Research Academy for Young Scientists

Machine Learning Methods for Lung Tumor Diagnosis

Carl Viggo Nilsson Gravenhorst-Lövenstierne
carlviggo@icloud.com

under the direction of
Dr. Mehdi Astaraki

Stockholm University
Department of Medical Radiation Physics

July 12, 2023

*

Abstract

Lung cancer stands as one of the deadliest forms of cancer. In 2020 alone, the disease claimed the lives of more than 1.8 million individuals globally [1]. Improvements in the field of medical imaging technology has led to an accumulation of cancer image data worldwide. However, the traditional approach of identifying and diagnosing cancer still relies on manual evaluation of such data. This workflow is time consuming and prone to errors.

In an attempt to help clinicians, computer models for cancer classification have recently been developed. The objective of this study is to investigate these models by focusing on machine learning algorithms specifically developed for the diagnosis of lung cancer.

This study reveals that, based on the analysis of the Kaggle Data Science Bowl 2017 data set [2], the random forest algorithm outperforms the LeNet, AlexNet and VGG16 convolutional neural network architectures. However, if the deep learning models had undergone further training or if the data set had been more extensive in size, then the results might have been different.

Acknowledgements

I would like to express my gratitude to my mentor Dr. Mehdi Astaraki for his continuous guidance throughout the course of this project. I am also grateful for the many inspiring discussions with my co-worker, Daria Chamoun, and for the proofreading provided by Aleksander Purik. Furthermore, I extend my thanks to Stiftelsen Hierta-Retzius Stipendiefond and Stiftelsen Oscar och Maria Ekmans Donationsfond for making the summer research school possible. Finally, I am thankful to Rays — for excellence, for providing me with the opportunity to participate in the summer research program of 2023.

Contents

1 Introduction

In 2020, cancer caused more than 10 million deaths worldwide [1]. As our lifestyles change and the global population ages and grows, this number is expected to almost double by 2040 [3]. Among the various types of cancer, lung carcinoma, i.e. cancer in the respiratory region, is the most deadly [1]. If the disease is detected early, the chances of survival increase significantly [1].

Improvements in medical imaging technology have made it possible to capture vast quantities of high resolution medical images from patients. This has led to an accumulation of cancer image data worldwide. Conventional methods of assessing abnormalities in medical images include uncertain, labour intensive methods such as manual image observation. In the image data, the presence and type of cancer is not always easily detectable, which increases the risk of misdiagnosis. This uncertainty could prohibit patients from being diagnosed early in the disease course, and thereby impact their survival chance negatively.

When cancer is detected, the malignancy of any observed tumor is classified. The next step is to identify the tumor characteristics. For instance, there exists a complex microenvironment within cancerous regions which represent different level of aggressiveness [4]. In some cases, such characteristics can be studied by conducting an invasive surgery known as biopsy. When a tumor is incorrectly classified and its malignancy goes undetected, it poses additional risks to the patient.

Computer based models have recently been developed to assist clinicians with their daily work. This approach relies upon novel imaging processing techniques and machine learning models. In this project, conventional machine learning algorithms such as random forests and convolutional neural networks have been applied to classify benign and malignant tumors in the lung region.

Background

A list of prerequisites and their corresponding explanations can be found in this section.

1.1 Quantitative Imaging Biomarkers

Due to recent improvements in medical imaging technology, for example magnetic resonance imaging or computed tomography scans, the information content within medical images has undergone substantial growth in recent years [5]. This enhanced level of information can be utilized as a quantifiable indicator of potential disease.

Medical images provide 2D or 3D quantitative imaging data from the inside of human organs. Such information can be further processed to be used for diagnosis, prognosis and prediction purposes in a process is known as QIB, quantitative image biomarking. A subfield of QIB is radiomics, in which particular image features are extracted, processed and then analyzed. [6]

These features must be chosen with respect to the set of data and the purpose of the study, which in this project is lung tumor classification. Examples of extractable features are tissue density, image entropy — a metric of how diverse the pixel values are; and image energy — a measure of variance in pixel intensity . The intensity corresponds to the density of the tissue that is captured. One method of feature extraction is the appliance of filters. The sequential feature selection filter, SFS, is an effective technique to find and keep only the most informative features and remove the less predictive ones. Another filter is the least absolute shrinkage and selection operator (lasso) which assigns additional weights to relevant features based on their predictive powers. Lastly, the principal component analysis (PCA) is a dimensionality reduction filter which involve multiple matrix operations. Out of the thousands of features that exist, deep learning models can be trained to extract the most relevant features in respect to the data set. In other words, the neural networks do the feature extraction part automatically. [7]

1.2 Random Forests and Decision Trees

Decision trees are supervised machine learning algorithms that are implemented for regression and classification tasks. They consist of a series of conditional statements which split and thereby sort the data. The following schematic flowchart represents an example of a decision tree:

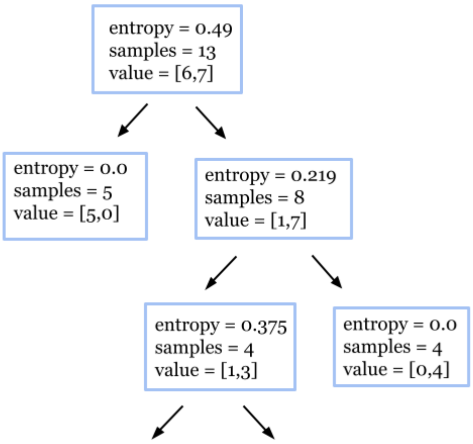

Figure 1: Subunit of a decision tree. The nodes are represented by boxes and the branches by arrows. The value parameter stands for [number of classes, number of samples].

The model determines the condition by comparing the entropy in the parent node against the entropy in the child node. The splitting condition that generates the largest difference in entropy, also called the information gain, is finally chosen. The information gain is defined as:

$$\text{Information Gain} = \Delta S = \sum_{i=1}^{c} p_i \log_2 p_i - \sum_{j=1}^{c} p_j \log_2 p_j \tag{1}$$

In which p is the probability of randomly choosing a data point in class i and c is the number of classes. The splitting continues until the entropy of all leaf nodes is 0 or until the parameters of the model are satisfied. To determine the information gain, all possible splitting combinations must be tested, resulting in decision trees to be relatively slow and inappropriate for large sets of data. Another weakness inherent to decision trees is that unless a maximum tree depth is defined, the tree will continue to split the data until all leaf nodes are in a state of zero entropy [8].

Random forests are supervised machine learning models that are generally less prone to overfitting than decision trees. In the initialization of the model, randomized features are extracted from a sub section of the data set in a process called bootstrap aggregation. These features are fed to a set of n decision trees that are inherent to the model. The class selected by most trees becomes the final output from the random forest. The adoption of randomness from the bootstrapping improves the generalization capabilities of the model and thus improves its robustness [9].

1.3 Artificial Neural Networks

Artificial neural networks are data structures inspired by the biological nervous system used for function approximation purposes. The most primitive unit of a neural network is the perceptron, see figure 2.

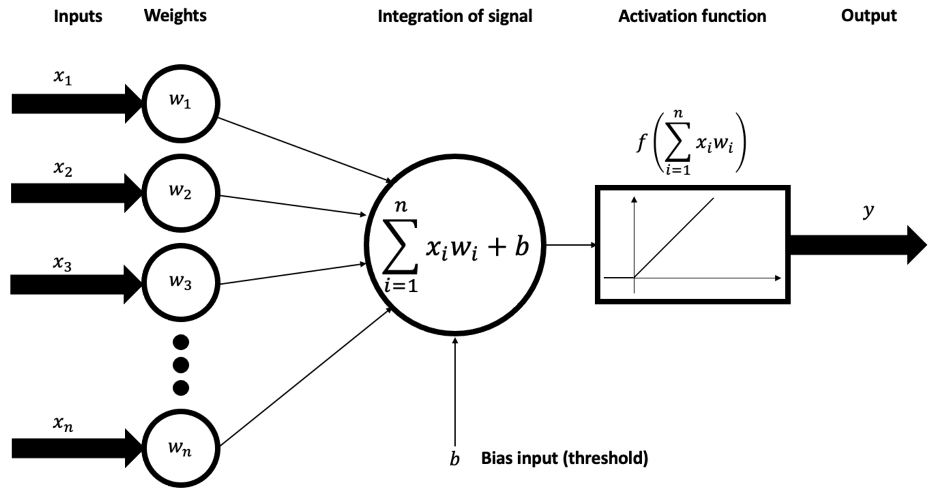

Figure 2: Schematic representation of a perceptron [10], including input data, x_n; weights, w_n; biases, b; weighted sum and activation function. (CC BY-SA 4.0)

In the instantiation of the model, each node-connection is assigned a randomized weight and bias. The input in the form of a image-matrix is then multiplied with the weights and summarized in the following mathematical operation called the weighted sum, where b is the bias:

$$\sum_{i=1}^{n}(x_i w_i + b) \tag{2}$$

The model must be introduced to a dimension of non-linearity to be able to adapt itself to the input data. This is achieved by the activation function which takes the weighted sum as input. Its output either serves as input to a new perceptron or as the final classification of the model. If the activation function had been linear, then the model would only be able to learn linear relationships between the input and outputs.

A neural network is composed of an input layer, at least one hidden layer with n neurons and an output layer. In a densely connected neural network, every node is interconnected by weights. The parts of a neural network which must be manually configured, e.g. the number of nodes per layer or the activation function, are called hyperparameters. The hyperparameters make up the model architecture and can influence the performance of the model significantly. Choosing the right hyperparameters is often the main challenge in constructing deep learning networks.

1.4 Loss Functions and Optimizers

Loss functions and optimizers are components of neural networks that are essential to their training process. In supervised neural networks, the loss function, for example mean squared error or cross entropy loss, is used to calculate the difference between the predicted class and the true class. An optimizing function numerically derives the loss function, with respect to its hyperparameters. This derivative helps the optimizer to tune the weights of the neural network in a process called backpropagation. Simultaneously, the input data is passed to the model n times, also called the number of epochs. This cycle is known as model training, and ideally continues until the loss function converges at one of its local minimum values.

1.5 Convolutional Neural Networks

A convolutional neural network (CNN) automatically extracts features that are relevant for the classification of the image data. A typical convolutional neural network architecture consists of convolutional layers, see figure 3, that are alternated by pooling layers. This structure is finally interconnected with a densely connected neural network which gives the final classification.

The convolutional layer, a subunit of the convolutional neural network, initially convolves a tensor into feature maps that contain the extracted features from the initial tensor. In the domain of greyscaled medical images, this tensor is a two dimensional matrix, see figure 3.

The convolving operation involves a filter matrix, also known as a kernel, that discretely moves over the input matrix. For each step, the sum of the dot product between the input matrix and the kernel is calculated and saved in the feature map. In the initialization of the model, the kernel values are randomized. Yet, as the model backpropagates, these values are tuned by the optimizing function in the training process to extract increasingly complex attributes. The convolving operation is summarized in equation 3.

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(T_{ij} \cdot K_{ij}) \tag{3}$$

Where K_{ij} is the kernel and T_{ij} is the input tensor with the dimensions $m \times n$. The position of the kernel is represented by $i \times j$. An approach used in this project is to apply a kernel of dimensionality 3×3. This example is visualized in figure 3, where the variables i, j, m, n equal 3.

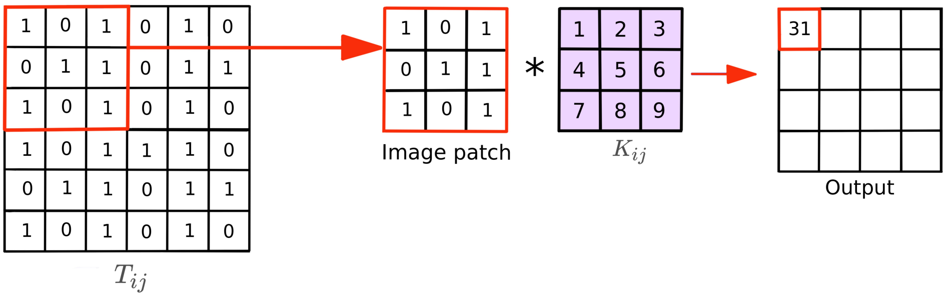

Figure 3: Structure of a convolutional layer. The sum of the cross products between the image patch and the kernel is calculated and projected onto the feature map.

Multiple feature maps can be extracted from a single input matrix. The feature maps are either transferred to another convolutional layer or to a neural network, where they are ultimately classified. To minimize computing time, the feature maps can be passed through a pooling layer, where their dimensionality is reduced.

Figure 4 visualizes the transformation process of an image after two convolutions, g_1 and g_2, in the lung cancer data set:

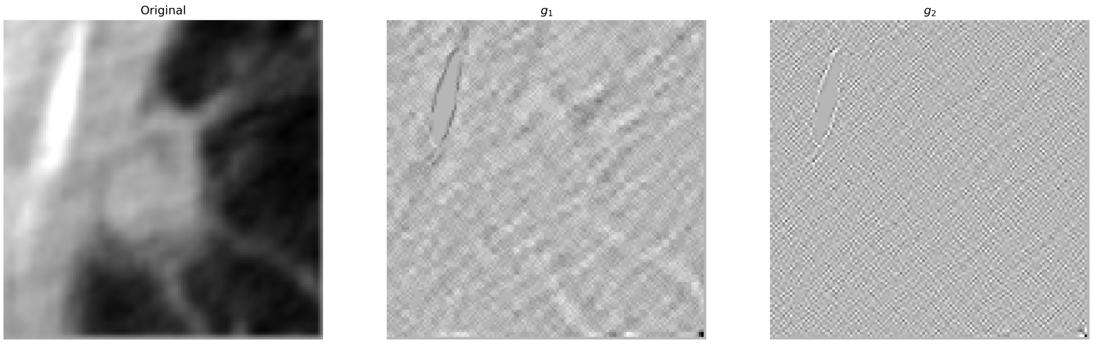

Figure 4: Application of two convolutional layers.

1.6 Image Augmentation, Feature Learning, Transfer Learning

A common phenomena for machine learning models is to develop more parameters than what is realistic for the data they are trained on. Just like a polynomial can be modified to fit n data points, a machine learning model can overadjust itself and thereby lose its ability to accurately classify new data. This phenomena is called overfitting, and can be partly counteracted by image augmentation, feature learning and transfer learning.

Image augmentation is a technique that can be implemented to increase the size of the training data set. This is achieved by applying geometrical or intensity transformations to the data with techniques such as filter application, rotation, flipping and zooming. When dealing with limited data sets, large number of features increases the risk of overfitting. By reducing features that are associated to one another, the authenticity of the data set increases.

As the architecture of a convolutional neural network gets increasingly intricate, the model transitions from being able to detect primitive features such as edges, corners and diagonals to more complex features. The better a model is at detecting these primitive structures, the better suited it will be for more advanced tasks. Pre-trained models trained on extensive datasets such as the ImageNet [11] can be imported and integrated to a preexisting neural network to improve its generalization capabilities.

1.7 Model Evaluation: Metrics

When validating a machine learning model, various metrics are used to quantify its performance. In the manual optimization of the model, this information gives an indication of what hyperparameters that needs to be tuned. While a single metric can be misrepresentative, combining multiple can give a more accurate representation of the model performance. One method for model optimization is hyperparameter tuning, where hyperparameters are adjusted while measuring the model's performance. Another method for model evaluation is K-fold cross-validation. This algorithm trains the model multiple times, using different validation data each time, and thus reduces the risk of biased datasets.

A prerequisite for calculating many metrics is the creation of a confusion matrix, which includes the four possible outcomes of a binary classifier: true positive, false positive, true negative, and false negative. To clarify, a false positive diagnosis occurs when a patient's tumor is predicted as malignant when it is actually benign.

1.7.1 Accuracy

Accuracy is a metric used to measure the proportion of correctly classified predictions. The definition of accuracy is found in expression 4.

$$Accuracy = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \tag{4}$$

Where T_P, T_N, F_P and F_N correspond to true positive, true negative, falsely positive and falsely negative.

1.7.2 AUC: Area Under the Curve

The AUC value is a measurement of the model's ability to distinguish between positive and negative classes. The AUC is identical to the area under the graph that is created as the false positive rate, $\frac{F_P}{F_P+T_N}$, is plotted against true positive rate, $\frac{T_P}{T_P+F_P}$. For each epoch in the training process of the model, a confusion matrix is generated, from which the graph is plotted. The closer the AUC value is to 1, the better the value. See figure 5.

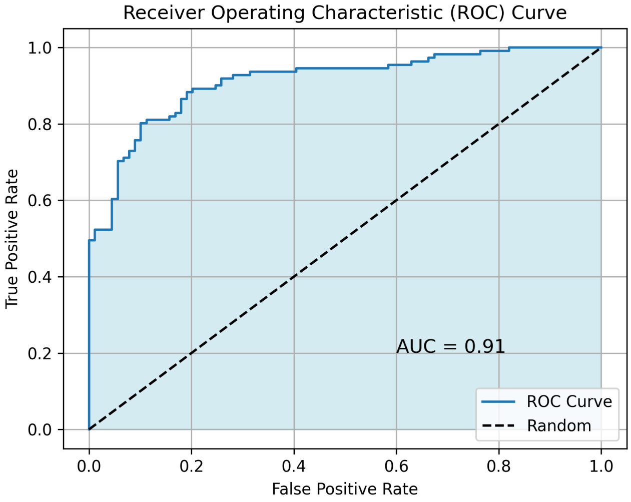

Figure 5: Area under the curve, AUC, generated from a logistic regression task.

1.7.3 Loss

The loss-function is an essential part of the learning algorithm which quantifies the mismatch between the model's predicted outputs and the actual true values. Furthermore, loss can serve as a certification of the accuracy metric, particularly in scenarios involving imbalanced datasets.

1.8 Previous Studies

The field of deep learning as an extention of machine learning emerged in 2006. In 2012, the AlexNet architecture was invented by computer scientist Alex Krizhevsky and soon thereafter, the VGG16 architecture was introduced at Oxford University. [12]

Since then, a variety of increasingly efficient convolutional neural network (CNN) architectures have been developed. Figure 6 visualizes the progress in performance of different CNN architectures on the ImageNet data set.

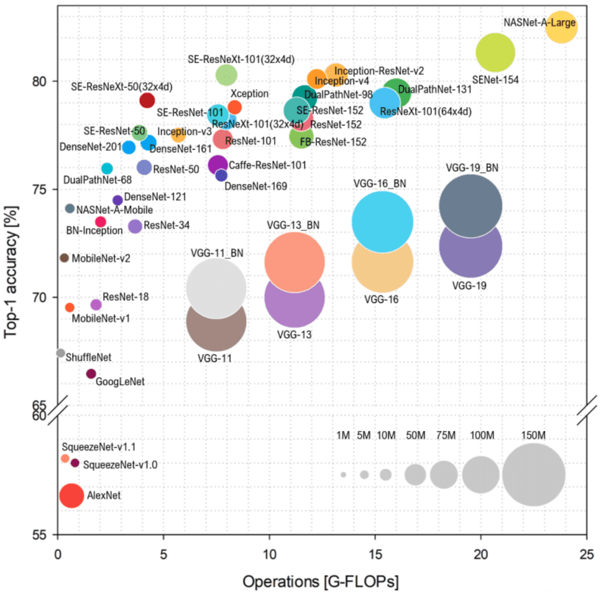

Figure 6: The graph compares CNN architectures based on their accuracy (y-axis) and the computational power measured in gigaflops (x-axis) [13]. (ArXiv License)

2 Method

The project pipeline included data preprocessing, radiomic feature extraction, hyperparameter tuning, synthetic minority oversampling techniques, k-fold cross validation analysis, dimensionality reduction, convolutional neural networks with AlexNet, LeNet and VGG16 architectures and model validation.

2.1 Data Preprocessing

The Kaggle Data Science Bowl 2017 data set [2], containing 1297 images of benign and malignant pulmonary nodules, was used in this project. Figure 7 contains 32 samples of the data set, where the the labels 0 and 1 correspond to the classes benign and malignant respectively.

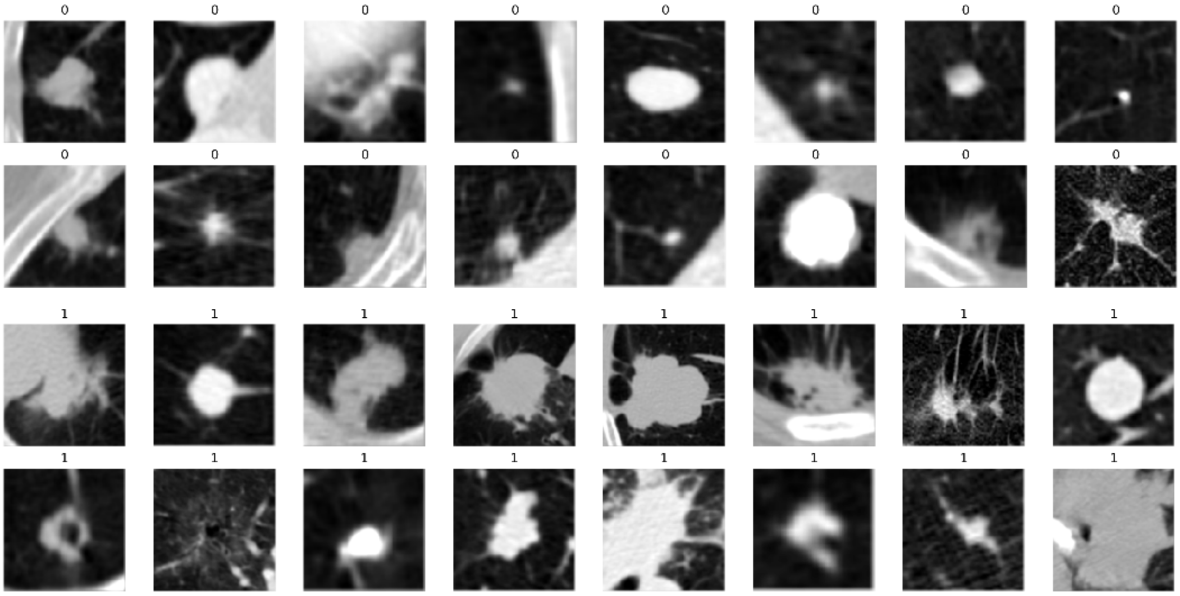

Figure 7: 16 samples from the Kaggle Data Science Bowl 2017 data set [2] used in this study for visualization purposes. A title of 1 corresponds to the malignant class, while a title of 0 corresponds to the benign class.

2.2 Radiomic Feature Extraction

A feature extracting method from the radiomics library was implemented to extract 1070 radiomic features from the Kaggle Data Science Bowl 2017 data set, containing low-dose computer tomography images of the chest region from lung cancer patients.

2.3 Hyperparameter Tuning

The random forest machine learning model was instantiated. The hyperparameters maximum tree depth and number of estimators were tuned combination wise via the GridSearchCv module to optimize the performance of the model. See results in figure 12.

2.4 Synthetic Minority Oversampling Techniques

A synthetic minority oversampling technique (SMOTE) was applied to counteract the imbalance between the two classes of the data set, benign (67.5%) and malignant (32.5%). This imbalance had a degrading effect on performance for the minority class. The SMOTE algorithm creates synthetic data points by interpolation to even the relationship between the classes.

2.5 K-fold Cross Validation Analysis

Five-fold cross validation was applied to the data set to confirm the absence of statistical errors when splitting the data into training and validation sets. For each validation, the mean accuracy and AUC were measured. See results in table 2.

2.6 Dimensionality Reduction and Feature Selection

To reduce the risk of overfitting, the following filter methods were applied to the data set: least absolute shrinkage and selection operator (lasso), dimensionality reduction with principal component analysis (PCA) and sequential feature selection (SFS). Five-fold cross validation was finally applied to evaluate the performance of the model. See results in table 2

2.7 Convolutional Neural Network - LeNet

A convolutional neural network with the LeNet architecture, see figure 8, was configured via TensorFlow Keras [14] and applied to the data set. The loss value and the binary accuracy were measured.

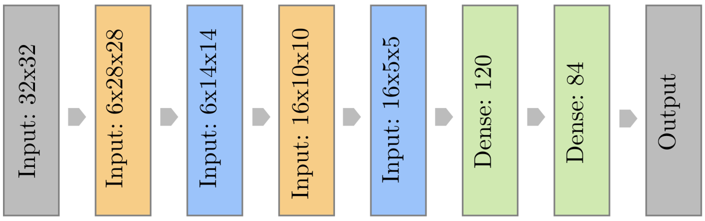

Figure 8: LeNet architecture composed of 2 convolutional layers, 2 max pooling layers and 2 densely connected layers.

2.8 Convolutional Neural Network - AlexNet

The AlexNet architecture was instantiated, see figure 9, and validated on accuracy and loss. See results in figure 11.

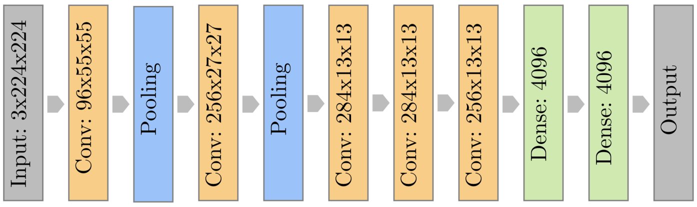

Figure 9: AlexNet architecture composed of 5 convolutional layers, 2 max pooling layers and 2 densely connected layers.

2.9 Transfer Learning with VGG16

A convolutional neural network with the VGG16 architecture, see figure 10, pre-trained on the ImageNet data set [11], containing 14,000,000 images was imported from the keras library [14]. The VGG16 model was integrated with a three layer densely connected neural network.

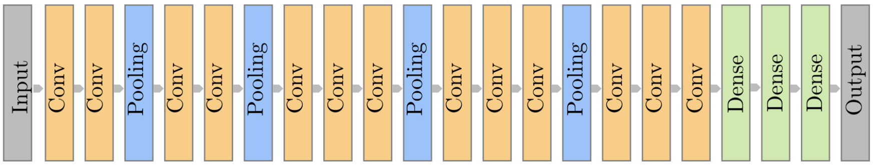

Figure 10: VGG16 architecture composed of 13 convolutional layers, 4 max pooling layers and 3 densely connected layers.

3 Results

Results from the project are presented in this section. In most graphs, the mean value after the graph has reached a plateau is plotted with decision boundaries of ± one standard deviation.

3.1 Random Forest Metrics

The AUC of the random forest model with extracted features was measured in an interval of 0-100 estimators, i.e. number of decision trees, and 0-50 maximum depths.

In the left graph of figure 11, the variance around the mean value seems to follow a gaussian distribution, since 80% of the data points are found within the first standard deviation and the rest within the second. This suggests convergence around the mean AUC of 0.764.

In the right graph of figure 11, the AUC initially increases rapidly and reaches a plateau after ∼20 estimators. At that point, roughly 70% of the data points are positioned within the first boundary from the mean. As the number of estimators increase, the number of data points that exceed the first standard deviation appear to decrease. These two aspects suggests convergence around the mean AUC of 0.781.

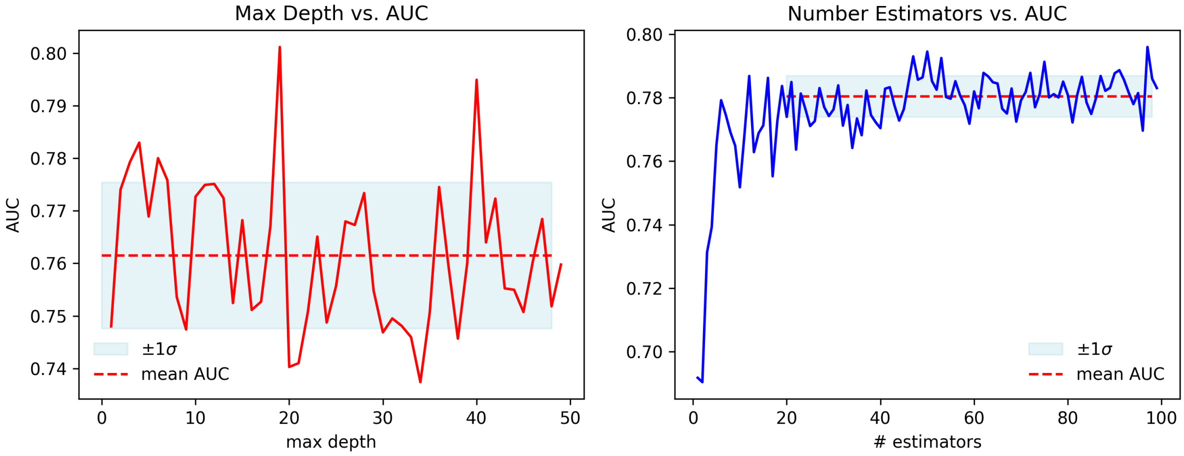

Figure 11: Random forest learning curves. The model to the left is configured with 20 estimators while the model to the right is configured with a maximum depth of 5. The data in both graphs originates from the validation set of the data.

A summary of the metrics in figure 11 is presented in table 1.

Table 1: Random forest - statistics summary.

Hyperparameter	Mean AUC	Mean Accuracy
Number Estimators	0.781 ± 0.006	0.780 ± 0.007
Max Depth	0.764 ± 0.014	0.725 ± 0.019

Figure 12 visualizes the graph that is generated when the two hyperparameters from figure 11 are combined. An optimal AUC value of 0.91 is achieved when the number of estimators is 41 and the maximum depth is 21. This could be a statistical anomaly. What's of importance is that the plane appears to reach a plateau as both hyperparameters exceed a value of 20, indicating that no more training is needed.

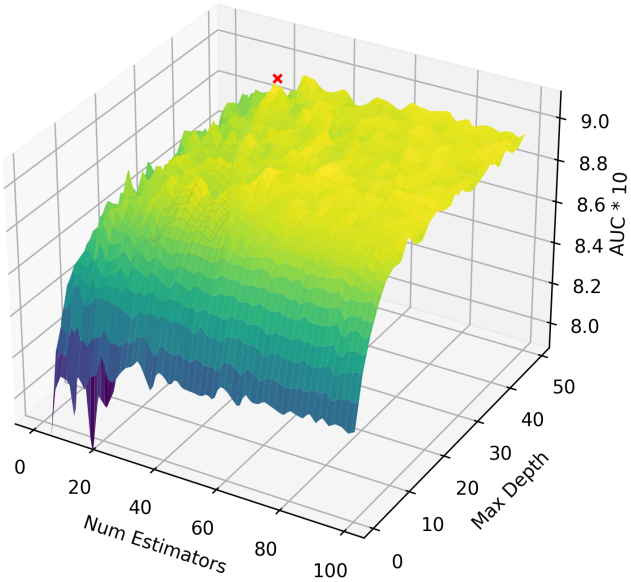

Figure 12: The mesh visualizes the surface plot created when the number of estimators and maximum depth is plotted against AUC. The AUC values are multiplied by 10.

The mean AUC from five fold cross validation after application of lasso, PCA and SFS filters can be found in table 2. During the validation process, a range of 2-30 features were tested for the PCA, while 10 features were tested for the SFS.

Table 2: Mean AUC after filter application.

Filter Method	**Mean AUC**
Lasso	0.777
PCA dimension reduction	0.810
SFS algorithm	0.876

3.2 Convolutional Neural Network Metrics

The accuracy and loss values from n epochs of training the LeNet convolutional neural network are found in figure 13.

In the left graph of figure 13, 76% of the data points in the domain $[20, 50]$ are positioned within one standard deviation from the mean accuracy. The distribution of points can therefore be deviated from a gaussian distribution, indicating that the graph has reached convergence. Yet, in a more narrow interval, e.g. $[40, 50]$, a majority of the data points have moved towards the second deviation. Ultimately, this indicates that convergence has not been reached and that further training is needed. The loss learning curve confirms this, since as the number of epochs increases, the loss continues to decline. Subsequently, there is an overrepresentation of data points in the lower boundary.

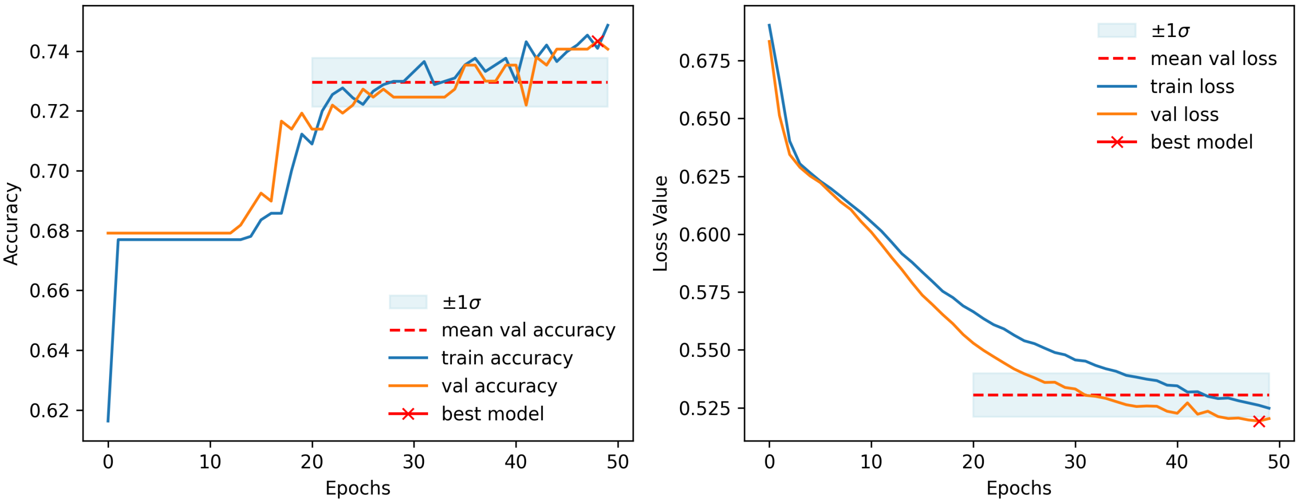

Figure 13: LeNet CNN learning curves. The orange graphs represent validation scores, while the blue graphs represent training scores. As the validation graphs stabilize, the mean value with standard deviations boundaries is plotted.

The convolutional neural network with the AlexNet architecture was validated with identical configurations as those of the LeNet model.

The graphs in figure 14 appear to follow the same trend as their LeNet counterparts, see figure 13. Yet, the noisiness of the graphs in figure 14 seems higher despite the learning rate being the same. This is confirmed by the slight fluctuation in standard deviation, $\Delta\sigma$, between the graphs in figure 13 and 14, where $\Delta\sigma_{\text{left}} = 0.00894$ and $\Delta\sigma_{\text{right}} = 0.00816$, see table 3.

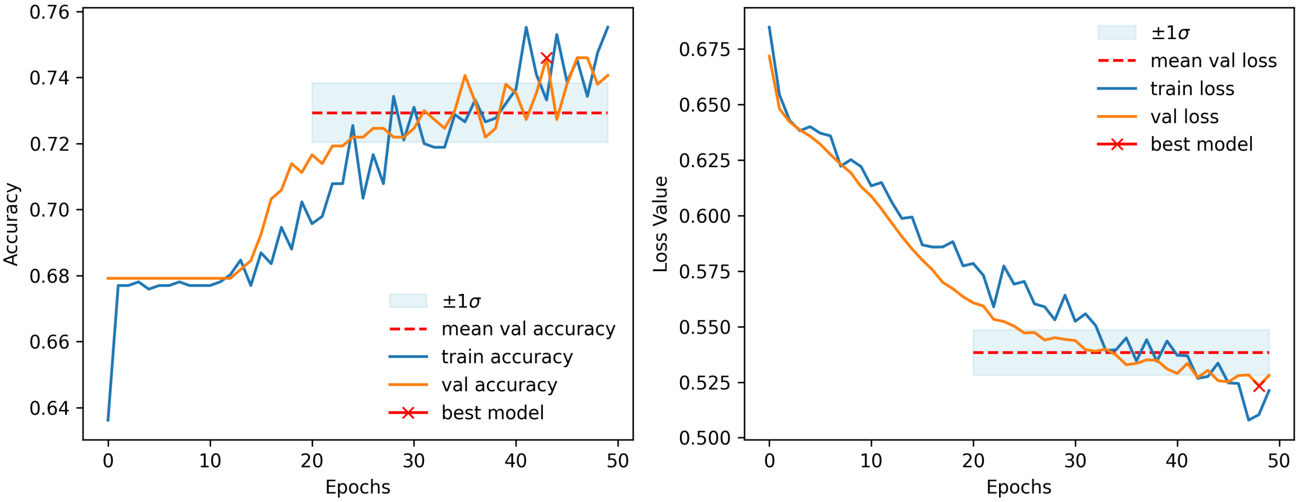

Figure 14: AlexNet CNN learning curves. The orange graphs represent validation scores, while the blue graphs represent training scores. As the validation graphs stabilize, the mean value with standard deviation boundaries is plotted.

The convolutional neural network with the pre-trained VGG16 architecture was validated with identical configurations as those of LeNet and AlexNet.

In the left graph of figure 15, 76.7% of the data points in the domain $[15, 50]$ are positioned within one standard deviation from the mean accuracy of 0.729. The data points may therefore deviate from a gaussian distribution. Thus, they can be considered to converge. Simultaneously, the graph of the training accuracy continues to rise, indicating that the model is becoming increasingly overfitted. This is supported by the corresponding loss graph in figure 15, in which the impact of training appears to cease after 15 epochs.

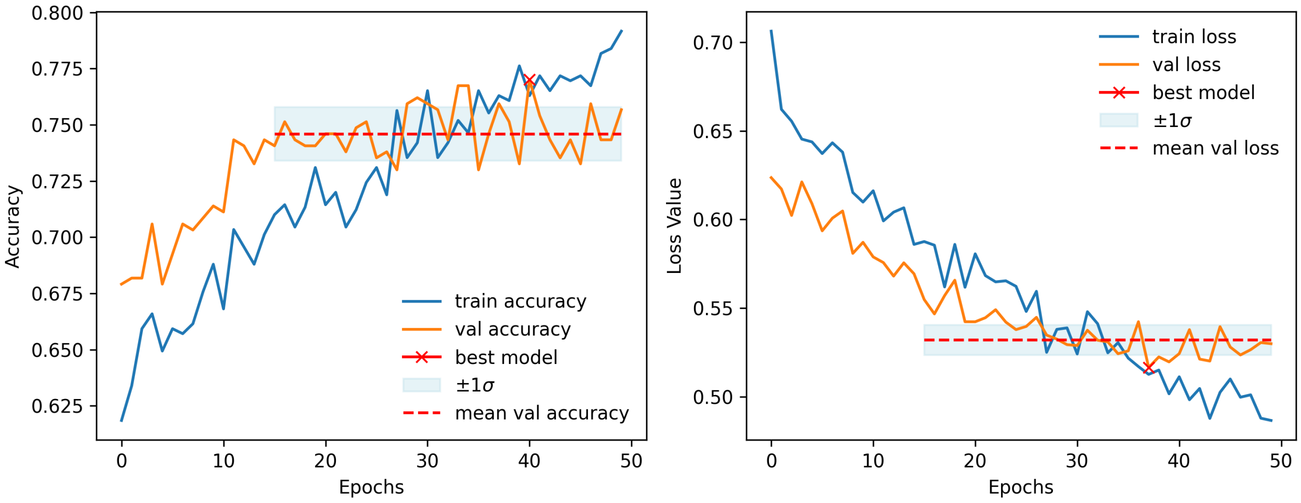

Figure 15: Learning curves from the VGG16 CNN model with transfer learning. The orange graphs represent validation scores, while the blue graphs represent training scores. As the validation graphs stabilize, the mean value with standard deviation boundaries is plotted.

Table 3 provides a summary of the mean validation accuracy and loss from the LeNet, AlexNet, and VGG16 models. Standard deviations are included.

Table 3: CNN model summary.

Architecture	**Validation Accuracy**	**Validation Loss**
LeNet	0.729 ± 0.008	0.530 ± 0.009
AlexNet	0.729 ± 0.009	0.538 ± 0.010
VGG16 with pre-training	0.746 ± 0.025	0.535 ± 0.008

The information in table 1 and 3 is visualized in figure 16.

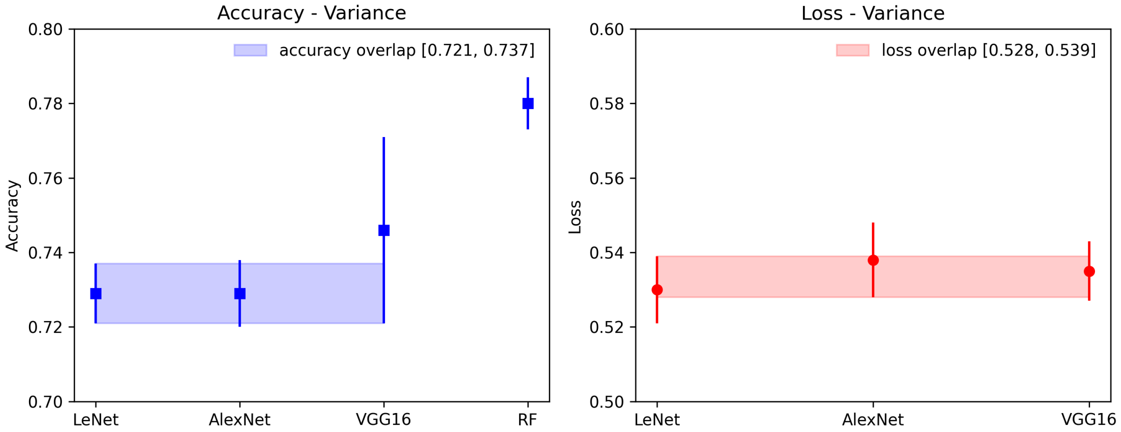

Figure 16: Visualization of table 1 and 3. The squares represent the mean value for the respective models while the error bars represent the boundary of one standard deviation from the mean. The highlighted area visualizes the overlap between the standard deviations.

4 Discussion

4.1 Model Comparison

Neither the testing graphs of the LeNet model nor the AlexNet model appear to converge within the interval of 50 epochs, suggesting that more training had been necessary in prior to validation. While the testing graph of the VGG16 model appears to converge around its mean accuracy of 0.746 and mean loss of 0.535, the corresponding accuracy and loss of the training graph continues to improve. Consequently, the model becomes increasingly overfitted to the training data.

The similarities between the mean AUC of the random forest model, 0.781 ± 0.006, and the corresponding mean accuracy of 0.780 ± 0.007 are notable. The relatively high values indicate that the model persistently discriminates the two classes from one another and predicts the correct classes sufficiently. After application of the sequential feature filter, i.e. SFS, the accuracy reached a maximum of 87.6%.

4.1.1 Standard Deviation Overlap

The mean accuracy among the CNN models varies slightly, with a variance of 2.3% within the range of accuracy $[0.729, 0.746]$. As can be observed in the left graph of figure 16, the standard deviation of the LeNet and AlexNet models fully overlap one another. Thus, there is at least a 68.3% chance that the models will perform within the same distribution. Because of this, it is not possible to nominate one of the model architectures as the best with certainty.

Further, there is overlap between the mean accuracy of the VGG16 model and the LeNet and AlexNet models, resulting in a 45.3% chance of performing within the same distribution, as shown in equation 5 in the appendix. The exception to this trend is the mean accuracy of the random forest, which is further than one standard deviation away from the upper boundary of the VGG16 model. Consequently, the likelihood of the VGG16 and the random forest model to generate an accuracy belonging to both distributions is very low: 0.771%, see equation 6 in the appendix. This suggests that the random forest model consistently generates higher levels of accuracy than its VGG16 counterpart.

4.1.2 Random Forest vs. Convolutional Neural Networks

The random forest frequently outperforms the deep learning models with a few percentage units. Also, the LeNet, AlexNet and VGG16 models show more or less prevalent indications of overfitting. This suggest that the relationship between the amount of data in the data set and the amount of parameters in the model architecture is insufficient. By adjusting hyperparameters such as batch size, model architecture, number of layers or neurons per layer, this relationship could somewhat have been balanced. Yet, in contrary to the random forest, no clear signs of convergence was observed in the deep learning models. This could imply that the performance of the deep learning models would improve if the number of epochs increased. Also, the extensive complexity of the deep learning models entails that if the number of data points were to increase, the performance of the deep learning models would likely improve too.

4.2 Applicability in Healthcare

A study conducted in Turkey, specifically about the Bahcesehir Mammographic Screening program, evaluated the breast cancer detection rate among radiologists. The study reported an average accuracy of 67.3% [15]. When supervised by the Lunit deep learning model [16], this accuracy improved to 83.6%.

4.2.1 Degree of Discrimination

The pre-trained VGG16 model scored an average accuracy of 74.7%, see table 3. The uneven distribution of classes, where 67.5% of samples belonged to the benign class and the remaining 32.5% to the malignant class makes this score rather unreliable. For example, classifying all data points as benign would generate an accuracy of 67.5%. Therefore, an accuracy of 74.7% is just a 9.6% improvement in model performance, compared to random guessing.

In contrast, the pre-processed data that was provided to the random forest model was perfectly balanced, where 50% of the data belonged the respective classes due to the implementation of synthetic minority oversampling techniques. Consequently, a mean accuracy of 78.1% would correspond to a 56.2% improved model accuracy, compared to randomly guessing.

4.2.2 Clinical Implementations

Although the VGG16 accuracy of 74.7% outperforms that of the average radiologist in the study, it is still too low to be applicable for clinical implementations involving more critical types of cancer [17]. For example, the chances of surviving more than five years of lung cancer is approximately five times worse than the corresponding survival rate for breast cancer [18] [19]. Further, due to the nature of the normal distribution, the risk for the VGG16 model to perform worse than its current mean accuracy of 74.7% is 47.5%. Also, the risk of the VGG16 model to perform worse than its lower first standard deviation, 72%, is 15.8%. Despite the accuracy of the random forest being notably better than that of the VGG16 model, see figure 16; 22% of the images are still falsely classified, suggesting that further improvements are needed before clinical implementation.

4.3 Further Studies

In the pre-processing step of the data, the original three dimensional NIFTI files were converted to the 2D JPEG format.

NIFTI images contain unique features such as dimensionality, bit depth and tissue density. For example, the tissue density of lungs, represented by pixel intensity; typically vary within the shade-range $[-400, 1000]$. When converting from NIFTI to JPEG, the intensity range is adjusted to fall within the shade range of $[0, 255]$. This adjustment is needed because of the 8-bit JPEG compression, but it results in the loss of information, which can negatively impact the model's performance.

Also, in the conversion from the three dimensional NIFTI format to JPEG, only the axial view, the head-to-foot axis, was converted. If the sagittal view, parallel to the sides of the patient; and the coronal view, perpendicular to the front of the patient; had been included, the data would better have represented the human respiratory region.

In addition, the original resolution of the images was reduced to $244 \times 244 \times 1$ to be compatible with the VGG16 model. As the pulmonary nodules take up very few pixels, information about their internal heterogeneity and shape could be lost. This loss of information could have had further negative effects on the model performance.

To avoid these deficiencies, the pydicom library could have been implemented to immediately convert the DICOM files to readable numpy arrays. Also, the model architectures could have been modified to avoid the issue of information loss. These measures would likely improve the model's performance but would also considerably extend its training time. If more time and processing power had been available, this would have been the optimal approach to take.

5 Conclusion

The study reveals that, based on the analysis of the Kaggle Data Science Bowl 2017 data set [2], the random forest algorithm outperforms the LeNet, AlexNet and VGG16 convolutional neural network architectures. However, if the deep learning models had undergone further training or if the data set had been more extensive in size, then the results might have been improved.

References

[1] Kleber, H., D., and Gold, M., S., , "Use of psychotropic drugs in treatment of methadone maintained narcotic addicts," *Ann N Y Acad Sci*, vol. 311, pp. 81–98, 1978.

[2] "Data science bowl 2017." https://www.kaggle.com/c/data-science-bowl-2017. Accessed: 07 08, 2023.

[3] Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R. L., Torre, L. A., Jemal, A, "Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *A Cancer Journal for Clinicians*, vol. 68, pp. 394–424, 2018.

[4] Xu, J., Liao, K., Yang, X. et al, "Using single-cell sequencing technology to detect circulating tumor cells in solid tumors," *Molecular Cancer*, 2021. https://doi.org/10.1186/s12943-021-01392-.

[5] Otero, Hansel J. and Rybicki, Frank J. and Greenberg, Dan and Neumann, Peter J., "Twenty Years of Cost-effectiveness Analysis in Medical Imaging: Are We Improving? ," *Radiology*, vol. 249, no. 3, pp. 917–925, 2008. https://doi.org/10.1148/radiol.2493080237.

[6] Park, J. E. and Kim, H. S. , "Radiomics as a Quantitative Imaging Biomarker: Practical Considerations and the Current Standpoint in Neuro-oncologic Studies," *Nucl Med Mol Imaging*, vol. 52, pp. 99–108, Apr 2018.

[7] M. Yogeshwari and G. Thailambal, "Automatic feature extraction and detection of plant leaf disease using GLCM features and convolutional neural networks ," *Materials Today: Proceedings*, vol. 81, pp. 530–536, 2023. https://www.sciencedirect.com/science/article/pii/S2214785321028029.

[8] Wikipedia contributors, "Decision tree — Wikipedia, the free encyclopedia." https://en.wikipedia.org/w/index.php?title=Decision_tree&oldid=1164832164, 2023. [Online; accessed 11-July-2023].

[9] David O'Connor and Evelyn M.R. Lake and Dustin Scheinost and R. Todd Constable, "Resample aggregating improves the generalizability of connectome predictive modeling," *NeuroImage*, vol. 236, no. 11, p. 118044, 2021. https://www.sciencedirect.com/science/article/pii/S1053811921003219.

[10] N. Brunello, "Example of a neural network's neural unit." https://upload.wikimedia.org/wikipedia/commons/3/39/Example_of_a_neural_network%27s_neural_unit.png, August 2021.

[11] "Imagenet." https://www.image-net.org/index.php. Accessed: 07 08, 2023.

[12] Cai, L. and Gao, J. and Zhao, D., "A review of the application of deep learning in medical image classification and segmentation," *Ann Transl Med*, vol. 8, p. 713, Jun 2020.

[13] Bianco, S., Cadene, R., Celona, L., Napoletano, P., "Benchmark analysis of representative deep neural network architectures," *arXiv*, 2018. https://doi.org/10.1109/ACCESS.2018.2877890.

[14] "Keras." https://keras.io/. Accessed: 07 08, 2023.

[15] Kizildag Yirgin, I. and Koyluoglu, Y. O. and Seker, M. E. and Ozkan Gurdal, S. and Ozaydin, A. N. and Ozcinar, B. and lu, N. and Ozmen, V. and Aribal, E. , "Diagnostic Performance of AI for Cancers Registered in A Mammography Screening Program: A Retrospective Analysis," *Technol Cancer Res Treat*, vol. 21, p. 15330338221075172, 2022.

[16] "lunit." https://www.lunit.io/en. Accessed: 07 09, 2023.

[17] Rishi P. Singh; Grant L. Hom; Michael D. Abramoff; J. Peter Campbell; Michael F. Chiang, "Current Challenges and Barriers to Real-World Artificial Intelligence Adoption for the Healthcare System, Provider, and the Patient," *transitional science teaching*, August 2020. https://tvst.arvojournals.org/article.aspx?articleid=2770632.

[18] Di Girolamo, C. and Walters, S. and Benitez Majano, S. and Rachet, B. and Coleman, M. P. and Njagi, E. N. and Morris, M., "Characteristics of patients with missing information on stage: a population-based study of patients diagnosed with colon, lung or breast cancer in England in 2013," *BMC Cancer*, vol. 18, p. 492, May 2018.

[19] DeSantis, C. E. and Bray, F. and Ferlay, J. and Lortet-Tieulent, J. and Anderson, B. O. and Jemal, A. , "International Variation in Female Breast Cancer Incidence and Mortality Rates," *Cancer Epidemiol Biomarkers Prev*, vol. 24, pp. 1495–1506, Oct 2015.

[20] M. Astaraki, G. Yang, Y. Zakko, I. Toma Dasu, Smedby, and C. Wang, "A comparative study of radiomics and deep-learning based methods for pulmonary nodule malignancy prediction in low dose ct images," 2021. https://github.com/Astarakee/Radiomics_pipeline.

A Code Access

For the part of the project that involved radiomics feature extraction, part 2.1-2.6 of the method, the following in-house material was modified and optimized:

`https://github.com/Astarakee/Radiomics_pipeline` [20]

For the part of the project that involved deep learning models such as convolutional neural networks and densely connected neural networks, the following code was used:

`https://github.com/CarlViggo/Carl-Viggo---Rays-2023/blob/main/Rays%2C_DL_Project%2C_Carl_Viggo_2023.ipynb`

B Complementary Data

In equation 5 the probability of the AlexNet and the VGG16 model to generate an accuracy belonging to both distributions is calculated.

$$\begin{aligned} P_1 = &\int_{-\infty}^{0.71} \frac{1}{0.009\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-0.729}{0.009}\right)^2} dx + \int_{0.71}^{0.742} \frac{1}{0.025\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-0.746}{0.025}\right)^2} dx \\ &+ \int_{0.742}^{+\infty} \frac{1}{0.009\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-0.729}{0.009}\right)^2} dx = 0.45319... \approx 0.453 \end{aligned} \tag{5}$$

In equation 6 the probability of the VGG16 and the random forest model to generate an accuracy belonging to both distributions is calculated.

$$P_2 = \int_{-\infty}^{0.7561} \frac{1}{0.007\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-0.71}{0.007}\right)^2} dx + \int_{0.7561}^{+\infty} \frac{1}{0.008\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-0.729}{0.008}\right)^2} dx = 0.0006724... \approx 0.00067 \tag{6}$$