Universiteit Leiden

Master Computer Science

Predicting Treatment Response for a Malignant Pleural Mesothelioma (MPM) Patient Based on the 3D Medical Imaging.

Name: Ertugrul Bozcal

Student ID: 2969971

Date: 30/11/2022

Specialisation: Data Science

1'st Supervisor: Dr. Lu CAO

2'nd Supervisor: PhD Stefano Trebeschi

Daily Supervisor: MSc Kevin Groot Lipman

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science

Leiden University

Niels Bohrweg 1

2333 CA Leiden

The Netherlands

ABSTRACT

Malignant Pleural Mesothelioma (MPM) is a cancer type that develops around the lung, and the life expectancy for MPM patients is about one year. Some treatments, such as chemotherapy and radiotherapy, are available to slow down tumor growth. However, the treatment process is mainly painful and does not always yield a positive result. Therefore, being able to predict the outcome of the treatment for any newly diagnosed patient enables doctor and patient to make informed decisions about the treatment and might prevent potential suffering.

We aim to use 3D CT scans to predict the response of MPM treatment with machine learning (ML) methods. Two experiments were conducted to determine whether a patient's CT scans have predictive features for the treatment response: Unsupervised and supervised ML methods.

In the first part, an unsupervised ML method was implemented on the features extracted from CTs using radiomics. The result was compared with labels created by looking at the volume change on the follow-up CT scans. In the second part, we employed supervised ML methods using Convolutional Neural Network(CNN) based deep neural network models. We sought to predict the treatment result by applying trained supervised ML models, which perform well on the test set.

Within the scope of this research, we could not find any promising result for predicting the treatment result of MPM patients based on the CT scan through ML. This result may be due to the relatively small data size and the unavailability of the optimal labeling method. Nevertheless, given relatively better transfer learning results in the second part and there is yet room to find optimal labeling and data types, promising results can still be achieved using sufficient data and optimal labeling and data types.

Keywords: Malignant Pleural Mesothelioma (MPM), Computed Tomography(CT), Radiomics, Machine Learning(ML), Unsupervised ML, Supervised ML, Convolutional Neural Network (CNN), ResNet, Transfer Learning(TL), Hyperparameter Optimization(HPO).

Contents

1 Introduction **4**

2 Background/Preliminaries **5**
2.1 Malignant Pleural Mesothelioma (MPM) . . . 5
2.2 Radiomics . . . 6
2.3 Harmonization . . . 8
2.4 Clustering Algorithms . . . 8
2.5 Principal Component Analysis . . . 9
2.6 Convolutional Neural Networks . . . 9
2.6.1 Simple CNN . . . 10
2.6.2 ResNet . . . 10
2.7 Hyperparameter Optimization . . . 11
2.8 Transfer Learning . . . 12

3 Related Works **12**

4 Methods **13**
4.1 Data . . . 13
4.2 Radiomics Feature Extraction and Data Prepossessing . . . 14
4.2.1 Feature extraction . . . 14
4.2.2 Data Preprocessing . . . 15
4.3 Data Prepossessing for CNN and ResNet . . . 15
4.4 Tumor Volume and Labelling . . . 17
4.5 Unsupervised Methods . . . 17
4.6 Supervised Methods . . . 18
4.6.1 Hyperparameter Optimization . . . 18
4.6.2 CNN . . . 19
4.6.3 ResNet . . . 20
4.6.4 Transfer Learning . . . 20

5 Results **20**
5.1 Unsupervised Methods . . . 20
5.1.1 Feature extraction and preprocessing . . . 20
5.1.2 Result of clustering algorithms . . . 21
5.2 Supervised Methods . . . 23
5.2.1 Hyperparameter Optimization . . . 23
5.2.2 Results with Different Labelling Methods and Data Types: . . . 24
5.2.3 Transfer Learning . . . 26
5.2.4 Results on the Test Set with Optimal Data, Labels and Hyperparameters 27

6 Discussion **28**

7 Conclusion **31**

References **31**

8 Appendix **34**
8.1 Hyperparameter Optimization Results 34

1 Introduction

MPM is aggressive cancer that grows around the lung with an average life expectancy of one year[1]. Although specific treatments are available to reduce tumor volume growth, these do not work for all MPM patients and might cause suffering side effects. Since the biological activity of MPM is unpredictable, and a patient's prognosis differs based on their specific situation, it is difficult to predict whether the treatment will be effective in advance[2].

Monitoring and assessing the volume change of the tumor is crucial for determining the treatment response. Rusch et al. observed that with the evaluation of the volume change based on the volume measurement, the treatment outcome for the MPM patients can be predicted[3].

Recent technology developments have ushered in a new era of precision medicine by merging ML with biological research to analyze diseases using data. The application of ML aids in extracting meaningful conclusions from large amounts of data, enhancing treatment outcomes[4]. ML is frequently utilized in cancer research and is gaining attraction in cancer identification and predicting the treatment result. The precision medicine aim is to provide medications that not only raise patients' chances of survival but also improve their quality of life by decreasing undesired side effects[4].

Radiomic signature in the images is predictive in a variety of tumor forms[5]. However, the CT-based radiomics model for predicting an MPM patient's progression-free survival and overall survival could not be successfully trained[1]. To the best of our knowledge, a CT-based model for MPM patients for predicting treatment response using an ML model has not yet been investigated. This study aimed to evaluate the predictive ability of CT-based radiomics models and CT-based Deep Learning models for treatment response prediction of MPM patients.

The aim of this paper involves answering the following research question by using ML methods: How can we predict whether treatment will work for MPM patients based on their 3D CTs?

Two main ML methods were studied to answer this research question: unsupervised clustering and supervised deep learning. The first part applied clustering algorithms to the selected radiomics features extracted from CT scans. The result was then compared with the labels created based on the tumor volume change in the subsequent CT scans. On one side, we have cluster labels based on the radiomics features of CT scans, and on the other, we have labels based on volume change with follow-up CT scans. We aimed to analyze how much of our labeled data coincides with the cluster's labels. Suppose that the majority of the response classes are grouped in one of the clusters, say the first, and we observe a newly diagnosed patient's CT scan in the same cluster. We could recommend that patient can proceed with treatment because the radiomics features indicate that the treatment result will most likely be positive for that patient.

The supervised deep learning algorithms were applied in the second part to answer the same research question. Simple CNN and ResNet versions were implemented as models in this part. After optimizing network architectures through hyperparameter optimization, we trained them with the optimal hyperparameters, data types, and labels. The models' per-

formance was then measured on the test set. As the last step of this part, to mitigate the effect of the relatively small data size on the model's performance, the fine-tuning method was implemented to the ResNet as transfer learning. We aimed that if the model performs well on the test set, it could be applied to unseen data, newly diagnosed MPM patients' CT scans in our case, and we could predict the treatment outcome for that patient.

Concerning data, in the first part, a total of n=561 CT scans belonging to n=133 MPM patients were used. In the second part, there were a total of n=1003 scans belonging to n=236 MPM patients.

Regarding the result, in the first part, we could not observe most of the response classes in one of the clusters after comparing the result of the clustering algorithms and the labels. Likewise, in the second part, while the CNN and ResNet trained from scratch gave results no better than a random predictor on the test sets, ResNet slightly outperformed the others using transfer learning.

Within this study's scope, we could not identify any promising results for predicting the treatment outcome of MPM patients based on CT scans using ML. However, given relatively better transfer learning results and the possibility of finding optimal labeling and data types, promising results can still be obtained in the future with sufficient data and ideal labeling and data types.

We organized the rest of the report as follows. In the next section, the background of this study was provided with a more in-depth overview. Subsequently, related works are presented in section 3. Section 4 presents the methods and experimental setup for the data and models used in the experiments. The results of the experiments for both ML models are stated in section 5. Section 6 reports how we evaluated the results of the models explained in the previous section and analyses the results. Finally, the paper is concluded in section 7, followed by the references and appendix in section 8.

2 Background/Preliminaries

This chapter provides background information on this study's topics: MPM, Radiomics, Harmonization, Clustering Algorithms, Principal Component Analysis, CNN, ResNet, Hyperparameter Optimization, and Transfer Learning.

2.1 Malignant Pleural Mesothelioma (MPM)

MPM is rare cancer that develops around the lung membrane, known as the pleura, and is caused by long-term exposure to asbestos. The fibers can become embedded in the pleura after inhalation, producing inflammation and scarring. These mechanisms can eventually lead to the development of mesothelioma. Since this tumor has an uneven form and widespread growth throughout the pleura, it isn't easy to contour it[1].

Figure 1: One slice of the CT scan and mask for an anonymous MPM patient

Because of the time lag between exposure and the onset of MPM, about 40 years, and the continued usage of asbestos in some regions of the world, the number of MPM patients is projected to climb in the future years. Therefore, the management of the treatment is essential for MPM patients.

The life expectancy of MPM patients is low, about 12 months. Although some treatments are available for MPM patients, such as chemotherapy and radiotherapy, this procedure carries a toxicity risk[6]. Therefore, carefully selecting acceptable candidates for this kind of treatment is crucial.

In this research, the effectiveness of the treatment is assessed in terms of the volume reduction of the tumor. The tumor volume is measured for each CT scan for any patient with the help of AI, and the treatment response is evaluated by comparing the current CT scan and looking up the subsequent CT scans.

2.2 Radiomics

Radiomics is a technique that uses data-characterization algorithms to extract features from medical images. These features can potentially reveal tumoral patterns and characteristics that the naked eye cannot detect. Radiomics uses texture, shape, and grey-level statistics within images to identify various relationships[7]. Radiomics features can be categorized as; First Order Statistics Features, Morphological or Shape-based Features, and Texture Based Features.

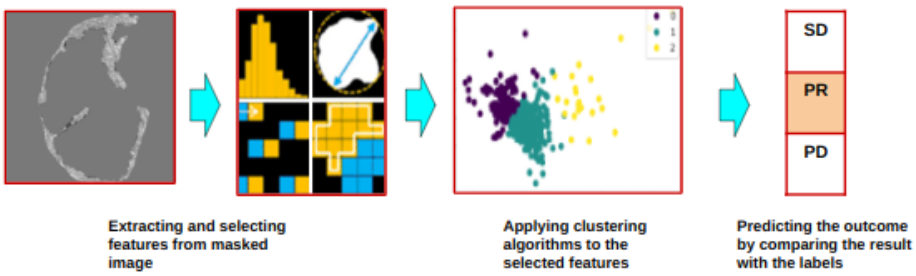


Figure 2: Unsupervised ML pipeline. After taking images and segmentation as input, radiomics software extracts features from masked images, and powerful ones were selected by high correlation and low variance filters. In the last step, Unsupervised ML models were implemented on the selected radiomics features.

First order statistics depict the voxel intensity distribution in the tumor part. The most known features in this category are *energy, entropy, minimum, maximum, median, mean,*

standard deviation, skewness, kurtosis, variance, and uniformity. As an example, if H is a first-order histogram with B bins, then the entropy and uniformity can be calculated using the following equations:

$$Entropy = -\sum_{i=1}^{N_g} p(i) log(p(i) + \epsilon)$$

$$Uniformity = \sum_{i=1}^{N_g} p(i)^2$$

where P(i) is the first order histogram, N_g represent the number of bins and p(i) is first order histogram which is normalized and can be written as $\frac{P(I)}{N_g}$ and ϵ define a small positive number which can be specified arbitrarily. While entropy assesses the inherent randomness in an image's grey level intensities, uniformity assesses the uniformity of grey level intensities within an image or Region of Interest(ROI)[7].

Morphological(Shape-based) Features describe the ROI's 3D size and shape and are calculated based on the original images and masks. Some of the features in this group are *sphericity, compactness, elongation, flatness, and maximum 3D and 2D diameters.* These features are used to define the shape of an object in many contexts. The tumor's shape can be quantified using fractal dimension, a measurement of irregularity in the tumor's shape. The triangle mesh defines the approximated shape to derive shape-based features. For example, sphericity measures the roundness of the tumor region's shape in relation to a sphere.

$$Sphericity = \frac{\sqrt[3]{36\pi V^2}}{A}$$

where A is the mesh's surface area and V is the mesh volume[7].

Texture-based features are considered one of the most important characteristics of the images. While second-order statistical features (shape-based) are calculated based on the neighboring voxel's positions, texture-based features account for the distributions across three or more voxels and evaluate the parameters such as complexity and roughness. In other words, texture analysis is used to analyze the spatial distribution of pixel intensities, focusing on the interaction between each voxel and its nearby areas. While first-order statistics give more global characteristics, texture-based features highlight the local ones.[8]. These texture characteristics are formed as matrices which are described as follows:

- The Gray Level Co-occurrence Matrix(GLCM) is used to extract the spatial distribution of grey-level intensities within an image
- The Grey Level Run Length Matrix (GLRLM) specifies how many neighboring voxels have the same grey-level value. It characterizes the grey-level run lengths of different grey-level intensities in any direction.
- The Neighborhood Grey Tone Difference Matrix (NGTDM) analyses texture using an image's visual properties.
- A Gray Level Size Zone Matrix (GLSZM) is used to quantify grey-level zones in an image, and

- A Gray Level Dependence Matrix (GLDM) calculates the grey-level relationships in an image.

2.3 Harmonization

Since using one hospital's dataset leads to models overfitting to characteristics of the hospital scanner/reconstruction, image analysis models require large and multicenter datasets to make them heterogeneous. Therefore, data is collected from different centers to ensure the models generalize better to out-of-domain data. However, one side effect of this solution is the issue of inter-scanner variability. Harmonization eliminates these systematic differences between images mainly caused by acquisition protocols, image post-processing, and reconstruction in the different platforms. If not eliminated, those differences affect the model's performance built on the extracted features from the medical images[9].

The Combat Harmonization method is used in this study as a harmonization software. It is an empirical adjustment for location (mean) and scale (variance) based on empirical Bayes estimation. First, the data is modeled as a linear combination of the biological variables of interest and scanner effects, and then it is adjusted to be consistent across sites/scanners. Using ComBat, the value for each feature f , i.e. Y_{ijf}, for subject j for site/scanner i is first modeled as follows:

$$Y_{ijf} = \alpha_f + X_{ij}\beta_f + \gamma_{if} + \delta_{if}\epsilon_{ijf}$$

where α_f is the average value for feature f , X_{ij} is the design vector of biological variables, β_f is the vector of regression coefficients corresponding to X_{ij}, and γ_{if} and δ_{if} are the additive and multiplicative terms for site/scanner i and feature f respectively [10].

2.4 Clustering Algorithms

Four different clustering algorithms are experienced in this study: K-Means, Gaussian mixture, Fuzzy-Cmeans, and Affinity propagation. When selecting these algorithms, we aimed to choose algorithms that use different clustering approaches. For example, K-means and Affinity Propagation use hard clustering while Fuzzy_Cmeans uses soft and Gaussian mixture uses both. On the other hand, while we can determine the number of clusters in advance for the first three algorithms, this is not required by Affinity Propagation, which determines the number of clusters on its own.

In **K-Means** clusters, the criterion is minimized by distributing the N observations among the K clusters in such a way that the average dissimilarity of the observations from the cluster means, as determined by the points in that cluster, is minimized within each cluster. After assigning all observations to the closest center, the mean is calculated within each cluster and the current set of means is minimized by assigning each observation to the closest (current) cluster mean again. This process is iterated until the assignments do not change[11].

Gaussian mixture model is a probabilistic model in which all data points are believed to be the result of a finite number of Gaussian distributions with unknown parameters being mixed. Gaussian Mixture models may be thought of as generalizing K-means clustering by including information on the data's covariance structure as well as the centers of the latent Gaussian distributions[11].

Fuzzy-Cmeans algorithm assigns each point to each cluster with a probability of belonging to that cluster, rather than rigidly belonging to only one cluster as the classic K-means algorithm does. Each point in Fuzzy-C Means clustering has a weighting connected with a certain cluster; therefore, a point does not sit "in a cluster" as much as it has a weak or strong relationship to the cluster[12].

Affinity propagation algorithm relies on "message transfer" between samples. Unlike K-Means, affinity propagation does not need to determine or predict the number of clusters before running the algorithm. Affinity propagation identifies "exemplars," individuals of the input set indicative of clusters[13].

2.5 Principal Component Analysis

Principal component analysis (PCA) is a dimensionality reduction method used to analyze datasets with many dimensions/features per observation. It improves data interpretability while preserving much information by retaining as much variance as possible and allows multidimensional data visualization[14].
Because it is hard to display our high-dimensional features, we implemented PCA on the selected features to reduce our data dimensions to two, so they could be easily visualized.

2.6 Convolutional Neural Networks

Convolutional Neural Network(CNN) is a deep neural network type that uses convolution in at least one of its layers and is commonly used for visual analysis. A convolution is a mathematical operation that slides one function over another and measures the integral of their pointwise multiplication[15]. CNN is mostly used in image recognition and was designed to handle pixel data.

Convolutional layers carry out convolution, a linear multiplication of weights with the input. Through convolutional layers, the features are extracted from the image as a feature map using filters or kernels consisting of the network's weights. Each filter passes through the input image to cover the whole image. This operation results in the feature maps, the outcomes of estimating the dot product between these filters' weights and the image's filter-sized part. This process is conducted using a different number of kernels with the same dimensions as the input image but a smaller size. An activation function is applied to feature maps to capture non-linearity.

Pooling layers lower the size of the feature map by merging the result of neuron clusters at one layer into a single neuron at the next layer. The two most widely used types of pooling are maximum and average. Average pooling employs the average value, while in max pooling, the largest value is chosen in each local cluster of neurons of the feature map[16].

In the final part of the CNN, fully connected layers are connected to all the information gathered in earlier layers and pass the information to the output layer. The most often used activation functions in the output layers for classification tasks are softmax and sigmoid, which are used to calculate the probabilities of an input belonging to a specific class. The sigmoid is used mainly for binary classification, while softmax is used for multi-class classification tasks.

The network is trained by using backpropagation. Backpropagation computes the gradient of the loss function with respect to the network weights and accomplishes this automatically in just two network passes(one forward, one backward). The backpropagation algorithm determines the gradient of the network's error for each parameter used in the model. In other words, it can identify how to change each link's weight and bias term to eliminate errors. After obtaining these gradients, it executes a standard Gradient Descent step, and the procedure is continued until the model converges.[15].

2.6.1 Simple CNN

This study used a simple CNN structure consisting of input layers, several convolutional layers, pooling layers, fully connected layers, and outcome layers.

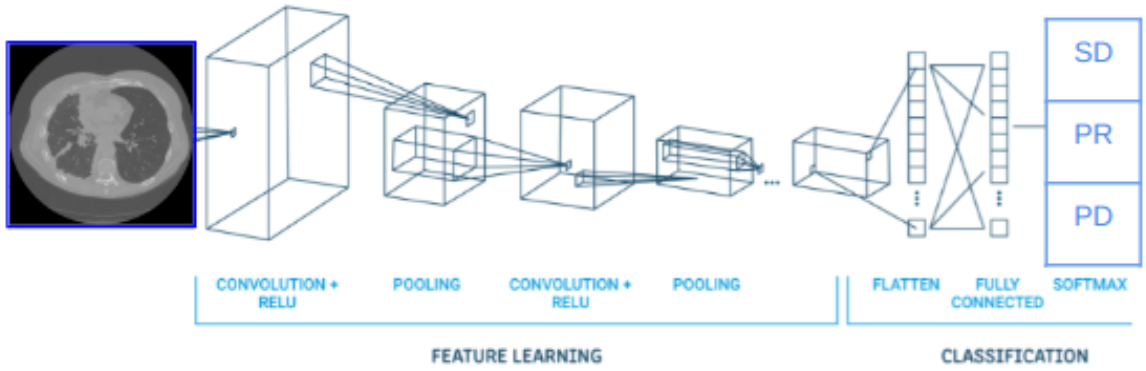


Figure 3: 3D CNN structure. A simple CNN can be separated into two main parts: feature extraction and classification. The feature extraction consists of the input layer, several convolutional layers, and pooling layers. The classification consists of fully connected layers and an outcome layer with an output neuron and activation function corresponding to the classification type.

2.6.2 ResNet

ResNet is a deep and special version of CNN that consist of different depths, which are the number of convolutional layers in the architecture ranging from 10 to 200. ResNet uses skip connections or shortcuts, which connect the layer's activations to subsequent layers by bypassing certain layers in between, resulting in a residual block. This skipping simplifies the network by employing fewer layers in the initial training stages and prevents the vanishing gradient and accuracy saturation problems. It also speeds up learning by lowering the influence of vanishing gradients due to having to propagate through fewer layers. The network gradually recovers the skipped layers as it learns the feature space. In other words, instead of layers learning the underlying mapping, ResNet allows the network to fit the residual mapping[17].

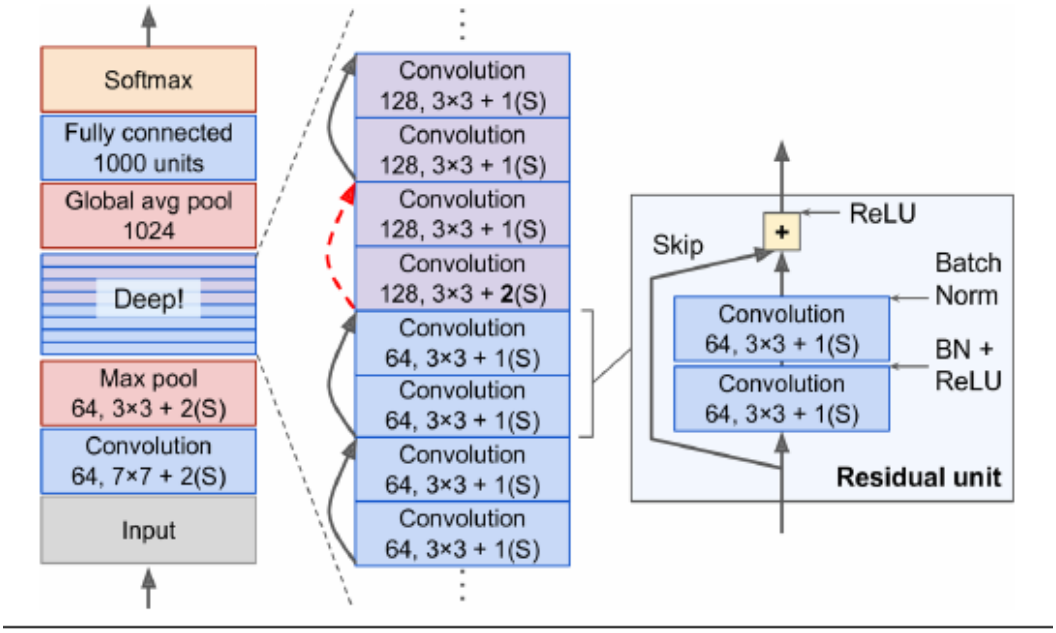


Figure 4: ResNet architecture. In ResNet, models go deeper with fewer parameters. The key to training such a deep network is skipping connections. The feature maps obtained from the convolutional layers are added to the info that passes over the skip connection. As a result, direct feature maps can be learned by relying on skip connections[15]

.

2.7 Hyperparameter Optimization

Hyperparameters are parameters whose values regulate the model's learning at the start of the training phase, and the process of selecting those sets of ideal hyperparameters for an ML algorithm is called hyperparameter optimization. The same ML model may require different hyperparameter configurations in different data sets. Since it is difficult to determine what values to use for a particular algorithm's hyperparameters on a given dataset, they must be tuned before training the model and using it on unseen data[18].

The most popular hyperparameter optimization methods are Grid Search, Random Search, and Bayesian Optimization. Evaluating the performance of a neural network is computationally costly, and Bayesian optimization finds good hyperparameter configurations in the search space with a few function evaluations. Therefore, this study chooses Bayesian optimization as the hyperparameter optimization method[18].

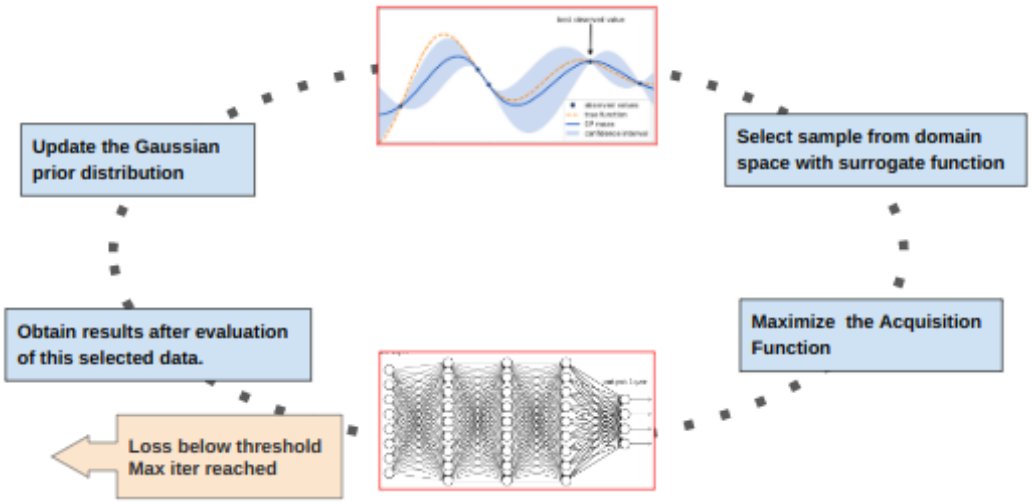


Figure 5: Bayesian optimization circle. Bayesian optimization identifies the set of hyperparameters by constructing a probability model based on the previous evaluation outcomes. As can be seen in the figure, this optimization process can be explained in 4 main steps: The prior distribution is initialized in the first step, and the surrogate function selects a sample from domain space. In the second step, several data points are chosen from this sample such that the acquisition function is maximized. And then, the objective cost function evaluates these selected hyperparameters to obtain the results. As the last step, the Gaussian Process prior distribution is updated in the light of the new results to produce a posterior, and in the following phase, this will become the precedence. These steps are repeated until the result is below the threshold or the maximum iteration is reached[19].

2.8 Transfer Learning

Transfer learning tries to enhance learning new problems by leveraging previous knowledge with a matching problem. In transfer learning, some trained layers or parameters are copied to the network for a new job, and fine-tuning was conducted to increase the model's performance on the new problem using less data. In computer vision tasks, we can transfer lower layers of the pre-trained model since lower layers of convolutional neural networks have generic characteristics. In comparison, higher layers have more particular to the original data[20].

There are two ways to use transfer learning. The feature extraction method can be used if the pretrained models and new data are highly comparable. In this method, after freezing lower layers from a previously trained model to avoid destroying any knowledge gained from subsequent training rounds, some fresh, trainable layers are put on top of these frozen layers. These additional layers are then trained on the new dataset. If our data sets differ from the pretrained ones, we can use fine-tuning, which entails unfreezing the entire model (or a portion of it) and training it on the new dataset.

3 Related Works

To our knowledge, no radiomics model has ever been developed successfully yet to predict the treatment outcome of MPM patients using CT scans. Therefore, this study would be the first

in this scope. However, some related works are similar to this study in predicting the treatment response for different cancer diseases by using ML methods and CT/FDG-PET images.

Pavic et al. [1] tried to predict progression-free survival(PSF) and overall survival(OS) of MPM patients by comparing CT and FDG PET images. After extracting 1404 CT and 1410 FDG PET radiomics features from each image, they used PCA to select only robust, non-redundant features and implemented cox regression to predict PFS and OS. While they could not successfully train a model based on the CT scans, they could build a model using FDG PET images to predict PSF and OS for MPM patients. According to the writers, the reason for this result is that due to variability in the contouring of the tumor, they chose more stable features in training the model, and FDG PET gives a higher percentage of stable features due to the imaging modality itself.

Rusch et al. [3] observed that the treatment outcome for MPM patients might be predicted by evaluating volume change based on volume measurement in CT scans. They discovered that tumor volume was independently correlated with OS by performing univariate and multivariable analyses that included clinical factors.

Lu l. et al. [21] used deep learning in their research for predicting the treatment response of metastatic colorectal cancer(mCRC) based on the serial of medical imaging from 1,028 patients. They considered the morphological change in the tumor instead of tumor volume change for assessing tumor response since morphological changes in the tumor might occur before tumor size change. They employed a deep learning network to define tumor morphological change for response evaluation in mCRC patients and discovered that their research might improve individualized early-on therapy decision-making.

Li H. et al. [22] developed an Unsupervised ML model to predict treatment response and survival of Non-Small Cell Lung Cancer(NSCLC) patients by extracting features from FDG-PET. After defining each patient with 722 radiomic features, an unsupervised two-way clustering algorithm was implemented to identify groups of patients and radiomic features simultaneously. The patient groups were compared regarding survival and freedom from nodal failure. They observed that the prediction models built upon radiomic features from medical imaging had better prediction performance than those built upon clinical measures.

4 Methods

This section explains the methods applied in each part and the related experimental setups. First, the common experimental setup is touched upon for the data, and then we go into depth about each of the methods we experimented with.

4.1 Data

The dataset used in this study included 3D CT scans of patients with advanced MPM who were under treatment. Most of the CT images were obtained at Netherlands Cancer Institute (NKI), but some in the dataset were taken in different hospitals around the country. Each patient's initial CT scan, as well as subsequent CT scans, were included. CT images have $3mm$ slice thickness; $512X512$Xnumber of slices matrix size.

In the first part, a total of n=561 CT scans, including baseline and follow-up scans belonging to n=133 MPM patients, were used, while in the second part, there were a total of n=1003 CT scans together with baseline and follow-up scans belonging to n=236 MPM patients.

In the second part, during the hyperparameter optimization, the dataset was divided into two parts as 70% train and 30% validation set while training the model with best parameters, data was divided into three parts as 60% train, 20% validation, and 20% test sets. In the final training phase with simple CNN and ResNet, the train set had n=142 patients and 601 CT scans, the validation set included n=48 patients and n=205 CT scans, and the test set included n=46 patients and n=197 CT scans. However, because both CT scans at the time of diagnosis and follow-up CT scans were included, the split was based on the number of patients to avoid having one patient's CT scans in numerous sets and to ensure the independence of each set.

While only masked CT was used as a data type in the unsupervised part, three different data types were used in the supervised part. The first two are masked CT and CT cropped by mask in a rectangular shape. As a third data type, two consecutive CT scans from one of the data types were fed into the model in two channels using the labels of the second CT. In the data loader, while the regular CT scans were fed in the first channel, the follow-up CT scan was loaded as a second channel by using the label of the second one. The model's performance was tried to increase by diversifying the data in this way.

4.2 Radiomics Feature Extraction and Data Prepossessing

4.2.1 Feature extraction

Pyradiomics were chosen as software to extract radiomics features from the CT scans. It is an open-source python library and can be used to compute every feature value for the ROI and voxel-based calculation to generate feature maps[23]. Pyradiomics allows feature extraction customization by specifying image and feature types. In our experiment, in addition to the original image type, all the derived image types were included in the parameter settings of pyradiomics and all feature types as well, as seen in the table below. Thus, more information aimed to be extracted from images by using mathematically defined filters.[6].

While 110 features were extracted for the original image types, 96 were extracted for each derived image type since shape-based features are extracted only from the original image type. LoG-derived features types extracted $5 * 96$ features since all the feature types are extracted for each sigma value. As a result of these parameter settings, a total of 1262 features were extracted per CT scan.

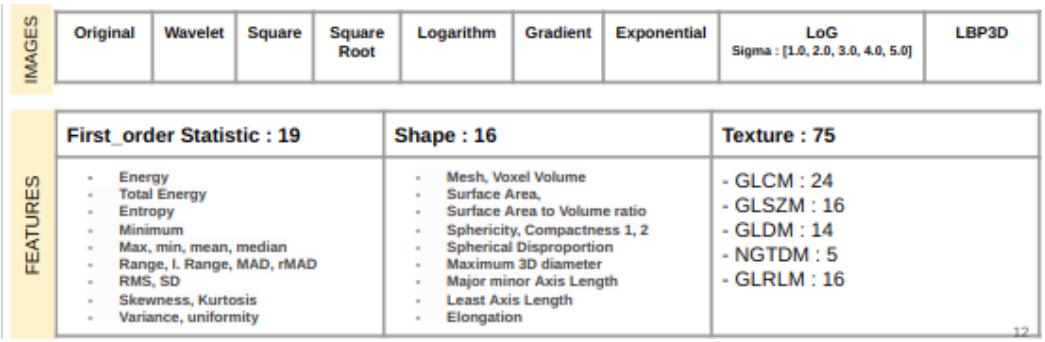


Figure 6: Pyradiomics setup. The table shows how image and feature types were defined in the pyradiomics parameters setup. 8 derived image types were defined in the parameter settings together with the original image, which is a total of 9 image types, as seen in the figure's upper part. All 3D feature types are defined in the parameter settings, and the distribution of the feature types is seen in the lower part of the figure.

4.2.2 Data Preprocessing

ML requires cleaning and preprocessing of data, including removing noisy data, handling missing data, removing outliers, and resizing data so that it can be used efficiently in the model. In the unsupervised part, the preprocessing was implemented for the features extracted from the images and masks.

As a first step of preprocessing in the unsupervised part, the harmonization algorithm neurocombat was applied to the features extracted by pyradiomics to solve the heterogeneity problem in CT scans. Then feature selection methods which are *high-correlation filters and low-variance filters*, were implemented to eliminate the noisy and redundant features which might deteriorate the model's performance. By applying these methods, features with a correlation higher than 80% and a variance lower than 20% were removed from the data set.

Correlation refers to how close two variables are to each other linearly. If two variables have a high correlation, it means they are linearly dependent on each other, and it makes no sense to use both in the model since both have the same effect on the dependent variable. 80% threshold was used to eliminate correlated features, which means if any feature correlates higher than 80%, that feature was removed from the data set[24].

The second method used to eliminate redundant features is low-variance filters. When a feature's value is constant in all observations or doesn't change considerably in different observations, this feature is not considered a discriminative variable for any ML model. To eliminate this kind of low variance redundant features, a 20% threshold is used. Any features whose variance didn't meet this threshold were removed from the dataset by using Scikit-learn VarianceThreshold function. And finally, the remaining feature values were standardized using Scikit-learn StandardScaler before applying the clustering algorithms.

4.3 Data Prepossessing for CNN and ResNet

While the preprocessing was implemented for the features extracted from the images and masks in the unsupervised part, the images and masks were subject to preprocessing in the supervised part. The feature extraction process was not implemented since CNN-based models handle the whole process, from feature extraction to classification, as an end-to-end

learning method. However, before feeding the images as input to the CNN, some preprocessing was applied to get the valuable and proper information from the images by eliminating their noisy parts.

There are three different data types created during the preprocessing. First, the masked CT scan and CT cropped by mask in a rectangular shape were used to train the models as two different data types. After that, the third data type was created by feeding any of the first two data types in two channels in the following way. In the data loader, while the normal CT scans were fed as a first channel, the follow-up CT scan was loaded as a second channel by using the label of the second one. We aimed to increase data size and diversify the data to get a better model performance in this way.

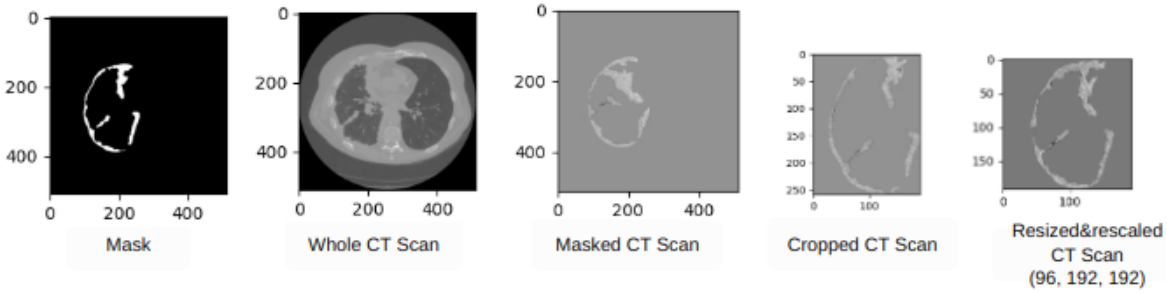


Figure 7: One of the slides of 3D Masked CT scans. As the figure illustrates, the mask overlapped the CT scan, and the tumor was segmented from CT scans. By doing this, only the information in the tumor part was used in training the model. Then, CT was preprocessed and rescaled in the size of (96, 192, 192).

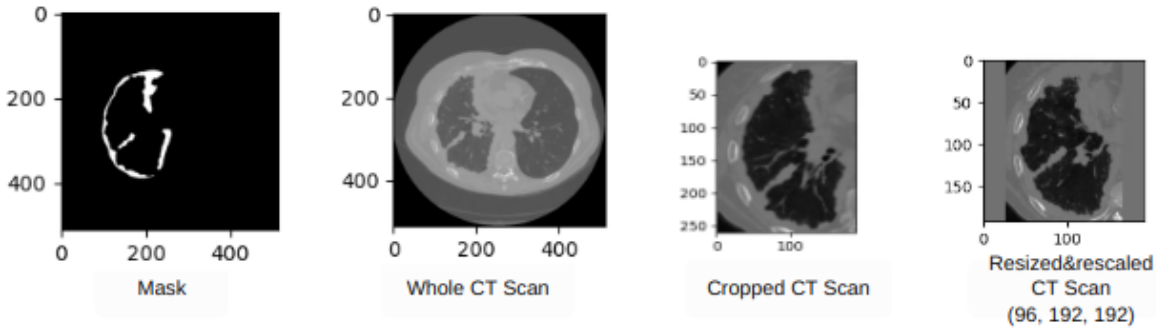


Figure 8: One of the slides of 3D CT scans cropped by mask in a rectangular shape. The figure illustrates that the CT scan was cropped as a square to cover the entire tumor area. By doing this, while as much information as possible was kept, the unrelated CT part was removed. Then, CT was preprocessed and rescaled in the size of (96, 192, 192).

After creating two different data types, as explained above, the data preprocessing continued in the following way for both data types. All CT scans consisted of a Hounsfield scale, a quantitative scale used to describe radiodensity (raw voxel intensity). Since HU values above 400 are bones and below 1000 air, the Hounsfield Unit(HU) threshold window of [-1024, 400] was set to normalize all the values between 0 and 1.

The dimension of the original CT scans are 512 x 512 x The number of slices. CT scans were resized to 96x192x192 for computation efficiency by keeping the maximum resolution.

As the last step of preprocessing, the CT scans are augmented by rotating at random angles during training, such as flipping the image and rotating it randomly in a range of [-20 ∘ , 20 ∘]. This process exposes the model to diverse parts of the training data while delaying overfitting.
The data loader was used to create and load the data set by splitting the data into batches instead of loading the data at once. Using the data loader provides easy access to the data set and speeds up the model's training.

4.4 Tumor Volume and Labelling

The data consists of n=133 baseline CT scans in the first part and n=236 in the second part. Because this is an insufficient quantity of data points for training a deep learning model, more data points were included by treating every CT scan as the baseline except the last ones for any patient. These CT scans were labeled by considering the tumor volume change in the follow-up CT scans. They were labeled as categorical for both unsupervised clustering and supervised classification models.

- If the tumor volume of any follow-up scan is decreased more than 20% from the subject CT scan, the CT is labeled as a response (PR)
- If the tumor volume of any follow-up scan is increased more than 20% from the subject CT scan, the CT is labeled as a progression (PD)
- If the tumor volume increases or decreases less than 24ML and the tumor does not grow or shrink more than 20%, and the CT is labeled as stable (SD).

Since the research is still going on in the NKI about the optimal labeling method, the 10% threshold was also used in the second classification labeling method. As a third labeling method, the CTs were labeled by looking at the volume change only in the follow-up CT scans by using the 10% threshold again. In addition to the categorical labeling for the classification models, the continuous labels are created in line with the above explanation by directly using the volume change in percentage to be used in the CNN regression training.

4.5 Unsupervised Methods

In the first part of this study, unsupervised clustering methods were implemented to discover the solution to the research question. First, the clustering algorithms were implemented to the selected features. The result was compared with the labels created by looking up the volume change in follow-up CT to check the extent to which those clusters overlap with these labels.

More specifically, we have cluster labels based on radiomics features of CT scans on one side and labels based on volume change with follow-up CT scans on the other. We aimed to determine how much our labeled data overlaps with the cluster's labels. For example, if the majority of response classes are grouped in one of the clusters, and we find a newly diagnosed patient's CT scan in the same cluster, that patient can move with treatment because the radiomics features display that the treatment result will most likely be positive for that patient.

To get accurate and reliable results, four different clustering algorithms were implemented for the selected features to see specific patterns in the data. Those are K_Means, Fuzzy_Cmeans, Gaussian Mixture, and Affinity Propagation. For K_Means, Gaussian Mixture, and Affinity

Propagation algorithms, the Scikit-learn library was used while the Fuzzy_Cmeans algorithm was imported from the python fuzzy-c-means library. After trying a different number of k *clusters*, three were defined as parameter k for K_Means, Fuzzy_Cmeans, and Gaussian Mixture algorithms as the higher number did not provide more clusters. For Affinity Propagation, damping and preference are defined as 0.9 and -500, respectively.

As evaluation metrics, Silhuette_Coefficient and Calinski_Harabasz were used from the Scikit-learn library to evaluate the performance of the clustering algorithms. Principal Component Analyse was implemented for the selected features, and the result of four clustering algorithms was displayed in 2D using two components from this PCA result.

As the last step, the labels created by tracking the volume change in the subsequent CT scans were compared with labels created by the optimal clustering algorithm and measured how many labels were dropped in which cluster. The result of this comparison is then displayed using the Pie chart.

4.6 Supervised Methods

This section used CNN-based algorithms, simple CNN and ResNet, as supervised methods. After finding the best hyperparameters for both algorithms and training them, their performances were evaluated on the test set. The aim is that if we have a model with proper test sets score, we can employ our model for newly diagnosed patient CT scans to predict the treatment response.

4.6.1 Hyperparameter Optimization

As a first step of the supervised method, hyperparameters were optimized for both CNN and ResNet models using the Bayesian Optimization method. The hyperparameters subject to optimization are seen in the table below.

	CNN	ResNet
Optimization Method	Bayesian	Bayesian
Optimizer	Adam, SGD	Adam, SGD
Learning Rate	0.01 - 0.0001	0.01 - 0.0001
Batch size	4, 8, 16	4, 8, 16
Number of Layers	2, 3, 4	-
, Number of Filters	8, 16, 32	-
Model Depth	-	10, 18, 34, 50, 101
Activation Function	Softmax vs Sigmoid	-
Regularization (dropout rate)	0.1-0.3	-
Batch Normalization or not	+	+

Table 1: Hpyperparameters configuration setting for CNN and ResNet. As the table illustrates, model depth is used only for ResNet, and the number of layers, filters, and activation functions on the output layer, were used in the CNN as hyperparameters. This activation function was not included in the ResNet since softmax is embedded in the cross entropy loss function.
The number of filters was defined by multiplying one of the values defined in the table by 2 in the first convolutional layer, with an increase of multiplication of 2 per convolutional block. The CNN configuration in the table is related to the classification. The same configuration was used for the CNN regression model except for the activation function in the outcome layer.

In line with the above configuration, 20 runs were executed with 200 epochs for each model by monitoring the *validation loss* with *early stopping* (*patience* = 30) to find the optimal hyperparameter setting.

4.6.2 CNN

A simple 3D CNN architecture was used as the first model of the supervised part. Since our data structure allows us to build our model in two different types of supervised methods, the models were defined in both classification and regression, and the hyperparameters were optimized for each model.

In line with the result of the hyperparameter optimization process, since we optimized the number of layers and the activation function in the output layer, the optimal CNN was structured with three convolutional and pooling layers, including the batch normalization after each convolutional layer, one fully connected layer, and an output layer with a softmax activation function on it. Then the CNN classification model was trained with the optimal configuration settings: the *Adam* optimization function, a 0.007 learning rate, and 8 batch size.

Although the classification and regression model structures are almost identical, the only

difference between them is that in the output layer of the regression models, one output neuron is used instead of three, and no activation function is defined on it. Mean Squared Error is used as a loss function for the regression model instead of Categorical Cross-entropy.

4.6.3 ResNet

As a result of hyperparameter optimization, we optimized the model depth and 10 was found as an optimal depth. However, during the experience of finding the optimal data types and labeling methods, when we checked the training and validation loss, we realized that ResNet was performing better with model depth 50 than with model depth 10. Then, the experience continued with ResNet50. ResNet50 was then trained with the remaining optimal configuration settings: the *SGD* optimization function, a 0.004 learning rate, and 8 batch size.

4.6.4 Transfer Learning

Fine-tuning was used as a transfer learning method. The parameters of 3D ResNet50 from MedicalNet were used, which was created in Pytorch and trained on the chest CT images. After loading the pretrained model's parameters (weights and biases) to our ResNet50 model, a fully connected layer and output layer were added. Our model's training was started with these new parameters instead of random ones, and all layers were trained.

5 Results

This chapter presents the outcomes of the experiments. The findings will be discussed in the same order as the preceding chapter.

5.1 Unsupervised Methods

5.1.1 Feature extraction and preprocessing

Pyradiomics extracted n=1262 features per CT scan, and after eliminating the redundant and noisy features by using high correlation and low variation filters, the number of features was reduced to n=104. The high correlation filter eliminated most of the features as the number of features dropped to n=240. The low variance filter also eliminated n=136 features. The distribution of the selected features is seen in the below table.

Original	Wavelet	Square	Square Root	Exponential	LoG	LBP
17	43	11	9	7	11	6

First_order Statistic	Shape	Texture
58	6	40 - GLCM : 12 - GLSZM : 15 - GLDM : 6 - NGTDM : 3 - GLRLM : 4

Figure 9: Distributions of the pyradiomics selected features. The upper part of the figure shows the distributions of the selected features in terms of image types, while the bottom part is related to feature types. As seen in the figure, in terms of image types, most of the selected features belong to wavelet images; regarding the features types, most are from first-order statistics and texture-based features.

5.1.2 Result of clustering algorithms

The results of 4 clustering algorithms are seen in the below figure. Principal Component Analyses (PCA) were applied to reduce the dimensions of the features, and the result is displayed using 2 Principal Components(PC).

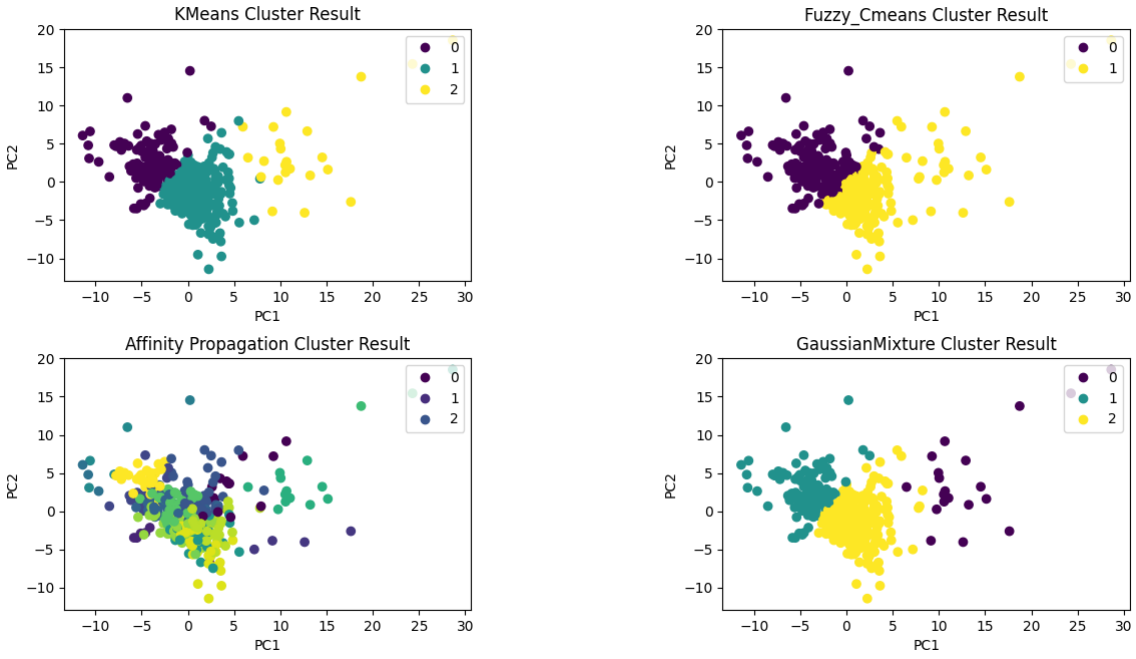


Figure 10: The result of 4 different clustering algorithms implemented to the 104 selected features of 426 CT scans. The figure illustrates that the Kmeans and Gaussian Mixture separated clusters better than the others.

	K-Means	Fuzzy-CMeans	Affinity Prop	Gaussian Mixture
Silhuette-Coeff	0.13	0.09	0.06	0.13
Calinski-Harabasz	59.05	32.40	25.56	58.97

Table 2: The result of Silhouette-Coeff and Calinski-Harabasz evaluation metrics for four different clustering algorithms. The result is consistent with our observation in the previous figure as the K-Means and Gaussian Mixture's scores are better than the other two.

Since the K-means score is slightly better than Gaussian Mixture, the result of the KMeans is used to compare the clustering result with the labels.

The result of 426 CT scan labeling, calculated in line with the previous section's description (using 20% thresholding and considering all follow-up CTs), is seen in Table 3 below.

Labels	Number of Labels
Stabel(SD)	248
Response(PR)	52
Progress(PD)	126

Table 3: The result of labeling. Most samples are from the stable class.

The result of 426 CT scans is also displayed on the two PCs using labels created by looking at the volume change in the follow-up CT scans. As can be seen, there is no clear border between the three classes.

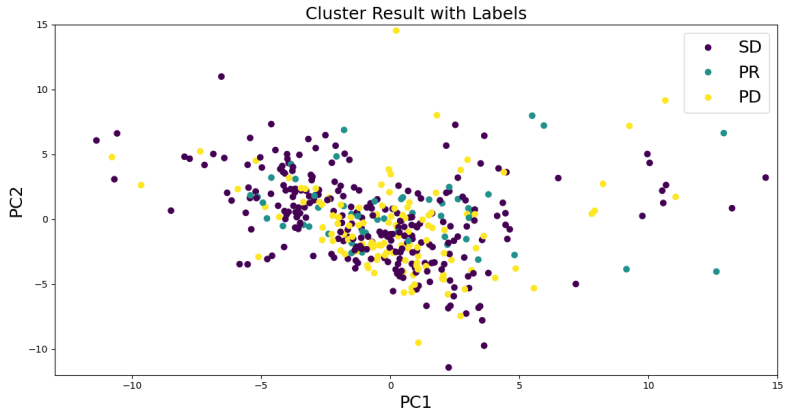


Figure 11: The result of labeled 426 CT scans. As the figure shows, there is no separation between the three classes.

After calculating the cluster labels and the classification labels for 426 CT scans, the results were compared to determine whether we could observe any overlap between the two labels. The below table shows to what extent the labels overlap with the clusters.

	1^{st} Cluster	2^{nd} Cluster	3^{rd} Cluster
Stable(SD)	99	142	13
Response(PR)	14	34	4
Progression(PD)	22	98	6

Table 4: Distributions of the labels among the clusters. The table shows that most of the classes are distributed in the first two clusters. Although the response class is more in the second cluster, it is almost the same as the first cluster in ratio.

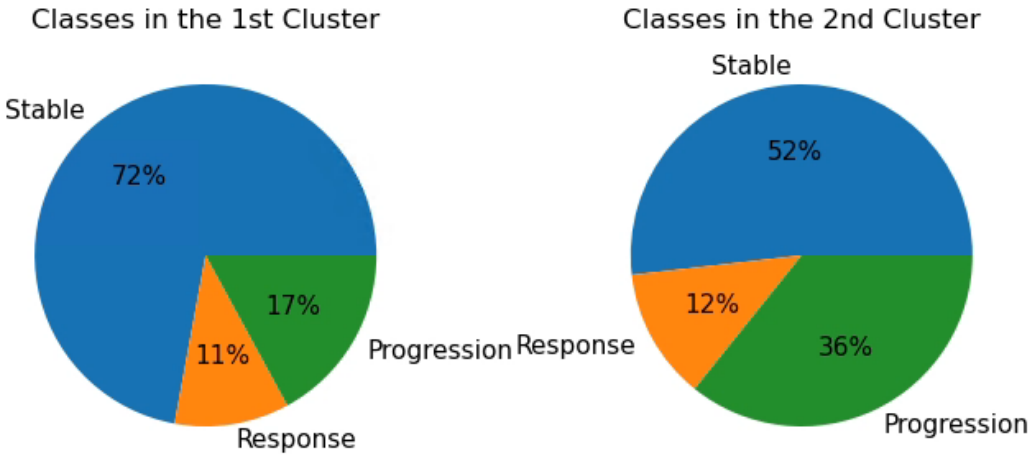


Figure 12: Distributions of the labels between two main clusters. As the pie chart depicts, although there are some differences between the two clusters in terms of stable and progress classes, the response classes are almost equally distributed between the two clusters.

When we look at the Pie chart that compares the two main clusters with classes, overall, the stable class dominates both clusters since most of the classes belong to stable. Although the progress rate is slightly higher in the second cluster, there are no significant differences in response rate between the two clusters.

5.2 Supervised Methods

In this section, after the result of hyperparameter optimization, the results of the CNN and ResNet models were explained, followed by ResNet transfer learning. In the last part, the model's performances were evaluated on the test sets using optimal hyperparameters, labeling methods, and data types.

5.2.1 Hyperparameter Optimization

Three hyperparameter optimizations were implemented for CNN classification, CNN regression, and ResNet. Models were created with the PyTorch library[25], and Wandb was used as a hyperparameter optimization platform[26].
The results of hyperparameter optimization for the CNN classification and ResNet are seen below.

	CNN_class	ResNet
Optimizer	Adam	SGD
Learning Rate	0.007	0.06
Batch size	8	8
Number of Layers	3	-
Number of Filters	8	-
Model Depth	-	10
Activation Function	Sigmoid	-
Regularization (dropout rate)	0.1	-
Batch Normalization or not	+	-

Table 5: The optimal hyperparameters configuration setting for CNN classification and ResNet.

The accuracy and loss figures 16- 20 and the tables 17- 21 that show hyperparameters and best evaluation scores for each run (sweep) are in the appendix. After analyzing the results in the appendix, the above configuration was decided as the optimal configuration of the hyperparameters.

While choosing the best model from the sweeps, the focus should be on the trade-off between the validation loss and training loss. Usually, the runs in which the validation loss kept decreasing concerning the training loss should be selected as optimal. But in our results, it is difficult to apply this rule since there is no improvement in the validation loss. Therefore, the optimal configurations were chosen by looking at the best training loss-validation loss combination.
Since the CNN regression results were very unstable, as seen in the appendix, the experience continued only with CNN classification and ResNet models.

5.2.2 Results with Different Labelling Methods and Data Types:

The models were trained with optimal hyperparameters, three different labeling methods, and three different data types to find the optimal data types and labeling methods. The first two data types are masked CT scans, and the CT scans cropped by a mask in a rectangular shape. As explained in the Method part, the third data type was created by feeding one of these two data types into two channels.

Three labeling methods used in this part are:

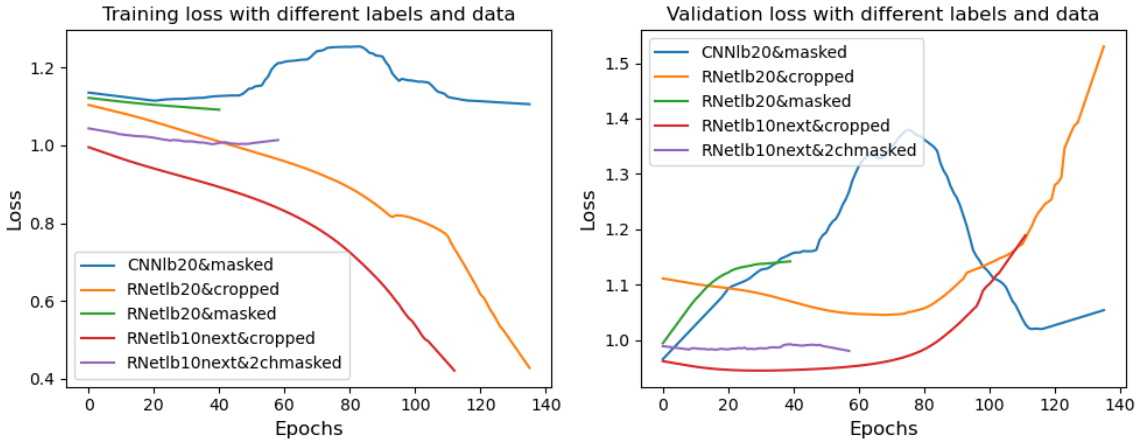
- **20&all**:20% thresholding by considering all follow-up CT scans
- **10&all**:10% thresholding by considering all follow-up CT scans
- **10&next**:10% thresholding by considering only the next CT scans.

The result of labeling 1003 CT scans in three methods is seen in Table 6 below.

Labels	10&all	20&all	10&next
Stabel(SD)	204	302	387
Response(PR)	192	152	109
Progress(PD)	395	337	266
Total	791	791	762

Table 6: The number of classes with different labeling methods. Although the class distributions are not unbalanced in the first two methods, they change in favor of the stable class when the threshold window widens. The total number of labeled CT scans was reduced from 1003 to 762 and 791 for the first two methods and third methods respectively since we could not labels the last CT scans.

Apart from the labeling methods, we aimed to find an optimized data type to reach the maximal model's performance. The results of the two models' performance with three different labeling techniques and data types are seen in the figures below.



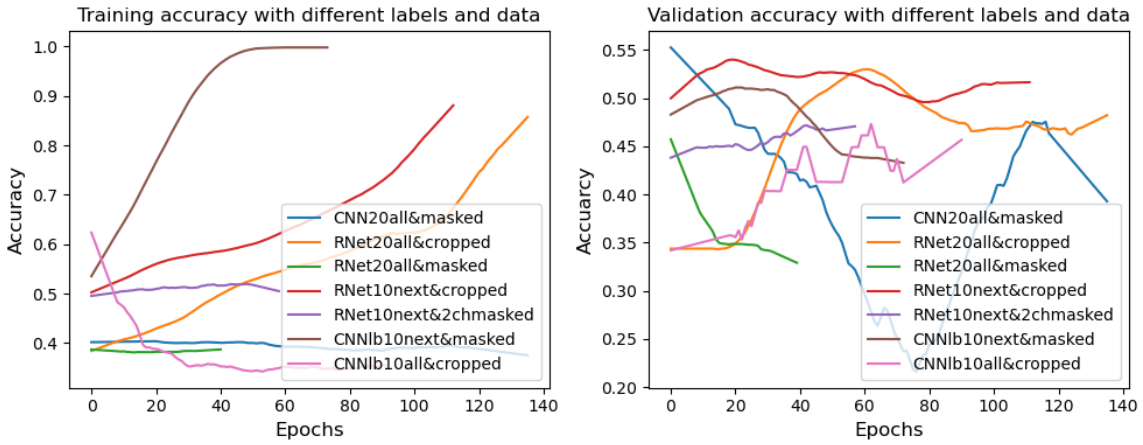


Figure 13: The two models' results using different labeling and data preprocessing methods. The figure shows the CNN classification and ResNet results on different labeling and data types. While the upper figures show how the loss value changes in each episode, the lower figures depict the accuracy value changes in each episode. The left figures illustrate the training results, while the right figures are related to the validation change.
As seen in the figure, concerning the labeling methods, the models trained using the 10%next labeling method performed better than those that used the 10%all and 20%all labeling. Regarding the data types, CT cropped by mask in a rectangular shape outperformed the models that used the masked CT scans and two channels data types.

In contrast to our expectations, the models did not perform well using two-channel data types. First, we created two-channel data types with CT cropped with a mask in a rectangular shape, and our model quickly overfits in this experiment. On the other hand, when we created a two-channel data type with masked CT, models did not learn from the data, as training and validation losses did not decrease enough.
Since the models trained by using CT cropped by mask in a rectangular shape as the data type and the labels 10%next outperformed the others, the model's performances were measured and compared on the test sets using this configuration.

5.2.3 Transfer Learning

After determining the optimal labeling and data type as 10%next and the CT cropped by mask in a rectangular shape, the transfer learning was implemented with this optimal configuration using the ResNet50 network and transferring pretrained parameters from MedicalNet's pretrained ResNet50 model. The result is displayed below.

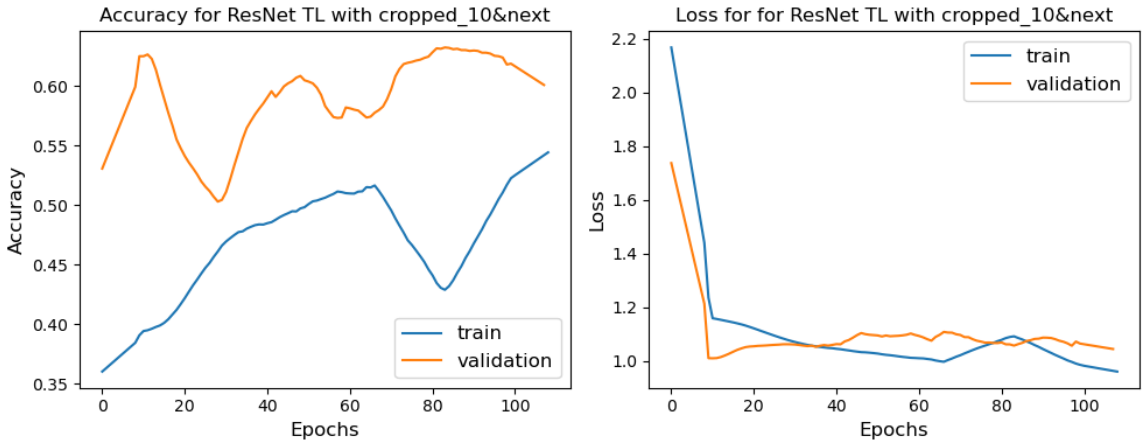


Figure 14: The result of Transfer Learning with ResNet 50. Graphs illustrate the change of train and validation sets accuracy and loss values through 100 epochs. As the figure illustrates, although transfer learning performance is not sufficient to be evaluated as promising, it gave more proper results than the other methods as accuracy increases, and loss values decrease for both training and validation sets when the episode number increases.

5.2.4 Results on the Test Set with Optimal Data, Labels and Hyperparameters

After training the three models using the best hyperparameters, optimal data type, and labeling methods, they were employed on the test set. The model's performances on the test set are seen in the table.

Network	Accuracy	Precision	Recall	f1-score	AUC	p-value
CNN_class	0.40	0.51	0.40	0.43	0.57	0.483
ResNet50	0.34	0.42	0.34	0.35	0.50	0.023
ResnNet_TL	0.60	0.48	0.60	0.52	0.70	0.423

Table 7: Performance of the optimal models on the test set concerning the accuracy, recall, precision, f1-score, and AUC. Since our models are multi-class classification, the precision, recall, and f1-score were calculated as weighted class average scores.

The table shows that transfer learning with ResNet50 outperformed the other two models regarding the accuracy, recall, f1-score, and AUC scores. Although the result of the ResNet50 trained from scratch seems to have a statistically significant p-value, it does not make sense since its accuracy is close to random prediction, 0.34.

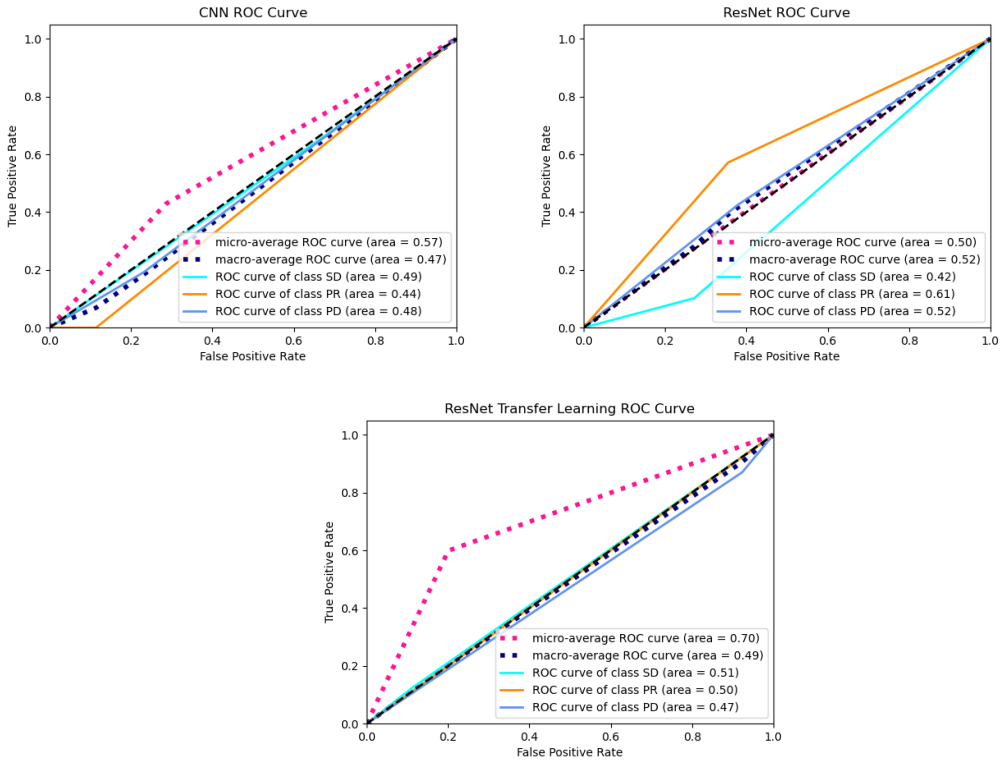


Figure 15: ROC curve of CNN, ResNet50, and ResNet50 Transfer Learning with the FPRs (False Positive Rate) on the x-axis and the TPRs (True Positive Rate) on the y-axis. As can be seen, although the CNN and ResNet models perform around a random classifier, which represents the diagonal line, the transfer learning result is better than the other two.

6 Discussion

This study aimed to predict whether the treatment for MPM patients will be effective based on baseline CT imaging. Supervised and unsupervised models were implemented in two parts.

While only one data type and one labeling technique were used in the unsupervised part, three different data types and three different labeling methods were used in the supervised part. Due to insufficient unique patients' CT, not only the baseline CT scans but any CT scan with at least one follow-up CT scan was employed for the labeling.

Regarding the unsupervised part, after extracting features by radiomics, most features were eliminated with the high correlation filter. The high correlation rate might be caused by the fact that most of the features belong to image types derived from the original image, leading to a high correlation among features.

Another notable point of the feature selection process is that most shape-related features were eliminated. It is challenging to segment MPM's tumor because of its heterogeneous shape and overall development along the pleura[1]. The less shape-related features in the selected features are consistent with the MPM CT's tumor shape heterogeneity. Therefore, using mostly first-order statistics and texture-based features was reasonable to get the proper model's performance.

Although those selected radiomics features led CT scans to be perfectly grouped into two main clusters, we couldn't observe a similar separation when we displayed our CT scans based on the labels created by looking up tumor volume change in the follow-up CT scans. Likewise, these labels did not overlap with those created by the cluster algorithms. The chosen cut-off for the labeling might be one of the possible reasons for this result.

When we compared the two main clusters with classes, there were no significant differences in response rate in the two clusters, as seen in the pie chart of Figure 12. Although the progression percentage is increased in the second cluster, this is not significant for predicting the treatment result of the MPM patient since we are looking for the response to dominate one of the clusters. As a result of the unsupervised part, our model could not predict whether the treatment would work for the newly diagnosed MPM patient based on its baseline CT image.

In the supervised part, we sought to predict the result of treatment by using trained ML models. For this, we first optimized the model's hyperparameters by using the training and the validation set, and we measured the model's performance on a separate test set. We aimed that if the model performs well on the test set, it can be employed on the unseen data, newly diagnosed MPM patients' CT scans in our case, and we can predict the treatment result for that patient.

After CNN and ResNet versions were deemed the best networks, hyperparameter optimization was performed to find the optimal configuration set. However, all configurations underperformed, leading to ambiguous results. Although it was difficult to choose the best configuration by looking at the trade-off between the validation and train loss since the validation losses were unstable, the best hyperparameter configuration was decided by looking at the optimal train-validation losses.

Concerning the data types, different results were observed using different data types. The optimal performance was achieved with the data type CT cropped by the mask in a rectangular shape. This data type not only consists of the tumor area but its surroundings as well within the rectangle. Although there can be some helpful information around the tumor area to predict the treatment result, it might contain noise, which deteriorates the model's performance and lead to the model's overfitting. On the other hand, its low accuracy on the train and validation test set indicates that using only the masked CT could not give enough information to the model. Therefore, experimenting with other data types which reduce the noise furthermore and contain the most informative parts of the image can increase the performance of the model.

Regarding the labeling, although we aimed to predict the treatment results by looking at the baseline CT of the MPM patients, due to data size constrain, we treated all the CT scans as a baseline which might affect the result of our methods. There are two variables in determining the labeling methods: the threshold that creates intervals to define categorical labels and how many follow-ups CT scans are to be considered to track the volume change. Our results indicate that using the 10% threshold and looking only at the next CT scans gives better results than the other methods. However, using only the baselines and finding a better labeling method might also increase the model's performance.

After determining the optimal hyperparameters, labeling method, and data types, the mod-

els were trained, and their performance was evaluated on the test sets. Although in most of the experiments, CNN and ResNet models quickly reach nearly 100 percent of training accuracy, which is an indication of overfitting, the validation accuracy score does not exceed the random prediction, which is around 37% in our case.
On the other hand, while the model trained from scratch with CNN and ResNet could not exceed the random prediction on the test set, transfer learning could achieve around 60% accuracy on the validation and test sets. The main reason for implementing transfer learning is to make up for insufficient data, as the deep learning method is data-hungry and does not perform well with inadequate data size. Therefore, we can make out from this result that if there is enough data, similar or even better results might be achieved with CNN and ResNet, which will be trained from scratch.

Consequently, the result of the second part, in general, was similar to the first part in predicting the treatment results of MPM patients. Although none of the results could be assessed as promising, the ResNet with transfer learning outperformed the other two.

We assessed that the following limitations might cause the above results. First, due to time constraints and a prolonged time to complete one run, the hyperparameter optimizations were run only 20 times for each model. Although bayesian optimization was used as the most reasonable method when the time is limited, given that the CNN model has nine different hyperparameters to be optimized, 20 running is too low to optimize all of them. On the other hand, since HPO is time-consuming, around 30 days, we started our optimization while still trying to determine the optimal data type and the labeling method. Therefore, the result of the HPO was not with the optimal data type and labeling method, which might affect the result of hyperparameter optimization.

Secondly, due to limited baseline CT scans, all the CT scans except the last ones were treated as a baseline and labeled by looking up the subsequent CTs. Since we are trying to estimate the treatment result of the newly diagnosed MPM patient based on the baseline CT, using only the baselines not only gives better features predictability but also provides a more realistic labeling process since we will have more follow-up CT scans to take into consideration.

Another limitation is the labeling method. An adequately labeled data set used as the objective standard to train and test a given model is sometimes called "ground truth" in ML. Because the quality of the trained model depends on the ground truth's accuracy, it is critical to have accurate data labeling. In our case, it is challenging to reach the ground truth since our labels can not be easily categorized, as is the case in many ML classifications, like classifying different objects or animals. Creating it depends on two variables, the threshold that creates intervals and the number of follow-up CT scans. We labeled data in line with the method and threshold currently used in the NKI, but research is still going on at the NKI to find optimal labeling methods. Although three different labeling methods were used in line with the advice from NKI to get reasonable labeling, this might be another reason for the result of our experiments.

Concerning future works, in the first part of the project, the features were extracted only from the masked CTs, and clustering algorithms were applied to these features. In future research, features can be extracted not only from the segmented part but from the whole scans and cropped scans, and the same experiment can be done by using features from these different data types.

Apart from the data types, in the first part, only the 20% threshold was used by considering the volume change in all follow-up CT scans when calculating the labels, and clusters were compared with these labels. Since the research is still going on about the proper labeling method, difference thresholding for the labeling can also be tried, which might increase the model's performance.

Likewise, although more options were tried in the second part, using optimal labeling and data types might increase the model's performance to get a promising result.
For example, as different data types, the multi-input models can be used by feeding the network with additional features. First, the volume change in percentage by looking at the next CT scan of the patients can be used as a second input. It can be fed to the classification layer directly with the features extracted from CT scans by convolutional layers. However, in this case, the treatment result can be predicted after treatment starts and getting at least one follow-up CT scan.

In addition to the volume change in percentage, the clinical data of the same patients, such as Electronic Health Records, can be used as third input directly to the classification layers.

Suppose promising results can be taken for estimating the treatment results of MPM patients in general. In that case, the experiment can be extended to predict the result of different treatment types, such as chemotherapy and immunotherapy.

7 Conclusion

We investigated how we can predict the MPM patient's treatment result based on the CT image by implementing two ML methods. We could not observe satisfactory results for the Unsupervised Clustering and Supervised Deep Learning models we implemented. However, some models' performance was improved by using different data types and labeling methods, signifying better results might be achieved by finding optimal labeling and data types. Moreover, the performance of the ResNet was increased by using transfer learning, which indicates that using more data also might give a better result.

To conclude, within the scope of this study, we could not uncover any promising results for predicting the treatment outcome of MPM patients based on CT scans using ML. However, given the relatively better transfer learning results and the possibility of discovering optimal labeling and data types, promising results with adequate data, ideal labeling, and data types might be achieved in future works.

References

[1] Pavic, M., Bogowicz, M., Kraft, J., Vuong, D., Mayinger, M., Kroeze, S., Friess, M., Frauenfelder, T., Andratschke, N., Huellner, M., et al. (2020). FDG PET versus CT radiomics to predict outcome in malignant pleural mesothelioma patients.

[2] Opitz, I. Weder, W. (2018). Pleural mesothelioma: is the surgeon still there? Annals of Oncology.

[3] Rusch, V. W., Gill, R., Mitchell, A., Naidich, D., Rice, D. C., Pass, H. I., Kindler, H. L., De Perrot, M., Friedberg, J., Ginsberg, M., et al. (2016). A multicenter study of volumetric computed tomography for staging malignant pleural mesothelioma. The Annals of thoracic surgery.

[4] Rafique, R., Riazul Islam, S. M., Kazi, J. U. (2021). Machine learning in the prediction of cancer therapy. Computational and Structural Biotechnology Journal.

[5] Keek SA, Leijenaar RT, Jochems A. Woodruff HC (2018). A review on radiomics and the future of theranostics for patient selection in precision medicine.

[6] Thieke C, Nicolay NH, Sterzing F, Hoffmann H, Roeder F, Safi S, et al (2015). Long-term results in malignant pleural mesothelioma treated with neoadjuvant chemotherapy, extrapleural pneumonectomy and intensity-modulated radiotherapy. Radiation oncology (London, England).

[7] Parekh V, Jacobs MA (2016). "Radiomics: a new application from established techniques". Expert Review of Precision Medicine and Drug Development.

[8] Zhang X., Zhang Y. et al(2022). Deep Learning With Radiomics for Disease Diagnosis and Treatment: Challenges and Potential.

[9] Horng H, Singh A, Yousefi B(2022). Generalized ComBat harmonization methods for radiomic features with multi-modal distributions and multiple batch effects.

[10] Torbati M.E., Minhas D.S., Ahmad G. et al(2021)., A multi-scanner neuroimaging data harmonization using RAVEL and ComBat.

[11] Hastie T., Tibshirani R., Friedman J.(2008). The Elements of Statistical Learning.

[12] Banerjee, Tanvi (2014). "Day or Night Activity Recognition From Video Using Fuzzy Clustering Techniques".

[13] Rendan J. Frey; Delbert Dueck (2007). "Clustering by passing messages between data points".

[14] Jolliffe, Ian T.; Cadima, Jorge (2016). "Principal component analysis: a review and recent developments". Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences.

[15] Géron A(2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.

[16] Venkatesan, Ragav; Li, Baoxin (2017). Convolutional Neural Networks in Visual Computing: A Concise Guide.

[17] He, K., Zhang, X., Ren, S., Sun, J. (2016). Identity mappings in deep residual networks.

[18] Matthias Feurer and Frank Hutter. Hyperparameter optimization. In: AutoML: Methods, Systems, Challenges, pages 3–38.

[19] Snoek, Jasper; Larochelle, Hugo; Adams, Ryan (2012). "Practical Bayesian Optimization of Machine Learning Algorithms"

[20] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson(2014). How transferable are features in deep neural networks? In Neural Information Processing Systems, pages 3320–3328,

[21] Lu, L., Dercle, L., Zhao, B. et al(2021). Deep learning for the prediction of early on-treatment response in metastatic colorectal cancer from serial medical imaging.

[22] Lia, H., Galperin-Aizenberga,M., Prymaa, D., Simone II, C.B., and Fana Y.(2018). Unsupervised machine learning of radiomic features for predicting treatment response and overall survival of early-stage non-small cell lung cancer patients treated with stereotactic body radiation therapy.

[23] Van Griethuysen, J. J. M., Fedorov, A., Parmar, C., Hosny, A., Aucoin, N., Narayan, V., Beets-Tan, R. G. H., Fillon-Robin, J. C., Pieper, S., Aerts, H. J. W. L. (2017). Computational Radiomics System to Decode the Radiographic Phenotype.

[24] Ye X, Ji K, and Sakurai T(2016). Unsupervised Feature Selection with Correlation and Individuality Analysis.

[25] Paszke, A. et al., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. Available at: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[26] L. Biewald, "Experiment Tracking with Weights and Biases," Weights Biases. [Online]. Available:https://wandb.ai/site.

8 Appendix

8.1 Hyperparameter Optimization Results

CNN Classification Hyperparameter Optimization Result.

Name (19 visualized)		bn	bs	dr	lr	nf	nl	opt	acc	batch loss	loss	val_acc	val_loss
ethereal-sweep-6	moid	bn	16	0.1097	0.006698	8	4	SGD	0.7872	-	0.6159	0.4896	1.505
stellar-sweep-11	ftmax	no_bn	4	0.2427	0.002042	16	4	SGD	0.7282	-	0.6556	0.3646	1.809
floral-sweep-9	moid	bn	4	0.1824	0.0008847	8	3	SGD	0.8423	-	0.456	0.3438	2.051
sparkling-sweep-1	moid	bn	16	0.1722	0.00113	16	4	SGD	1	-	0.0008355	0.4219	2.295
royal-sweep-7	ftmax	bn	4	0.1803	0.003158	16	4	SGD	0.8775	-	0.3061	0.4427	2.356
rare-sweep-3	moid	no_bn	4	0.1246	0.008987	8	4	Adam	0.6544	-	1.177	0.25	2.606
bumbling-sweep-2	moid	no_bn	16	0.2153	0.007022	8	3	Adam	0.9139	-	0.2577	0.4479	3.042
rare-sweep-5	moid	no_bn	16	0.1079	0.0002365	16	3	Adam	0.9713	-	0.1662	0.4063	3.277
silvery-sweep-4	ftmax	no_bn	16	0.1066	0.003145	16	3	Adam	0.7956	-	1.12	0.4271	3.517
vibrant-sweep-8	ftmax	no_bn	8	0.2585	0.0003249	8	3	Adam	0.8345	-	0.6463	0.4271	4.757

Figure 16: The result of the first 10 runs with Bayesian optimization for CNN classification Hyperparameter Optimization. Although it was difficult to choose the optimal settings as the validation loss is unstable, the bumbling-sweep2 was selected as the optimal run by choosing the most acceptable training loss and validation loss combination.

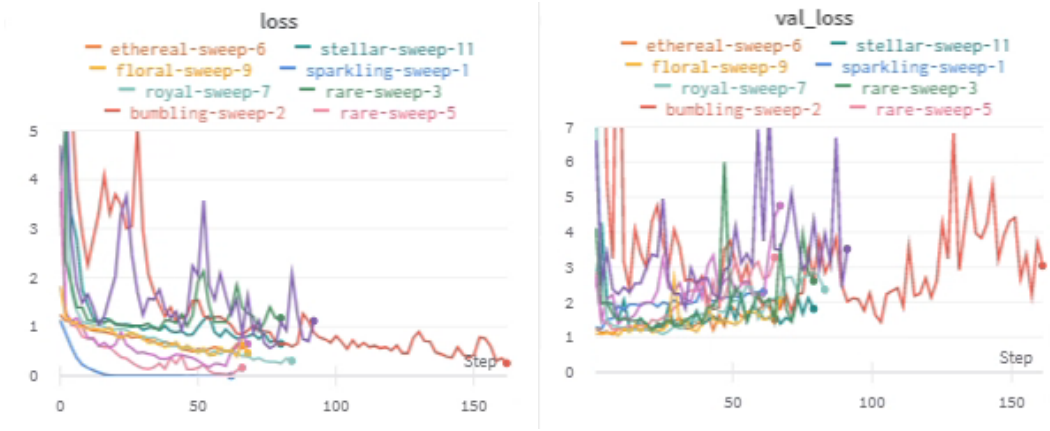


Figure 17: The result of the first 10 runs for CNN classification hyperparameter optimization. The figure shows the performance of the CNN classification model with different hyperparameter configuration settings. The left figure shows how each episode's training loss value changes with different hyperparameter settings. The right figure shows the change in the validation loss in each episode for different hyperparameter settings. As seen in the figure, the best performance is observed with bumbling-sweep2

CNN Regression hyperparameter Optimization Result

Name (20 visualized)	intime	Sweep	bn	bs	dr	lr	nf	nl	opt	train_acc	train_loss	val_acc	val_loss
fresh-sweep-6	h 5m 43	ufxspe5g	-	8	0.2717	0.04206	8	4	-	0.3786	0.4899	0.2961	0.6904
fluent-sweep-2	h 24m 5	ufxspe5g	-	4	0.1625	0.01117	16	4	-	0.346	6.377	0.3269	7.143
rare-sweep-1	h 25m 4	ufxspe5g	-	4	0.2531	0.03097	8	4	-	0.3442	93.421	0.2628	57.358
daily-sweep-4	2h 41m	ufxspe5g	-	4	0.2451	0.03727	8	4	-	0.3514	4627.505	0.3013	6411.643
good-sweep-3	h 48m 5	ufxspe5g	-	4	0.2814	0.04697	8	3	-	0.3659	20165.231	0.3397	15321.744
fallen-sweep-5	1h 25m	ufxspe5g	-	4	0.224	0.04567	16	3	-	0.3025	976207.90	0.2756	1043670.8
unique-sweep-11		ufxspe5g	-	4	0.289	0.04368	8	4	-	-	-	-	-

Figure 18: The result of 7 runs for CNN regression hyperparameter optimization. As seen in the figure, since the model's performance is extremely volatile, it is not easy to choose the optimal run

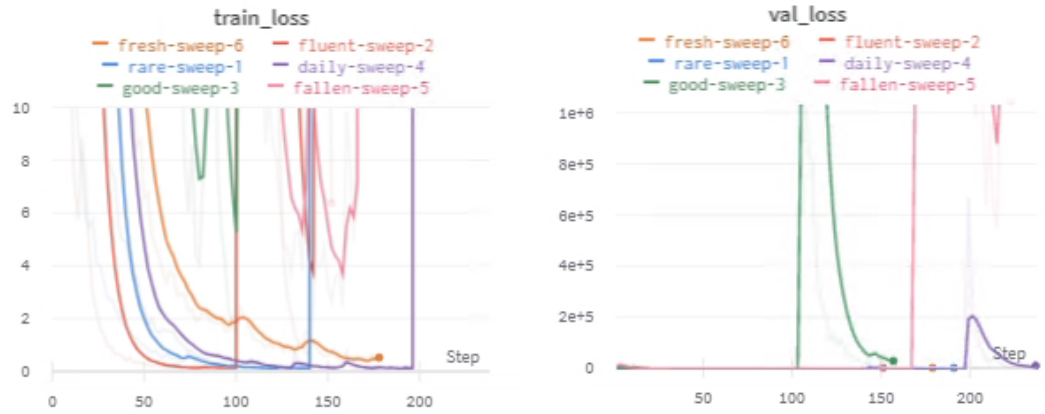


Figure 19: The result of 6 runs for CNN regression hyperparameter optimization. The figure shows the performance of the CNN regression model with different hyperparameter configuration settings. The left figure shows how the training loss value changes with different hyperparameter settings. The right figure shows the change in the validation loss in each episode for four different hyperparameter settings. As seen in the figure, since the model's performance is extremely volatile, it is not easy to choose the optimal run

ResNet Hyperparameter Optimization Result

Name (20 visualized)	Tags	Created	Runtime	Sweep	bs	lr	model_dep	opt	acc	batch loss	loss	val_acc	val_loss
earthy-sweep-6		3w ago	10h 12m 4	l9yssbzv	8	0.06992	101	Adam	0.4155	1.21	1.073	0.5417	0.9928
visionary-sweep-1		3w ago	8h 48m 56	l9yssbzv	4	0.02078	50	Adam	0.3674	0.9899	1.079	0.5208	1.011
chocolate-sweep-2		3w ago	9h 8m 52s	l9yssbzv	8	0.08569	101	Adam	0.3885	1.093	1.082	0.3854	1.039
chocolate-sweep-4		3w ago	7h 51m 18	l9yssbzv	8	0.002576	101	Adam	0.7601	0.4337	0.5921	0.4323	1.37
glamorous-sweep-3		3w ago	10h 27m 1	l9yssbzv	8	0.06499	101	SGD	0.8142	0.3719	0.4188	0.3333	2.347
effortless-sweep-12		2w ago	9h 10m 52	l9yssbzv	4	0.06064	10	SGD	0.9983	0.0004328	0.01316	0.3802	2.662
driven-sweep-20		2w ago	6h 24m 38	l9yssbzv	8	0.06639	10	SGD	1	0.003355	0.01265	0.4427	3.106
vague-sweep-14		2w ago	8h 2m 49s	l9yssbzv	8	0.08145	10	SGD	1	0.0007521	0.00384	0.3906	3.224
fallen-sweep-15		2w ago	8h 1s	l9yssbzv	8	0.06868	10	SGD	0.9983	0.003838	0.01013	0.3958	3.283
sleek-sweep-8		2w ago	6h 57m 37	l9yssbzv	4	0.07272	10	SGD	1	0.0002821	0.005671		

1-20 of 20

Figure 20: The result of the first 10 runs with Bayesian optimization for ResNet hyperparameter optimization. Although it was difficult to choose the optimal settings as the validation loss is unstable, the Effortless-sweep12 was selected as the optimal run by choosing the most acceptable training loss and validation loss combination.

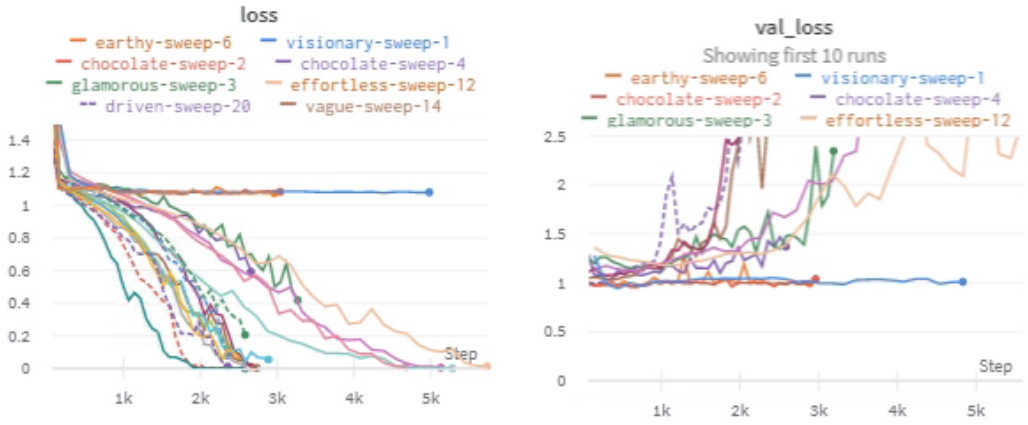


Figure 21: The result of the first 10 runs for ResNet hyperparameter optimization. The figure shows the performance of the ResNet model with different hyperparameter configuration settings. The left figure shows how each episode's training loss value changes with different hyperparameter settings. The right figure shows the change in the validation loss in each episode for four different hyperparameters settings. As seen in the figure, the best performance is observed with Effortless-sweep12