opportunity to participate in such interesting research and experience the joy of accomplishment that doing research brings to me. I would like to thank the members of my defense committee – Prof. Chen Li, Prof. Jun Sheng and Prof. Peter Greaney, for their time and advice on the thesis. I am grateful to Qingan Cai for teaching me how to use the equipment for Raman spectroscopy, to Chau Minh Giang for measuring the Raman spectra, and to everyone who gives me valuable advice on writing.

I would also like to thank my family for their support in all aspects. To my husband, Yongda: In the vastness of space and the immensity of time, it is my joy to share a planet and an epoch with you.

Table of Contents

List of Figures

List of Tables

1. Introduction

Tissue identification plays an important role in a variety of applications. For example, in many forensic cases, the identification of biological materials found at crime scenes can be very informative [1]. In the biomedical field, the identification of abnormal tissues in patients allows clinicians to analyze diseases and provide appropriate treatments [2] [3]. In particular, tumor identification is essential for early detection and prevention of tumor deterioration and expansion. Glioma is one of the most common and life-threatening brain tumors and can occur in a variety of brain cells [4]. To predict the evolution of glioma and its response to different therapies, it is crucial to know its type and grade [5]. Since the symptoms of most types of gliomas are similar, tumor tissue classification becomes necessary.

When patients report symptoms, several imaging-based diagnostic approaches are available for the preliminary diagnosis of brain tumors, such as computed tomography (CT) and magnetic resonance imaging (MRI). However, it is not always possible to identify brain tumors from medical images, and the process is usually time-consuming and requires a skilled radiologist [6]. Recently, image processing and machine learning algorithms have been developed to aid this process, such as random forest for MRI [7] and support vector machine (SVM) for optical coherence tomography (OCT) [8]. However, these techniques are still in the early development stage, and the current paradigm involves needle biopsy after imaging-based diagnosis for the definitive diagnosis of brain tumors. Although needle biopsy has been widely used in the diagnosis of a variety of medical conditions, it faces unique challenges in glioma diagnosis due to intratumor heterogeneity. Due to the finite resolution of medical images and the error in needle placement, tissue samples extracted by a biopsy needle could misrepresent the true condition of the glioma and misguide subsequent treatment.

To improve the diagnosis of glioma, our objective is ultimately to combine spectroscopy with needle biopsy to guide the selection of the tissue biopsy location. Among spectroscopic techniques, Raman spectroscopy is one of the most widely used to analyze materials [9], [10], [11] and has been applied to detect and classify several types of tumors in open procedures [12], [13], [14]. The peaks in Raman spectra indicate the feature of the measured material, allowing for the identification of different materials in a more quantitative way compared to the images [15]. However, manual identification of the spectra requires training and false identification can be common. Therefore, it is necessary to develop a reliable method to analyze spectra autonomously and to improve the efficiency and accuracy of spectral identification.

In the medical field, the combination of Raman spectroscopy and machine learning algorithms is gaining popularity. For example, four machine learning methods (k-Nearest Neighbors algorithm, Decision Tree, Support Vector Machine, and Probabilistic Neural Network) combined with Raman spectroscopy were used to identify cerebral ischemia and cerebral infarction [16]. As reported in [17], partial least squares regression (PLS-DA) was used with Raman spectroscopy to identify the risk of Alzheimer's disease. In addition, the combination of artificial neural network (ANN) and Raman spectroscopy allows tracking of human neural stem cells [18].

This work explores the random forest as a robust tool for the classification of brain tissue. As a supervised machine learning algorithm, the random forest is widely used for classification and regression problems [19]. A prominent benefit of using random forest is that it limits overfitting during training to some extent, as random forest is based on the aggregation of several decision trees [20] using bagging, which is more robust against overfitting[21]. Another benefit is that it allows one to train the model based on small data sets. Although random forest has been used to diagnose several types of cancer, such as lung cancer [22] and laryngeal carcinoma [23], its efficacy in identifying various types of brain tissue and tumor tissue has not yet been explored.

The aim of this work is to validate the feasibility of combing the Raman spectroscopy and machine learning algorithm for a prediction task using tissues. Therefore, several critical questions were raised: i) How to decide a proper prediction objective? ii) How to optimize Raman spectroscopy to achieve sufficient data quality? iii) What is the appropriate procedure for data preprocessing? iv) How to build and train a reasonable model? v) What is an effective way to evaluate model performance? In this thesis, the mock spectra were used to decide the prediction task and machine learning algorithm used for predicting. The choice of configurations and the pipeline for training were elaborated. The training results were presented and discussed.

2. Materials and Raman Spectroscopy

2.1 Mock Spectra Acquisition

Mock spectra were used for the choice of configurations before acquiring Raman spectra of tissues. We synthesized 100,000 spectra with 1000 pixels. Each spectrum contains up to 10 peaks synthesized with Gaussian, adding a noise level of up to 0.2 of the maximum intensity to each pixel. Then the generated data were divided into three groups randomly, as the training set, validation set and test set, respectively. Figure 1 is an example of the mock spectrum. The distribution of the number of peaks generated in each spectrum was shown as Figure 2 to verify whether the mock spectra are as expected.

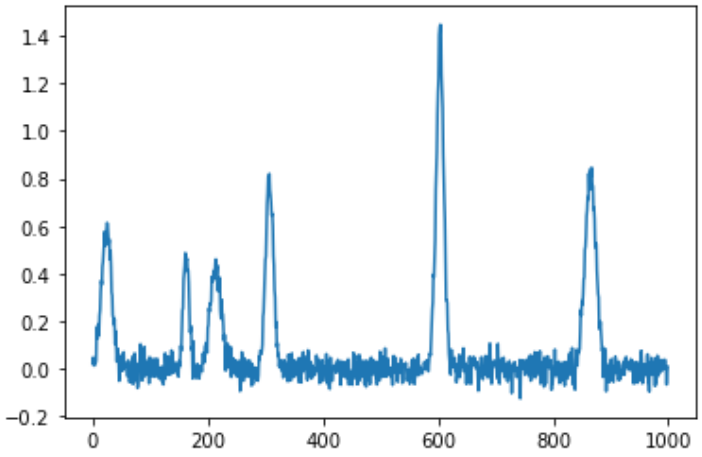

Figure 1 An example of the mock spectrum.

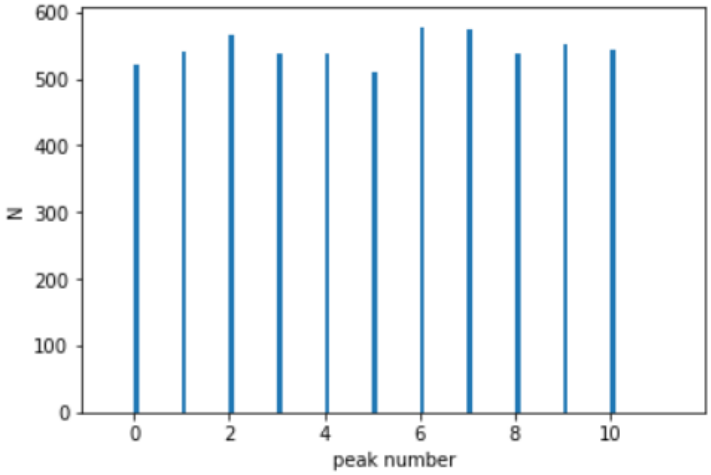

Figure 2 the distribution of the number of peaks generated in each spectrum.

2.2 Raman Spectra Acquisition

Three types of biopsies were prepared, including ectocinerea (grey matter), alba (white matter), and blood vessels. The freshly dissected goat brain (Figure 3) was obtained from the slaughterhouse. As shown in, the brain was kept at -18°C for 24 hours, and cut to a thickness of about 2 mm (Figure 3). The samples were fixed on glass slide using super glue and acclimated to room temperature. Since the blood vessels contained in sliced tissue are very small, we developed

our method by slicing the goat aorta because the blood vessels contained in sliced tissue are very small.

Raman spectra of the tissue samples were acquired under 532 nm laser excitation using a custom-built open bench Raman spectrometer. The tissue slide was mounted on a stage which can be adjusted precisely in three directions (Figure 3). A long working distance microscope objective was used in reflection geometry for focusing and light collection, and a notch filter was used for removing Rayleigh scattering. We manually adjusted the stage to obtain Raman spectrum on individual points. To ensure light focus on the tissue surface, the microscope objection was adjusted for each point. The ground truth of classification was manually observed from the microscope and recorded. The optimal equipment setting was determined by applying different setting combinations on a relatively small data quantity, including different wavenumber ranges and exposure time.

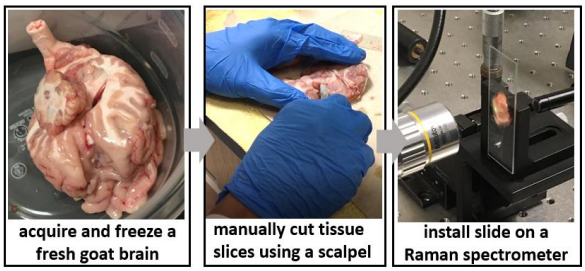

Figure 3 Preparation of tissue samples and installation on an open-bench Raman spectrometer.

2.3 Optimization of Data Pre-Processing and Choice of Configurations

2.3.1 Feature Number Reduction

We took Raman spectra from 21.43 to 3744.25 cm^{-1} (wavenumber) using white matter and grey matter mentioned above. Due to the characteristic of the instrument, four measurements with different wavenumber ranges were needed, which are 21.43 to 1324.25 cm^{-1}, 886.72 to 2066.01

cm^{-1}, 1856.37 to 2903.52 cm^{-1}, and 2822.03 to 3744.25 cm^{-1}. As shown in Figure 4, there is a wavenumber range overlap and an intensity gap between the two measurements in the raw spectrum.

The data pre-processing process of the four-part spectrum includes stitching the four parts, averaging the overlap part, removing the sporadic signal, rebinning the wavenumber to 1 cm^{-1} interval, cutting off the edge (i.e., wavenumber < 400 cm^{-1}), and removing baseline. The latter three parts of the spectrum were shifted to the same intensity range to the first part of the spectrum. Then the intensity of the overlapped range was calculated using the mean of the two intensities. Generally, the sporadic signal is from a single pixel with high counts on the detector, and inevitable from background radiation. To remove the sporadic signal, the standard deviation of the intensity of each pixel was calculated using the 20 neighbouring pixels. Next, a threshold, which is the calculated standard deviation, was set for the detection of sporadic signals. The intensity of a pixel will be replaced by the average of its two adjacent pixels if it is greater than the threshold. Different multiples of the standard deviation were set as the threshold to test the reasonableness of threshold, but we find no significant differences in performance when using multiples of the standard deviation, i.e., two to five times of the standard deviation. This algorithm detected and removed most of the sporadic signals with negligible effects. Then, all spectral wavenumbers were rebinned into 1 cm^{-1} intervals using a weighted rebinning method, which is SpectRes, given that the original wavenumber intervals are not uniform [24].

The classification might be misled by the background of Raman spectra. Figure 5 is the comparison of the Raman spectrum and the calculated baseline. The black spectrum is the pre-processed spectrum except for removing baseline. The red curve is the baseline that will be subtracted. It was estimated using improved asymmetric least squares (IAsLS) [25], which is based on polynomial fitting. The optimal smoothing parameter λ and the penalizing weight factor p were 10^5 and 0.02, respectively.

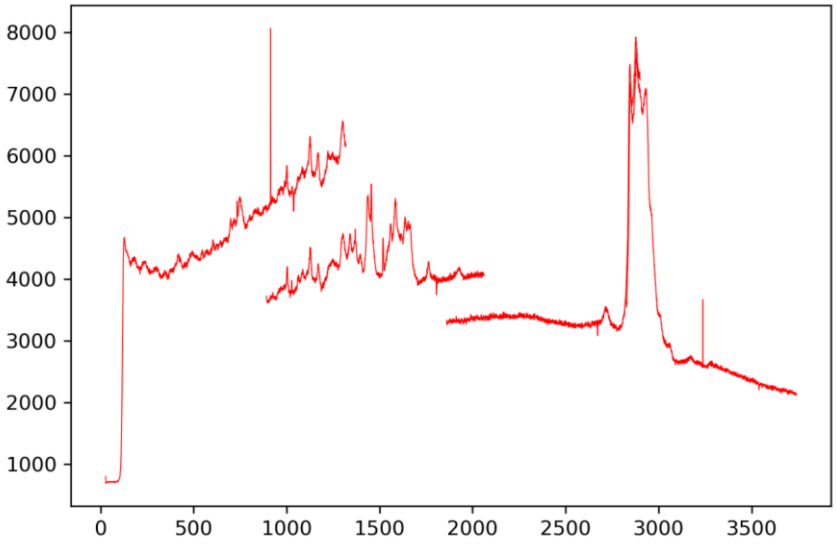

Figure 4 an example of the raw Raman spectrum from the white matter with a wavenumber range of 21.43 to 3744.25.

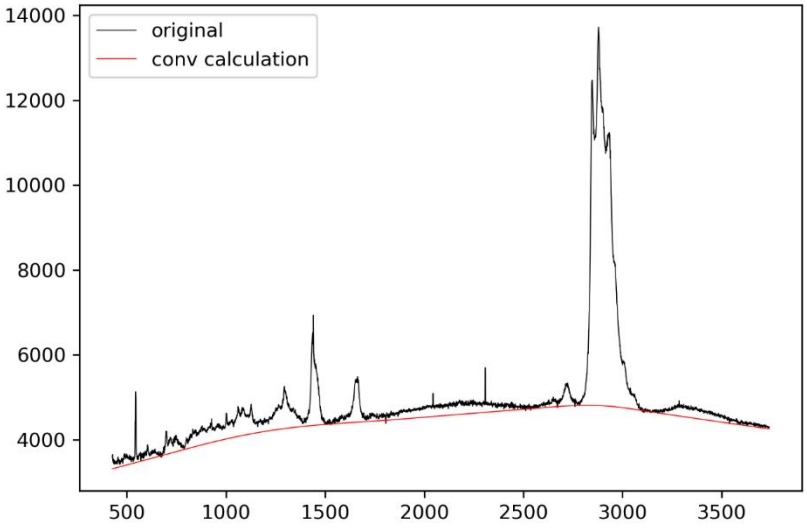

Figure 5 an example of the pre-processed Raman spectrum except for removing baseline.

Reducing the number of features is necessary because irrelevant features may negatively affect the training and increase computation cost [26]. To determine if the desired training results can be achieved using a relatively small wavenumber range, the four parts of the spectrum were trained separately using Random Forest Classifier. Table 1 compares the training score and the test score

using the four parts of the spectrum separately. From the table, we can see that using the second part of the spectrum and the fourth part of the spectrum for training get a better test score. However, the fourth part of the spectrum contains only one peak with high intensity, while the second part of the spectrum contains multiple peaks, so the second part of the spectrum was used for a large amount of data acquisition to save the measuring time.

Table 1 comparison of training score and the test score using four parts of the spectrum

	Training score	Test score
Part 1	0.97	0.82
Part 2	0.97	0.99
Part 3	0.94	0.72
Part 4	0.99	0.96

Feature importance was used as another way to determine the narrowed wavenumber range. It is calculated as the mean and standard deviation of the accumulation of the Gini impurity decrease in each tree [27]. Figure 6 shows a semi-log plot of the feature importance of Raman spectra feature by the pixel index.

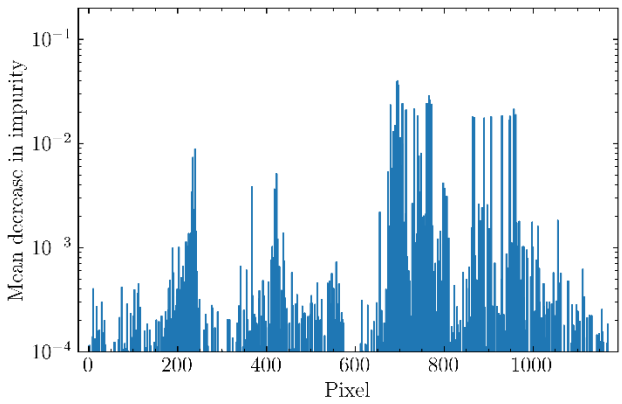

Figure 6 The semi-log plot of feature importance. The mean decrease in impurity on the y-axis denotes the Gini impurity.

2.3.2 Experiment Parameters Optimization

The exposure time when measuring the Raman spectrum affects the data quality and the model performance. Figure 7 shows the test scores of grey matter and white matter using different

exposure times. The prediction accuracy of the testing set decreases with the shortening of exposure time. Furthermore, the scores are reasonably high (above 0.9) when the exposure time is higher than 60-s. When the exposure time is shorter than 10-s, the model performs poorly. An exposure time of 120-s with a wavenumber range of 892 - 2062 cm^{-1} was used for later data collection. The 120-s exposure time guarantees the data quality, and the reduced feature number shortens the experimental measurement time of Raman spectra.

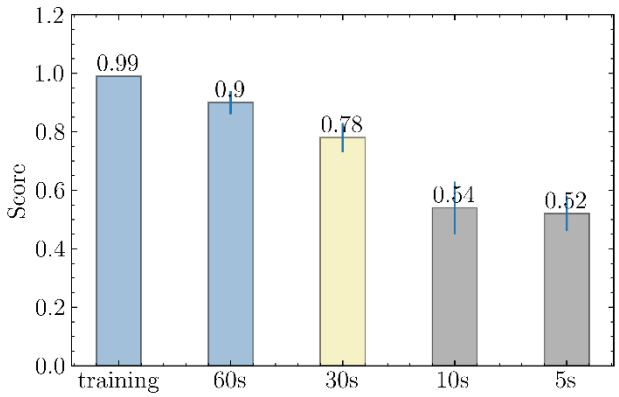

Figure 7 Comparison of test scores between data sets based on different Raman collection time. The training in this plot denotes a data set with a 120-s exposure time. The blue error bar is the standard deviation of the test score.

2.3.3 Spectral Augmentation

To increase the data size, a spectral augmentation algorithm needs to be implemented. The data augmentation algorithm includes shifting each spectrum by a few pixels randomly based on a normal distribution, and adding a Gaussian noise to each pixel at 5% of the maximum intensity. Shifting was used to simulate small changes in alignment, and adding noise was used to simulate statistical fluctuations.

For data augmentation, we also tried to combine two spectra after shifting and adding noise, and named this process as linear combination, which consists of four methods. The first method is to combine two spectra of the same label randomly, and then mix the two spectra at a random

combination factor ranging from 0 to 1. Compared with the first method, the second method mixes the two spectra at a fixed combination factor, which is 0.2. The third method randomly chooses one spectrum from the grey matter and one from the white matter, and then mixes the two spectra at a random combination factor ranging from 0 to 1. The fourth method mixes the two spectra at a combination factor of 0.2 compared with the third method.

According to the comparison of the training result of the four linear combination methods, it is found that most of the wrong predictions were predicting white as grey. This might be because of the huge intensity difference between the white and grey data. Therefore, the data normalization was added in the later data pre-processing process. Comparing with linear combination between two types of tissue, the overfitting problem was severe in same-label linear combination, which means linear combination is not feasible for data augmentation. For the linear combination using two types of spectra, Random Forest Classifier is not a suitable algorithm for training because the output of the Random Forest Classifier is the label of the spectrum, which are white matter and grey matter.

2.3.4 Low-Quality Data Removal

We remove noisy spectra using a custom Python routine automatically. It computes pixel to pixel fluctuation, the standard deviation of the fluctuation, and compares the median of the standard deviation with the median of overall intensity. The parameter and threshold are chosen based on our tests. Figure 8 is an example of the removed spectrum.

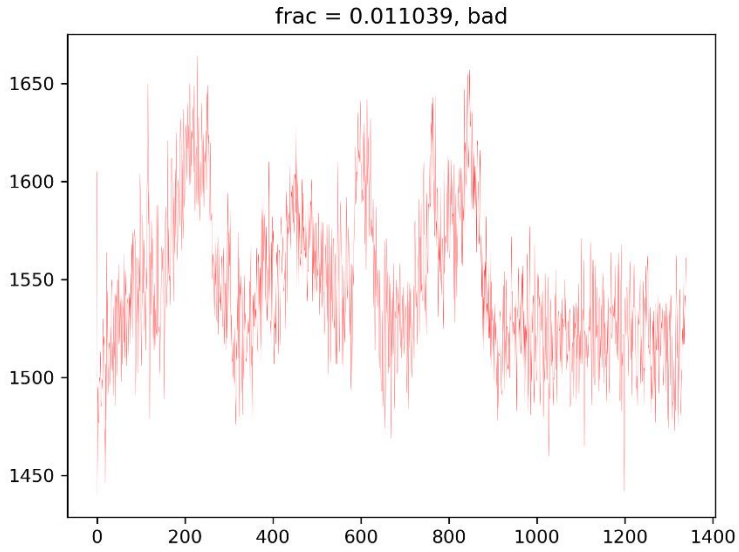

Figure 8 an example of the removed spectrum using a filter.

3. Construction of the Pipeline

3.1 Exploration of the Objectives and Machine Learning Algorithms

Before deciding to use the Random Forest Classifier, we also explored other machine learning algorithms and experimented with different prediction objectives. A naïve guess is that the number of peaks might be informative. Therefore, we first attempted to use Random Forest Regressor to identify the number of peaks in mock spectra. The average error between the number of peaks predicted by the model and the real peak numbers was calculated. As Figure 9 shows, the result is not satisfactory. One reasonable assumption is that the model assumes the vast majority of spectra contain five peaks, so the average error of five peaks is the smallest. When noise was removed, the averaged error decreases from 1.68 to 1.0, but no regular pattern can be found (as shown in Figure 10).

In order to figure out what influences the prediction result of the model, the number of peaks in each spectrum was decreased to 1, i.e., the task of the model was simplified to predict whether a

spectrum contains a peak or not. The averaged error decreased from 1.68 to 0.47 (as shown in Figure 11 the average error between the number of peaks predicted by the model and the real peak numbers when reducing the maximum number of peaks to 1.), which means the model predicted whether the spectrum contains peaks or not randomly, i.e., the model is not effective in predicting the number of peaks in the mock spectrum.

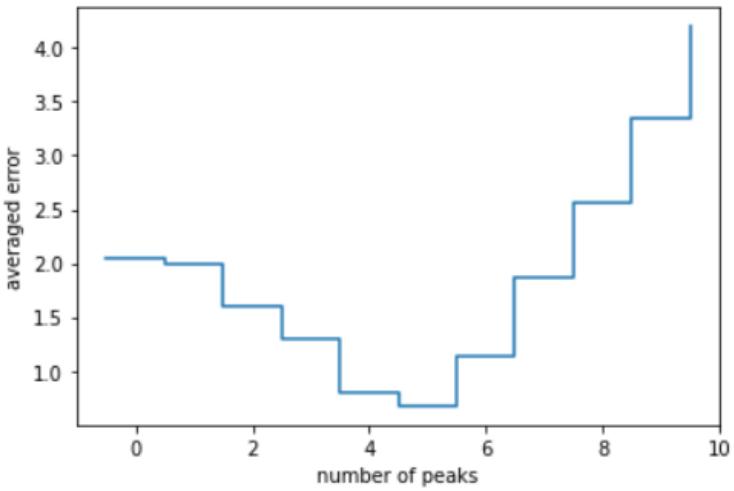

Figure 9 the average error between the number of peaks predicted by the model and the real peak numbers.

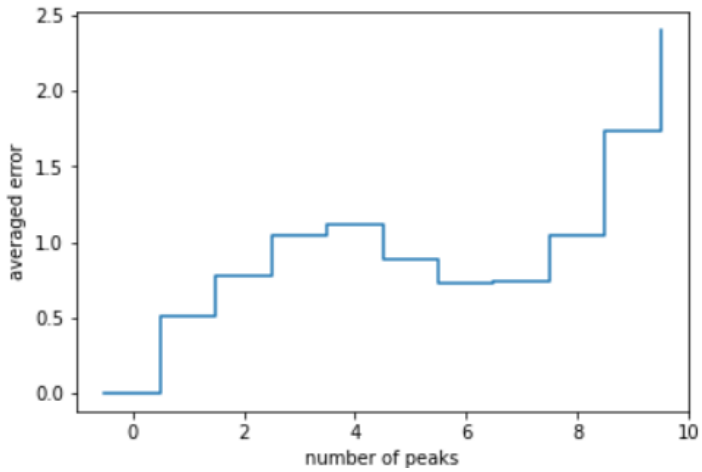

Figure 10 the average error between the number of peaks predicted by the model and the real peak numbers when removing noise.

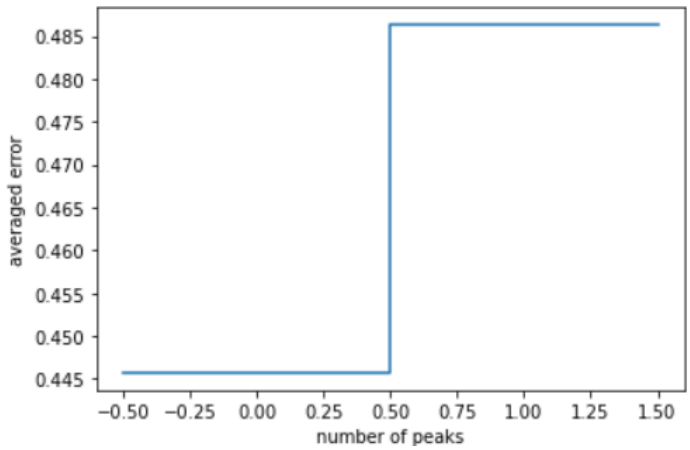

Figure 11 the average error between the number of peaks predicted by the model and the real peak numbers when reducing the maximum number of peaks to 1.

A Neural Network algorithm, which is MLPClassifier, was used for training as well. However, the model failed to achieve the desired result, i.e., a training score over 0.85. Then it came to mind that the method of generating mock spectrum might be unreasonable. When the intensity of the peaks in the spectrum was set to constant and no noise was added to the spectrum, the training results of the model were good. Figure 12 shows when the number of training examples increased, both the training score and the cross-validation score were reasonably high. A similar result was found when adding noise at a level of 5% of the maximum intensity.

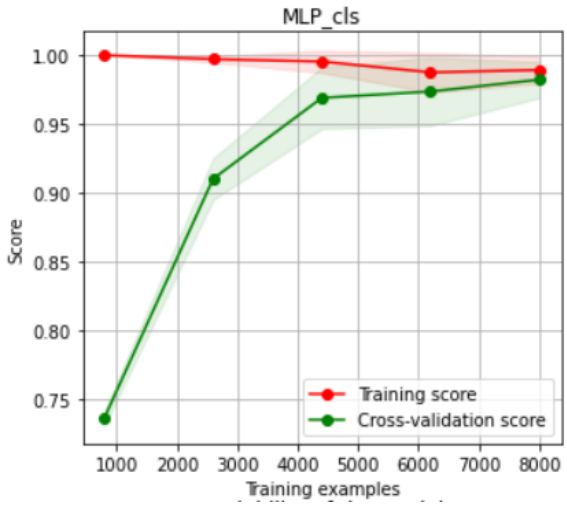

Figure 12 the plot of training score and the cross-validation score using MLPClassifier. The intensity of the peaks in the mock spectrum was set to constant and no noise was added.

Another way to generate mock spectra was to set the intensity and width of the peaks to obey a Gaussian distribution, make sure the width of the peaks is at least 5 pixels, and add no noise. The result in Figure 13 shows that the model performs well, with evidence that both scores are higher than 0.96.

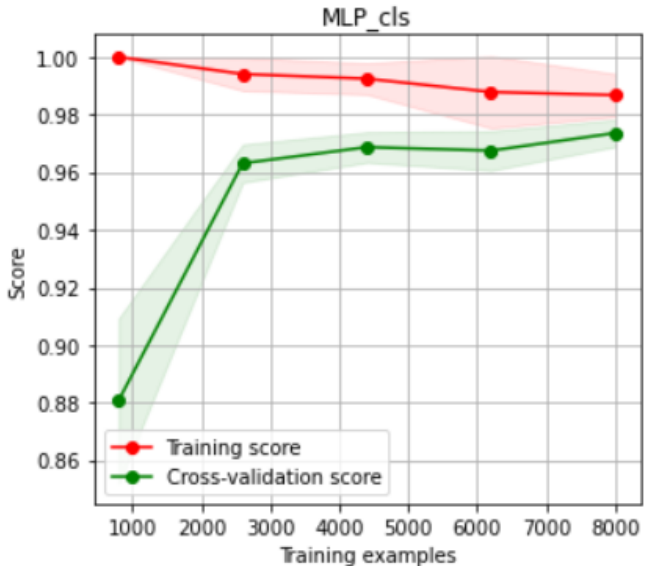

Figure 13 the plot of training score and the cross-validation score using MLPClassifier. The intensity and width of the peaks obeyed a Gaussian distribution. The width of the peaks was at least 5 pixels, and no noise was added.

When adding noise at a level of 5% of the maximum intensity, the training result stays satisfactory. As Figure 14 shows, when reaching sufficient training examples, both the training score and the cross-validation score reached over 0.95.

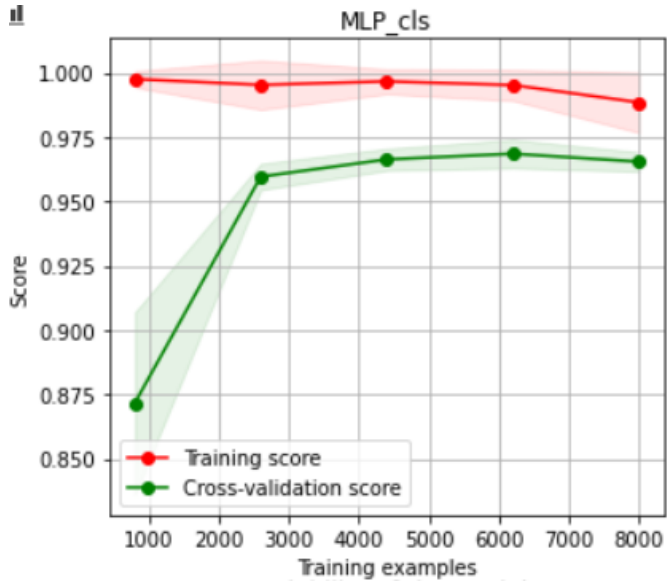

Figure 14 the plot of training score and the cross-validation score using MLPClassifier. The intensity and width of the peaks obeyed a Gaussian distribution. The width of the peaks was at least 5 pixels, and noise at a level of 5% of the maximum intensity was added.

We also used MLPClassifer to identify two different types of mock spectra. The main difference between the two kinds of spectra is the distance between two neighboring peaks in the spectrum. According to the comparison of different hyper-parameter configurations, the best configuration is to use tanh as the activation function for the hidden layer, lbfgs as the solver for weight optimization, and two hidden layers with the number of neurons as 100 and 2, respectively. As Figure 15 shows, the training score kept as 1, but the cross-validation score never exceeded 0.95. When the number of samples increased to about 380, the cross-validation score started to decrease. Although the explorations mentioned above did not reach satisfactory results, the result led us to consider using a new machine learning model to classify different types of spectra, i.e., Random Forest Classifier (see Section 3).

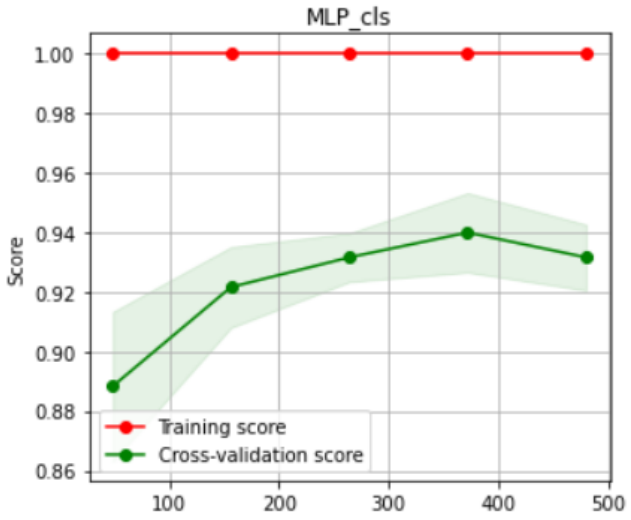

Figure 15 the plot of training score and the cross-validation score using MLPClassifier to identify two different types of mock spectra.

3.2 Further Data Pre-Processing

Data pre-processing can improve the accuracy of classification [29]. After the choice of configurations, data pre-processing includes sporadic signal removal, rebinning, background removal, data normalization, and data augmentation. To prevent high-count pixels from dominating the training, the Raman spectra were normalized to 0 to 1.To avoid leakage of the training example, which means the training data are used in the test set [30], all data were shuffled and then divided into the training set and test set in a ratio of 8: 2 before data preprocessing. Figure 16 shows the Raman spectrogram of data pre-processing in 892 to 2062 cm^{-1} wavenumber range, and 120-s exposure time.

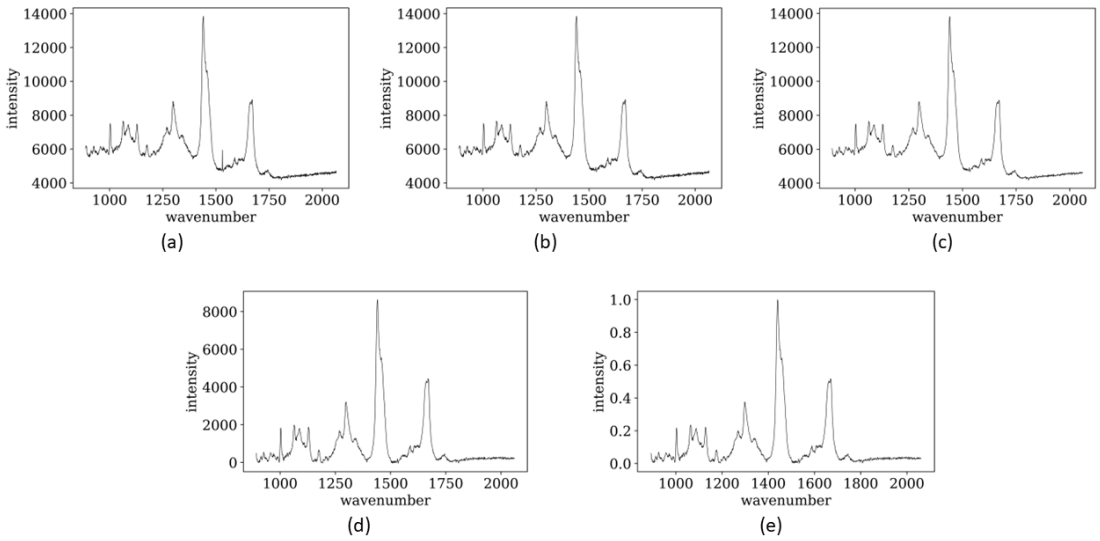

Figure 16 Example data preprocessing of the Raman spectrum: (a) original Raman spectrum, (b) Raman spectrum after the sporadic signal is removed, (c) Raman spectrum after rebinning, (d) Raman spectrum after background removal, and (e) normalized Raman spectrum. After removing the spectra with insufficient statistics, 654 spectra were used for spectral augmentation. There are 269 Raman spectra from the grey matter, 264 from the white matter, and 121 from blood vessels. Data augmentation increased the number of the training and test data to 24300 and 8400, respectively. Finally, the Raman spectra of the white matter, grey matter, and blood vessels were labeled as 0, 1, 2, respectively.

3.3 Hyperparameters Selection

Hyperparameters (e.g., the number of trees in the forest , the maximum depth of each tree , etc.) are arguments passed to the constructor of the random forest classifier class [31]. The choice and values of the hyperparameters in the random forest will significantly influence the result [32]. The GridSearchCV algorithm, which performs an exhaustive search within a specified range of parameter values [33], was used to determine the approximate hyperparameter range, and then we manually tune the hyperparameter combination for the best test score. The number of trees in the forest and the maximum depth of each tree were set to 12. The function that measures the quality of each split was Gini impurity, which is a criterion commonly used in random forest.

3.4 Model Training

Random Forest Classifier is an ensemble model for classification. When building the model, bootstrap is used as the criterion to divide the training data into each decision tree [34], which is a binary tree. At each node, Raman spectra are put into one of the two subgroups using a randomly

chosen feature [35]. The splitting criterion is to maximize the Gini impurity decrease from each node to its child node [36]. The decision tree grows until it reaches the maximum depth of the classifier, which can be set as a constant or none. If setting the maximum depth as none, the nodes will expand until all leaves are pure or contain less samples than the minimum number of samples required to split a node. The random forest algorithm combines classifiers by averaging their probabilistic prediction, rather than letting each classifier vote for a single class [31]. Figure 17 is a schematic showing how the random forest algorithm classifies a Raman spectrum.

The classifier was implemented using Python 3.8.3 [37], Scikit-learn 0.23.1 [31], matplotlib v3.2.2 [38] and SciPy 1.0 [28].

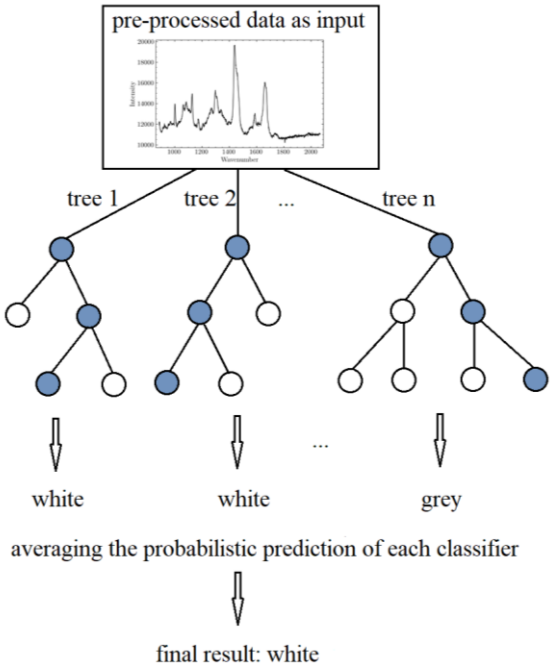

Figure 17 Schematic of random forest identification. The circles represent the nodes of the tree. The blue circles denote a possible trial when a piece of data is put in the model.

4. Results and Discussion

The performance of the training model is evaluated by confusion matrix, precision, recall, f1 score, accuracy, and the probability of each prediction.

The confusion matrix helps to obtain detailed information about the performance of the model. When comparing the number between the predicted and actual labels, the confusion matrix not only determines the number of wrong predictions, but also exposes the distribution of errors [39]. As shown in Table 2, confusion matrix summarizes the amount of data between each true label and predicted label. The result indicates that the spectral difference between white matter and blood vessels is sufficient to achieve confident predictions with zero misidentification. The model tends to identify grey matter as white matter according to the misidentified results.

Table 2 confusion matrix of the test set

Predicted / True	White matter	Grey matter	Blood vessels
White matter	3145	55	0
Grey matter	220	3265	15
Blood vessels	0	5	1695

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$f1 - score = 2 \times \frac{\mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

$$accuracy = \frac{\mathrm{TN} + \mathrm{TP}}{\mathrm{TN} + \mathrm{FP} + \mathrm{FN} + \mathrm{TP}}$$

Further analyses are available based on the confusion matrix. To be specific, precision provides the classifier's ability not to predict a negative labelled sample as positive, and recall gives the ability of the classifier to find all positive samples. The f1 score is the weighted harmonic mean of precision and recall. Accuracy represents the predictive power of the model on the given test data and labels. The support is the gives the data volume of each class in the test set. The macro-average is the average of the unweighted mean of each label, while the weighted-average is the average of

the support-weighted mean of each label. In an imbalanced data set, the weighted average is more useful since it considers the influence of imbalanced data volume.

The evaluation result in Table 3 shows that the model performance is satisfactory with high accuracy in classifying the Raman spectra of grey matter, white matter, and blood vessels. Furthermore, the result shows that the pre-processed data contain sufficient and significant information for building a reliable classification model.

The above analysis is based on the output given by the model; however, it is also important to understand the model's confidence level when making predictions. The predicted probabilities of one type of tissue are the mean predicted probabilities of trees in the forest for that type of tissue [31]. Table 4 provides the predicted probabilities of the three tissue types for part of the test set as an example. The values stand for the probabilities of the data being identified as one type of tissue. The class with the maximum value in each row indicates the model preference.

Table 3 the main identification report.

	precision	recall	f1 - score	support
White matter	0.928+/-0.006	0.981+/-0.002	0.954+/-0.004	3200
Grey matter	0.982+/-0.002	0.926+/-0.007	0.953+/-0.004	3500
Blood vessels	0.991+/-0.003	0.999+/-0.001	0.995+/-0.002	1700
Accuracy			0.962+/-0.003	8400
Macro avg	0.967+/-0.002	0.969+/-0.002	0.967+/-0.003	8400
Weighted avg	0.963+/-0.003	0.962+/-0.003	0.962+/-0.003	8400

Table 4 examples of class probabilities predicted for the test set

	White matter	Grey matter	Blood vessels
Example 1	0.121	0.794	0.085
Example 2	0.169	0.83	0.001
Example 3	0.825	0.175	0
Example 4	0.084	0.083	0.833
Example 5	0	0.083	0.917

When using MLPClassifier and mock spectra to identify peak numbers, two ways were used to generate the mock spectra to test the proper activation function. When setting the peak width ranging from 3 to 30 pixels randomly, with peak height at normal distribution, spectral intensity integrated into overall spectrum, only tanh works well as the activation function. When setting the peak width as a normal distribution, spectral intensity ranging from 3 to 30, and peak height associated with peak width and intensity, both relu and tanh work well in the predicting task. This is strong support for the importance of the way to generate mock spectra. The result reminds us of a reasonable result in mock spectra may not be effective in experimental Raman spectra. It is also found that there is not much difference in model performance between setting the noise level at 0.05 and making it associated with intensity at normal distribution. It might be because the added noise level is not high relative to the intensity, regardless of which method is used.

When using the mock spectra to identify two types of spectra, GaussianProcessClassifier was used but failed to achieve a reasonable result. It might be because this algorithm is restricted to using the logistic link function, which is not suitable for spectral classification.

When using random forest for classifying the Raman spectra, it is not necessary to use too many decision trees to achieve the desired result. The default configuration of the maximum depth of the decision tree is none. The reason for tuning this hyper-parameter is that this will significantly decrease the running time without influencing the model performance.

As part of the data augmentation process, when the added noise level is increased, the performance of the model decreases. A similar result could be seen in shifting. The result shows that applying data augmentation on the training set gives better test scores comparing with not applying data augmentation for more than 3σ. Furthermore, the test score becomes less stable if not applying data augmentation, as evidenced by the standard deviation of 0.023, which is ~8 times higher than the augmented data. Therefore, it is believed that data augmentation can improve and stabilize model performance.

By training different machine learning models, we believe the Raman spectra contain enough information to make a good classification result for a specific wavenumber range. Compared to other machine learning algorithms we have tried, Random Forest Classifier trains fast, with ~10 seconds for each run. It is believed one of the reasons is that it does not require cross-validation to prevent overfitting due to its nature of assembling several decision trees.

5. Conclusion and Future Work

In this thesis, we started with using mock spectra to determine the machine leaning algorithm to be used in the classification objective. Then several approaches of data-preprocessing were tested for an optimal configuration. The results show that the pipeline to identify the preprocessed Raman spectra of the white matter, grey matter and blood vessels works well by using the Random Forest Classifier.

However, according to the analysis of the incorrectly predicted data in the test set, Random Forest Classifier is sensitive to shifting. This may be because the model relies on individual pixels as features for identification. Therefore, future studies may consider using a convoluted model to avoid locating the feature importance to a single pixel. Furthermore, future experiments may use brain tumors to obtain Raman spectra for further study. In reality, the composition of the sample

may be complex and may not be a single type of tissue, so the main limitation of this study is that random forest does not support this case.

References

[1] D. Frumkin, A. Wasserstrom, B. Budowle, and A. Davidson, “DNA methylation-based forensic tissue identification,” *Forensic Science International: Genetics*, vol. 5, no. 5, pp. 517–524, Nov. 2011, doi: 10.1016/j.fsigen.2010.12.001.

[2] S. F. Shariat and C. G. Roehrborn, “Using Biopsy to Detect Prostate Cancer,” p. 19.

[3] D. Chatterjee *et al.*, “An autoantibody profile detects Brugada syndrome and identifies abnormally expressed myocardial proteins,” *European Heart Journal*, vol. 41, no. 30, pp. 2878–2890, Aug. 2020, doi: 10.1093/eurheartj/ehaa383.

[4] D. M. Park and J. N. Rich, “Biology of glioma cancer stem cells,” *Mol Cells*, vol. 28, no. 1, pp. 7–12, Jul. 2009, doi: 10.1007/s10059-009-0111-2.

[5] P. Kleihues, F. Soylemezoglu, B. Schäuble, B. W. Scheithauer, and P. C. Burger, “Histopathology, classification, and grading of gliomas,” *Glia*, vol. 15, no. 3, pp. 211–221, Nov. 1995, doi: 10.1002/glia.440150303.

[6] J. S. Joy, R. Thomas, and J. Johnson, “A REVIEW OF BRAIN TUMOR SEGMENTATION ON MRI IMAGE USING MACHINE LEARNING ALGORITHM,” vol. 1, no. 1, p. 6, 2021.

[7] K. Usman and K. Rajpoot, “Brain tumor classification from multi-modality MRI using wavelets and machine learning,” *Pattern Anal Applic*, vol. 20, no. 3, pp. 871–881, Aug. 2017, doi: 10.1007/s10044-017-0597-8.

[8] J. Möller *et al.*, “Applying machine learning to optical coherence tomography images for automated tissue classification in brain metastases,” *Int J CARS*, vol. 16, no. 9, pp. 1517–1526, Sep. 2021, doi: 10.1007/s11548-021-02412-2.

[9] G. McLaughlin, K. C. Doty, and I. K. Lednev, “Raman Spectroscopy of Blood for Species Identification,” *Anal. Chem.*, vol. 86, no. 23, pp. 11628–11633, Dec. 2014, doi: 10.1021/ac5026368.

[10] M. D’Acunto, R. Gaeta, R. Capanna, and A. Franchi, “Contribution of Raman Spectroscopy to Diagnosis and Grading of Chondrogenic Tumors,” *Sci Rep*, vol. 10, no. 1, p. 2155, Dec. 2020, doi: 10.1038/s41598-020-58848-0.

[11] P. Rösch, M. Harz, M. Schmitt, and J. Popp, “Raman spectroscopic identification of single yeast cells,” *J. Raman Spectrosc.*, vol. 36, no. 5, pp. 377–379, May 2005, doi: 10.1002/jrs.1312.

[12] B. W. D. de Jong *et al.*, “Discrimination between Nontumor Bladder Tissue and Tumor by Raman Spectroscopy,” *Anal. Chem.*, vol. 78, no. 22, pp. 7761–7769, Nov. 2006, doi: 10.1021/ac061417b.

[13] R. E. Kast *et al.*, “Raman spectroscopy can differentiate malignant tumors from normal breast tissue and detect early neoplastic changes in a mouse model,” *Biopolymers*, vol. 89, no. 3, pp. 235–241, Mar. 2008, doi: 10.1002/bip.20899.

[14] E. M. Kanter *et al.*, “Multiclass discrimination of cervical precancers using Raman spectroscopy,” *J. Raman Spectrosc.*, vol. 40, no. 2, pp. 205–211, Feb. 2009, doi: 10.1002/jrs.2108.

[15] “Raman spectroscopy in cultural heritage: Background paper,” *Anal. Methods*, vol. 7, no. 12, pp. 4844–4847, 2015, doi: 10.1039/C5AY90036K.

[16] Y. Fan *et al.*, “Rapid noninvasive screening of cerebral ischemia and cerebral infarction based on tear Raman spectroscopy combined with multiple machine learning algorithms,” *Lasers Med Sci*, May 2021, doi: 10.1007/s10103-021-03273-6.

[17] N. M. Ralbovsky, G. S. Fitzgerald, E. C. McNay, and I. K. Lednev, “Towards development of a novel screening method for identifying Alzheimer’s disease risk: Raman spectroscopy of blood serum and machine learning,” *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 254, p. 119603, Jun. 2021, doi: 10.1016/j.saa.2021.119603.

[18] J. Geng, W. Zhang, C. Chen, H. Zhang, A. Zhou, and Y. Huang, “Tracking the Differentiation Status of Human Neural Stem Cells through Label-Free Raman Spectroscopy and Machine Learning-Based Analysis,” *Anal. Chem.*, vol. 93, no. 30, pp. 10453–10461, Aug. 2021, doi: 10.1021/acs.analchem.0c04941.

[19] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[20] A.-L. Boulesteix, S. Janitza, J. Kruppa, and I. R. König, “Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics: Random forests in bioinformatics,” *WIREs Data Mining Knowl Discov*, vol. 2, no. 6, pp. 493–507, Nov. 2012, doi: 10.1002/widm.1072.

[21] C. Strobl, J. Malley, and G. Tutz, “An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests.,” *Psychological Methods*, vol. 14, no. 4, pp. 323–348, Dec. 2009, doi: 10.1037/a0016973.

[22] N. D. Magee *et al.*, “Raman microscopy in the diagnosis and prognosis of surgically resected nonsmall cell lung cancer,” *J. Biomed. Opt.*, vol. 15, no. 2, p. 026015, 2010, doi: 10.1117/1.3323088.

[23] S. K. Teh, W. Zheng, D. P. Lau, and Z. Huang, “Spectroscopic diagnosis of laryngeal carcinoma using near-infrared Raman spectroscopy and random recursive partitioning ensemble techniques,” *Analyst*, vol. 134, no. 6, p. 1232, 2009, doi: 10.1039/b811008e.

[24] A. C. Carnall, “SpectRes: A Fast Spectral Resampling Tool in Python,” 2017, doi: 10.48550/ARXIV.1705.05165.

[25] S. He *et al.*, “Baseline correction for Raman spectra using an improved asymmetric least squares method,” *Anal. Methods*, vol. 6, no. 12, pp. 4402–4407, 2014, doi: 10.1039/C4AY00068D.

[26] A. Alsaafin and A. Elnagar, “A Minimal Subset of Features Using Feature Selection for Handwritten Digit Recognition,” *JILSA*, vol. 09, no. 04, pp. 55–68, 2017, doi: 10.4236/jilsa.2017.94006.

[27] Z. Wang, C. Lai, X. Chen, B. Yang, S. Zhao, and X. Bai, “Flood hazard risk assessment model based on random forest,” *Journal of Hydrology*, vol. 527, pp. 1130–1141, Aug. 2015, doi: 10.1016/j.jhydrol.2015.06.008.

[28] P. Virtanen *et al.*, “SciPy 1.0: fundamental algorithms for scientific computing in Python,” *Nat Methods*, vol. 17, no. 3, pp. 261–272, Mar. 2020, doi: 10.1038/s41592-019-0686-2.

[29] J. Huang, Y.-F. Li, and M. Xie, “An empirical analysis of data preprocessing for machine learning-based software cost estimation,” *Information and Software Technology*, vol. 67, pp. 108–127, Nov. 2015, doi: 10.1016/j.infsof.2015.07.004.

[30] S. Chakrabarti, Ed., *Data mining: know it all ; [a 360-degree view from our best-selling authors ; key concepts, practices, and protocols fully explained ; the ultimate desk reference for researchers, scientists, engineers, and practitioners]*. Amsterdam Heidelberg: Elsevier, Morgan Kaufmann, 2009.

[31] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[32] A. Callens, D. Morichon, S. Abadie, M. Delpey, and B. Liquet, “Using Random forest and Gradient boosting trees to improve wave forecast at a specific location,” *Applied Ocean Research*, vol. 104, p. 102339, Nov. 2020, doi: 10.1016/j.apor.2020.102339.

[33] Pirjatullah, D. Kartini, D. T. Nugrahadi, Muliadi, and A. Farmadi, “Hyperparameter Tuning using GridsearchCV on The Comparison of The Activation Function of The ELM Method to The Classification of Pneumonia in Toddlers,” in *2021 4th International Conference of Computer and Informatics Engineering (IC2IE)*, Depok, Indonesia, Sep. 2021, pp. 390–395. doi: 10.1109/IC2IE53219.2021.9649207.

[34] T.-H. Lee, A. Ullah, and R. Wang, “Bootstrap Aggregating and Random Forest,” in *Macroeconomic Forecasting in the Era of Big Data*, vol. 52, P. Fuleky, Ed. Cham: Springer International Publishing, 2020, pp. 389–429. doi: 10.1007/978-3-030-31150-6_13.

[35] R. Garreta and G. Moncecchi, Eds., *Learning scikit-learn: machine learning in Python: experience the benefits of machine learning techniques by applying them to real-world problems using Python and the open source scikit-learn library*. Birmingham, UK: Packt Publishing Ltd, 2013.

[36] C. Strobl, A.-L. Boulesteix, and T. Augustin, "Unbiased split selection for classification trees based on the Gini Index," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 483–501, Sep. 2007, doi: 10.1016/j.csda.2006.12.030.

[37] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[38] T. A. Caswell *et al.*, "matplotlib/matplotlib: REL: v3.2.2." Zenodo, Jun. 17, 2020. doi: 10.5281/ZENODO.3898017.

[39] S. M. Piryonesi and T. E. El-Diraby, "Data Analytics in Asset Management: Cost-Effective Prediction of the Pavement Condition Index," *J. Infrastruct. Syst.*, vol. 26, no. 1, p. 04019036, Mar. 2020, doi: 10.1061/(ASCE)IS.1943-555X.0000512.

Appendix

(a summary of the glossary associated with the model)

MLPClassifier: Multi-layer Perceptron classifier.

Tanh: the hyperbolic tan function used as activation function for the hidden layer in MLPClassifier.

Relu: the rectified linear unit function used as activation function for the hidden layer in MLPClassifier.

Lbfgs: an optimizer in the family of quasi-Newton methods used as the solver for weight optimization in MLPClassifier.

GridSearchCV: This algorithm performs an exhaustive search for the specified parameter values of an estimator, such as Random Forest Classifier, MLPClassifier, etc. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid.

GaussianProcessClassifier: a classification algorithm based on Laplace approximation.