Early detection of lung cancer through Deep Learning

Gabrielle Gantos Xiaoxia Lin Ekaterina Putintseva

Abstract

In the future, algorithms will play a major role in how we practice medicine. This project attempts to automate lung cancer detection in patient CT scans taken a year prior to diagnosis. An important first step in the analysis of lung cancer CT images is the detection of pulmonary nodules. Even among highly-trained medical professionals, malignant pulmonary nodules are difficult to identify in the early stages of disease when diagnosis would have the most impact. Using several Convolution Neural Network architectures and machine learning classifiers, here we propose two models for lung nodule analysis and early lung cancer detection.

1 Introduction

With recent advances in deep learning, research has made a significant leap to help identify, classify, and quantify patterns in medical images. Particularly, improvements in computer vision inspired its use in medical image analysis such as image segmentation, image registration, image fusion, image annotation, computer-aided diagnosis and prognosis, lesion/landmark detection, and microscopic imaging analysis, to name a few [6]. For instance, 2-D skin cancer images were successfully classified by a single CNN architecture in [3].

In this same vein, Kaggle selected the topic for the 2017 Data Science Bowl competition in order to focus teams of data scientists on improving lung health through early lung cancer detection. In recent months, sophisticated attempts to solve this problem have been released. Beginning with AlexNet and continuing with the model described by one of the winners of the Kaggle Data Science Bowl competition [2] as a foundational framework, we propose a simple and elegant approach to the challenge.

2 Methods

2.1 Data Cleaning and Preprocessing

Two datasets were used to explore early lung cancer detection: Kaggle Data Science Bowl CT [4] scans and LUng Nodule Analysis 2016 challenge (LUNA16) [5] CT scans. Both CT scan datasets are high resolution, represent a patient's lung tissue at a single point in time, and are representative of a heterogeneous range of scanner models and technical parameters. Kaggle data were provided by the National Cancer Institute while LUNA16 data are a subset of the publicly available LIDC/IDRI dataset.

Each patient in the Kaggle competition is represented by a single, 3-D CT scan without diagnostic metadata such as age, weight, or sex. Cancer and no-cancer labels were confirmed by pathology diagnosis and provided for most patients. CT scans range in resolution from 0.49 mm - 0.98 mm per pixel and have inter-slice spacing ranging from 0.625 mm - 3.0 mm.

LUNA16 patients are represented by a 3-D CT scan and a file of $1,172$ nodule locations annotated with likelihood of malignancy, spiculation, lobulation, and diameter (mm). Likelihood of malignancy represents an estimation of malignancy under the assumption that the patient is a 60-year-old smoker. Lobulation and spicualtion characterize nodule shape and all three estimates are classed by independent radiologists on a range from 1 to 5 [**?**].

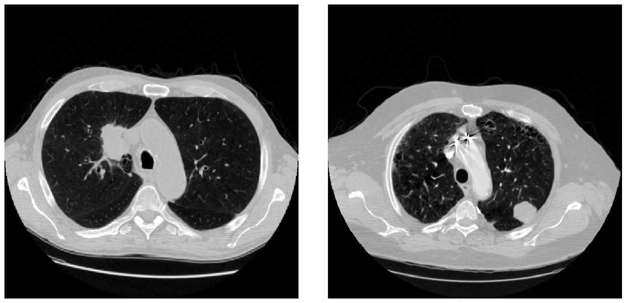

Figure 1: Largest LUNA nodule sized 32.3 mm (left) and large Kaggle nodule (right)

It should be noted that the nodules represented in both datasets differ meaningfully in size and location. Generally, annotated LUNA nodules are smaller than those in the Kaggle dataset as seen in (Figure 1) [7].

2.1.1 Whole Image Preprocessing

Kaggle data were initially preprocessed in order to load whole 3-D patient scans as inputs to a CNN architecture. Various resolutions and image spacings were standardization through the following:

- Conversion of original scan intensity scale to Hounsfield Units (HU)
- Lung segmentation and masking
- Resampling such that each pixel is representative of $3mm^3$

2.1.2 Patch Preprocessing

For nodule analysis and cancer prediction through image patches, Kaggle scans were augmented by annotated LUNA scans. Kaggle preprocessing for patch extraction resembles whole image preprocessing and is summarized as follows:

- Conversion of original scan intensity scale to Hounsfield Units (HU)
- Resample such that each pixel represents $1mm^3$
- Extract lung tissue mask without segmenting the image
- For patches containing lung tissue, extract via sliding window

Each LUNA16 patient is represented in the training set by nodule-centered voxels, random, non-nodule-centered voxels, and false positive candidate-centered voxels (Figure 2). If the patient does not have a nodule or false positive candidate then the patient is included only by random, non-nodule-centered voxels. LUNA data have already been scaled to HU, so extracted patches are only resampled such that each pixel represents $1mm^3$ before CNN training.

2.2 CNN architectures

Three fundamentally different CNN architectures were constructed in order to predict whether or not a patient will be diagnosed with cancer within one year.

2.2.1 AlexNet

The first model was a replica of the classic AlexNet convolutional neural network model, developed by Alex Krizhevsky [1]. The inputs to this baseline model were whole 3D CT scans (see Section 2.1) and outputs were 'cancer'/'no-cancer' patient labels.

A classical AlexNet was constructed with a 3x3x3 convolutional kernel sizes, two convolution/max pooling blocks, three subsequent convolution blocks and three concluding fully connected layers. To reduce overfitting, we implement dropout in the dense layers with a 50% rate. Moreover, we introduce

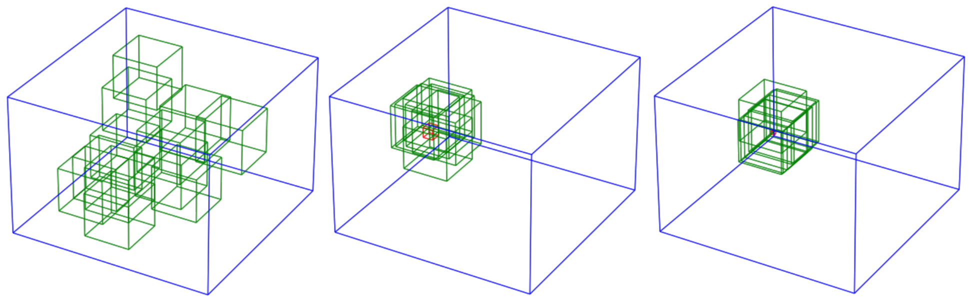

Figure 2: For each patient, random voxels (left), voxels randomly shifted around a 13.6 mm-sized nodule (red) (middle), and voxels randomly shifted around a nearby, 4.3 mm-sized nodule (red) (right).

an L2-regularization term with a degree of 0.001 to each of the layers. ReLU activation functions were used at each layer, except for the output layer, where a softmax function was implemented instead. Since the output of the CNN was 0 or 1 class ('non-cancer' or 'cancer' label), we used a categorical cross entropy loss function. Stochastic gradient descent optimizer with the learning rate 0.01 and Nesterov momentum were used as the model's optimizer function.

2.2.2 Patch-based CNN regressor for nodule analysis

The general framework for the second model was adapted from the paper by Daniel Hammack [2]. Overall, 16 different models with various hyperparameters are constructed for this part. All the architectures consist of 5 convolutional blocks, followed by a global max pooling and two dense layers. Each junction between convolutional blocks is complemented with a parallel stream of average pooling layers of the intact initial patches, augmenting the information being transferred (see Figure 3).

2.2.3 Patch-based cascade nodule CNN classifier and feature regressor

In order to separate the process of nodule detection and characterization as well as to facilitate the latter, our third model includes two parts: a CNN classifier and a CNN regressor. Both of these parts are trained using the LUNA16 dataset (see Section 2.1).

For the nodule classifier, only the positive and negative subsets were used, as the presence of false positive samples on this stage introduced inter-subset confusion and increased the loss function. On the second step, however, the negative subset was composed of false positive samples and negative samples predicted to be positive by the classifier. A consistent 20% validation set was used across both cascade stages.

The overall architecture is depicted in Figure 3B with each of the 2 blocks utilizing the same core architecture as the model described in the previous section. In total, 7 different models with various hyperparameters were built and analyzed.

The obtained optimized models from Sections 2.2.2 and 2.2.3 were used to predict the corresponding values of extracted patches from Kaggle patients. Based on the obtained predictions, a classification model was built for discriminating between no-cancer and cancer patients (see Section 2.3).

2.3 Patient-level cancer classification and prediction

2.3.1 Feature selection

The final step in our early lung cancer detection pipeline is to predict diagnosis using predictions of patch characteristics. Kaggle patches extracted in Section 2.1 utilize the LUNA-trained CNN model (see Section 2.2.2) to predict malignancy, spiculation, lobulation, and nodule diameter for each patch. From these predicted attributes, we select 14 relevant features for classification of patients into cancer and no-cancer:

- Max malignancy, spiculation, lobulation, and diameter for all nodules (4 features)

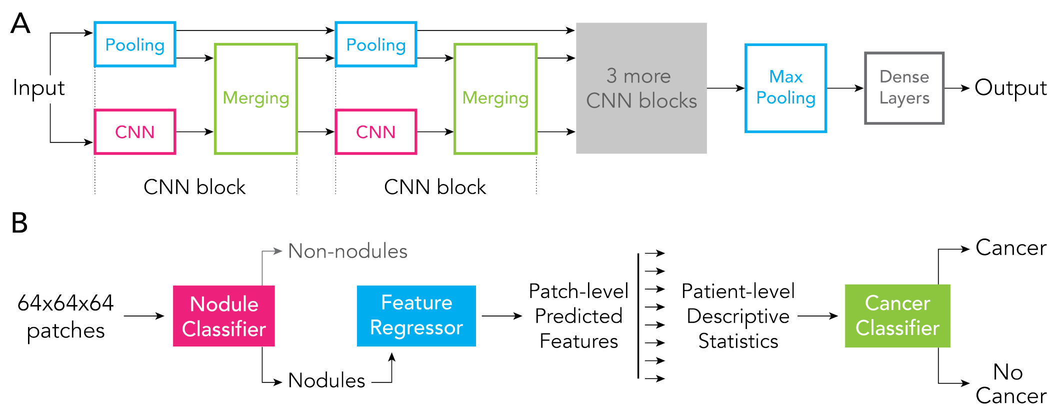

Figure 3: Schematic representation of machine learning work flow. (A) Principal deep learning architecture, (B) Cascading nodule classifier and feature regressor.

- Standard deviation of malignancy, spiculation, lobulation, and diameter for all nodules (4 features)
- Location of most malignant nodule (3 features, one for each dimension in the scan)
- Standard deviation of nodule locations in each dimension (3 features)

2.3.2 Cancer Classifiers

The selected 14 features are fed into different classifiers (Logistic regression, Support Vector Machine, Random Forest, etc.) with different parameter tuning (e.g. class-weight, C, penalty, cv-folds etc) to predict disease diagnosis. We also iterated through different feature combinations before ultimately selecting to use all 14 features. We implemented a classifier consisting of an L1-penalized logistic regression model followed by an Extremely Random Trees regressor fit on the residuals from the linear model [2].

We adjust for very unbalanced data (363 cancer patients versus 1071 no-cancer patients) using class-weights for all classifiers. Furthermore we used Stratified K-Folds cross-validator to preserve the percentage of samples for each class and all training and test data were standardized before fitting into classifiers.

2.3.3 Evaluation Metrics

In the Kaggle competition, the aim is to predict the likelihood that a patient will be diagnosed with lung cancer a year from when the scan was taken. Log loss is used to evaluate the prediction result.

We decide to use the same loss function in order to compare the results with the participants in the competition. We have to mention that this comparison is not 100% fair, since the second stage label is not available, our loss is computed only based on stage1 data using cross validation.

Furthermore, we compute the confusion matrix of the classification result that enable us to define other important metrics such as recall. In cancer detection because the cost of missing one patient in a trial is very large, we want the predictor to have very large recall (we do not accept false negatives) even when this means accepting more false positives. These false positives can be discarded in subsequent tests.

3 Experimental Results

We divide all the experimental results into three major subsections according to the different deep learning architectures implemented.

3.1 AlexNet

The first explored approach was constructing a model using AlexNet CNN architecture. Whole 3D CT patient scans are fed into the CNN to predict whether or not a certain patient is likely to develop cancer within a year. While trying to optimize this architecture, GPU memory constraints were encountered. Attempting to reduce scan resolution and batch sizes did not mitigate this limitation so we were unable to generate appropriate model results. In any case, when taking into account the ratio between a lung CT scan (~$30cm^3$) and a small nodule (~$1cm^3$), there was not much room for scan resolution reduction without dramatically affecting the chances of cancer prediction.

3.2 Patch-based CNN regressor

3.2.1 Nodule Analysis

To overcome the memory limitations, we use a similar dataset (LUNA16), in which malignant nodules are identified and annotated (see Section 2.1.2). The training dataset contains 50% nodule and 50% non-nodule patches. In the process of training, for some models, the non-nodule set of patches includes 50% false positive candidate patches. This was done in order to increase the sensitivity and the predictive power of the model.

The predicted features were: malignancy, spiculation, lobulation, and diameter. The convolutional part of the architecture (Figure 3) was shared across all the four feature models. Then, the CNN branched out to optimize and predict all the different features. The branching point was optimized and chosen to be on the level before the fifth convolutional block, since there is no overfitting. In general, overfitting was the major problem (Figure 4). In order to control for it, a 20% portion of the dataset was used as a validation set. A number of parameters was tuned in order to eliminate overfitting:

1. *Dropout*. Across different models, the dropout rate was increased from 0 to 20% in the dense layers of the architecture and set to 0% in the convolutional blocks.
2. *Regularization.* L2 regularization term was introduced on all the levels of the neural network with a maximum of 0.001 degree.
3. *Parameters reduction.* The number of neurons in the first dense layer was reduced from 100 to 32. The number of filters, corresponding to different convolutional blocks, was reduced from 8-24-64-72-72 to 8-24-48-64-64.
4. *Data augmentation.* Since the amount of patches containing malignant nodules was less than benign ones, for the former type of patches we used data augmentation. It included jittering around nodules (see Section 2.1.2) and flips over random axes.
5. *Early stopping.* Early stopping was implemented with patience of 4 epochs and validation data loss change of 0.001.
6. *The right choice of an optimizer.* Although the training loss was much lower with Nadam optimizer, for instance, the validation loss started to increase steadily already after several epochs. The optimizer used in the final model was stochastic gradient descent with Nesterov momentum.
7. *Learning rate scheduler.* For the final model, the following scheduler was used (<2 : 0.01, <5 : 0.001, <10 : 0.0005, else : 0.00005).

It is interesting to note that predicted features had different levels of overfitting. The most susceptible feature of all was malignancy, while all the others looked comparatively stable across a large variety of different hyperparameters.

Among other parameters, different activation functions were tested. In the final model, a Leaky ReLU was used as an activation function for convolutional blocks and a softmax was implemented in the output dense layer.

3.2.2 Cancer Prediction

In order to get an intuition of how well the model was predicting, we analyzed paired box plots and a scatter plot of the feature matrix for cancer and non-cancer patients and the feature importance

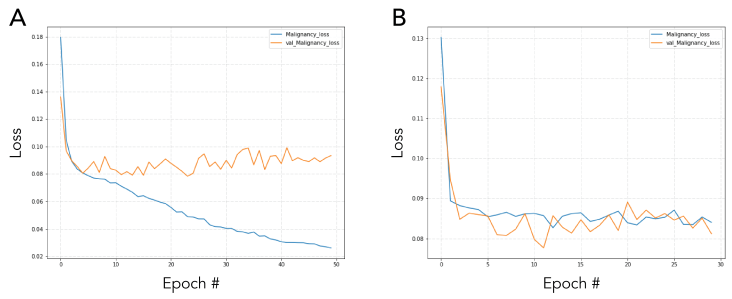

Figure 4: Overfitting illustration for the malignancy feature prediction model. (A) The learning curves of one of the initially built models, (B) The learning curves of the final model.

histogram. To our surprise, the features for both cancer and non-cancer patients are extremely similar, and all the features seem to have almost equal importance.

We suspect that it would be very challenging for our classifiers to distinguish two classes that are so similar. Our suspicion is confirmed in the result (see Table 1).

We want to analyze and understand the possible causes of the poor prediction. As already mentioned in section 2.1, LUNA has some very small annotated nodules, which may be less relevant to cancer prediction. In our model, we do not exclude these nodules and so we suspect these small nodules may cause noise. The other significant difference between LUNA nodules and Kaggle nodules as seen in 1 indicates that a network trained on LUNA alone will struggle to adequately learn to detect large nodules with such different characteristics.

As a positive control, we used LUNA patients to validate that the trained CNN model performs well on LUNA patches. Therefore we followed the same methodology as for Kaggle1 to classify and predict cancer and no-cancer in LUNA patients. Ideally, we would split LUNA data into training and test sets. However, due to the limited number of nodules and patients on which to predict, the same patients used to train were used to validate our model's predictive capabilities and estimate training error.

For each LUNA patient and annotated nodule, we generated predictions using patches extracted using randomness different from the randomness used to generate patches for the training dataset. We predict on these patches and build a feature matrix similar to that of Kaggle1 except for that we exclude patch location. And we plotted the feature importance histogram, a paired box plot and a scatter plot from 30 cancer and 30 no-cancer patients to compare features relevancy. We see that the max malignancy, spiculation, lobulation, and diameter appear to be the most important important in early cancer prediction. Also, the max malignancy, spiculation, lobulation, and diameter are higher in cancer patients than the no-cancer patients.

3.3 Patch-based cascade nodule CNN classifier and feature regressor

3.3.1 Nodule Analysis

The third model consists of a cascade of two CNN predictors (Figure 3B). The first CNN is specialized to discriminate between patches containing malignant nodules from the patches that do not contain malignant nodule tissue. The second CNN uses the predicted malignant patches to predict four features of interest mentioned above: malignancy, lobulation, spiculation and diameter.

Both parts of the model were optimized sequentially. For the nodule classifier, 7 models with different hyperparameters were built. By decreasing the number of neurons in the dense layer to 10, increasing the droupout rate, tuning kernel initializer, loss function and optimizer, we successfully reduced the level of overfitting (Figure 5) and managed to obtain a comparatively accurate nodule classifier.

After tuning a number of different parameters, the second part of the cascade ended up being a close replica of the model implemented in Section 3.2. The higher dropout rate (0.3) and a 0.3:0.7 ratio between negative and positive samples, correspondingly, substantially improved our results (Figure 6).

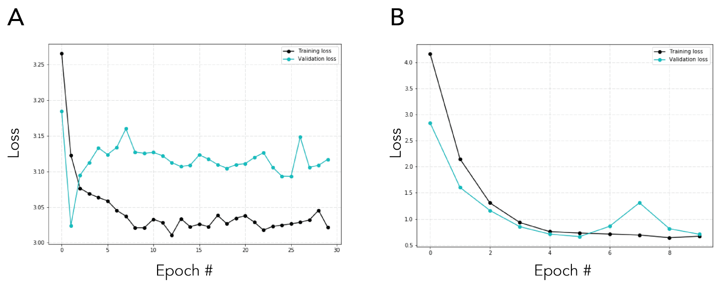

Figure 5: (A) The learning curves of the initially built model, (B) The learning curves of the final model.

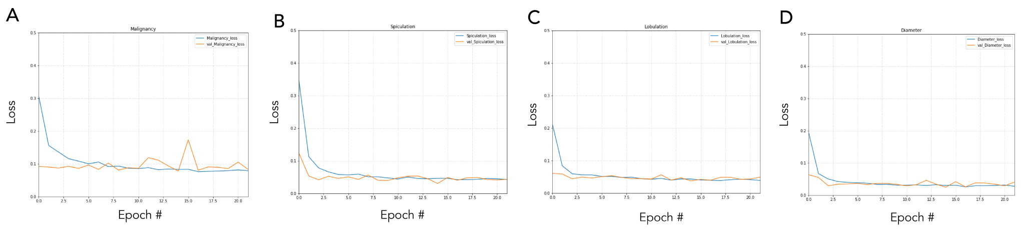

Figure 6: The resulting regression model learning curves for four predicted features.

3.3.2 Cancer Prediction

We used the constructed models in the same pipeline as described in Section 3.2.2 to predict whether a patient will develop cancer within the next year. To our surprise, although the loss function in the course of training the regressor was lower than the one of model 2 (6), the prediction accuracy was not improved 1. We believe that this is due to the data inconsistency problem presented in Section 3.2.

Table 1: Classification by Logistic Regression

Model	Sklearn CF	Recall (TP/T)	Specificity (FN/F)	Accuracy	Log-Loss
Kaggle V2	$\begin{bmatrix} 578 & 493 \\ 183 & 180 \end{bmatrix}$	0.50	0.54	0.53	0.69
LUNA V2	$\begin{bmatrix} 238 & 55 \\ 98 & 497 \end{bmatrix}$	0.84	0.81	0.83	0.39
Kaggle V3	$\begin{bmatrix} 553 & 518 \\ 183 & 180 \end{bmatrix}$	0.50	0.52	0.51	0.70

As a point of reference, we would like to highlight that the winning team achieved a log-loss score of 0.39975 in the Kaggle competition (lower scores are better).

4 Conclusion and Future Work

Medical image recognition for diagnostic prediction is difficult due to computation resources required, non-standardized data collection practices, and potential for signal minimization due to human-to-human variational noise. We proposed three CNN architectures for early cancer detection using patient CT scans. Although our model was not able to significantly predict cancer in Kaggle patients, the positive control test performed on LUNA patients demonstrated learning for nodules of that dataset's size and our Log-Loss places us in the top quartile of competitors in the Kaggle DSB. There is a lot of work that can be done to improve our model such as thresholding nodule diameter in training, extended data augmentation, ensemble learning, and further exploration of the feature space and CNN architectures.

5 Individual Contributions

I started this project by exploring and collecting information about Kaggle and LUNA datasets, I spent time following some great tutorial in the Kaggle so I can fully understand the data and contribute in the data processing and the 3-D patch extraction. We worked together to ensure the development environment was robust and well organized. We had a memory issue when we attempted to download, store and utilize the immensity of both LUNA and Kaggle datasets. I overcame this problem by transferring the data to a bigger disc and symlink the data.

Later on I mainly focus on the machine learning classifier portion, selection features and tuning parameters. I built the feature and classification visualizations that allowed me to analyze the accuracy of the entire pipeline. This is the first feedback we received about our model. I used and tuned number of different classifiers included Logistic Regression, Random Forest, SVM, and so on. And finally I assessed the model accuracy using different metrics across different classifiers. On the process of continuing to research, I found out a hypothetical reason for our model no be able to accurately predict the nodule on Kaggle patients using the model trained in LUNA data (fundamental difference in nodule size and location).

References

[1] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. *NIPS 2012*, pages 115–118, 2017.

[2] Hammack D. 2nd place solution to 2017. 2017.

[3] Novoa RA Ko J Swetter SM Blau HM Thrun S. Esteva A, Kuprel B. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118, 2017.

[4] Kaggle. Data science bowl 2017. 2017.

[5] Luna. Luna 16 grand challenge. 2016.

[6] Suk HI Shen D, Wu G. Deep learning in medical image analysis. *Annu Rev Biomed Eng*, 19(3):221–248, 2017.

[7] ‘grt123’ team. Solution of the ‘grt123’ team. 2017.