Detection of Lung Cancer Using Convolutional Neural Networks

Dosbol Aliev daliev@cougarnet.uh.edu
Kurmanbek Bazarov kbazarov@cougarnet.uh.edu
Michael Moorman mmoorman@cougarnet.uh.edu
Abraar Patel aipatel2@cougarnet.uh.edu
Nouhad Rizk njrizk@uh.edu

ABSTRACT

Cancer detection in the early stages is a significantly useful tool for decreasing cancer mortality and improving outcomes for patients. This research focuses on comparing implementations of machine learning algorithms to classify different types of lung cancer. The model utilizes pre-processed and regularized axial chest CT scans of patients with either one of three non-small-cell lung carcinomas, or no cancer. Traditional supervised learning models Support Vector Classifiers (SVC) and Random Forest Classifiers (RFC) are compared to deep learning models using Convolutional Neural Networks (CNN) by comparing their respective model accuracies and weighted F1 scores, with special care given to false positive and false negatives in cancer detection. The study uses different techniques involving hyperparameter tuning, Dropouts, and regularization techniques to control overfitting in the CNN model to get an efficient model which makes accurate predictions of lung cancer detection. The SVC and RFC models are also tuned using k-fold cross-validation to counteract overfitting. Moreover, the research also involved using transfer learning models such as InceptionV3, Xception, EfficientNetB2, ResNet50 and VGG-16. All transfer learning models used a different number of layers and trainable parameters, with the majority of the transfer learning models obtaining satisfactory model accuracies on the testing data with minimal loss. The results of all the models is compared to that of the current standard of care.

Keywords: Chest CT-Scans, lung cancer, machine learning models, Support Vector Classifiers, Random Forest Classifiers, deep learning models, Convolutional Neural Networks, hyperparameter tuning, Dropouts, overfitting, transfer learning, InceptionV3, Xception, EfficientNet, ResNet50, VGG-16.

INTRODUCTION

Cancer detection is traditionally done by trained oncologists interpreting the results of many tests, including those of medical scans such as CT, MRI, and PET scans. However, the possibility of using deep learning models to recognize the presence or absence of cancer, based on training on large datasets of scan images, with accuracy equal or greater than that of medical professionals, would be a significant asset. This would lead to improved patient outcomes as well as freeing up resources of medical professionals, which could have significant benefits in areas of the world where experts in cancer detection are scarcer and their time is more precious.

Fig. 1. An example image from the dataset

An examination of literature regarding the use of CT scan screening of non-small-cell lung cancer reveals that there is a range of accuracies of the screenings within those studies. In particular, an analysis by the US Preventive Services Task Force analyzing the results of low-dose CT scan screening studies performed in various locations in the United States which screened for the presence or absence of tumors, found that all of these studies had a false negative rate no greater than 3%, and between the studies, false positive rates ranged between 3.3% and 43.5% [7]. As the most critical metric of cancer screening is the false negative rate, since false negatives correspond to cancer missed and thus more negative patient outcomes relative to false positives, we will use this 3% as a benchmark. By virtue of the classes in the model, it is easiest to relate this to the specificity of the no-cancer diagnosis: this specificity is the ratio of true negatives (people who have cancer that the model predicted as having cancer) to the sum of true negatives and false positives (all people who had cancer). This value is equal to one minus the false negative rate mentioned above; thus the benchmark of success in this will be a specificity roughly equal to 0.97.

Besides cancer detection, the research also seeks to evaluate the ability to distinguish between three different kinds of non-small cell lung cancer. In particular, the Chest CT-Scan dataset contains 4 categories of chest CT-scan images. These are adenocarcinoma, large cell carcinoma, squamous

cell carcinoma, and non-cancerous. Adenocarcinoma is the most common form of lung cancer accounting for 30 percent of all cases and around 40 percent of all non-small cell lung cancer occurrences. Adenocarcinoma of the lung is found in the outer region of the lung. Large cell carcinoma is a type of lung cancer which accounts for 10 to 15 percent of all cases of lung cancer. Large cell carcinoma tends to grow and spread rapidly and can be found anywhere in the lungs. Squamous cell carcinoma is a type of lung cancer that is generally linked to smoking and is found in the center of the lung, where larger bronchi join main airway branches. A normal chest CT scan image shows that no lung cancer is detected. These different types of cancer have different presentations and different prognoses; in particular, adenocarcinomas often have a mutation in ALK proteins that have been targeted with specific cancer medications [1]. Conversely, squamous-cell carcinomas have been harder to treat and require different methods [2]. The utility of being able to distinguish different kinds of cancerous tumors in the class of non-small-cell lung carcinomas is therefore quite useful as well.

Related Work

Techniques of convolutional neural networks have been used in medical imaging for quite some time [10]. In 2018, early devices for performing diagnosis of a condition using the output of convolutional neural networks were approved for sale by the FDA [6]. In this case, the device is attempting to use retinal scans to diagnose diabetic retinopathy at a point earlier than doctors typically are able to discern it. It is surely likely that such devices will become more common as neural network technology becomes more accurate. Chest CT is the best imaging modality that detects different parenchymal patterns and disease severity in COVID-19 patients [8] . The most endorsed screening test for detecting lung cancer is the CT scan. CT Scans use X-ray and computer technology to display full images of structures inside the chest and provide more information about chest-related injuries. Back in 2008, lung cancer accounted for 18 percent of deaths worldwide. As the lung cancer epidemic has grown and spread, ways of detecting the disease earlier, to improve the cure rate, have been explored. These have mainly been based around imaging using the chest radiograph (CXR) and computed tomography (CT) [4].

Proposed Methods

Pursuant to the goal of comparing a range of deep learning techniques for image classification on this lung cancer dataset, the research begins with a simpler CNN model and adds complexity. Support vector machines and random forest classifiers are trained to provide a baseline comparison of the strength or weakness of these basic CNN models compared to other performant machine learning models. Finally, these basic CNN models are compared to sophisticated transfer learning techniques involving very complex CNN models that were trained on the ImageNet dataset and which are included in Tensorflow. Three different metrics will be considered in evaluating the models: the accuracy, the weighted average F1 score of all categories, and the specificity of the cancer detection.

Dataset Preparation

The dataset that was used for the research was the Chest CT-Scan dataset, a public domain dataset compiled by Mohamed Hany and made available on Kaggle [5]. The dataset consists of 1000 RGB images of axial chest CT scans, of varying sizes and orientations, that was pre-sorted into training, validation, and testing sets of 613, 72, and 315 images respectively.

Preprocessing initially involved converting the images to grayscale and resizing them to a common size of 300 by 200 pixels. In the process of preprocessing, we discovered one blank image in the training set and one in the testing set, which were removed, yielding 998 total images.

For comparison, we also used the ImageDataGenerator function from Keras to produce modified images for the training and validation. These images were not made grayscale or resized, but were zoomed, sheared, or flipped using the ImageDataGenerator. The testing images were the same for all of the models regardless of whether they were trained on modified or unmodified images. The SVC and RFC models, because they took a very long time to train, were not retrained on the ImageDataGenerator images.

The Basic CNN Model

The Tensorflow and Keras libraries permit the creation of simple sequential convolutional neural network models. The simplest models thus produced for this research utilized image input that was scaled to 300x200 pixels and was converted to grayscale.

A single round of convolution is performed using the Keras Conv2D layer, using a set of eight 3x2 pixel kernels. These kernels are then passed through a ReLU activation. Finally, pooling is performed on the output, with 2x2 pools that reduce the number of outputs by a factor of four. This set of convolution, activation, and pooling is then repeated. The output of these convolutional layers is then reshaped and passed to a fully-connected layer of 64 neurons, also with ReLU activation. These then output to a final layer of four neurons with softmax activation, providing classification values.

This model was trained using the Adam optimizer, minimizing the categorical cross-entropy loss. This optimizer was introduced in 2014 and is a fairly standard optimizer for stochastic gradient descent in Tensorflow. Adam utilizes momentum to dampen the changes in weights due to gradient calculation; specifically, it uses first and second moments, which describe the rate of change of the gradient with respect to time, and its square respectively. [9]

The categorical cross-entropy loss function is used here as the classification is multi-category:

$$L = \sum_{i}^{C} y_i \log \hat{y}_i$$

where the sum runs over every class, and y and $\hat{y}$ are the ground-truth and predicted values for each class. With

softmax-activated outputs, the model produces a probability distribution that can be input to this loss function. The softmax activation and categorical cross-entropy loss will be used in all neural network models.

CNN model with Dropout

To improve the efficiency of the basic CNN model, dropout layers were added. The Dropout layer of the convolutional neural network is a regularization method that reduces the number of hidden layers to control over-fitting. 50% Dropout rate was added to the basic CNN model after the convolutional layer and before the fully connected layer. The Dropout layer randomly deactivates specific neurons in a layer. Due to this, the neural network cannot rely on specific activations in a given feed-forward pass during training. Thus, the neural network will gain several redundant representations since it cannot depend on a specific combination of neurons. The advantage of using the Dropout layer is that training will be faster.

Tuned CNN model using Keras Tuner

Keras Tuner is an easy-to-use, scalable hyperparameter optimization framework available for Tensorflow and Keras [12]. The Keras Tuner model definition allows one to specify ranges of different hyperparameter values that are included as variables in the model. It creates a search space that can be thought of geometrically as an n-dimensional volume, where each hyperparameter represents a different dimension and the scale of the dimension are the values that the hyperparameter may take on, such as real-valued, integer-valued, or categorical. It searches this space to train different models and saves the one that provides the highest validation accuracy.

The hyperparameters that were tuned included the number of convolutional kernels and their size, the number of convolution and maxpooling layers, the level of dropout (ranging from no dropout to 70% dropout) and the size and topology of the dense layers. Finally, we trained Keras Tuner models using both the Adam optimizer, as well as the Nadam optimizer which is an extension of Adam version of gradient descent that includes Nesterov momentum [3]. CNN model with Keras Tuner using Nadam optimizer yielded better results than the CNN model with Keras Tuner using the Adam optimizer.

Support Vector Machines

To strike a comparison between the CNN model and machine learning models, a Support Vector Machine (SVM) workflow was created to solve the image classification task. SVM is a supervised machine learning algorithm used to solve classification and regression problems. The Support Vector Classifier (SVC) model was created and trained using hyperparameter tuning. To determine the optimal set of hyperparameters for the SVC model, RandomizedSearch cross-validation was implemented. A range of hyperparameters such as different values of C (penalty parameter), γ hyperparameter, and different types of kernel, were used to carry out RandomizedSearch cross-validation on the SVC model.

Random Forests

A Random Forests workflow was created to solve the image classification problem to compare the SVC model. Random Forests is a supervised machine learning algorithm that constructs decision tree ensembles to solve classification tasks. Random Forest Classifier (RFC) model was created and trained using hyperparameter tuning. To determine the optimal set of hyperparameters for the RFC model, RandomizedSearch cross-validation was implemented. A range of hyperparameters such as different tree depths, number of trees in the forest, the maximum number of levels in each decision tree, and the number of features for the best split were used to carry out RandomizedSearch cross-validation on the RFC model.

Transfer Learning Models

Transfer learning is a machine learning method that involves taking a pre-trained neural network and adapting it to a new dataset by transferring or repurposing the learned features. A model trained on ImageNet is trained, and the learned weights in that model are used to begin the training and classification on a completely new dataset. Transfer learning is extremely beneficial in deep learning and results in improved efficiency when training new models.

The following transfer learning models were used for this research:

- **InceptionV3:** InceptionV3 is a convolutional neural network model that employs a transfer learning framework for image classification. It is 48 layers deep and has 23,626,728 trainable parameters. The input image used for this model is a 224×224 RGB image. The convolutional layers of the InceptionV3 model are fed into an initially untrained fully-connected network composed of two dense layers of 512 neurons each, a 2-D global average pooling layer, with Dropout layers scattered at a dropout rate of 30%. Initially, all of the layers in the pre-trained InceptionV3 model are frozen, rendering all of the layer weights untrainable. After freezing all of the layers of the pre-trained InceptionV3 model, the model was optimized using the Adam optimizer and trained for 8 epochs at a learning rate of 0.001 until the dense layers began to converge. After that, a fine-tuning workflow was implemented by freezing only the last 15 layers of the pre-trained InceptionV3 model and unfreezing the remaining layers. The model was then retrained for an additional 8 epochs at the same learning rate of 0.001 to attain more accurate predictions of different types of lung cancer. The InceptionV3 function returned a Keras image classification model, loaded optionally with pre-trained weights on ImageNet.
- **Xception:** Xception is a convolutional neural network model that employs transfer learning framework for image classification. It is 71 layers deep and has 22,855,952 trainable parameters. The input image used for this model is a 299×299 RGB image. The convolutional layers of the Xception model are fed into an initially untrained fully-connected network composed of two dense layers of 1024 neurons each, a 2-D global average pooling layer with Dropout layers scattered at a dropout rate of 30%. Initially, all of the layers in the pre-trained Xception model are frozen, rendering all of the layer

weights untrainable. After freezing all of the layers of the pre-trained Xception model, the model was optimized using the Adam optimizer and trained for 8 epochs at a learning rate of 0.001 until the dense layers began to converge. After that, a fine-tuning workflow was implemented by freezing only the last 15 layers of the pre-trained Xception model and unfreezing the remaining layers. The model was then retrained for an additional 8 epochs at the same learning rate of 0.001 to attain more accurate predictions of different types of lung cancer. The Xception function returned a Keras image classification model, loaded optionally with pre-trained weights on ImageNet.

- **EfficientNetB2:** EfficientNetB2 is a convolutional neural network model for image classification that uses a transfer learning architecture. It includes 11 million trainable parameters and contains 342 layers. This model's input image was a 224x224 RGB image. The convolutional layers of the EfficientNetB2 model are fed into an initially untrained fully-connected network composed of two dense layers of 1024 neurons each, with Dropout layers dispersed at a dropout rate of 30%. Initially, all of the layers in the pre-trained EfficientNetB2 model are frozen, rendering all of the layer weights untrainable. After freezing all of the layers of the pre-trained EfficientNetB2 model, the model was optimized using the Adam optimizer and trained for 8 epochs at a learning rate of 0.001 until the dense layers started to converge. Following that, the layers of the pre-trained EfficientNetB2 model were partially unfrozen, and the model was retrained for an additional 8 epochs using the same Adam optimizer and learning rate of 0.001 to classify the images more efficiently. Using pre-trained weights from ImageNet, the EfficientNetB2 function returned a Keras image classification model.
- **ResNet50:** ResNet50 is a 50 layer deep convolutional neural network model that employs a transfer learning framework. It consists of 5 stages, each with a convolution and identity block. Each convolution block and identity block contains three convolution layers. The ResNet50 model has over 23 million trainable parameters. The input image used for this model is a 224×224 RGB image. The model's convolutional layers are fed into an untrained fully-connected network composed of two dense layers of 1024 neurons each, a 2-D global average pooling layer, and Dropout layers scattered at a dropout rate of 30%. All of the layers in the pre-trained ResNet50 model are initially frozen, making all of the layer weights untrainable. Following the freezing of all layers in the pre-trained ResNet50 model, the model was optimized using the Adam optimizer and trained for 8 epochs at a learning rate of 0.001 until the dense layers began to converge. Following that, the layers of the pre-trained ResNet50 model were partially unfrozen, and the model was retrained for an additional 8 epochs using the same Adam optimizer and learning rate of 0.001 to classify the image features of the dataset more efficiently.
- **VGG-16:** The VGG-16 neural network is a convolutional neural network that was introduced as a submission in the ImageNet Challenge 2014 competition, for which it subsequently won second-place for classification [15]. VGG-16 was designed for depth of layers, utilizing a small kernel size in its convolutions. In this respect, it resembles the basic CNN model used in this research, but with far more convolution cycles (16 versus 2). Like the other models trained on ImageNet, it accepts 224x224 RGB images. These convolutional layers of the transfer model are input to an initially-untrained fully-connected network consisting of two layers of 512 neurons each, with Dropout layers interspersed, at a dropout rate of 30%. The model was optimized using Adam, on a frozen model, for 8 epochs at a learning rate of 0.001. The transfer layers were then partially unfrozen and trained for an additional 18 epochs at a learning rate of 0.0005, to help retrain the final convolutional layers to more directly match the image features of the dataset.

Evaluation of Models

A primary goal of the research is to compare traditional machine learning to deep learning techniques like Convolutional Neural Networks and analyze whether a Convolutional Neural Network has superior accuracy over non-deep learning techniques. One aspect of a successful outcome for this project overall is for the models to attain higher than 70% accuracy; that is, for the model to classify at least 70% of previously-unseen testing images correctly. This would demonstrate that these techniques are sufficient to perform at this degree of accuracy with real world data.

To rate the models on their strength across different classes, the weighted F1 scores for each model are compared to their accuracies. The F1 score for a particular class is the harmonic mean of its precision and recall, where recall describes the rate at which ground truth instances of this class were correctly identified, and precision describes the rate at which predicted instances of this class were correct identifications. These can thus be widely different for different classes, and a low F1 score for that classification illustrates that the model had trouble correctly distinguishing it from other classes. The weighted F1 score provides a weighted average of the F1 scores for each class, yielding a single number for each model. The closeness of this number to the accuracy is somewhat indicative of how evenly it handles the different categories. These metrics weigh heavily on the ability of the models to discern the three different types of non-small cell lung cancer from each other.

Finally, the model should have extremely high accuracy in classifying cancerous versus non-cancerous lungs, as the determination that cancer is present is the most consequential aspect of the model's determination. A model with a high false negative rate for cancer detection would be useless as a diagnostic tool. As stated above, the models which can provide a specificity greater than 0.97 on the no-cancer classification

are the models which are comparable in accuracy for this specific detail as trained physicians have been, on average. This specificity describes the rate at which people who actually had cancer, were diagnosed by the model as having cancer.

Results and Discussion

Model	Accuracy	Weighted F1 Score	Specificity
Basic CNN	0.4857	0.47	0.828
Basic CNN with Dropout	0.5365	0.55	0.946
Tuned CNN using Adam	0.45	0.29	0.754
Tuned CNN using Nadam	0.55	0.41	1.0
SVC	0.5015	0.55	0.985
RFC	0.5238	0.52	1.0
InceptionV3	0.8412	0.84	0.989
EfficientNetB2	0.381	0.0	1.0
Xception	0.7396	0.75	0.961
ResNet50	0.467	0.344	0.781
VGG-16	0.6063	0.60	0.981

TABLE I
ACCURACY, WEIGHTED F1 SCORES, AND SPECIFICITY ON THE TESTING DATA FOR THE DIFFERENT MODELS TESTED

The summary version of the model accuracy, weighted average F1 score values, and specificity can be seen in Table I.

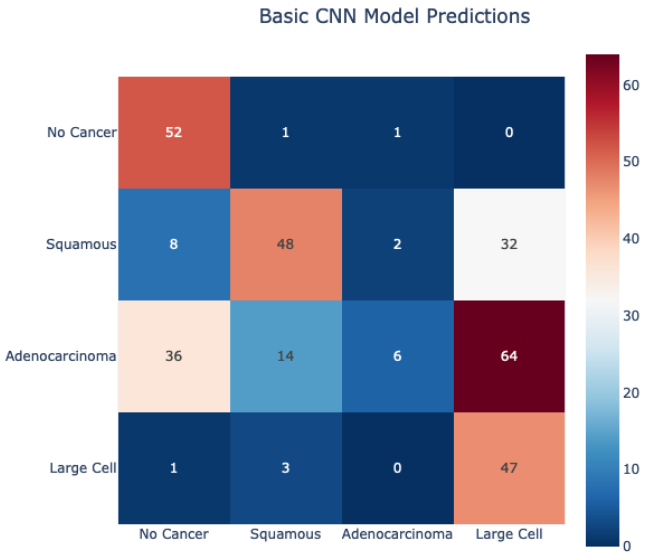

Fig. 2. Confusion matrix for predictions of the Basic CNN

Basic CNN: The basic CNN model was able to converge quickly; however, it could not achieve the desired testing accuracy. The model achieved an accuracy of 48.57% on the testing set. Examining the weighted F1 score of 0.47, closely mirroring the accuracy, suggests that it performed about as well as one could hope, across the different cancer categories in terms of precision and recall. The confusion matrix for the basic CNN model is shown in Figure 2. The specificity of the basic model was 0.828, very poor compared to the standard of care. The best model was trained on the unmodified images.

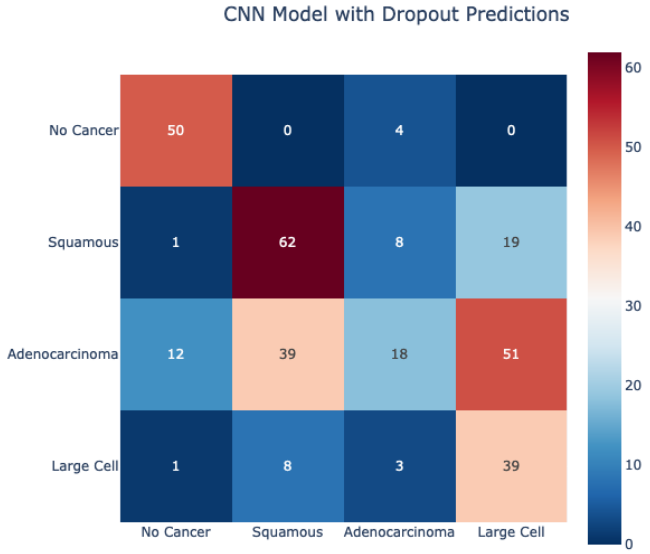

Fig. 3. Confusion matrix for predictions of the Basic CNN model with Dropout added

Basic CNN with Dropout: In the case of the CNN model with a single dropout layer added, the best-trained model was trained using a dropout rate of 0.5. This model achieved 53.65% accuracy and a weighted F1 score of 0.55. This model thus outperforms the basic model without any dropout yet still falls well short of the mark in terms of total model accuracy. No class was particularly well predicted. However, there is significant improvement with diagnosing the presence of cancer, with the model achieving a specificity of 0.946 and approaching the levels of trained physicians. The confusion matrix in Figure 3 shows that while the model had poor performance distinguishing between different kinds of lung cancer, it was relatively strong at distinguishing the presence or absence of cancer. The best model was trained on the unmodified images.

Basic CNN with Keras Tuner Using Keras Tuner that was tuning the number of convolutional cycles, the size of the kernels, the size of the pooling, the number of the convolutional kernels, the presence or absence of dropout, and the size and number of the fully-connected classification layers, the strongest-performing network using the Adam optimizer had the following topology and hyperparameters:

- Two convolutional layers with ReLU activation
- Each convolution layer used 8 filters of shape 3x2
- Each convolution was followed by a maxpooling layer with pools of size 2x2
- Each maxpooling was followed by a dropout layer with a dropout rate of 0.2
- Two layers of dense neurons were used for classification, with 256 neurons each.
- The model was trained on modified images

The topology was representative of the range of hyperparameters that were tested for these CNN models. The results

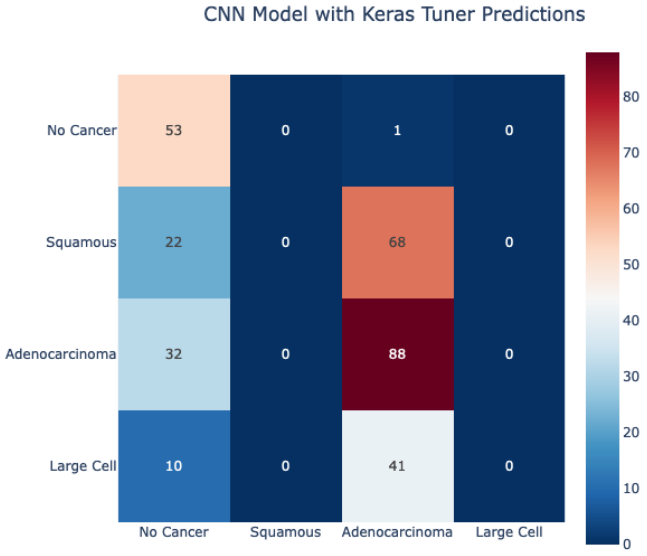

Fig. 4. Confusion matrix for predictions of Basic CNN model using Keras Tuner with Adam

from this model were not impressive; however the model achieved only 45% accuracy on the testing data, with a weighted average F1 score of 0.29. This F1 score is low because the model fails to predict all classes; all diagnoses were either for the no-cancer class or for adenocarcinoma, which was the most frequently-occurring cancerous class in the dataset (see Figure 4). The specificity is 0.757, the lowest value attained by any model presented here. The tuner models in general ran into a wall like this, with this model being more successful only insofar as it could distinguish between its two classes at a higher rate than the others.

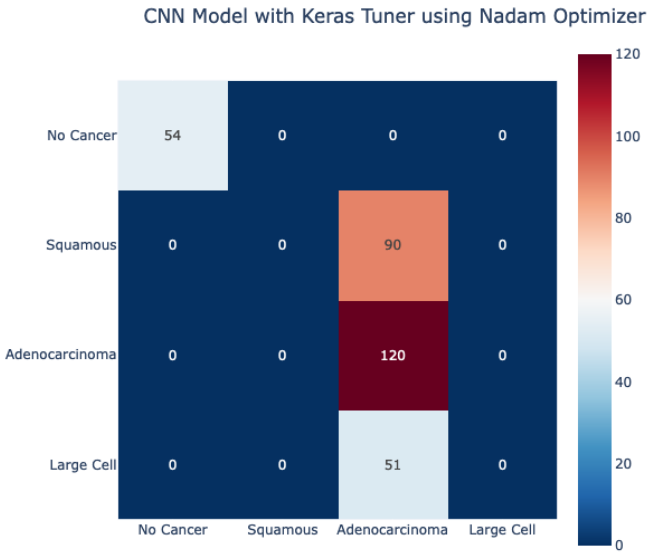

Fig. 5. Confusion matrix for predictions of Basic CNN model using Keras Tuner with Nadam

The attempt to use the Keras Tuner with dropout and a different optimizer, the Nadam optimizer, yielded models with better accuracy overall than the tuned models with Adam. The best-performing model of this type had the following topology and hyperparameters:

- One convolutional layer with ReLU activation
- This convolution layer used 8 kernels of shape 3x2
- One maxpooling layer with pools of size 2x2
- One dropout layer with dropout rate 0.5
- Two layers of dense neurons were used for classification, with 64 neurons each.
- The model was trained on modified images

This ended up being a very simple topology in comparison to the large search space that the Keras Tuner considered, which was surprising. This tuned model achieved 55% accuracy on testing data, with a weighted F1 score of 0.41. This F1 score is lower than the other models that achieved similar accuracy, which is reflected in the fact that the model only predicted two of the four prediction classes: the no-cancer class and the adenocarcinoma class (see Figure 5), similarly to the other Keras Tuner model presented above. However, this model was interesting as it achieved 1.0 specificity in predicting cancer versus no cancer, thus not missing a single instance of cancer in the testing data.

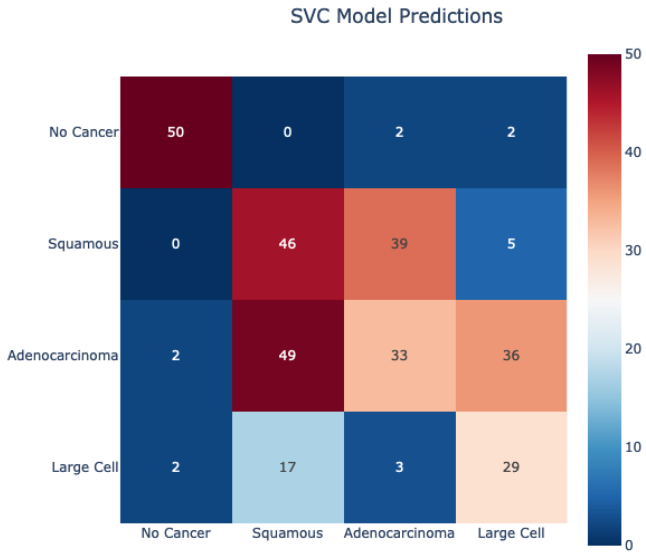

Fig. 6. Confusion matrix for predictions of Support Vector Classifier

SVC: The Support Vector Classifier (SVC) model took a long time to train with hyperparameter tuning implemented using RandomizedSearch cross-validation. Using a range of hyperparameters such as different values of C (penalty parameter), γ hyperparameter, and different types of kernel, the tuned SVC model had the following best set of hyperparameters:

- 100 as the C value
- 0.00001 as the γ value
- Rbf as the Kernel

The SVC model failed to achieve significant accuracy, with 50.15% accuracy on the testing set. In terms of precision and recall, the weighted F1 score of 0.55 indicates that the model performed about as well as one could hope, across the various cancer categories. The model seems to perform about equally in all classes (see 6). Like the Keras Tuner model above, it also achieves an astonishingly good specificity of 0.985.

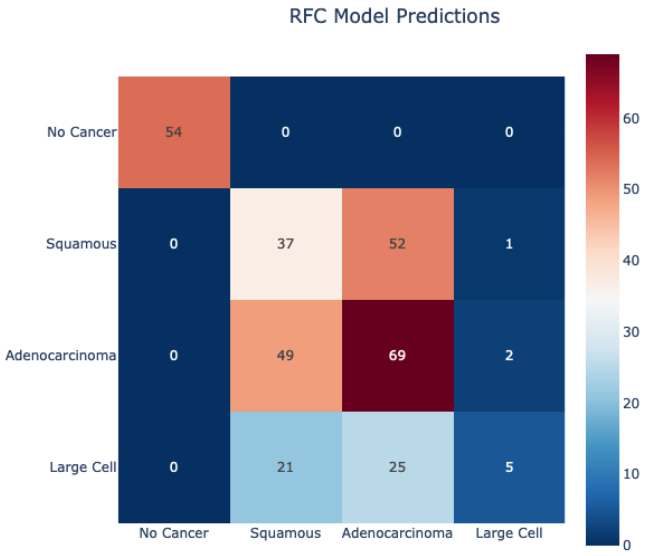

Fig. 7. Confusion matrix for predictions of Random Forest Classifier

RFC: The Random Forest Classifier (RFC) model also took a long time to train with hyperparameter tuning implemented using RandomizedSearch cross-validation. Using a range of hyperparameters such as such as different tree depths, number of trees in the forest, maximum number of levels in each decision tree, and number of features, the tuned RFC model had the following best set of hyperparameters:

- 25 trees in the forest
- 1 maximum sample to build each tree
- sqrt of the maximum number of features for the best split
- Maximum depth of tree being 7

Although the RFC model did not achieve significant testing accuracy, it did achieve slightly higher testing accuracy than the SVC model. The model attained an accuracy of 52.38% on the testing set. Examining the weighted F1 score of 0.52, which closely mirrors the testing accuracy, suggests that the model performed about as well as one could hope in terms of precision and recall, across the various cancer categories. In classifying cancer patients, the RFC model achieved a specificity of 1.00, indicating that it correctly classified 100% of all cancer patients (see Figure 7).

In comparison to the CNN models and non-neural network models such as SVC and RFC, most of the trained transfer learning models performed better, achieving satisfactory testing accuracies. All of the transfer learning models performed better on the ImageDataGenerator-modified images.

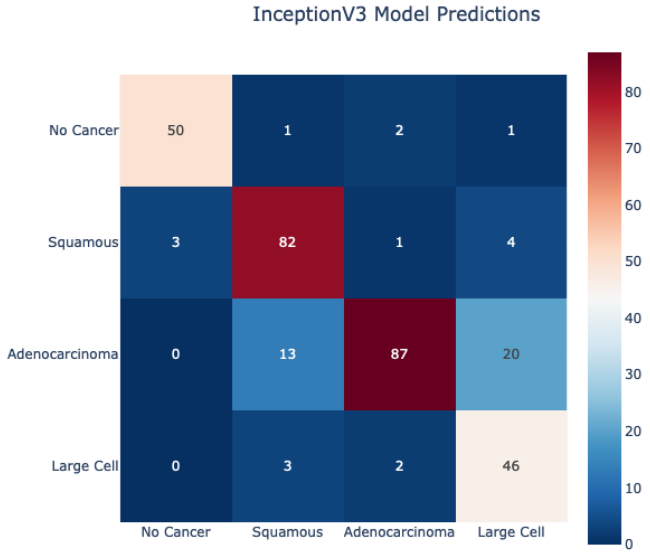

Fig. 8. Confusion matrix for predictions of InceptionV3 transfer model

InceptionV3: InceptionV3 transfer learning model was able to converge nicely and successfully achieve significantly higher accuracy, with 84.12% accuracy on the testing set. In terms of precision and recall, the weighted F1 score of 0.84, which closely mirrors the testing accuracy, indicates that the model performed very nicely across the various cancer categories. The model was able to predict all four categories of non-small cell lung cancer. The specificity was 0.989, well within the accuracy of trained physicians. The confusion matrix for InceptionV3 can be seen in Figure 8. While this is not the best of all models examined, combined with the stellar accuracy overall, this would be considered the best model for general classification.

Xception: Xception transfer learning model was able to converge nicely and successfully achieve higher accuracy, with 73.96% accuracy on the testing set. In comparison to the InceptionV3 model, the Xception model achieved slightly lower testing accuracy. In terms of precision and recall, the weighted F1 score of 0.75 indicates that the Xception model performed very nicely across the various cancer categories. The model was able to predict all four categories of non-small cell lung cancer. The Xception model achieved a specificity of 0.961 in classifying cancer patients, indicating that it correctly classified 96% of all cancer patients. The confusion matrix for the Xception model can be seen in Figure 9.

EfficientNetB2: The EfficientNetB2 transfer learning model failed to converge smoothly and performed the worst out of all the transfer learning models. With 38.1% accuracy on the testing set, the model failed to achieve satisfactory accuracy. The weighted F1 score of 0.0 demonstrates how poorly EfficientNetB2 performed across the various cancer categories in terms of precision and recall. The model was not able to train well enough and could only predict adenocarcinoma, the most common type of non-small cell lung cancer.

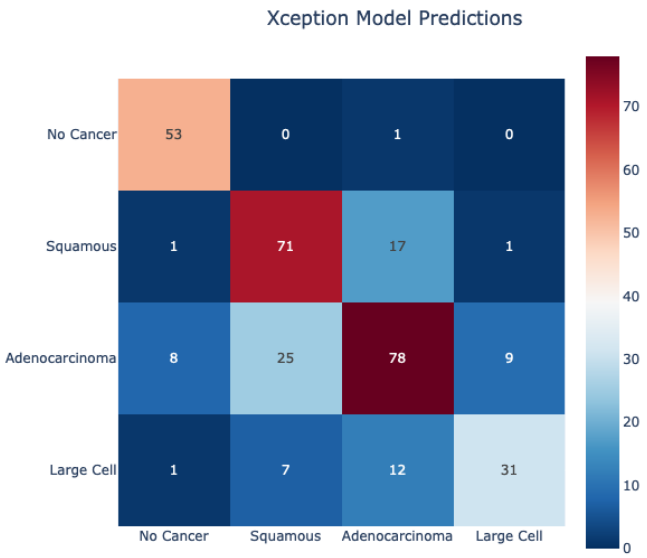

Fig. 9. Confusion matrix for predictions of Xception transfer model

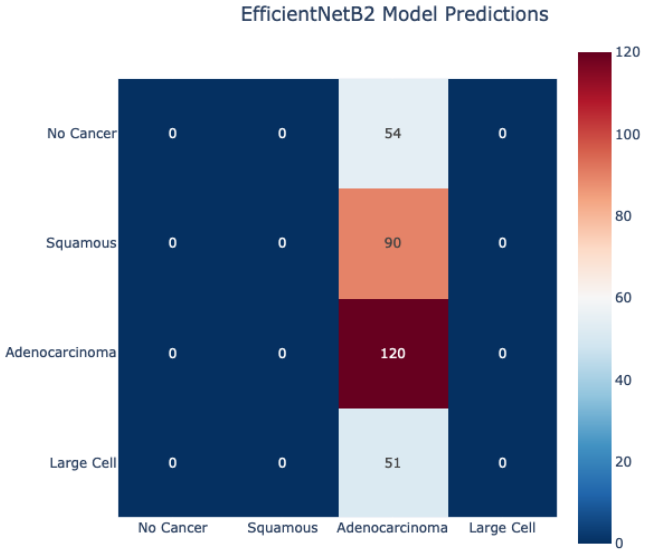

Fig. 10. Confusion matrix for predictions of EfficientNet transfer model

It failed to predict the other subsets of non-small cell lung cancer. The EfficientNetB2 model achieved a specificity of 1.0 in classifying cancer patients, indicating that it correctly classified 100% of all cancer patients. The confusion matrix for the EfficientNetB2 model can be seen in Figure 10.

ResNet50: The ResNet50 transfer learning model unfortunately didn't succeed greatly in terms of the performance itself. With 47% accuracy on the training data, the model was not up to satisfactory accuracy. The weighted F1 score of 0.34 showed very poor results of ResNet50 in terms of precision and recall. The model was able to train less than 50% of the available trainable parameters and it failed to predict some

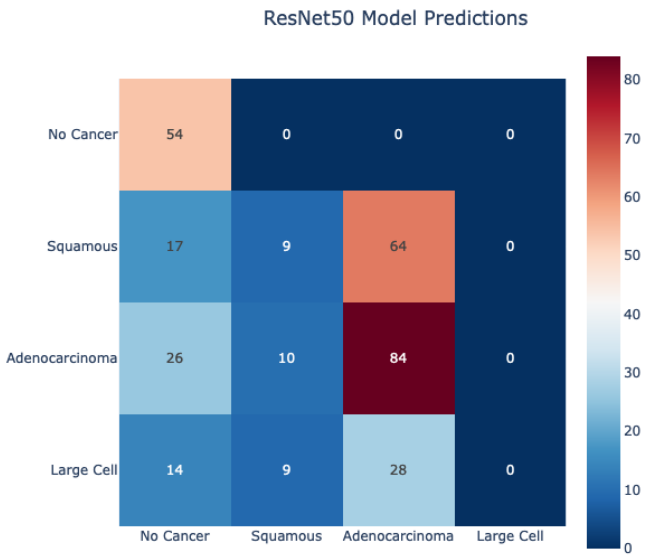

Fig. 11. Confusion matrix for predictions of ResNet50 transfer model

subsets of non-small cell lung cancer. The confusion matrix can be found in Figure 11.

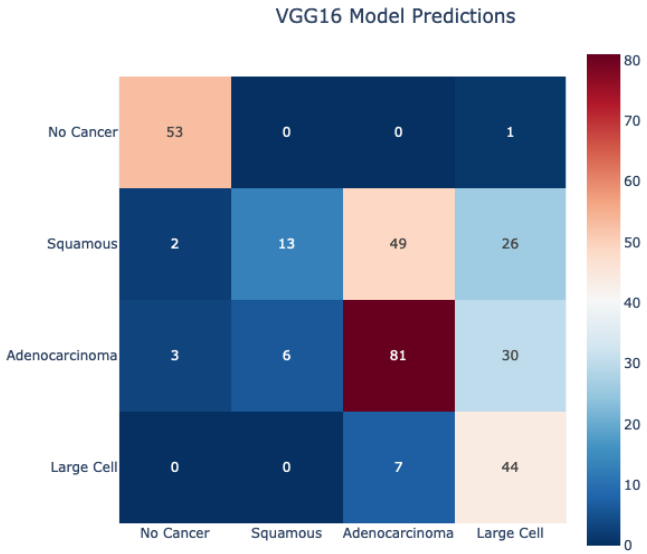

Fig. 12. Confusion matrix for predictions of VGG-16 transfer model

VGG-16: The final transfer learning model tested was VGG-16. VGG-16 converged slowly and managed only an accuracy of 0.606 and a weighted F1 score of 0.6. This is better than the basic CNN models, but not by large margins. It demonstrates that having a larger version of the basic CNN was able to achieve better results; perhaps with more time, it could be improved further. The specificity was found to be 0.981, which is exceptional and demonstrates that VGG-16 is as good as the other models at cancer detection. The confusion matrix is shown in 12

Summary

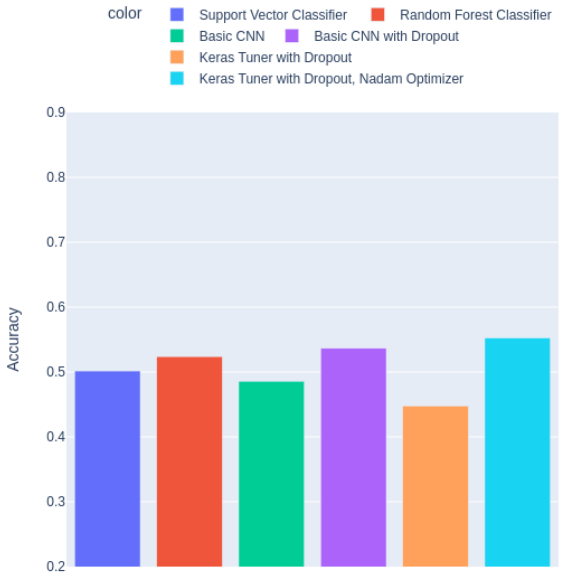

Fig. 13. Accuracies for the most-performant basic CNN models, compared to accuracies of Support Vector Classifier and Random Forest Classifier

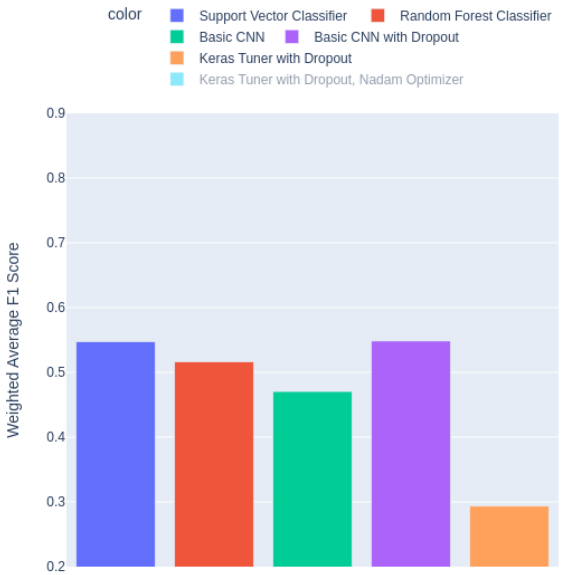

Fig. 14. Weighted F1 scores for the most-performant basic CNN models, compared to Support Vector Classifier and Random Forest Classifier

Figure 13 illustrates the trend of accuracies for basic CNN models. There is a wide variation, but a clear pattern emerges that introducing more features through dropout and hyperparameter tuning both improved model accuracy slightly but also introduced variability in the model's handling of different classes. This variability is seen in the lower weighted F1 scores of the tuned models, visible in Figure 14. The comparison of specificities can be seen in Figure 15.

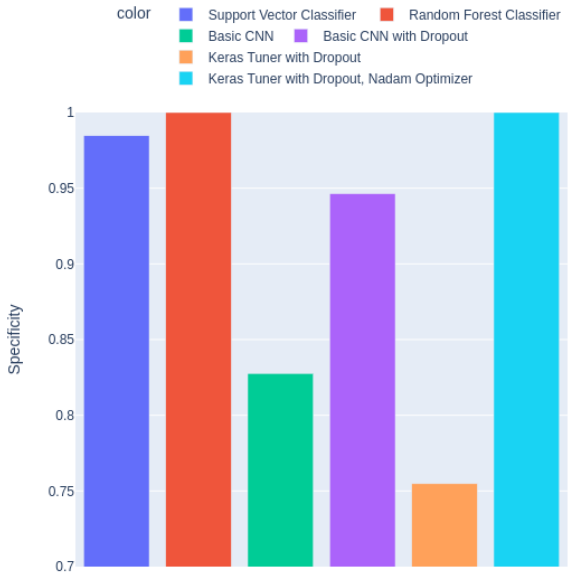

Fig. 15. Specificities for the most-performant basic CNN models, compared to Support Vector Classifier and Random Forest Classifier

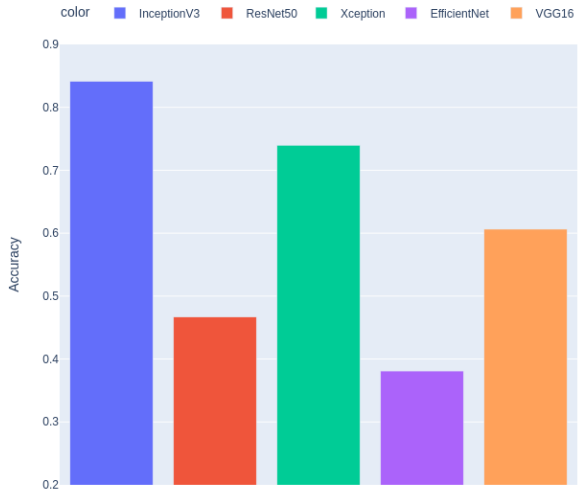

Fig. 16. Accuracies for the transfer learning models

The corresponding accuracies, weighted F1 scores, and specificities for transfer models can be seen in Figures 16, 17, and 18 respectively. As stated above, there was a wide range of results in the transfer learning models, with some training exceptionally well and others exceptionally poorly.

Conclusions

The results of the research suggest that convolutional neural networks, when designed and trained correctly, are quite powerful and can perform very well at cancer detection, and at distinguishing the different kinds of non-small cell lung carcinomas. The most basic CNN was easy to train; however, it did not achieve the desired accuracy. Adding dropout could occasionally yield much stronger accuracy, as shown by the

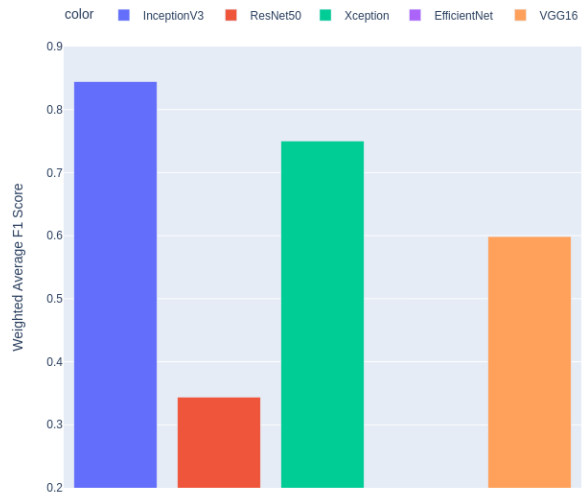

Fig. 17. Weighted F1 scores for the transfer learning models

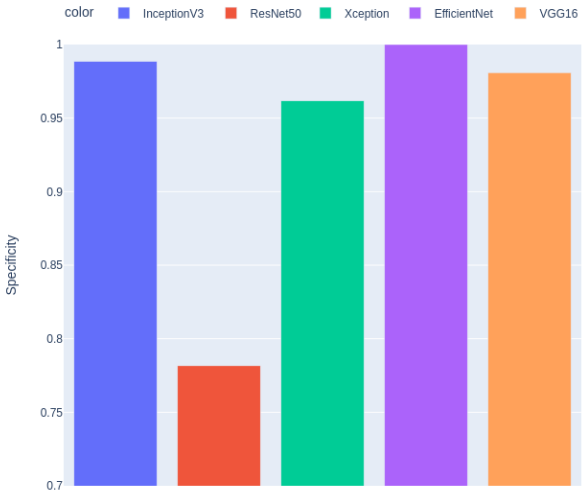

Fig. 18. Specificities for the transfer learning models

dropout model that was presented. However, training was difficult and yielded models which often converged at flawed classifications that dropped entire classes. These models, however, did seem to manage to perform very well at cancer detection. The tuned models also favored simplicity, with fewer filters with smaller kernels and smaller dense classification networks, which seemed counter-intuitive given the success of the large, complex pre-trained models.

The relative performance of SVC and RFC was also a surprise. They managed to achieve better results than the simpler CNN models overall. However, the randomized cross-validation search was extremely expensive in time, as the implementations on scikit-learn were not optimized for dealing with the massive feature sets we considered [14]. Also, the results were only good in comparison to the poor performance of the basic CNN models.

The pre-trained models generally performed very well on this dataset. The techniques used to stabilize the classification, and then to utilize fine-tuning to hone the convolutional layers towards detecting the specific features of the images of this dataset, were able to at least match those of simpler CNN models in most of the cases. In the case of the failure of ResNet50 and EfficientNet, it is likely due to the sheer size of the former, and the complex architecture of the latter; there was not adequate time put into training these networks to recognize the features of this particular dataset. By contrast, InceptionV3 and Xception took very little training to completely surpass all other models, and were relatively straightforward. This suggests that there is no ”best” transfer learning model for a particular use case, as EfficientNet is both newer and more sophisticated than InceptionV3, yet took much longer to train, and trained much worse in this instance.

Another conclusion is that the size of the dataset was a limiting factor to the accuracy of the basic CNN models. A dataset with a greater variety of images could be used to recognize more nuance, more readily. Further research into this problem would seek a wider variety of ground truth images to train the models on. By contrast, the transfer learning models had been trained on extremely large datasets, which provided a great deal of partial image features which needed to be discerned in the various layers of convolutional kernels. Somewhere in the models, it was relatively easy to repurpose these pre-trained partial features from distinguishing cars and planes, to distinguishing different kinds of lung tumors.

The success of the advanced classification models also suggests the possibility of extending the model to perform object detection as well; it would not only return a classification, but also provide a bounding box that locates the tumor in the image. There are many intriguing methods for performing this, such as R-CNN, which would be worth considering; however, the ground-truth data in this dataset does not contain this information, nor does it seem likely that there is a public-domain dataset that does for this particular problem.

Resources

The Chest CT-scan image dataset was compiled by Mohamed Hany from public domain sources, and made available under the Open Data Commons Open Database License (ODbL) v1.0 [5]. Dataset preprocessing was done using the Numpy [11] and OpenCV [13] libraries for numerical computing and computer vision. The construction and training of the CNN model utilized Google’s Tensorflow framework [16]. The InceptionV3, Xception, EfficientNetB2, ResNet50, and VGG-16 transfer learning models used by this research were made available through Tensorflow in the

`tensorflow.keras.applications`

library. Hyperparameter tuning was performed with the Keras-Tuner framework [12]. The implementations of Support Vector Machines, Random Forest and other classifiers for comparison with the CNN models, as well as the random search cross-validation functionality were all provided through the scikit-learn library [14].

CONTRIBUTIONS

Michael Moorman: Michael was involved in scheduling team meetings regularly and listing out the responsibilities of the team members. Lead data preprocessing by resizing the images. Michael identified a workflow for saving the models on Google Colab so that the models could be shared between team members efficiently without having to retrain them every time. Michael developed the basic CNN and VGG-16 models. In the writing part, Michael focused on listing out the resources used for the project, describing why the research is interesting and including visuals of the dataset, including the necessary references, the VGG-16 model architecture, and related work on chest CT scans. Moreover, Michael also worked on describing the outcomes of the proposed methods and provided conclusions based on the outcomes of the research. The outcomes of training the basic CNN model, CNN model with Dropout, CNN model with Keras Tuner, and VGG16 transfer learning model were described by Michael.

Kurmanbek Bazarov: Kurmanbek was involved in applying hyperparameter tuning on the CNN model and interpreting results to determine the best model which made accurate predictions of lung cancer detection and distinguish the different types of lung cancer correctly. Kurmanbek focused on training the ResNet50 transfer learning model. In the writing part, Kurmanbek concentrated on writing the abstract for the research, describing the tuned CNN model as well as the ResNet50 model architecture, and was also involved in describing the outcomes of training the ResNet50 transfer learning model.

Abraar Patel: Abraar was involved in setting up the Google Colab and Google Drive for the research. Abraar uploaded the dataset to Google Drive and set it up so that all members could easily access the folder through the Drive. Abraar also focused on training non-neural network models like SVC and RFC, and in training the InceptionV3 and Xception transfer learning models. Abraar worked on generating Plotly graphs and confusion matrices of the different machine learning models used. In the writing part, Abraar concentrated on describing the dataset, providing the background of the research and specifying the machine learning and deep learning techniques used to solve the image classification task. Abraar also described the workflow of the SVC and RFC models to solve the image classification problem, as well as the architectures of the InceptionV3 and Xception models. The outcomes of training the SVC, RFC, InceptionV3, and Xception models were discussed by Abraar. Abraar came up with the title of the research. Moreover, Abraar found some related work on chest CT scans and included the necessary references based on the related work.

Dosbol Aliev: Dosbol was involved in applying regularization techniques and using the Keras Dropout layer to control possible overfitting to improve the overall model accuracy and minimize the loss. Dosbol also trained the CNN model with Dropout layers and the EfficientNetB2 transfer learning model. In the writing part, Dosbol focused on describing the CNN model with Dropout layers as well as the architecture of the EfficientNetB2 model, and focused on describing successful outcomes of the research. Dosbol also discussed the outcomes of training the EfficientNetB2 model.

REFERENCES

[1] K. C. Arbour and G. J. Riely. "Diagnosis and Treatment of Anaplastic Lymphoma Kinase-Positive Non-Small Cell Lung Cancer". In: *Hematol Oncol Clin North Am* 31.1 (Feb. 2017), pp. 101–111. DOI: 10.1016/j.hoc.2016.08.012.

[2] B. A. Derman, K. F. Mileham, P. D. Bonomi, et al. "Treatment of advanced squamous cell carcinoma of the lung: a review". In: *Transl Lung Cancer Res* 4.5 (Oct. 2015), pp. 524–532. DOI: 10.3978/j.issn.2218-6751.2015.06.07.

[3] Timothy Dozat. "Incorporating Nesterov Momentum into Adam". In: 2016.

[4] Saeed Mirsadraee Edwin JR van Beek and John T Murchison. "Lung cancer screening: Computed tomography or chest radiographs?" In: *World J Radiol* 7.8 (Aug. 2015), pp. 189–193. DOI: 10.4329/wjr.v7.i8.189.

[5] Mohamed Hany. *Chest CT-Scan images Dataset*. URL: https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images.

[6] *IDX-dr*. June 2022. URL: https://www.digitaldiagnostics.com/products/eye-disease/idx-dr/.

[7] Daniel E. Jonas, Daniel S. Reuland, Shivani M. Reddy, et al. "Screening for Lung Cancer With Low-Dose Computed Tomography: Updated Evidence Report and Systematic Review for the US Preventive Services Task Force". In: *JAMA* 325.10 (Mar. 2021), pp. 971–987. ISSN: 0098-7484. DOI: 10.1001/jama.2021.0377. eprint: https://jamanetwork.com/journals/jama/articlepdf/2777242/jama_jonas_2021_us_210001_1614735444.90487.pdf. URL: https://doi.org/10.1001/jama.2021.0377.

[8] Mohamed N.E. Kassem and Doaa T. Masallat. "Clinical Application of Chest Computed Tomography (CT) in Detection and Characterization of Coronavirus (Covid-19) Pneumonia in Adults". In: *J Digit Imaging* 34.2 (Feb. 2021), pp. 273–283. DOI: 10.1007/s10278-021-00426-5.

[9] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[10] S.-C.B. Lo, S.-L.A. Lou, Jyh-Shyan Lin, et al. "Artificial convolution neural network techniques and applications for lung nodule detection". In: *IEEE Transactions on Medical Imaging* 14.4 (1995), pp. 711–718. DOI: 10.1109/42.476112.

[11] *numpy: Scientific computing with Python*. URL: http://numpy.org.

[12] Tom O'Malley, Elie Bursztein, James Long, et al. *KerasTuner*. https://github.com/keras-team/keras-tuner. 2019.

[13] *OpenCV computer vision library*. URL: http://opencv.org/.
[14] *scikit-learn*. URL: https://scikit-learn.org/stable/.
[15] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. DOI: 10.48550/ARXIV.1409.1556. URL: https://arxiv.org/abs/1409.1556.
[16] *Tensorflow Machine Learning Framework*. URL: https://www.tensorflow.org/.