Preparation of raman spectra for methylation-based classification methods in machine learning

Author: Joel Sjöberg 38686
Masters thesis in Computer Science
Supervisor: Luigia Petre
The Faculty Of Science And Engineering
Åbo Akademi University
2021

Contents

Foreword

Abstract

Chapter 1

Introduction

The mammalian brain contains so-called neurons and glial cells. Historically it was believed that the brain contained ten times more glial cells than neurons, but recent studies suggest the number of neurons are equal to the number of glial cells [1]. Glial cells were previously thought to be insignificant in terms of the brains computational functionality, only lending structural support to the neurons. Recent studies have disputed this and suggest their contribution to the nervous system is greater than once thought, though their actual function is still a matter of speculation. Glioma is a type of brain cancer which manifests within the glial cells and disrupts brain functions. The survivability of the cancer is extremely poor with a life expectancy of a few months without treatment to a few years depending on the patients health, the tumor type and grade; rarely do patients survive for five years [2, 3, 4]. Gliomas are categorized depending on their glial-cell of origin. There are four main types of glial cells (also called neuroglia or simply glia): oligodendrocytes, astocytes, ependymal cells and microglia. Oligendroglioma originates from oligodendrocytes, astocytoma from astocytes and ependymomas originate from ependymal cells. Furthermore, astocytoma-types may develop into glioblastoma multiforme (GBM), the most aggressive form of brain cancer; this may even communicate with microglia to increase tumor growth [5]. It is also possible for GBM to develop from other brain cells [2]. This cancer is particularly aggressive due to its quick reappearance in the brain only a short period after surgery [3]. The heterogeneity of GBM-cells further complicates the healing process, due to poor response to targeted treatments [6].

The World Health Organization (WHO) has defined four levels (or "grades") of cancer severity used to describe the cancer aggressiveness and tumor growth. Grades I and II are considered low-grade and grades III and IV are considered high-grade. Glioblastoma is cathegorized as a grade IV cancer [4, 7]; these grades are used to determine an appropriate prognosis and line of treatment. A study by Vigneswaran et al. [7] suggested these grades could be divided further to better describe the features of the tumor and express versions with poor prognosis. This suggestion is also supported by Hirose et al. [8]. Ceccarelli et al. [9] introduce alternative subdivisions of these classes, which show promise in expanding knowledge about glioma tumors and aid in treatment selection. Such evaluations require in-depth knowledge about the tumor tissue and further examination, which may last for weeks after extraction. Ceccarelli et al. define the subdivisions by six distinct classes, labeled LGm1-6. Their analysis showed IDH mutations in LGm1-3; as the name suggests, IDH mutations refers to mutations in the IDH1 or IDH2 genes which encode for the enzymes IDH1 and IDH2. These mutations are shown to be significant in a variety of cancers, including glioma [10]. Furthermore, LGm4-6 were IDH wild-type, where a majority of tumors could be labeled as glioblastoma. IDH wild-type refers to IDH genes with no mutations, but they are often correlated with poor prognosis in high-grade glioma. These clusters are reinforced by the results produced by Vigneswaran et al. The process of determining a prognosis and a line of treatment has great promise in improving patient outcome and is essentially influenced by classifying tumors into these subdivisions.

This thesis is the result of a project whose purpose is to optimize the categorization process, based on a deep learning model capable of predicting tumor-types in a matter of minutes. The project relies on tissue from tumors extracted from 45 patients and scanned using Raman spectroscopy. Raman spectroscopy was invented by Chandrasekhara Venkata Raman and measures the vibrations of molecules by spectral analysis. This method can be executed fairly quickly and can provide chemical information from the spectral light. A laser emits a ray unto the tumor tissue, causing the energy level of the molecules within to change, which in turn changes their vibration. This vibration is gathered by the instrument and provides information regarding the molecular properties of the material [11, 12]. This spectra is the data which the model uses as training and testing data. The choice of using Raman spectra in this way is due to the method's success in previous studies of Raman spectra using machine learning [13, 14]. The use of Raman spectra is further motivated by Liu et al. [15], whose work show promise for deep learning models trained on raw Raman spectra. The advantage of this method in the context of multilabel classification seem considerable, when compared to other machine learning methods such as Support Vector Machines, Random Forest and K-nearest neighbor [15].

This thesis aims to analyze the spectra extracted from all patient samples in an attempt to automate outlier detection. The samples are examined by statistical methods designed specifically for that purpose. Hierarchical clustering and K-means clustering are applied to the samples to divide the spectra into subsets which we find identifies many outliers. These results are examined in contrast to the results of a criterion for finding outliers in the data by the provider, after which the method most suitable for his purpose is used to remove the outliers. Following the removal of the outliers, we present a preprocessing pipeline which will be used to prepare the data for machine learning applications such as Artificial Neural Networks or Support Vector Machines. The features which best divide the data into the six LGm classes are extracted using f-classif. These features drastically reduces the size of the spectra which are analyzed for prognosis which in turn reduces the examination time. The thesis then aims to provide a clear way of preparing Raman spectra for machine learning applications and provides the most important features those spectra consist of. Suggestions and a discussion for how these methods may be improved, which alternative methods could be tested instead and eventual limits to this project are also given for future consideration.

The thesis is structured as follows, Chapter 2 presents the preliminary background for the statistical methods used in the project, along with the necessary mathematical definitions by which these methods are defined. Among these are the notion of the mean and standard deviation which are central to methods of analysis we use, this includes the analysis of variance (ANOVA) which is applied by the f-classif method for extracting features from the data. Understanding the underlying definitions and consequences is necessary to validate and confirm the results and as such, the chapter also presents the definition of supervised and unsupervised learning with short definitions on machine learning terminologies. The formal definitions of K-means clustering and Hierarchical clustering is given. In Chapter 3, we discuss the exploration methods in detail, to give further understanding of the data on which this project is based. The chapter begins by introducing the concrete shape of the data. Feature selection is applied to the data by the f-classif method and the results are examined. The majority of this chapter is based on the visual analysis of the outlier detection. This is done by applying the statistical methods and the clustering methods to the samples. The results of each method are analyzed in contrast to the criterion defined by the dataprovider in greater detail to form an argument for or against the method in question. The chapter ends by removing the outliers by the optimal method and performing feature selection once more on the data devoid of outliers. In Chapter 4, we present our suggestion as well as arguments for the preprocessing pipeline to prepare the data for machine learning. An Artificial Neural Network is created and trained on the curated data. The performance of the architecture is measured and presented. The thesis is concluded in Chapter 5, where we discuss improvements and suggestions to out methods. We also provide suggestions for alternative methods used for feature selection and preprocessing for future study and tests.

Chapter 2

Preliminaries

In this chapter we review the concepts on which this thesis is based. We first cover the statistical theory for understanding the methods of analysis used in Chapter 3 and use of Machine Learning (ML) in Chapter 4. We then proceed by introducing statistical concepts required for understanding the methods employed in the project. We then review common concepts and specific methods within machine learning which is the central theme in this thesis.

2.1 Statistics

In this section we discuss the primary statistical concepts required for understanding the methods used in this thesis. The first concepts necessary for the later chapters are the mean and the standard deviation. The mean is a value used to describe the average value of a population. A population is the term used to describe the complete collection of elements in some context e.g. all people alive, all numbers in an interval etc. The mean is an important concept in statistics as it is often used as a characteristic of the elements found in the population under analysis. The mean is calculated by the sum of the each element in the population divided by the number of elements in said population. The mean μ of a population of n numbers L, is calculated as expressed in formula (2.1).

$$\mu = \sum_{i=1}^{n} \frac{L_i}{n} \tag{2.1}$$

The mean of a population is usually used in association with the standard deviation. The standard deviation is the square root of the variance of that population. The variance is an expression for the scaled summed squares of differences from the mean of the entire population. Calculating the square root then produces a value expressing the the dispersion of elements within the population around the mean. The variance is calculated by measuring the summed squared distance between each element within the population and the mean, divided by the number of elements in the population to scale the sum. In the case where the entire population is impossible to analyze or unknown, samples are extracted from the population (i.e. subgroups of elements randomly taken from the population). The sample variance changes slightly form the population variance; it divides the summed squares of differences by the number of elements in the sample subtracted by one. Subtracting the original denominator by one is called "Bessel's correction". The correction is based on the fact that, if the population mean is unknown (as is often the case when samples are gathered), then the mean used in calculating the variance will only be an estimation. Subtraction by one is performed to avoid bias towards the sample mean and get an unbiased estimation of the population variance [16, 17]. The standard deviation σ of a sample is thus calculated as expressed in formula (2.2).

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(\mu - L_i)^2}{n-1}} \tag{2.2}$$

2.1.1 The standard deviation test

The mean and standard deviation can be used to detect outliers in a sample of data points drawn from an unknown population. The Gaussian distribution (also called the normal distribution) is used in association with the mean and the standard deviation. If the frequency which elements appear within a sample are more likely to be close to the mean than far away from it, we say that the elements within that sample are normally distributed. If an element differs from the mean by more than three standard deviations, the possibility of that element not belonging to that distribution is extremely high. Such elements, which likely do not belong to the population, are called outliers. The method which detects outliers by measuring the distance between each element and the mean is referred to, in this thesis, as the standard deviation test (SDT). The test assumes that elements within the collection are normally distributed with some mean μ_c and some standard deviation σ_c. By computing the *z-score* of the elements, the data is transformed into a standardized form through standardization. *Z-score* standardization subtracts the mean from all elements in the collection and divides the difference by the standard deviation, as expressed in equation (2.3) [18].

$$\frac{L_i - \mu_c}{\sigma_c} \tag{2.3}$$

Following standardization, the entire sample will have a mean of zero and a standard deviation of one. Each element in the sample can then be measured using the standard deviation (one) as unit; outliers can then be discarded from the sample by removing elements which have an absolute value of 3 or higher.

2.1.2 The interquartile range method

An alternative method suited for outlier detection is the interquartile range method (IRM). IRM is based on analyzing the sample by its median, which is the center of the sorted sample. It is applied by sorting the sample elements in ascending order and organizing the sample into four percentiles, each percentile distanced from the others by 25% of the sample. Two quartiles center around 50% of the sample contents, this center is referred to as the interquartile range. IRM requires two values from the sample, those are the highest value of the 25th percentile and the highest value of the 75th percentile. The first value is gained by taking the biggest value of the q_{25} first elements from the sorted collection, where q_{25} is 25% of the number of elements in L (calculated by $0.25 \cdot |L|$). The last percentile is gained in similar fashion, with the exception that the highest value is taken from the first 75% of the sorted data. Two "cut-off" points are then defined, by multiplying each of the two values by a constant k (which is called the "cut-off" constant). Elements can then be labeled as outliers if those elements fall below the lower "cut-off" point or above the higher "cut-off" point. This method is well suited for sample sizes which contain approximately 100 elements but may prove cumbersome when the number of elements exceed that value [19, 20, 21].

2.1.3 Analysis of variance

The analysis of variance (ANOVA) is a method for statistical analysis developed by Ronald Fisher to analyze the difference among sample means in a collection of samples. It is based on the null-hypothesis stating the sample means of two or more samples are the same. The analysis then yields a *F-value* and a *p-value* which are used with the F-distribution to accept or reject the null-hypothesis. The analysis assumes the population the samples are drawn from in normally distributed and that the samples are independent of each other. It also assumes the standard deviations and variances are roughly equal among the samples.

The analysis is performed by calculating the mean of each sample. The summed square of differences from each mean are then calculated for their respective sample, the result is then subtracted by the squared sum of the elements within the sample divided by the number of elements in said sample. This calculation for sample S is formally expressed i equation (2.4).

$$\sum_{s}^{S}(s-\mu_S)^2 - \frac{(\sum_s^S s)^2}{|S|} \tag{2.4}$$

This is performed once for all samples drawn from the population, the results are then summed together into a sum of sample sums (SS). The calculation is also performed once on the total collection of all elements from all samples, the result is stored in the total sum of sample sums (SST). The total sum of squares is the sum of SS and the sum of squared distances from each sample mean to the total mean (SSM). SSM is therefore calculated by subtracting SST and SS. The analysis also require divisions by two values called the degrees of freedom. They are calculated as follows: d_1 is the number of samples subtracted by one and d_2 is the number of elements in the total collection subtracted by the number of samples. The *F-value* is calculated by a fraction of two fractions. Let K_1 be the fraction $\frac{SSM}{d_1}$ and K_2 be the fraction $\frac{SS}{d_2}$, the *F-value* is defined as expressed in formula (2.5).

$$F = \frac{\frac{SSM}{d_1}}{\frac{SS}{d_2}} = \frac{K_1}{K_2} \tag{2.5}$$

The quotient yielded by the fraction in formula (2.5) can then be inserted into a pre-calculated table of the F-distribution provided the degrees of freedom to yield the *p-value*. A small *p-value* (i.e. lower than 0.05) indicates that the null-hypothesis may be rejected with relative certainty whereas a high *p-value* indicates the null-hypothesis holds and should be accepted with relative certainty [22].

2.2 Machine Learning

Machine learning is the practice of computing models for relationships between sets of data. The field has garnered significant interest within academia and industry alike due to the promising results in applications for which deterministic algorithms have proven difficult or impossible to make. Examples of such applications are computer vision, natural language processing and personalized advertising, to name a few [23, 24, 25]. There are two main paradigms for learning: Supervised learning (using labeled data to approximate models) and unsupervised learning (finding patterns within the data itself).

Models are used to great length within many scientific domains. In the context of machine learning, a model can be seen as a data structure made out of constant parameters combined with an algorithm which utilizes the data structure to produce predictions given an input vector (the input can also be in the form of a multi-dimensional matrix).

The model can be represented mathematically as a collection of structures in the form of vectors or matrices, the elements of which are referred to as parameters. A model can consist of learnable parameters θ and non-learnable parameters (often generated by stochastic initialization if used). The model computes a function f to yield a prediction y by applying the algorithm to the parameters given an input example x (which can be a vector or a matrix) drawn form the data set X. Let the dimensionality of the input x be equal to the dimensionality of θ. An example of a model prediction, where the algorithm produces predictions through addition, is given by formula (2.6).

$$y = x_0\theta_0 + x_1\theta_1 + ... + x_n\theta_n \tag{2.6}$$

Machine learning then, is the practice of changing (also known as tuning) θ by introducing small changes to the elements within θ. This is done to minimize a loss function L which computes the error (or loss) given y. The process of changing θ is known as the training process and is central to machine learning. In the training process for supervised learning, the data gathered for the model is separated into three sets. These sets are referred to as the training set, validation set and test set. They are randomly collected samples from the common data set such that the intersection between the sets is empty. The purpose of the training process is to train the model on the training data and use the validation and test sets as a means to validate the model performance on data not encountered during the training process. Supervised learning requires that the examples used have a label which the model tries to predict (data which possess labels are called labeled data). We say that a model generalizes well to the data if the training process allows the model to perform well on unseen data. If the model manages to perform well on the training set but fails to generalize, the model is said to overfit to the data. Unsupervised learning is a learning paradigm which does not rely on the use of labeled data. Instead, the paradigmn focuses on organizing the data in a way that minimizes L. Predictions can then be performed by evaluating the way the data has been organized by some method related to the problem context. The problem context is usually framed by two problem categories. These categories are regression and classification. Regression is the task of predicting continuous values given either continuous or discrete data i.e. Predicting stock prices given information about the current economical situation or predicting the number of patients in a hospital during a pandemic given the density of the population. Classification aims to group different examples into categories (or classes) i.e. Predicting whether an image contains a dog or a cat or which methylation type a given tumor belongs to. Both problem contexts are encountered in this project.

2.2.1 K-means Clustering

Clustering is an unsupervised learning method whose primary use is in grouping data into sets. In this thesis we consider the *K-means* clustering algorithm. The following is a formal definition of *K-means* clustering as defined by MacQueen [26]. Given a set of n-dimensional points (where $n \in \mathbb{N}$) E_N and a desired amount of partitions k of E_n, partition the elements of E_n into k sets. The partitions are stored in a superset S such that $S = \{S_1, S_2, ...S_k\}$. The partitioning of E_n is performed by randomly initializing k n-dimensional points as randomly selected points within E_n. We define the set V with elements v, where v_i is the i:th cluster center and $i \in [1,k] \cap \mathbb{N}$. The partitioning of the elements $x \in E_n$ into their respective partition S_i is performed by computing the closest cluster center for all elements in E_n. Let T_i where $i \in [1,k] \cap \mathbb{N}$ be the set of elements $x \in E_n$ such that the distance from the element to the relevant cluster center is minimal, T_i is defined by formula (2.7).

$$T_i = \{x : x \in E_n | (|x - v_i| \leq |x - v_j|)\}(i \in [1,k] \cap \mathbb{N}) \tag{2.7}$$

For centers that share equal distance to any given x, the cluster with the smallest index is chosen as the containing set. This is performed by iteratively defining S_i as the intersection of T_i and the points which are not in any prior partitioned sets i.e. for S_j where $j < i$. This is denoted by the set complement S_i^c for all elements not in S_i. Let S_1 be defined by T_1, then he partitions $S_i \in S$ for $i \in [2,k] \cap \mathbb{N}$ are defined by formula (2.8).

$$S_i = T_i \cap \bigcap_{j=1}^{(i-1)} S_j^c \tag{2.8}$$

A consequence to this definition is that outliers have a potential to drastically change the quality of the clustering outcomes [27]. To remedy this and the stochastic nature of the initialization process, the method is run several times on the same dataset, yielding the optimal solution from those runs. This does not guarantee the best solution for the problem, but the solution is approximated. The problem *K-means* clustering attempts to solve is proven to be NP-hard [27, 28] but the algorithm itself has a time complexity of $O(n^2)$ [29].

2.2.2 Hierarchical clustering

Hierarchical clustering is a deterministic clustering method. Each cluster formed is based on the entire dataset, in contrast to *K-means* which approximates clusters by performing small changes to the cluster centers. The method produces clusters by iteratively combining the closest clusters according to the given linkage criterion (defined in section 2.2.2.2). The two primary strategies for forming clusters are *agglomerative* and *divisive*. Agglomerative clustering initializes one cluster for each data point and combines them in a hierarchy according to the linkage criterion until all clusters are part of the hierarchy. Divisive strategies initializes one universal cluster for all data points and proceeds to separate the points into distinct clusters according to the linkage criterion. The method proceeds until all data points are separated to their own cluster within the unifying hierarchy. The project described in this thesis uses the *agglomerative* strategy. All strategies rely on specific distance metrics and linkage criteria [30].

2.2.2.1 Distance metrics

Let u and v be vectors of the same dimension $n \in \mathbb{N}$. The *Euclidean distance* (also called *L2-distance*) metric can be used to measure distance between the vectors in Euclidean space. The *Euclidean distance* between u and v is defined by formula (2.9).

$$d(u,v) = \sqrt{\sum_{i=1}^{n} (u_i - v_i)^2} \tag{2.9}$$

The *Manhattan distance* (also called *L1-distance*) metric is also a viable alternative, if the distance is to be measured in blocks. The distance is akin to finding a shortest path among blocks and is therefore calculated as expressed in formula (2.10).

$$d(u,v) = \sum_{i=1}^{n} |u_i - v_i| \tag{2.10}$$

Cosine similarity measures similarity between vector angles and suits situations where certain vectors are expected to be similar. Should the vectors be sizable in terms of dimensionality, this method will yield varying results, especially if the elements have significant variance in each dimension. It is calculated as expressed in formula (2.11).

$$d(u,v) = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2}\sqrt{\sum_{i=1}^{n} v_i^2}} \tag{2.11}$$

2.2.2.2 Linkage Criteria

In order to measure distance between clusters it is essential to know between which points the distance should be measured, since clusters often consist of several points.

Linkage criteria describe the method for determining how the distance metric will be applied. In this project, we use the library SKlearn and the already defined methods within it to perform our analyses, the following criteria are therefore the only focus for this subsection. SKlearn define four criteria in the documentation: Single linkage, complete linkage, average linkage and ward linkage [31]. Depending on which criterion is applied, the results may differ considerably.

Single linkage goes through each pair of clusters measuring the distance among all points within one with respect to the other. The distance between these clusters is determined to be the distance between the two closest points. Let U be the elements in the first cluster and V be the elements of the second. The distance between the first and the second cluster is defined formally in formula (2.12).

$$d(U,V) = \forall_{u,v \in U,V} min(d(u,v)) \tag{2.12}$$

Single linkage tends to produce trivial results, forging a hierarchy in a chain where individual elements slowly merge with the bigger cluster. In contrast, complete linkage considers the largest distance between two points for every pair of clusters. The distance between two clusters then become the distance between the points which are the furthest apart, formally expressed in formula (2.13).

$$d(U,V) = \forall_{u,v \in U,V} max(d(u,v)) \tag{2.13}$$

By considering the largest possible distance between two clusters, this criterion bypasses the setback of single linkage, allowing more clusters to form before merging into one unifying cluster.

Average linkage calculates the average between all elements for every pair of clusters and merges the ones possessing to the minimal average distance. Formally described by formula (2.14).

$$d(U,V) = \frac{1}{|U||V|} \sum_{u \in U} \sum_{v \in V} d(U_u, V_v) \tag{2.14}$$

Ward linkage represents distance by how much the summed square would increase by merging them. The method aims to merge the clusters such that the within cluster variance is minimal. Let c_a be the center of cluster a, then ward linkage is expressed formally by formula (2.15) [32].

$$d(U,V) = \frac{|U||V|}{|U| - |V|} ||c_U - c_V||^2 \tag{2.15}$$

2.2.3 Feature Selection

In many cases, the data available contains numerous features, which often helps to building sufficient classifiers as the model may find non-trivial patterns among the features. To avoid expanding the dependence on large datasets, it is often necessary to strip the data of certain features which possess minimal correlation to other features or which lack that correlation entirely [33]. Features that possess the necessary expressive information are not always trivial, there are several ways in which they may be found. In this project we exclusively use one form of feature selection with the SKlearn library. The SelectKBest method is a method which ranks features by their significance according to some scoring function. In this project, we use the f-classif method to score the features in the data set. The method computes the F-value using ANOVA for each feature in the data provided, the features are then sorted according to the F-value after which SelectKBest returns the k features with the highest score.

2.3 Deep Learning

Deep learning (DL) is part of machine learning and concerns the use of massive models. DL is commonly used in association with artificial neural networks (or simply neural networks) which have been used to great success in classification and regression tasks alike. In this section, we review the preliminary methods central to neural networks in the context of deep learning. The concepts of activation functions, layers and optimizers are covered in context of what the project requires.

2.3.1 Neural Networks

Neural networks are supervised learning models which have been used to great success during the 20th century; in no small part due to the increase in computational power over the past decade. With the use of neural networks, several fields including Natural Language Processing, Encoding and Image classification have undergone revolutionary leaps in performance regarding optimization due to the predictive power of these networks [34, 35, 36, 37]. At the same time they are heavily criticized for their complexity, yielding a structure much more akin to a so called *"black box"* than a reliable and deterministic method for prediction. This complexity is due to numerous different structural typologies available at present and an awesome number of learnable parameters [38]. A consequence of this is hard skepticism in regards to the correctness of their functionality within practical use. While these models have shown great promise when compared to their human counterparts, the question remain whether or not perfect performance can be yielded from the constructed models.

Let $n = 1..k$, then a neural network consists of a set of learnable parameters W_n for a model possessing k layers. These parameters are commonly referred to as weights and are matrices with arbitrary dimensionality, with a set number of parameters for each element in x. The first set of weights have one dimension set to the shape of x and the other shapes are chosen according to the size of the layers specified by the user. The last set of weights W_k has the size of the expected output signal (usually the number of different categories available in the context of classification). The layers denote the size of the different shapes the input is transformed into as the input propagates through the architecture. The input is propagated through the architecture via the dot product of the weights and the layer signals, yielding a new vector of shape l_n. Layers can also be convolutional, meaning they are multi-dimensional structures which can be used in context of image classification and Natural Language Processing where input can be read in sequences, rather than giant data structures. This is managed by initializing smaller kernels which are able to compute signals from the input by only observing the defined size, they can then move over the entire input by steps called strides after which a pooling layer is used to summarize the final layer signal. Between each transformation, an activation function is used to transform the signal further in a non-linear fashion. Each layer transformation f_n is then the yielded signal from the activation function given the dot product between the current and preceding layer. The signal of layer l_i where $1 < i \leq k$ is then the result of the activation function σ of the dot product of the preceding layer signal l_{i-1} and the weights W_i. This layer function is denoted by the output of the nested function call of all functions $f_0, f_1, ...f_i$ of x as expressed in formula (2.16) [39].

$$f_i(f_{i-1}(...f_1(x))) = l_i = \sigma(W_i \cdot l_{i-1} + b_i) \tag{2.16}$$

The variable b_i is the bias term for the activation. It's inclusion allows the activation curve to be moved along the x-axis. This shift in position of the activation allows the model to further change it's own behavior through the learning process. It avoids bias towards a y-intercept of the activation function.

2.3.2 Activation functions

Activation functions are used in neural networks to transform the input in a non-linear fashion. The function can be any function on numerical elements, the only requirement is that it must be derivable for all possible inputs. The functions usually transform the signal to be in a certain interval such as the hyperbolic tangent function (tanh) or the sigmoid function. The sigmoid function was originally used due to the similarities with the activation of biological neurons. The "s"-shaped curve of the function transforms any signal to the interval $[0,1]$. It is comparable to tanh, which transforms signals into the interval $[-1,1]$. The functions are formally expressed in formula (2.17) and (2.18) respectively.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.17}$$

$$\sigma(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.18}$$

The choice of activation function then depends on what range the user wants the signals to fall into. Sigmoid and tanh work sufficiently well for many models, giving promising results for many different tasks within deep learning. One flaw is that they are computationally expensive to calculate. The rectified linear unit (ReLU) is an activation function which is easily computed for many elements without the necessity for many computational resources. The ReLU returns the input itself if the input is greater than zero, zero otherwise. The derivative of ReLU is similarly efficient to compute, the derivative is one for input greater than zero, zero otherwise. The function is shown to outperform sigmoid and tanh as activation function in many cases which has promoted its use in several applications. The drawback of ReLU is the derivative of 0 for signals of zero and below. The derivative is used during the training process to introduce changes to the model. With ReLU, the activation signal will become zero if the signal is smaller than zero. The neurons which suffer from this problem are referred to as "dead", since the weights used for their activation always bring the signal to or below zero. A possible fix for this is the leakyReLU function, whihc introduces a slight slope to the ReLU curve for values below zero. The ReLU method may then be defined through the leakyReLU function with a slope s of zero for signals below zero. The derivative then becomes one for values greater than zero, and s for values below zero [40, 41].

The final activation function introduced in this section is the softmax function. Softmax is a method which transforms the signal in the output into a probability distribution i.e. scales the signals according to the maximum element and transforms the data to have a sum of one. This is done using Eulers constant e as base, dividing e raised to the power of each element in the input signal and dividing the exponent by the sum of all exponents. This is formally expressed in formula (2.19) [42].

$$\sigma(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \tag{2.19}$$

2.3.3 Regularization

Regularization methods are used during the training process to aid in generalization for neural networks. One such method is dropout which assigns a dropout rate to specifically selected layers. Dropout randomly reduces signals of individual neurons in the selected layers to zero which prevents the model from enforcing connections which become heavily affiliated with certain types of predictions. This is especially important ion large architectures where layers can consist of hundreds of neurons where strong reinforcements are easily established [43]. Gaussian noise may also be added to all signals in any layer to shift the signals in them sporadically. However, this method requires some knowledge about the range of the signals within the layers, as large additive noise can remove any necessary information from the input which affects the gradient significantly. The model will learn to reduce dependency on noise during training, provided the noise is not "destructive". For example, adding Gaussian noise drawn from a normal distribution with a standard deviation of five to signals returned by the sigmoid activation function shift the distribution of signals which removes valuable signal information. Batch normalization is a regularization technique which normalizes the change applied to W during training over several batches of input and help regularizing the model [44].

2.3.4 Optimization

Neural networks have many usable loss functions depending on context. In context of classification with multiple categories, categorical cross-entropy is commonly used to measure error between the prediction of the network (usually produced with softmax or sigmoid activations) when the prediction is meant to categorize the input. The cross-entropy loss is calculated as the sum of negative elements of the true label y_t multiplied by the logarithm of the predicted label y. Let k be the number of elements in the output signal y, the cross-entropy loss is formally expressed in equation (2.20).

$$-\sum_{i=1}^{k} y_t log(y) \tag{2.20}$$

Equation (2.20) measures the entropy between two distributions, entropy being the loss of information between two different distributions. The equation assumes the values within y are elements of a probability distribution such that the sum of the elements within y equals one. The negation of the logarithm is used to bring the elements of the sum to a positive scope. This ensures the loss will be positive, since all elements in y and y_t are in the interval betrween zero and one. The sigmoid function may also be used in association with cross-entropy, as the values will be transformed into the zero-to-one interval. Other activation functions such as tanh and ReLU run the risk of bringing elements in the sum to the negative scope or undefined (as the logarithm is undefined for values less than zero) [45, 46].

The computed loss between the data labels and the predicted labels is then used by the optimizer to change the learnable parameters. The backpropagation algorithm calculates the partial derivative of the computed loss with respect to each learnable parameter. The collection of the derived parameters are called the gradient. The gradient is scaled by the learning rate (usually a value less than 0.01) defined in the optimizer, this allows for small changes to each parameter which allows the model to approach the minimum of the loss function at a speed proportional to the learning rate. Using the gradient in this way is called gradient descent and it is the common method of learning all optimizers use. Adam is an optimizer introduced by Kingma and Lei Ba [47] which has proven to be efficient in contrast to other optimization methods such as Gradient Descent, AdaGrad and RMSprop. Adam uses an adaptive learning rate to approach the minimum of the loss function. Adam requires four different parameters for the algorithm to run. These are the learning rate α, stochastic decay rates β_1 and β_2 and a small constant ε used to avoid division by zero for the update equation. The algorithm described by Kingma and Lei Ba also require two vectors m and v used to describe the "moment" of the gradient and the squared gradient respectively (initialized as zero vectors). The gradient update at timestep t is expressed by the following formulas:

$$\begin{aligned} t &= t+1 \\ g_t &= \nabla L_t(W_{t-1}) \\ m_t &= \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t \\ v_t &= \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t^2 \\ m'_t &= \tfrac{m_t}{1-\beta_1^t} \\ v'_t &= \tfrac{v_t}{1-\beta_2^t} \\ W_t &= W_t - \alpha \cdot \tfrac{m'_t}{\sqrt{v'_t}+\varepsilon} \end{aligned}$$

Each vector at timestep t uses the values for the "moments" at the previous timestep $t-1$. Each "moment" is then update using the previous timestep and the stochastic decay rates. Each "moment" vector is normalized by element-wise division of the "moment" vectors and their respective decay rates to the power of t. β_1 and β_2 are initialized to be 0.9 and 0.999 respectively, this ensures that subtraction by one will maintain the relative scope between the "moment" vectors and their corrected counterparts. The parameters themselves are then updated my subtracting the current parameters by alpha multiplied by the decay rate (calculated by the fraction of the "moment" vectors with the ε parameter added to the denominator). The "moment" vectors then give each feature a unique learning rate which improves training.

Chapter 3

Data Exploration

Deep learning models require tremendous amounts of data, to ensure the model works well; the data must also possess sufficient characteristics to approximate the sample population from which it was extracted. To satisfy this requirement we examine the data in attempt to remove outliers and determine whether the data is sufficient for classification. Moreover, tumors have been shown to be heterogeneous [48], which may be problematic for a classifier as heterogeneous samples lack in shared characteristics. In this chapter the data available to the project is examined in greater detail; details for how the Raman spectra were prepared is given to document the preprocessing of spectra for future use. First we describe the mathematical representation of the samples. Since the number of samples is too small to use in a deep learning model, we explain how each sample may be separated into individual spectra; this separation yields a drastic increase in the number of available training examples. We then explain how to balance the data; an unbalanced dataset would likely introduce bias in machine learning models, rendering their desired predictive capabilities uncertain. We achieve this by duplicating underrepresented sample classes in the dataset. Furthermore, this balancing is performed to maintain majority and minority classes, thus retaining some distributional information from the original dataset. The quintessential purpose of this chapter is to analyze the data using different methodologies for detecting spectra which has been altered due to non-tumor material affecting the spectra (henceforth denoted outliers). As a starting point for this analysis, we have defined the frequency criterion. The criterion is used to separate spectra from outliers in each sample. The outliers captured by this criterion is compared to outliers detected by other outlier detection techniques such as the Standard Deviation Test (SDT) and the Interquartile Range Method (IRM). The unsupervised machine learning methods K-means clustering and Hierarchical clustering are also used to detect outliers in each sample. The results from each method are then compared in effort to select the one which best separates outliers from the tumor spectra after which the data is curated by that method.

Another point of interest in this project is the identification of representative frequencies within the spectra. Each spectra belongs to a tumor which can be categorized by six different classes. The hypothesis states certain frequencies should be sufficient in determining which class the tumor belongs to. Feature selection is used for this purpose, representing each frequency within the spectra as distinct features. However, in order to extract such features reliably, the data must be devoid of outliers. Should outliers exist within the dataset, the features given by the methods used will be influenced and may yield conflicting results with the ground truth. To prepare for this the data is plotted for visual inspection using various methods to be discussed in the following section on feature selection. It is confirmed by the provider that the majority of samples include outliers. These outliers are influenced by a variety of other materials found on the tissues surface e.g. spectra of blood drops, plastic which may be reflected form underneath thin tissue or necrotic tissue, which is shown to affect the spectral signal significantly. Using the extracted features, a model can be trained on the data faster which can be essential many machine learning methods require a significant amount of computing resources, data and time. The features are extracted before and after the removal of problematic spectra. This is done to compare the impact of the outliers in feature selection. We expect the features vary significantly different subsets of the data available. Feature selection is not a good strategy for machine learning in this particular case if the features extracted from different subsets vary tremendously. In this case there are clear signs the machine learning models will fail to emphasize features which best represent the whole dataset; which is problematic since the model should have sufficiently accurate predictions for unseen data.

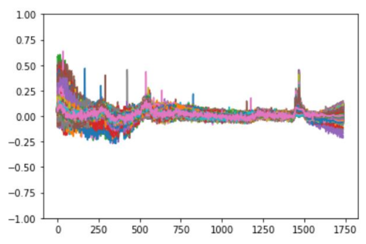
3.1 Data Representation

The data consists of the Raman-spectra extracted from the tissue of glioma tumors from 45 patients. Multiple samples of tissue were extracted from the same patient in some cases, yielding several samples for the respective patient. To maintain separation among the patients, the samples are sorted by their respective patient of origin. This is necessary due to the heterogeneity of each tumor. The data will be separated into three separate datasets. These sets are referred to as the training set, the validation set and the test set. Due to the expected heterogeneity, all datasets will consist of unique patients to avoid scenarios in which the model overfits to a patients tumor sample. This structure also allows for easier handling of the number of patients in each sample class, allowing for analysis on each class separately from the others.

There is also large variation with regard to the sample shape within the data. Each sample is a 3-dimensional array of shape $(w, h, 1738)$ where w and h are the width and height of the sample, respectively. This formalization is necessary, as width and height have non-zero variance among different samples. The shape is a result of how the tissue was scanned. In each case the tissue was placed inside the Raman spectrometer and scanned successively from side to side. This makes it possible to display each sample as an image, by substituting the third dimension (denoted above by 1738) by a color value denoted by which class the spectra belongs to. The number 1738 is constant through all samples and represents the length of a modified Raman-spectra. The spectra has been modified by the data provider; the data consists spectra resulting of a linear mapping from the extracted spectra. This was done to omit unnecessary frequencies and allow focus on parts of the spectra which we believe is sufficient for this project. Furthermore, each element inside these arrays is a real number without clear bounds. The largest absolute element found within the complete dataset is 79427.0625. Some values are negative, which is confirmed by the providers to have significance for the projects purpose. The project aims to prepare these spectra for use in machine learning methods. Predictions should be performed on individual spectra extracted form the dataset. We choose this strategy since each spectrum is independent of all other surrounding spectra, ideally sharing in some characteristics from the other spectra belonging to the LGm class we wish to predict i.e. one spectrum should describe which LGm class the sample belongs to. There are six distinct LGm classes as defined by Ceccarelli et al. [9], denoted LGm1 - 6. The model will take as input one vector of shape $(1, 1738)$ and produce one vector of shape $(1, 6)$. This strategy is inspired by Liu et al.[15], who managed to get satisfactory performance by training a model on raw spectra i.e. spectra without preprocessing or outlier removal. Restructuring of the data to represent all samples as a list of spectra rather than 3-dimensional matrices yields a dataset with more than $300,000$ datapoints, which better suits deep learning tasks.

Initially, we choose to examine each sample by plotting the spectral lines in a plot. This allows us to examine the general shape and visually deduce if any common patterns are present in the samples and identify problematic samples. As expected from analogue measurements, significant amounts of noise are present in each plot. Despite this, many spectra share in some general characteristics with a few spikes appearing on an mostly flat spectral line. An example of the plots created for this examination be seen in Figure 3.1.

In Figure 3.1 is shown two collections of spectra, the spectra belonging to sam-


(a) Spectra from patient HF-1293. The rest of the samples available share in this pattern with some deviations

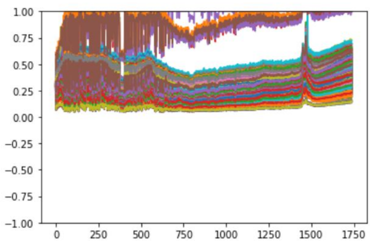

(b) Spectra from patient HF-1887. The frequencies tilt towards the upper part of the plot. The example is decidedly removed from the analysis.

Figure 3.1: Examples of samples drawn from the data, HF-1293 display a common pattern across all samples, HF-1887 is removed due to the skewed baseline. The range of the values are normalized between -1 and 1 to easier display the spectra.

ple HF-1293 are shown in Figure 3.1 (a) and spectra from sample HF-1887 are shown in 3.1 (b). The spectra form HF-1293 follow a general pattern visible in the vast majority of sample plots. Sample HF-1887 is an example of one sample which we deem too sporadic for this project, we remove it due to the skewed baseline of all spectra in the sample after confiding with the data provider, who agrees with our decision. After visual examination, we can confirm that some characteristics are present, but they are not sufficiently different among the different LGm classes to be classified by a human being in this case. Machine learning is needed to detect precise differences among the samples.

Another reason to remove samples are due to their size. The analysis we aim to perform requires considerable computational power and memory space. Many of the samples available have a manageable size, as they consist of approximately 3600 spectra i.e. width and height are approximately 60 and 60 respectively. In case the samples are too big for use we may simply extract a random sub sample from the larger sample and analyze those. But applying that method at this stage would potentially ruin the form of the samples which is used to evaluate the outlier detection methods in a later section. Fortunately only one sample suffers from this problem. Sample HF-3097, shown in Figure 3.2, shows a concerning number of spikes in contrast to the other samples and is the only sample which require considerable memory space (the number of spectra exceed 40000).

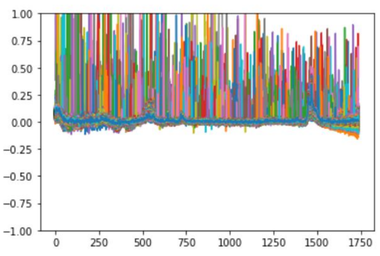


Figure 3.2: Sample HF-3097 from LGm2, the spectra spikes are extremely sporadic. The range of the values are normalized between -1 and 1 to easier display the spectra.

The values inside the spectra are also considerably bigger than the values found in the other samples. The methods SDT and IQM require statistical constants be extracted on the entire dataset, including this sample would affect the standard deviation and the mean of the frequencies tremendously (even if the constants were evaluated within the scope of the sample itself). K-means clustering is also a unsupervised machine learning model which can be trained on data before predictions are performed on unseen data. Since the aim is to use the training set for training the model, inclusion of the sample would affect the performance considerably. If the sample is put in the test set for testing, the noise in the signals would affect predictions considerably, which would affect testing accuracy due to what appears to be noise in the frequencies. To avoid these statistical issues and to reduce the computational requirements, the sample is discarded following the data providers approval.

3.2 Organization and Balance

The model will, as a consequence of it's learning-algorithm, become biased towards certain predictions if presented with frequent examples of a particular class. As the model encounters frequent training examples, the connections which produce such predictions will strengthen. Over exposure to examples of a certain class will force the model to associate features with that class, redirecting focus from classes for which that feature could be significant. This is why it is important to train models on balanced data. Having a balanced dataset means the number of examples in each class are uniform in the entire training set. Training on a balanced dataset also avoids bias towards data distribution, where certain classes may be predicted more frequently than others simply because the distribution was used in training. The initial data suffers heavily from this problem. Training on the data would result in a model which produces frequent predictions for the majority class (the class which has the most spectra) and perform inconsistently for the other classes. The data distribution is shown in Table 3.1.

Class	LGm1	LGm2	LGm3	LGm4	LGm5	LGm6
# of samples	5	11	4	10	11	4
# of spectra	37319	71846	31931	50660	62256	20176
percentage	14%	26%	12%	18%	23%	7%

Table 3.1: Table showing the distribution of data in the initial dataset after removing the problematic samples. The number of samples are displayed on the first row, the number of spectra in each class is shown on the second row. The percentage of the entire dataset is shown on the third row. The majority class is LGm 2 and minority is LGm 6. Classes LGm 1, 3 and 6 must be expanded to balance the data.

Table 3.1 shows the per class separation in the data, the header row shows the labels of each class. The first row shows the number of samples belonging to each class, these are the tumors which will be analyzed. The second row displays the total number of spectra across each class; these must be considered for balancing. Note the equal amount of samples in LGm3 and LGm6, but the difference in number of spectra within them. This is due to the varying size of all samples drawn from the tumors. Some samples share the same size, however the important fact is that the samples lack a uniform shape, which must be considered during the analysis. The last row shows the percentage each class makes of the entire dataset. Initially LGm2 is the majority class while LGm6 is the minority, consisting of only 7% of the entire dataset.

Before the data is balanced, the testing data is selected and separated from the rest manually. This is done by separating at least one patient and all their samples from the rest of the data. This way it will be possible to test if the model develops bias towards the patients in training and if the patient samples are heterogeneous with respect to the other samples of the same class. Samples are chosen with the criterion that approximately 30% of each class is represented in the test set. Balancing the classes which contain less elements by a factor larger than or equal to two compared to the majority class (LGm2) is done by repeating the spectrum in each sample by that factor. This methods does not perfectly balance the data to have a uniform distribution of classes, but it does make the underrepresented classes frequent enough to circumvent the issue of unbalanced datasets. The multipliers for each class are chosen so as to retain the majority class i.e. LGm2 will remain the majority class after balancing is done. There is also a need for the validation set, the validation will be used to monitor the models performance on unseen data during training. The samples for this set are chosen such that one sample from each class is allocated to the set. The separation of the entire dataset into three distinct sets is arbitrary and so can be performed automatically. However, we choose to perform this separation manually to maintain consistency in this analysis. The resulting separation is shown here as it is the training set from which the features are extracted in this project. We discover that the results will differ slightly depending on which samples are chosen for which set. The distributions of all datasets are shown in Table 3.2.

Class	LGm1	LGm2	LGm3	LGm4	LGm5	LGm6
# training	17689	47602	37557	30180	33396	9376
# validation	3600	4096	3600	4096	4096	3600
# test	14945	20140	11296	16384	24764	7200

[htb]

Table 3.2: Distribution of the three datasets

Table 3.2 shows the distribution of the different datasets used in this project. The training set is then balanced exclusively. This is not required in the test set or the validation set, since they will have no direct effect on how the model is developed through training. It is also important that the validation and test sets are not uniform, since it will prove whether the model can generalize to different prediction distributions. The training set is balanced by replicating each spectra in every patient of the classes which are under-represented. The number of sample replications per class can be computed by the following method. Let *m* be the LGm class which contains the majority of spectra in the set and $|LGm_n|$ be the number of spectra in LGm class *n*. The number of replications for each class can then be computed by $\lfloor \frac{|LGm_m|}{|LGm_n|} \rfloor$. The distribution of the training set after applying the replication method is shown in Table 3.3.

Class	LGm1	LGm2	LGm3	LGm4	LGm5	LGm6
# training	35378	47602	37557	30180	33396	46880

Table 3.3: Distribution of the testing data following balancing

As is shown, the data does not have a uniform distribution, LGm2 is still significantly larger than LGm4. The important thing is that distribution is in better balance relative to the original distribution. We believe this approach is good enough and proceed to the next stage.

3.3 Analysis

Following the balancing, the first step in the analysis is to find the frequencies which best describe the data with respect to the methylation-types. Each number in the spectra is a frequency at which the scattered light is gathered. This light is expected to be sufficient for predicting the methylation-type of the tumor-tissue. It is speculated that, to sufficiently categorize the spectra into the methylation-types, only certain frequencies are required. For this reason, the best features are extracted with SelectKBestfeatures [31] which is given the f-classif method for ranking the features in order of their significance using ANOVA. Following this step, we may pick any arbitrary number of features by extracting them in the order given by f-classif. The 70 best features are extracted from the training set in which there are outliers present still. The features are displayed in Appendix A. The extracted features show that regions of interest do exist on the spectra. This can be seen by the integers which have a successive difference of one, suggesting that the region of interest exist somewhere in specific parts of the spectra. It is worth noting here that the features selected might be correct provided the amount of outliers is sufficiently small to be ignored by the feature selection method. Due to this uncertainty, the data will be separated from the outliers and feature selection will be performed a second time at the end of this section.

All data is subject to this analysis as all samples must be curated before the model can be trained. In the methods where statistical constants must be calculated, the training data is used, this avoids bias towards the other datasets whose primary purpose is to evaluate the model. The goal of the analysis is to find a uniform criterion which each spectrum must fulfill to be considered *clean*. Spectra which fail to satisfy this criterion will be discarded form the project entirely. In this section the methods of analysis used are described and their results examined. The section begins by examining the frequency criterion, which is a confirmed criterion all spectra must satisfy to be considered "clean". This criterion was provided by the data provider. SDT and IRM are compared to the frequency criterion for validation, these are deterministic methods that rely solely on the values found within the data and are commonly used to detect outliers in data. *K-means* clustering and Hierarchical clustering are then performed on the data in attempt to capture potentially complex patterns within the data. These methods are specifically designed to allow for grouping of data based on the similarities between data points within the dataset. The section ends by selecting the method which best detect outliers, this method is then added to the preprocessing stage i.e. following this project, all samples must be curated using this method before they can be used by the machine learning model.

3.3.1 The frequency criterion

The frequency criterion is a criterion specified by Adrian Lita for separating outlier spectra from tumor spectra. The criterion states: "should any value of frequencies between 1463 and 1473 of any spectrum be below 5000, then that spectrum is defined to be and outlier". Given that this criterion is defined by the provider, and the lack of knowledge regarding where outliers may be positioned on the samples, we choose this criterion as the starting point of this analysis. The extracted features in Appendix A include parts of the range on which the criterion is based, which is promising. The range being present in the extracted features can also mean that the outliers work to influence the relationship between the spectra and the methylation types we want to predict. In which case the removal of the outliers are essential for building an unbiased model. We examine the results of the criterion to gain insight into where these outliers are positioned. The criterion is confirmed to miss some outliers and so it mainly functions as an initial approximation of the areas where outliers are present. An example of this is in sample HF-2849 of LGm3 shown in Figure 3.3.

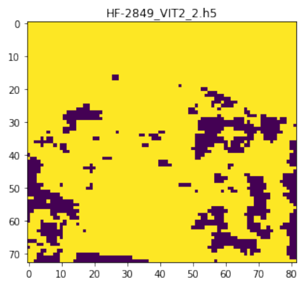


Figure 3.3: Sample HF-2849 from LGm3. The sample is confirmed to have necrotic tissue present in the upper part.

The criterion is satisfactory for capturing areas in the lower parts of the sample. However, the upper part of the sample is guaranteed to have spectra from necrotic tissue which is unreliable for describing the LGm class of the tissue. It appears the criterion is well suited for detecting liquid material on the tissue, since many samples show smaller spots of interest and fail to completely capture larger areas of outliers. This is evident in Figure 3.3, as multiple small spots are detected. This is one of several examples for how the criterion fails to detect all outliers reliably, but in the majority of the samples, the outliers are detected sufficiently well. One such case can be seen in sample HF-868 displayed in Figure 3.4.

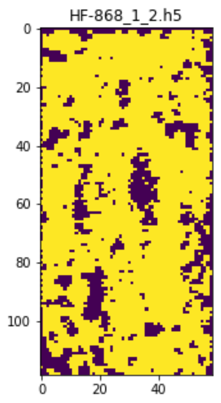


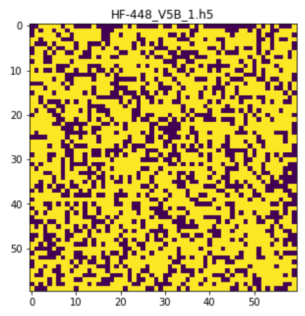
Figure 3.4: Sample HF-868 from LGm1. The areas detected are confirmed outliers.

According to the data provider, the outliers present in HF-868 are sufficiently captured by the frequency criterion. The outliers form as small, sporadic areas on the surface of the tissue. This shape is what the outliers are expected to have in the samples which are known to include them. However, since the frequency criterion fails to capture outliers in certain samples (possibly due to the material information of those outliers), we explore changes made to the criterion e.g. making the interval greater (checking frequencies ranging from 1458 to 1478) and checking for frequencies which are below 20000 within the interval the criterion concerns. The results vary greatly from the original criterion. Though some of the spectra originally ignored by the criterion now become visible, not all regions are sufficiently captured. This indicates the necessity to use other parts of the spectra which appear to be relevant for outliers. The spectra are too big for further manual inspection, which greatly motivates the application of machine learning methods. The frequency criterion will, however, serve as a ground truth in this analysis. Going forward in this section we aim to evaluate all following methods by comparing them to the results of the frequency criterion. Examples which best reflect the methods performance in outlier detection will be displayed for comparison with this criterion. This way, we have some knowledge about where outliers are detected. While the criterion is insufficient in detecting all outliers (as demonstrated in this subsection), the general patterns discovered here must be present in the results yielded by the other methods. If the method under analysis fails to produce results corresponding to the frequency criterion, we opt to disregard that method. The desired method optimally produces similar results as the frequency criterion and finds more areas on the samples where we know outliers are present. Ideally, the methods also aid in discovering new outliers.

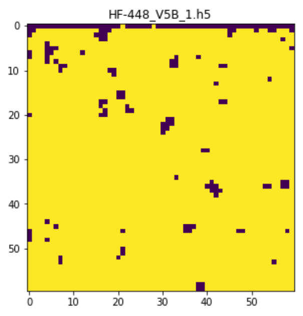
3.3.2 The standard deviation test

The standard deviation test is a test by which the data is centered around the mean and given a standard deviation of one. With this setup, outliers are defined as points which are separated from the mean by three standard deviations or more. We measure the mean and standard deviation on each frequency from the unbalanced training-set; we ignore the balancing to avoid changes to the mean and standard deviation which the balancing helps produce. The values are then used to standardize the spectra belonging to each tumor. A spectrum is deemed to be an outlier if the number of frequencies in that spectrum exceed an arbitrary value. We approximated the value by performing the test once while monitoring the average number of frequencies which lie three or more standard deviations from the mean. In any given sample, each spectrum includes on average 111 frequencies which fail the test. We specify that a spectrum which fails the test on more than 111 frequencies is deemed an outlier and removed.

Using this test, many small areas are detected in each sample, it suggests there is something present in those places, but they do not possess a clear shape by which we can decide whether to discard them or not. An example of such a sample is shown in Figure 3.5.



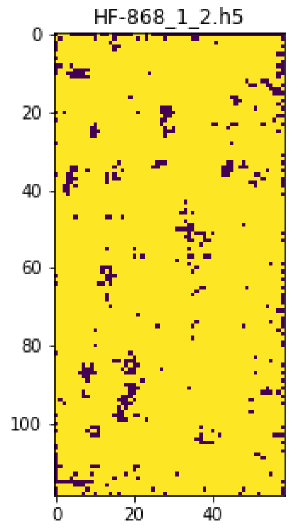
(a) The result of SDT for sample HF-448, Several points are labeled as outliers



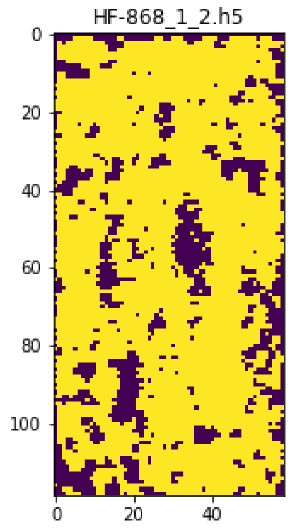
(b) The frequency criterion outliers for HF-448

Figure 3.5: A comparison between the SDT and the frequency criterion for sample HF-448.

SDT does not produce results similar to the frequency criterion, some spectra do correspond among the results, but the amount of outliers falsely labeled as outliers by SDT is troublesome. Roughly 50% of the samples in the entire dataset yield similar results with SDT. The method would as a result of this, discard too much from most samples, depraving the dataset of a great number of tumor spectra. Some areas are formed around the individual spectra, suggesting the presence of an unknown material, but the sporadic points in the surrounding area make it unclear where that material begins and ends. Furthermore, we must develop a criterion for which points to discard. Such a criterion would need to distinguish between sporadic points which are miss-classified and areas of real outliers. However, within some samples there are areas of outliers correctly labeled, but those areas are not sufficiently defined. One sample with this result is shown in Figure 3.6.



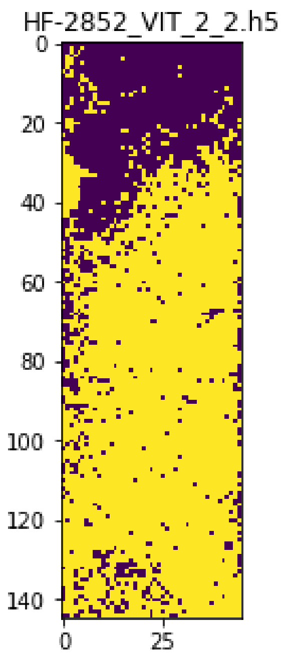
(a) The result of SDT for sample HF-868



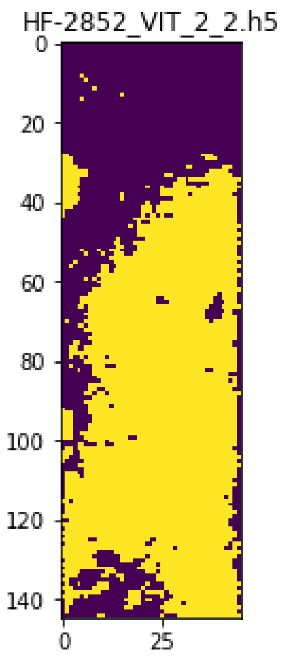
(b) The frequency criterion outliers for HF-868

Figure 3.6: A comparison between the SDT and the frequency criterion for sample HF-868. Many spectra correspond to the frequency criterion.

Sample HF-868 is less sporadic compared to HF-448, the outliers are instead formed around common points of interest which strongly suggests outliers are present in that area. The lack of definition in each area however, is not sufficient. The outliers must form concrete shapes with clear definition. The sample suggests outliers are present in the different areas, but all points are not present to make the shapes as defined as they are by the frequency criterion. The sporadic results show that this method is insufficient to detect the majority of outliers in all samples. It must be noted that despite this underwhelming result, some outliers are detected, suggesting further changes to the method could yield better results, though greedy application is not going to work for all samples. Throughout all the six LGm classes, the method yields the best results for LGm6 where it shares many patterns with the patterns produced by the frequency criterion. Another example of the methods promise is in sample HF-2852 of LGm3 shown in Figure 3.7.



(a) The result of SDT for sample HF-2852



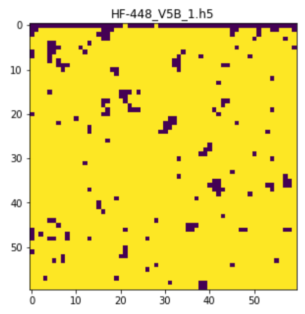
(b) The frequency criterion outliers for HF-2852

Figure 3.7: A comparison between the SDT and the frequency criterion for sample HF-2852.

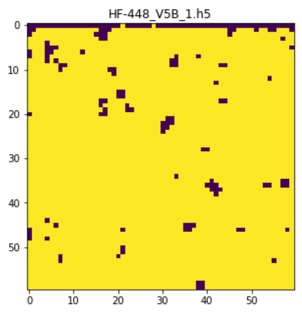
Sample HF-2852 has a large area in the upper part which consists of necrotic tissue. The spectra of that tissue differs from other spectra sufficiently well, allowing the method to distinguish between tumor and non-tumor with considerable accuracy. This is due to the amount of otherwise healthy tissue present in the sample. The methods inability to capture all outliers is also present despite this, as it allows several smaller spots in the area of the necrotic tissue to be classified as healthy tumor tissue. The area at the bottom of the sample is filled with outliers according to the frequency criterion. These results suggests that the method is best used in detecting necrotic tissue, but not in context of detecting other kinds of outliers.

3.3.3 The Interquartile range method

Similar to SDT, IRM is a purely statistical analysis method which detects outliers in terms of which percentile the spectra belong to. The 25th and 75th percentiles are calculated on each frequency for the entire training set. Like SDT, this method yields a varying amount of outliers for each sample. We instead define the allowed number of outlier frequencies within one spectra to be equal to the average number of outlier frequencies within the analyzed sample. Many regions are better represented by IRM, showing well defined areas where outliers are clearly present. The amount of individual spots are less frequent which shows promise in the method, as e.g. blood is expected to cover a larger area if present. The improvement from the standard deviation test is seen in Figure 3.8.



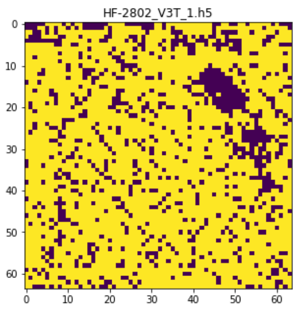
(a) The result of the interquartile range for HF-448, Several well defined areas are separated form the majority of spectra



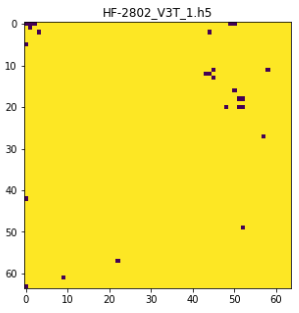
(b) The standard deviation test for sample HF-448

Figure 3.8: A comparison between IRM and the frequency criterion for sample HF-448.

SDT failed to separate outliers adequately in sample HF-448 while IRM is better suited to detect the outliers in this sample. The upper part of the sample has a well defined line where the sample is presumably cut, meaning the plastic underneath the tissue might be visible. We also note that many outliers detected by IRM also appear to be captured by SDT. While the areas are hard to distinguish among all sporadic spots, the larger spots in Figure 3.8 (a) seem to have some trace in Figure 3.5. The correspondence with the frequency criterion in Figure 3.8 (b) further shows the methods capability in contrast to SDT as the method seem to possess greater ability in defining the areas where outliers are present. There are still some spots appearing randomly around the sample surface which would ideally be ignored, but while we have some confirmation on where outliers are present in certain samples, we do not know exactly where the outliers are. The individual points being labeled as outliers suggests the method struggles with the same issue SDT suffers from. Though it appears to be less severe in all samples belonging to LGm1, there are still a considerable amount of spots through all samples within the class. Especially LGm4 have samples for which both methods perform poorly, with a considerable amount of sporadic spots appearing in some samples when applying IRM and a significant lack of spots when applying SDT i.e. None of the methods works sufficiently well to detect the outliers. It should be noted that both methods detect outliers in areas which the frequency criterion also produces. But the areas are not corresponding well in either method. This issue is apparent in sample HF-2802 of LGm4, shown in Figure 3.9.



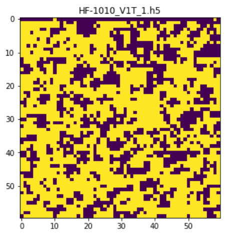
(a) The result of the interquartile range for HF-2802, some larger areas are detected with sporadic spots around the entire spectra.



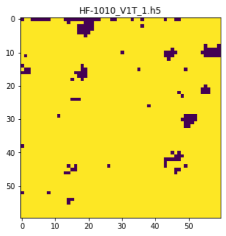
(b) The result of the standard deviation test for HF-2802. Few outliers are detected

Figure 3.9: A comparison between the interquartile range and the standard deviation test for sample HF-2802. In comparison, the interquartile range locates better defined areas than the standard deviation test, but many sporadic spots are present.

In Figure 3.9, the results of IRM and SDT are compared on sample HF-2802. Both methods yield poor separation between the outliers and tumor spectra. IRM is capable of detecting a large mass of some material on the sample, however, many sporadic spots appear around it, the majority of these spots are tumor spectra which would be discarded by the method. In stark contrast, SDT struggles to detect anything in the sample, only yielding small spots in the larger areas found with IRM. Fewer samples suffer from the stochastic results present in SDT, however all samples are not strictly improvements from SDT. One such example is sample HF-1010 belonging to LGm2, shown in Figure 3.10.



(a) The result of the interquartile range for HF-1010



(b) The result of the standard deviation test for HF-1010.

Figure 3.10: A comparison between the interquartile range and the standard deviation test for sample HF-1010, LGm2.

The comparison in Figure 3.10 show one of few samples where IRM fails to sufficiently separate outliers from tumor spectra. A considerable number of spectra are falsely labeled as outliers, and those spectra seem to form many smaller areas without sporadic points in the surrounding area. In this rare case, SDT exhibits sufficient separation of the outliers, as the detected areas correspond well with the the position of known outliers found by the frequency criterion.

Few samples follow the same conclusion however, these varying results show clear signs that neither of the methods are suitable to perfectly rid each sample of outliers. We therefore dismiss them from the analysis, while keeping the results for comparison with the other methods and continue the analysis by analyzing unsupervised machine learning methods for outlier detection.

3.3.4 Hierarchical clustering

The next method for outlier detection is hierarchical clustering, we choose to utilize agglomerative clustering as an arbitrary choice. Due to the algorithms demanding time complexity and memory constraints, we choose to analyze each sample separately to avoid memory errors. Each sample must be given one unique model which then divides the data into the number of clusters specified. This means there will not be a universal model designed to separate all samples. We compare the results of different distance metrics and choose to utilize Euclidean distance to measure distance among the clusters. This conclusion is due to the uncertainty in vector shape and angles on which Cosine similarity is dependent. The Manhattan distance would be a better choice compared to Cosine similarity. However, Euclidean distance magnifies long distances which should aid the algorithm in selecting clusters for agglomerative merging. Following the choice of distance metric we examine the linkage criteria available. In each case, the algorithm is set to run multiple tests where it separates the data into different amounts of clusters. We choose to initialize five different models to compute between two and seven clusters respectively. Due to the deterministic nature of the algorithm, the results in one cluster will be present in the other clusters as the the number of clusters increases.

Single linkage merges clusters by way of merging those clusters which posses points with minimal distance. Should the spectra within the samples be significantly "far apart" in the data, the criterion might start producing many unique clusters in a "chain". If the number of clusters are sufficiently small, the number of cluster separations will be few, and few spectra will be allocated to those clusters. This can result in few separations for some samples, which will render the method unusable for our purposes.

The criterion yields clusters which appear as individual points in the spectra and fail to detect areas where known outliers are present in the samples, suggesting the aforementioned flaw is present in this methodology. This is apparent as we see few cluster areas form in any of the initialized models, even if we allow the method to use more than two clusters. All different cluster models fail to separate the outliers and instead separate a small number of points. An example of this phenomenon is displayed in Figure B.2.

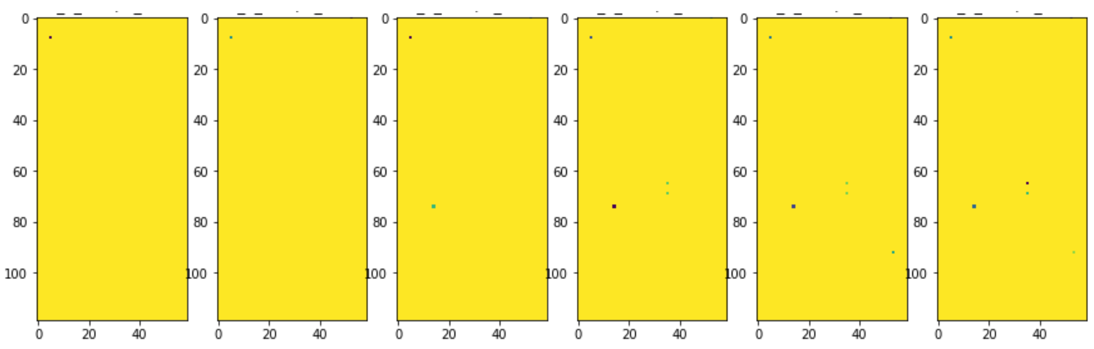


Figure 3.11: Single linkage on sample HF-868 from LGm1, The test fails to detect any outlier areas. Leftmost image is the result of the model computing two clusters. The number of clusters increase towards the rightmost image.

The models fail to detect any areas where outliers are present, only yielding an insignificant number of outliers in the entire sample. Models computing more clusters also fail to find significant areas and smaller spots appear as the number of clusters increase. As the number of clusters increase, several of the outliers seem to belong to their own clusters, which is a sign of the clusters being computed in a "chain" as previously stated. Due to this unsatisfactory result, Single linkage will not be used as linkage criterion in this project.

Average linkage yield superior results compared to single linkage since some areas become apparent as we increase the number of clusters. Moreover, the criterion does not have a set number of clusters which is guaranteed to include all outliers for all samples. For certain samples the outliers are visible when forcing the algorithm to agglomerate to two clusters and others only show them once five or more clusters are allowed. It is possible to use the criterion for discarding the outliers if the majority cluster is preserved when computing seven clusters, while the rest of the spectra are discarded. However, this would not remove all outliers and some problematic samples in LGm3 would have the majority of outliers preserved. The improvement from Single linkage is made apparent in Figure 3.12.

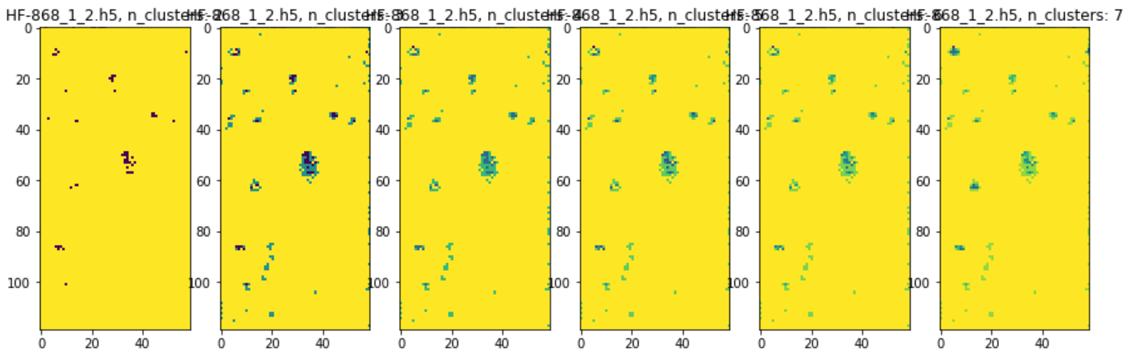


Figure 3.12: Average linkage on sample HF-868 from LGm1, Leftmost image is the result of the model computing two clusters. The number of clusters increase towards the rightmost image.

The model computing two clusters is capable of producing separations where the outliers are present, though it still misses the bigger areas. As the number of clusters increases, the areas become more defined and capture more of the outliers. Despite this, all outliers are not detected even with seven cluster computed. One area should be captured in the lower left corner of the sample. However, only smaller clusters form around that area. Capturing it sufficiently would mean adding more clusters, but the number of clusters which best capture the outliers for all samples is variable in this case. It is a clear improvement over Single linkage, but the criterion is still not sufficient.

Complete linkage merges the clusters which posses elements with the smallest possible maximal distance between them. The method produces similar results as average linkage, few areas with outliers are detected when fewer clusters are permitted. However, some outliers are present when computing two or three clusters. Computing seven clusters allows the algorithm to detect outliers which are also detected by the frequency criterion. Though these results are not strictly improvements from average linkage, the criterion does manage to detect certain sports of outliers with fewer clusters than average linkage in certain samples. The result of complete linkage on sample HF-868 is seen in Figure 3.13.

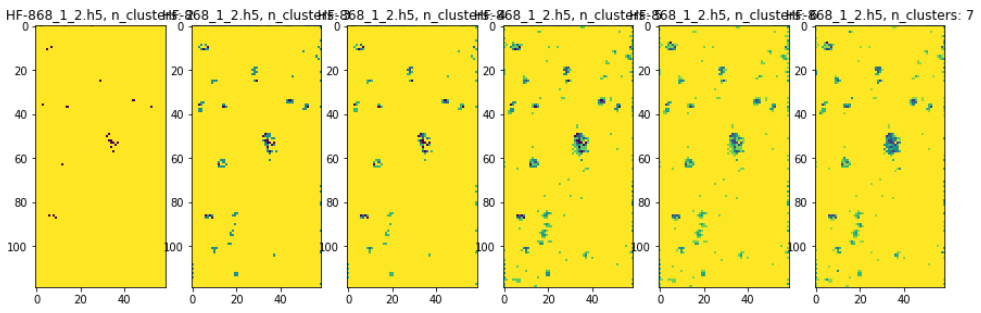


Figure 3.13: Complete linkage on sample HF-868 from LGm1. Leftmost image is the result of the model computing 2 clusters. The number of clusters increase towards the rightmost image.

As is evident in Figure 3.13, much like average linkage, complete linkage produces clearer areas as more clusters are computed. In HF-868, outliers are not detected earlier than average linkage, in fact, the outliers require more clusters to become apparent. A good approximation for the majority of samples appear to be the models computing seven clusters, if a few outliers are allowed to be included in the training phase of the model. The criterion's tendency to produce clusters earlier than average linkage is better displayed in Appendix B. Most samples display similar results; more outliers are detected overall with complete linkage in models computing two or three clusters than models using average linkage. Despite this, all five models fail to detect the outliers found by the frequency criterion. We therefore decide to disregard complete linkage for this project.

Ward linkage merges clusters which posses minimal variance between their respective elements and as such works well with Euclidean distance. We do stress that the clusters are merged by measuring variance among cluster elements and there is no guarantee the outlier spectra should share in characteristics which would result in low inter-cluster variance. Despite this lack of guarantee, the clusters form at the precise location of outliers detected by the frequency criterion. Allowing seven clusters produce a near picture perfect image of the biological tissue from which the spectra were measured. These results show that the algorithm is capable of organizing the spectra according to their visual information which aid us in understanding shape and state of the samples. The criterion works considerably well for outlier detection when compared to the other criteria analyzed thus far. It appears to be capable of detecting well defined outlier areas, the individual points which appear around those areas appear to form smaller groups which could very well be individual droplets of the same outlier material detected in the larger areas. The promising performance of ward linkage is well represented in Figure 3.14.

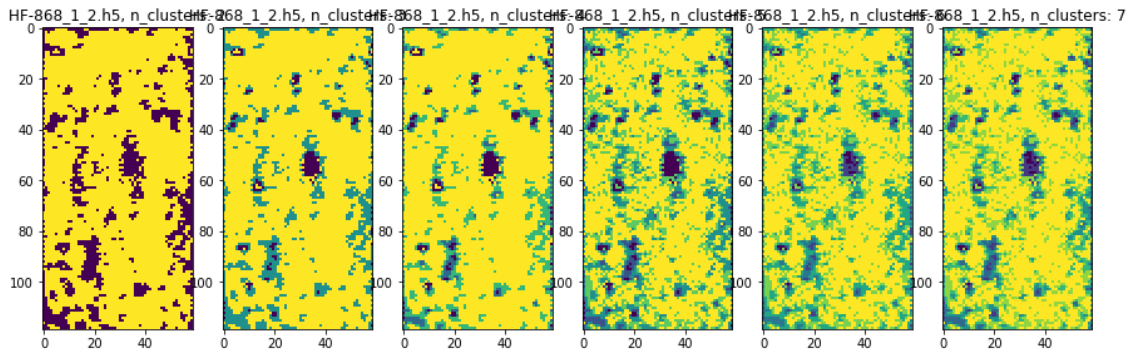


Figure 3.14: Ward linkage on sample HF-868 from LGm1. Leftmost image is the result of the model computing 2 clusters. The number of clusters increase towards the rightmost image.

The outliers identified correspond well with the frequency criterion, and as the number of clusters increase, the surface of the tissue starts to become visible by the different clusters. The model computing two clusters have near perfect resemblance with the frequency criterion. The models computing three and four clusters appear to perform subdivisions of the outlier cluster found in the model computing two clusters. This results tells us that the outliers appear to have significant differences from tumor spectra, which is a great result. We should expect similar results in samples which contain a great number of outliers. Once five clusters are allowed, spectra from healthy tissue starts to become apparent. These results appear to occur in the majority of samples.

The issue is finding a uniform criterion on which we can discard the outliers. We seek to define that criterion in terms of which clusters to discard from the samples following their analysis. Removing every cluster except for the majority cluster in the case where seven clusters have been formed would remove legitimate spectra which are suitable for training a model from all samples. In fact, there is no optimal choice in this case, as certain samples have their outliers sufficiently captured in a setup allowing for two clusters, while others show their outliers in arbitrary numbers of clusters. Furthermore, selecting clusters which contain close to 50% of the spectra would be undesirable in context of maintaining a sizable dataset. By visual inspection, we deem the optimal number of clusters to be three, since many outliers are present in this choice, though the problem is not completely remedied. In some samples, ward linkage separates areas not outlined by the frequency criterion when two clusters are computed. The outliers are instead detected when computing more clusters. This complicates the process of finding a uniform criterion, but the number of outliers which are be missed by our current paradigm is considerably small. In particular, sample HF-2485 suffers from this problem, the cluster results are shown in Figure 3.15.

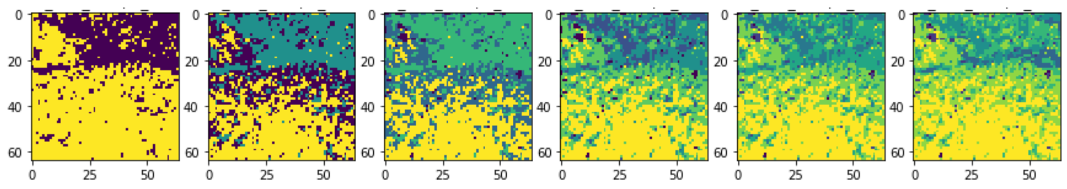


Figure 3.15: Sample HF-2485 from LGm5, clustered by Hierarchical clustering using ward linkage. From left to right, is shown the results of computing from 2 to 7 clusters respectively. The oultliers are present from image 4 and up.

The models computing two and three clusters actively avoid the concrete spots where the outliers are positioned. In the model computing four clusters, the outliers are detected as dark spots labeled as tumor spectra by the earlier models. The dark spots perfectly correlate with the position of outliers outlined by the frequency criterion. In the model computing seven clusters, the surface of the tissue starts to be replicated. Here, the outliers are displayed in stark contrast to the rest of the tumor material. In this case, we opt to continue with our paradigm of keeping the majority when computing four clusters. This will remove too much in a few samples, but will suffice should no other method yield better results. The alternative would be to use three clusters to preserve the othervise healthy spectra, but in the interest of training on data devoid of outliers, we opt for four clusters.

3.3.5 K-means clustering

We continue this analysis by performing K-means clustering in attempt to separate the outliers. We flatten the training set and reorganize it in random order to avoid bias towards recurrent LGm classes in the dataset. The examples are then used to fit five *K-means* models to compute two to seven clusters respectively as done with Hierarchical clustering. This method has the advantage of being trained on the entire training set whereas Hierarchical clustering possesses too great a time and space complexity which makes similar experiments impossible with our current hardware. Under this structure we may now compare the results between sample custers. We observe that the stochastic nature of the algorithm produce results of varying quality. In contrast to Hierarchical clustering, *K-means* do not produce clusters as subdivisions of previously seen clusters. This is due to the algorithm being computed several times with random initialization settings. The final result which the algorithm yields is the cluster state possessing minimal inertia compared to the other computations.

Contrary to all other methods of analysis, this method is able to capture small segments of the upper part of the sample HF-1293, shown in Appendix B. While the comparison of the same sample among the different models lose some credibility in this setting, the comparison between the model results among the different samples are promising. We find that the model in which two clusters are computed, the samples are divided in ways which corresponds with the frequency criterion, whereas samples where minimal amounts outliers are present the clusters seem to form around healthy tissue, which in turn create an image resembling the surface of the sample-tissue. In certain cases the clusters fail to capture known outliers but other models allowing for more clusters capture them sufficiently well. One example of this is sample HF-868 where the two cluster model fails to detect the relevant outliers but the model computing three clusters capture the outliers in near perfect resemblance to the frequency ctirerion. The clusters computed by the different models are displayed in Figure 3.16.

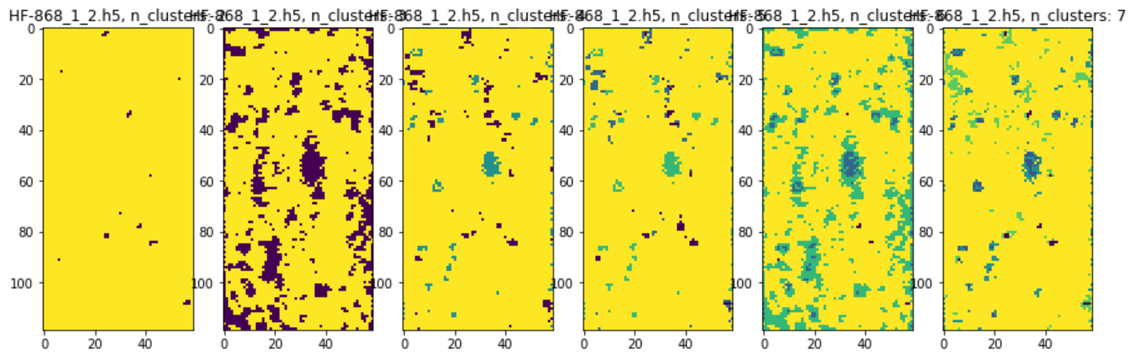


Figure 3.16: *K-means* clustering on sample HF-868 from LGm1. Leftmost image is the result of the model computing 2 clusters. The number of clusters increase towards the rightmost image.

The other models lose some of the outliers around the shapes present while still capturing the center of the shapes. The results of the frequency criterion returns in the model computing six clusters but this is again lost in the model computing seven clusters. Like ward linkage for Hierarchical clustering, there are some samples where the majority cluster is hard to evaluate, one such sample is HF-2544 which shifts the majority cluster between the model computing two clusters and the rest of the models. Due to the variety among the different models, finding a uniform criterion for detecting outliers is problematic. Many samples are such that the models steadily increase the number of outliers clustered, but this relationship is not constant through all samples, making it insufficient for use as criterion. The shifting of majority cluster in the aforementioned sample further complicates matters, since the majority cluster may not be capturing healthy tissue in some samples. For this reason we deem the method insufficient for separating outliers though we note the promise in capturing information about the tissues visual aspects, which makes the method comparable to ward linkage for Hierarchical clustering.

3.4 Post analytical feature extraction

Concluding this chapter we select the method of analysis best suited for separating outliers from the spectra and extract the features which best separate the data into the different LGm classes. The feature extraction is done by using the training set exclusively. Based on the analysis performed in this chapter we choose to apply Hierarchical clustering using ward linkage as linkage criterion. The samples within the training set are first rid of the outliers detected by performing the chosen clustering method. The model computes three distinct clusters of spectra for each sample separately and retain the cluster which contains the most spectra i.e. the spectra corresponding to the yellow colors in the figures shown in this chapter. We then repeat the procedure for feature extraction as done in the beginning of this chapter. We balance the training data, since the class distribution has changed due to removing spectra from each class. Feature selection is done with the f-classif method to score each frequency according to the ANOVA F-values. The 70 frequencies possessing the greatest scores after the method's application are then extracted. The index of the 70 most descriptive features of the training set are shown in Appendix A. In the set of features extracted from the curated data, twelve features are shared with the features drawn from the initial balanced training set. Of these intersecting features, none are the features which the frequency criterion concerns. This indicates that the outliers originally affected the Anova F-values computed by f-classif. The intersection of the features also show certain frequencies were correctly extracted from the non-curated training set. The features extracted from the curated training set are also positioned on different regions on the spectrum. The newer features show interest in frequencies starting early in the spectrum. More than half of the frequencies of interest are also positioned on the right end of the spectra, suggesting considerable amounts of information are located there. In case it becomes necessary to reduce the size of the data, these are the features we recommend be used instead. The spread across the entire spectrum suggests these features are descriptive enough to help machine learning models learn the sample-to-LGm relationship sufficiently well. In case the models struggle with training on these features exclusively, the algorithm may be easily adjusted to extract more features. We expect that 70 features should be enough for learning relationships within this data, especially since al-most all features lie in close proximity to other features i.e. the relevant frequencies appear to be grouped with other nearby frequencies. The most irregular features in this regard are the frequencies: 23 and 722. The fact these frequencies appear with no "neighbour" frequencies suggests the data would require more features to adequately be represented in this simplified form.

Chapter 4

Results

Following the curation of the data in the previous chapter, the number of spectra available in each data set decrease by approximately 25%. The distribution of the data remains the same, as the amount of spectra within each category retain the relative sizes to the other categories. In this chapter we present the proposed pre-processing methods for the data. We evaluate the data curation and pre-processing techniques by initializing a deep learning model and training it to predict the desired methylation types of each sample. The results are then presented and analyzed.

We proceed to make the data suitable for application with machine learning methodology. The first method of pre-processing is baseline correction. The baseline of a spectrum is the minimum value each wavelength frequency has. The extraction of the spectral signal from the spectrometer may, depending on the settings of the instrument, produce different minimal values for each frequency. This may lead to smaller spikes appearing in otherwise "flat" areas of the spectra. Baseline correction alleviates these issues by removing the excess baseline in all frequencies of the spectra, which simplifies analysis among the different frequencies. In context of the Raman spectra available in this project, the baseline by which the spectra are skewed is non-linear, and instead appear to have a polynomial baseline. This is apparent from the wave-like shape of the spectra where frequencies are minimal relative to the rest of the frequencies which portray spikes in the spectral wave. In turn some methods require a polynomial grade which is an approximation of the polynomial wave the baseline appears to originate from. This approximation of the polynomial grade may present issues, as it appears to require analysis of each spectrum in separation from the others. Nor is there any indication that the polynomial grade is uniform across all spectra in the data set. To avoid approximation of the polynomial grade we utilize the ZhangFit-method which has support for Python through the BaselineRemoval library. The method utilizes the adaptive iteratively rewheighted penalized least squares (airPLS) algorithm which iteratively approximates the baseline of the spectrum given without additional information regarding the structure of the spectrum [49]. For some spectra within the data set, the algorithm reaches the maximal amount of allowed iterations before the spectra are returned, This occurs in approximately 1000 spectra. However, this setback is ignored as it is less than one percent of the training spectra.

Machine learning methods are susceptible to spectra which have large variance among values. Frequencies inside the individual spectra vary tremendously, as spikes within each spectrum may differ from the baseline by a value of $10,000$ and more in many cases. It is clear that some method of normalization is needed to make the spectra applicable with machine learning methods. We opt to use *z-score* standardization and apply it to the frequencies within the training set. Standardization is performed as follows: Compute the mean and standard deviation of all frequencies through the whole training set. This results in two lists of length 1738 where the indices of those lists corresponds to the mean and standard deviation of respective frequencies. Each value of the spectra are then standardized by applying *z-score* standardization to each frequency by using the corresponding mean and standard deviation from the initialized lists. This method is preferred as it brings smaller values close to zero on each frequency while maintaining the larger values for the bigger spikes on the spectra within the training set. The features are however not sufficiently scaled down as many frequencies still exceed a value of 60 in some spectra.

In further attempt to scale the data into a range which works better for machine learning methods, we also apply normalization by the maximum absolute value. Following standardization, the maximum absolute value of each frequency frequency is measured and stored in a list. The data is then normalized by dividing each value in each frequency by the corresponding maximum absolute value stored. This retains the sparsity of the data while scaling all values down to a range between negative one and one.

We also define an augmentation method to better regularize the learning process. We train the model by selecting an equal amount of random spectra from each category for every batch used for training, this is done on the training set. Every batch will be balanced as a result which avoids the problem of unbalanced data sets. The batch is then augmented before the batch is passed to the pre-processor. Augmentation is performed by adding a skewed line to the batch of data which introduces a random, linear baseline to each spectrum in the batch. Each frequency in each spectra in the batch is then given additive normally distributed noise which is drawn from 1738 normal distributions, each generated from the means and standard deviations extracted from the data in the pre-processing step. This ensures the noise is within range of the usual values present for the respective frequency. To allow the model a chance to learn the real data patterns, this entire step is skipped randomly with 30% chance.

We create a deep convolutional neural network model and train it using the training set. The model architecture is based on the architecture presented by Liu et al. [15] and is four layers deep, consisting of two convolutional layers and three dense layers (one of which is the output layer). The input is always a one-dimensional vector consisting of 1738 elements (one neuron for each spectral frequency). The one-dimensional convolutional layers consist of 16 and 32 kernels respectively; both layers use the leakyReLU activation function and the kernel sizes are 21 and 16 respectively. Both layers use batch normalization and max pooling with a size of two. The model then flattens the result from the latter convolutional layer to allow for dense layers. Following the flattening layer, a dropout is provided with a dropout of 20%. The two dense layers before the output layer are sized 128 and 32 respectively and both use the tanh activation function. Both layers use batch normalization and the latter layer is followed by a dropout of 40% followed by Gaussian noise generated from a normal distribution with a mean of zero and a standard deviation of 0.15. The output consists of six neurons and uses the softmax activation function form a probability distribution of the input signal. The loss is then calculated using categorical cross-entropy which is optimized using the Adam optimizer with a learning rate of 0.003. All layer weights are initialized randomly from a normal distribution with a mean of zero and a standard deviation of 0.1. After compilation, the model consists of $1,742,374$ total parameters, of which 416 are non-trainable.

Several trials on the data suggests that the sizes of this architecture is arbitrary and provide little in terms of performance regarding the predictions of the model. We also create variations of the architecture, some of these variations are made with one type of activation function (the activation functions we test are tanh, LeakyReLU, ReLU and sigmoid). Changes to the convolutional layers are also arbitrary; strides and pooling sizes larger than two have been tried. The change to the convolutional layers drastically decreases the amount of learnable parameters in the model which makes the model faster to train. Before the model is initialized we set the random seed in numpy, os.environ and tensorflow to 6 (a value chosen arbitrarily). The resetting of the seeds is important, as it makes the random initializations predictable and, as a consequence, reproducible. The model is trained on balanced batches which consists of 600 spectra (100 spectra per unique category). The batch is generated and augmented before being pre-processed and given to the model where it is then trained on for a duration of 40 epochs.

The first test we perform is to deduce whether the data can be learned by the model or not. We perform this test by separating the training set into two smaller distinct sets. Before separation the entire set is sorted into a random order which scatters the spectra within randomly; the model is then trained on the first set and evaluated on the other. In this setting, the model manages to learn and generalize well to both halves of the training data. We manage to get these results by using the current pre-processing method discussed earlier in this chapter. Omitting the maximum absolute value normalization method when training in this setting prolong training drastically and prevent model generalization for the unseen part of the training set. While the accuracy for all spectra is insufficiently low (approximately 82% after training), letting the model train for longer periods of time appear to lead to better results. The overwhelming majority of correct predictions suggest the model is well suited for predicting the categories given all spectra from a sample. More spectra given to the model should increase the probability of correctness in the prediction. However this does not hold true for the validation data. The predicitions for the spectra within the validation data are sporadically scattered across many classes. LGm2 is severely underrepresented in the predictions generated by the model and few predictions appear correct through the majority of categories. LGm1 is the only category which is correctly predicted by the model. All other categories fail to be accurately represented, even the majority predictions fail as the majority of predictions are focused on other categories. The confusion matrix of the predictions for the training set and validation set are displayed in Appendix C. In hope of alleviating this issue, we rejoin the split training set into one complete data set and resolve to training on the entire training set while using the validation set for validating the performance of the model.

The model is fully capable of generalizing to the training set which is consistant with the previous result. The number of epochs must be increased further for the model to adequately learn the training data. We therefore increase the number of epochs from 40 to 80 which allows the model accuracy to reach over 90% accuracy on the training data. However, the accuracy is not reflected in the validation data. As in the setting with the split training data, the model achieves approximately 30% accuracy on the validation data. This result is constant through multiple runs in which we change the model in various ways i.e. change of activation functions, layer sizes and numbers and regularization layers such as dropout and Gaussian noise. By examining the confusion matrix made out of the predicted values and expected values we discover that the model tends to produce predictions mainly for categories LGm1, LGm4, LGm5 and LGm6. The model is seldom able to perform satisfactory predictions for more than two categories. The categories the model predicts seem to be a result of the order in which the model encounters spectra from the training set (which is dependent on the mini-batch parameter given to the model when training starts). The fact that the model can generalize to unseen training data but not to the validation data suggests that the spectra are heterogeneous among the samples which makes it hard to generalize to all categories. How to alleviate this issue and other possible sources this problem stems from is a speculative matter and elaborated upon in the next chapter.

Chapter 5

Conclusion

Bibliography

[1] C. S. von Bartheld, J. Bahney, and S. Herculano-Houzel, "The search for true numbers of neurons and glial cells in the human brain: a review of 150 years of cell counting," *Journal of Comparative Neurology*, vol. 524, no. 18, pp. 3865–3895, 2016.

[2] S. Jäkel and L. Dimou, "Glial cells and their function in the adult brain: a journey through the history of their ablation," *Frontiers in cellular neuroscience*, vol. 11, p. 24, 2017.

[3] O. Gallego, "Nonsurgical treatment of recurrent glioblastoma," *Current oncology*, vol. 22, no. 4, p. e273, 2015.

[4] F. E. Bleeker, R. J. Molenaar, and S. Leenstra, "Recent advances in the molecular understanding of glioblastoma," *Journal of neuro-oncology*, vol. 108, no. 1, pp. 11–27, 2012.

[5] S. L. Maas, E. R. Abels, L. L. Van De Haar, X. Zhang, L. Morsett, S. Sil, J. Guedes, P. Sen, S. Prabhakar, S. E. Hickman, *et al.*, "Glioblastoma hijacks microglial gene expression to support tumor growth," *Journal of neuroinflammation*, vol. 17, pp. 1–18, 2020.

[6] A. Dirkse, A. Golebiewska, T. Buder, P. V. Nazarov, A. Muller, S. Poovathingal, N. H. Brons, S. Leite, N. Sauvageot, D. Sarkisjan, *et al.*, "Stem cell-associated heterogeneity in glioblastoma results from intrinsic tumor plasticity shaped by the microenvironment," *Nature communications*, vol. 10, no. 1, pp. 1–16, 2019.

[7] K. Vigneswaran, S. Neill, and C. G. Hadjipanayis, "Beyond the world health organization grading of infiltrating gliomas: advances in the molecular genetics of glioma classification," *Annals of translational medicine*, vol. 3, no. 7, 2015.

[8] Y. Hirose, H. Sasaki, M. Abe, N. Hattori, K. Adachi, Y. Nishiyama, S. Nagahisa, T. Hayashi, M. Hasegawa, and K. Yoshida, "Subgrouping of gliomas

on the basis of genetic profiles," *Brain tumor pathology*, vol. 30, no. 4, pp. 203–208, 2013.

[9] M. Ceccarelli, F. P. Barthel, T. M. Malta, T. S. Sabedot, S. R. Salama, B. A. Murray, O. Morozova, Y. Newton, A. Radenbaugh, S. M. Pagnotta, *et al.*, "Molecular profiling reveals biologically discrete subsets and pathways of progression in diffuse glioma," *Cell*, vol. 164, no. 3, pp. 550–563, 2016.

[10] L. Dang, K. Yen, and E. Attar, "Idh mutations in cancer and progress toward development of targeted therapeutics," *Annals of Oncology*, vol. 27, no. 4, pp. 599–608, 2016.

[11] D. A. Long, "Raman spectroscopy," *New York*, pp. 1–12, 1977.

[12] P. Graves and D. Gardiner, "Practical raman spectroscopy," *Springer*, 1989.

[13] M. K. Maruthamuthu, A. H. Raffiee, D. M. De Oliveira, A. M. Ardekani, and M. S. Verma, "Raman spectra-based deep learning: A tool to identify microbial contamination," *MicrobiologyOpen*, vol. 9, no. 11, p. e1122, 2020.

[14] C.-S. Ho, N. Jean, C. A. Hogan, L. Blackmon, S. S. Jeffrey, M. Holodniy, N. Banaei, A. A. Saleh, S. Ermon, and J. Dionne, "Rapid identification of pathogenic bacteria using raman spectroscopy and deep learning," *Nature communications*, vol. 10, no. 1, pp. 1–8, 2019.

[15] J. Liu, M. Osadchy, L. Ashton, M. Foster, C. J. Solomon, and S. J. Gibson, "Deep convolutional neural networks for raman spectrum recognition: a unified solution," *Analyst*, vol. 142, no. 21, pp. 4067–4074, 2017.

[16] S. So, "Why is the sample variance a biased estimator?," *Griffith University, Tech. Rep.*, vol. 9, 2008.

[17] H. Nobach, "Practical realization of bessel's correction for the bias-free estimation of the auto-covariance and the cross-covariance functions," 2020.

[18] R. C. Geary, "The ratio of the mean deviation to the standard deviation as a test of normality," *Biometrika*, vol. 27, no. 3/4, pp. 310–332, 1935.

[19] H. Vinutha, B. Poornima, and B. Sagar, "Detection of outliers using interquartile range technique from intrusion dataset," in *Information and Decision Sciences*, pp. 511–518, Springer, 2018.

[20] S. Walfish, "A review of statistical outlier methods," *Pharmaceutical technology*, vol. 30, no. 11, p. 82, 2006.

[21] Y. Dovoedo and S. Chakraborti, “Boxplot-based outlier detection for the location-scale family,” *Communications in statistics-simulation and computation*, vol. 44, no. 6, pp. 1492–1513, 2015.

[22] R. Lowry, “Concepts and applications of inferential statistics,” 2014.

[23] N. Sebe, I. Cohen, A. Garg, and T. S. Huang, *Machine learning in computer vision*, vol. 29. Springer Science & Business Media, 2005.

[24] J. F. Allen, “Natural language processing,” in *Encyclopedia of computer science*, pp. 1218–1222, 2003.

[25] M. Malheiros, C. Jennett, S. Patel, S. Brostoff, and M. A. Sasse, “Too close for comfort: A study of the effectiveness and acceptability of rich-media personalized advertising,” in *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 579–588, 2012.

[26] J. MacQueen *et al.*, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[27] S. Chawla and A. Gionis, “k-means–: A unified approach to clustering and outlier detection,” in *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 189–197, SIAM, 2013.

[28] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, “The planar k-means problem is np-hard,” in *International Workshop on Algorithms and Computation*, pp. 274–285, Springer, 2009.

[29] M. K. Pakhira, “A linear time-complexity k-means algorithm using cluster shifting,” in *2014 International Conference on Computational Intelligence and Communication Networks*, pp. 1047–1051, IEEE, 2014.

[30] F. Murtagh, “A survey of recent advances in hierarchical clustering algorithms,” *The computer journal*, vol. 26, no. 4, pp. 354–359, 1983.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[32] C. Shalizi, "Distances between clustering, hierarchical clustering," *Lectures notes*, 2009.

[33] M. Dash and H. Liu, "Feature selection for classification," *Intelligent data analysis*, vol. 1, no. 3, pp. 131–156, 1997.

[34] O. Sharir, B. Peleg, and Y. Shoham, "The cost of training nlp models: A concise overview," *arXiv preprint arXiv:2004.08900*, 2020.

[35] M. Kukacka, "Overview of deep neural networks," *WDS '12 Proc. of Contributed Papers*, 2012.

[36] J. Zhang, C. Zong, *et al.*, "Deep neural networks in machine translation: An overview.," *IEEE Intell. Syst.*, vol. 30, no. 5, pp. 16–25, 2015.

[37] J.-G. Lee, S. Jun, Y.-W. Cho, H. Lee, G. B. Kim, J. B. Seo, and N. Kim, "Deep learning in medical imaging: general overview," *Korean journal of radiology*, vol. 18, no. 4, p. 570, 2017.

[38] F. Qiu and J. Jensen, "Opening the black box of neural networks for remote sensing image classification," *International Journal of Remote Sensing*, vol. 25, no. 9, pp. 1749–1768, 2004.

[39] S.-C. Wang, "Artificial neural network," in *Interdisciplinary computing in java programming*, pp. 81–100, Springer, 2003.

[40] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[41] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.

[42] D. Misra, "Mish: A self regularized non-monotonic neural activation function," *arXiv preprint arXiv:1908.08681*, vol. 4, 2019.

[43] N. Srivastava, "Improving neural networks with dropout," *University of Toronto*, vol. 182, no. 566, p. 7, 2013.

[44] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?," *arXiv preprint arXiv:1805.11604*, 2018.

[45] K. Krippendorff, "Mathematical theory of communication," *Departmental Papers (ASC)*, p. 169, 2009.

[46] C. E. Shannon, “A mathematical theory of communication,” *ACM SIGMOBILE mobile computing and communications review*, vol. 5, no. 1, pp. 3–55, 2001.

[47] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[48] D. Friedmann-Morvinski, “Glioblastoma heterogeneity and cancer cell plasticity,” *Critical Reviews™ in Oncogenesis*, vol. 19, no. 5, 2014.

[49] Z.-M. Zhang, S. Chen, and Y.-Z. Liang, “Baseline correction using adaptive iteratively reweighted penalized least squares,” *Analyst*, vol. 135, no. 5, pp. 1138–1146, 2010.

Appendices

Appendix A

Feature Selection

Features extracted from the original dataset: 509, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 532, 533, 538, 539, 540, 541, 545, 546, 547, 548, 549, 550, 551, 552, 553, 562, 563, 647, 1449, 1450, 1451, 1452, 1453, 1454, 1455, 1456, 1457, 1458, 1459, 1460, 1461, 1462, 1463, 1464, 1465, 1468, 1469, 1470, 1471, 1472, 1473, 1474, 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485, 1487, 1492, 1494, 1495, 1496, 1497

Features extracted from the curated training set: 23, 30, 31, 297, 298, 299, 308, 309, 310, 508, 509, 521, 522, 523, 524, 525, 526, 528, 529, 647, 648, 722, 1494, 1496, 1501, 1681, 1685, 1686, 1692, 1696, 1697, 1699, 1700, 1701, 1702, 1703, 1704, 1705, 1706, 1707, 1708, 1709, 1710, 1711, 1712, 1713, 1714, 1715, 1716, 1717, 1718, 1719, 1720, 1721, 1722, 1723, 1724, 1725, 1726, 1727, 1728, 1729, 1730, 1731, 1732, 1733, 1734, 1735, 1736, 1737

Intersection among the feature sets: 509, 521, 522, 523, 524, 525, 526, 528, 529, 647, 1494, 1496

Appendix B

Spectral Images For Outlier Detection

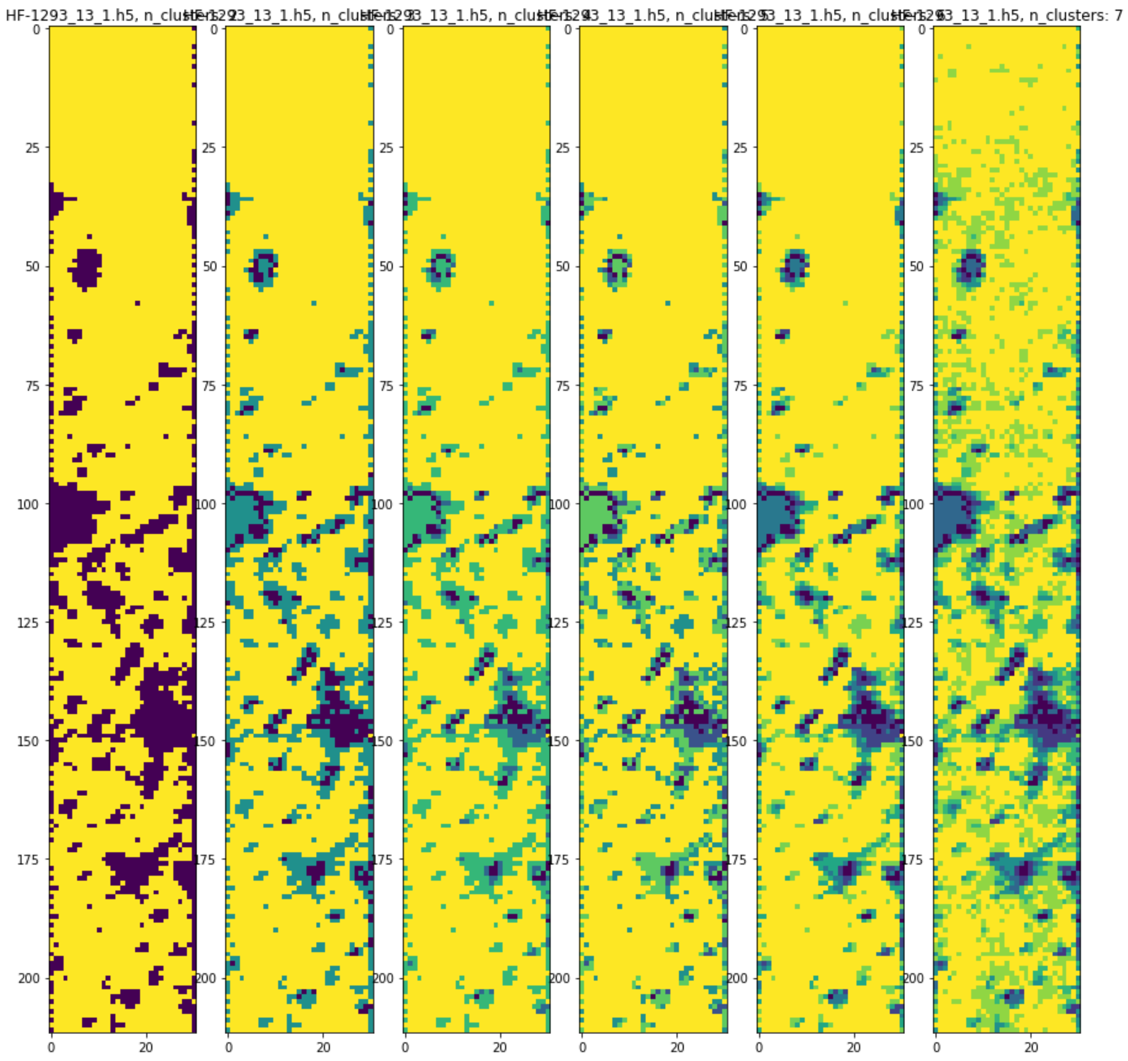


Figure B.1: Ward linkage on sample HF-868 from LGm1, The test fails to detect any outlier areas. LEftmost image is the result of the model computing 2 clusters. The number of clusters increase towards the rightmost image.

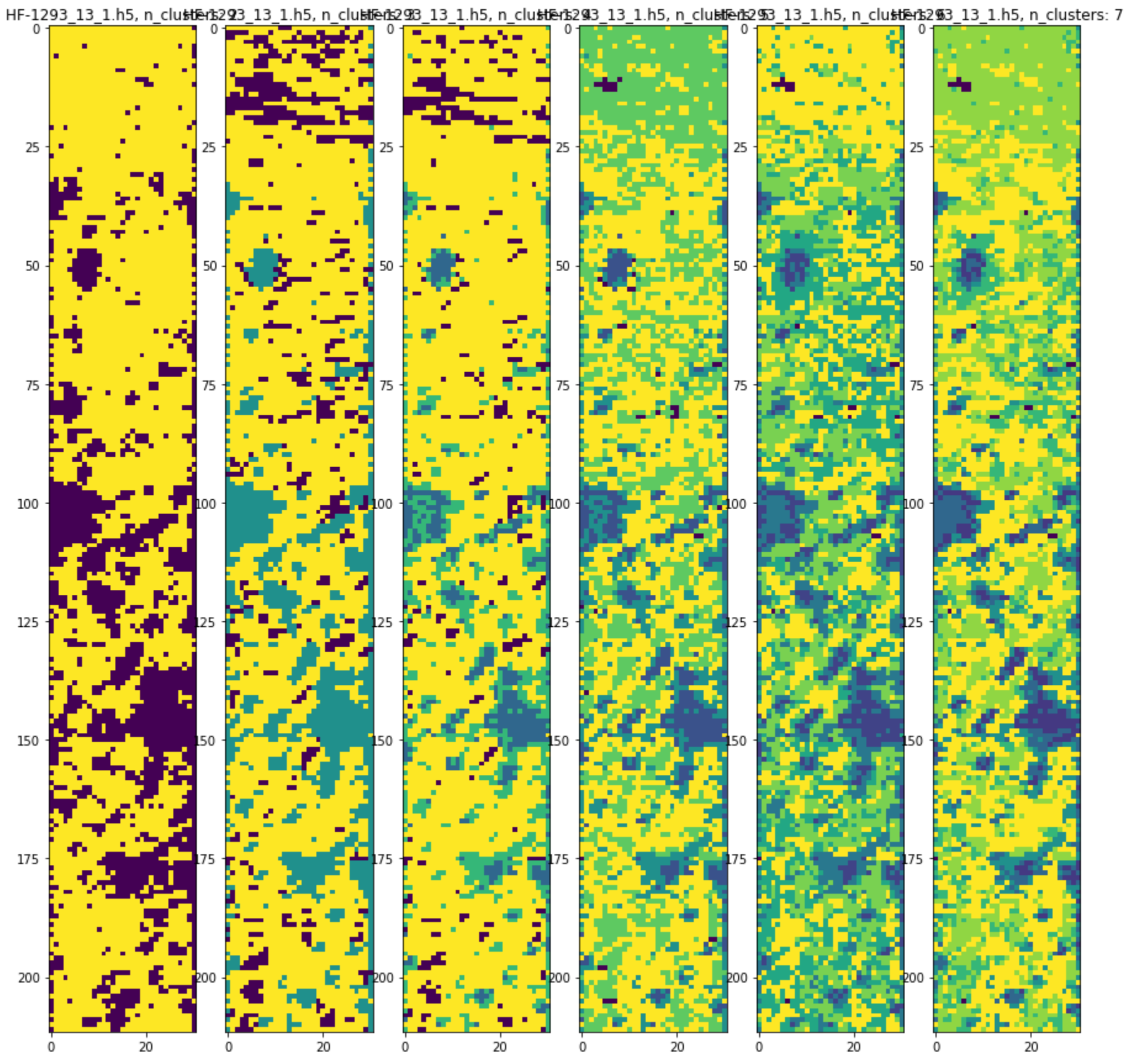


Figure B.2: *K-means* on sample HF-868 from LGm1, The test fails to detect any outlier areas. LEftmost image is the result of the model computing 2 clusters. The number of clusters increase towards the rightmost image.

Appendix C

Confusion matrix from model predictions

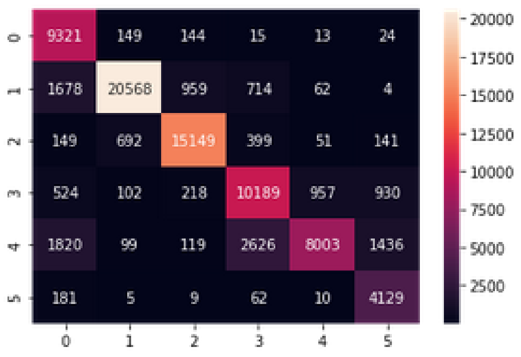


Figure C.1: The confusion matrix of the predictions for the separated training set. The diagonal shows the correct predictions of the models, wrong predictions are the surrounding area. The values within the cells are the number of predictions for the class. The model can generalize well to the separated training set.

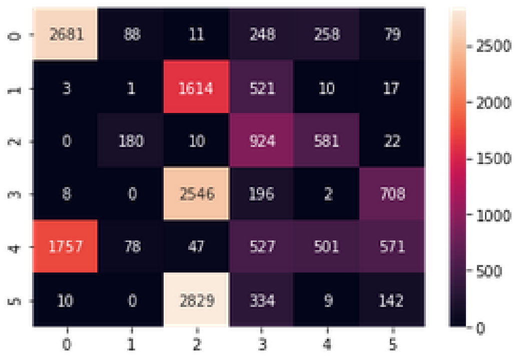


Figure C.2: The confusion matrix of the predictions for the validation set. The validation set consists of samples not included in the training data. The samples are hard to generalize to. Only LGm1 is correctly labeled by the majority of predictions.