

Predicting Lung Cancer Incidence from CT Imagery

Darren Baker
Stanford University
drbaker@stanford.edu

Jen Kilpatrick
Stanford University
jdkil@stanford.edu

Ali Chaudhry
Stanford University
alick@stanford.edu

Abstract

Lung cancer is one of the most common forms of cancer worldwide, and is responsible for a large number of deaths and significant health care costs. Human radiologists typically use low-dose CT (computed tomography) scans of patients' lungs to assess an individual's risk of lung cancer, usually by inspecting the images for the presence of tissue growths called "nodules" that are a common precursor to cancer. However, even for highly trained radiologists, detecting nodules and predicting their relationship to cancer are challenging tasks, leading to both false positive and false negative results that can adversely affect patient health.

Our initial objective in this project was to determine whether we could make direct inferences about a patient's risk of lung cancer based on the application of deep learning models to "raw" CT imagery, but our attempts to train both 2D and 3D convolutional networks over full lung images were largely unsuccessful, perhaps due to the low signal-to-noise ratio that also makes human diagnosis difficult. As a result, we modified our approach to focus on explicit identification of lung nodules, in the hope that a more specific learning task would lead to better results. Though we were still unsuccessful in training a 3D convolutional network from scratch on labeled nodule data, we ultimately found that a transfer learning approach using 2D image "slices" of nodules and other tissue produced classification accuracy of 85-90%. We believe that this approach might be successfully extended to predict cancer incidence through a two-stage classification process that first identifies nodules and then attempts to infer their malignancy separately.

1. Introduction

Lung cancer is one of the most common forms of cancer worldwide. In the United States alone, 225,000 new cases were diagnosed in 2016, and total health care expenditures on lung cancer treatment exceeded \$12 billion in that year. Up to 20% of deaths from lung cancer are estimated to be preventable with early detection and treatment. [1] To facilitate such detection, human radiologists use low-dose CT (computed tomography) scans like those shown in Figure 1 to look for tissue growths, commonly referred to as "nodules," that may develop into cancer.

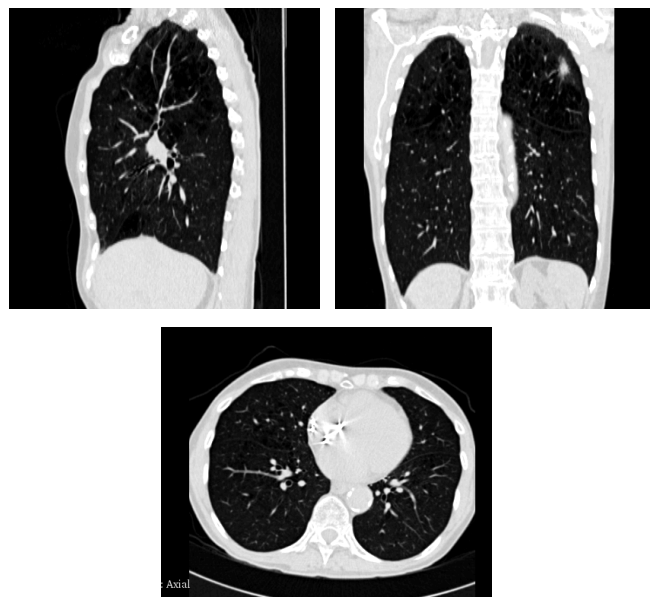


Figure 1. CT scan images from a single patient, showing tissue in the lungs from 3 different orientations.

Unfortunately, even for highly trained radiologists, potentially cancerous nodules can be very difficult to identify for several reasons. First, nodules are typically small, particularly in the pre-cancer stage; second, their appearance is not always distinct from that of other benign tissue formations in the lungs; and third, the resolution of CT imagery can vary in ways that make precise identification challenging. As a result, even expert diagnostic techniques can suffer from relatively high rates of false negative and false positive results. (False positives are a particular problem, since medical professionals are conservative and tend to recommend beginning cancer treatment even in cases where they believe that malignant nodules may be present but cannot determine their presence conclusively from the available imagery.) Of course, misdiagnosis in either direction can significantly affect the health and well-being of patients through either delayed or unnecessary treatments, and lead to higher mortality rates and costs in the health care system overall.

In light of these challenges, there is a need for better techniques to assess CT scans for cancerous lung lesions, with a broad goal of improving predictive precision. In recent years, neural network algorithms and deep learning

techniques have been applied very effectively to computer vision problems in both classification and object detection – especially where large, rich volumes of training data are available. Our goal in this project was to explore applications of these advanced machine learning approaches – especially convolutional neural networks – to the problem of automated lung cancer prediction over CT images. Specifically, we hoped to take as input a set of 3D pixel values produced by a CT scanner for a single patient and train one or more convolutional models to predict a binary label for the patient, with “1” meaning that the patient would be diagnosed with lung cancer within a year of the scan date, and “0” meaning no cancer diagnosis. As our experiments progressed and we found that model performance on this task was subpar, we shifted our strategy and attempted instead to classify 2D and 3D pixel regions as “nodule” or “not nodule.” The sections that follow describe our approaches and experiments in more detail.

2. Related work

We analyzed the existing research literature for work on both cancer prediction and nodule detection/classification over CT scans and other forms of medical imagery.

As demonstrated in the official video for the Kaggle competition [1], nodule detection is primarily done manually by trained pulmonary radiologists with the help of CAD (computer-aided diagnosis) systems. Existing CAD systems are designed to be highly sensitive to any potential nodules, so they capture numerous potential candidates for nodules and then a radiologist looks through the nodules to classify them. A typical CAD system segments the image to exclude tissues outside the lungs and uses nodule-enhancing filtering and thresholding to identify locations worth exploring. It then extracts hand-crafted feature data to generate candidates for review by a radiologist. [2]

Studies analyzing the efficacy of CAD systems look at the performance of a radiologist with and without CAD systems [3]. A prominent study by van Beek et al. found that CAD systems increase the sensitivity (recall for the positive class) of nodule detection from 64% to 93% while the decrease in specificity (recall for the negative class) was marginal - from 98% to 96%. [4] [5] Niemeijer et al. noted that CAD systems had different strengths and weaknesses, and as a result, different systems sometimes detected different nodules. These researchers combined multiple CAD systems and saw a significant increase in performance compared to the best individual CAD systems. [6]

Motivated by the desire to develop CAD systems with higher sensitivity and specificity, some studies have focused on using CNNs to classify nodules using the candidates generated by a CAD system. In the 1990s, researchers used shallow CNN network architectures for

this classification task. [7] More recently, deeper CNNs have been used for this task. [8] [9] In other domains, such as dermatology, deep CNNs have already shown performance on par with medical experts: for instance, in identifying common forms of skin cancer. [10]

Transfer learning from proven deep learning models like Google’s “Inception” can be an effective strategy for many computer vision tasks, because the parameters learned in the lower layers of the network can generalize even to image domains other than the one on which the network was originally trained. [11] More recently, studies have explored whether transfer learning can be used to help a network trained for an object detection task learn to perform other tasks as well. [12] Ginneken et al. used “OverFeat” trained for object detection in natural images as a starting point to train the network to distinguish between candidates and actual nodules. [13]

Instead of relying on CAD systems, some studies propose the use of methods to identify candidate regions automatically without having to hand-design features the way CAD systems do. Anirudh et al. propose unsupervised segmentation to “grow” 3D regions around weak labels that contain the central pixel of a nodule. [14]

For the purposes of classifying a patient’s risk of developing cancer, nodule detection is typically a prerequisite for that task because the signal-to-noise ratio in the raw CT scan images is too low and those raw images cannot directly be fed into a classifier for risk prediction. Nodule identification models help detect regions of the image that are likely to contain nodules, and those “suspect” regions can then directly be fed into a cancer classifier, which is designed to look closely at those regions that have already been identified as having a higher signal-to-noise ratio than the raw CT image. [15]

For our approaches to the problems of classifying nodules and cancerous patients, it was not necessary for us to obtain the precise positions of the bounding boxes around the nodules, because we were primarily interested in capturing the general region around the nodule. Therefore, we chose not to use models like Faster R-CNN [16], which focus on object localization. Instead, we focused on methods that classify small regions of the image as containing the nodule or not.

3. Modeling & prediction approaches

We pursued four related but distinct approaches to making lung cancer predictions using 2D and 3D data from patient CT scans.

3.1. 2D convolution on individual slices

The CT scans in the Kaggle dataset (described in more detail in section 4 below) consisted of a variable number of 2D image “slices” for each patient. Our baseline model

trained a 2D convolutional network on individual slices, using the single label for the corresponding patient (“1” if the patient had been diagnosed with cancer in the following year, “0” otherwise) to determine the proper class of the training image. To classify a test patient as cancerous or not, we ran the model on each of the slices for that patient separately to generate a predicted label for each slice; then, if the percentage of cancerous slices was greater than or equal to a threshold value that we chose, we would classify that patient as cancerous. We calculated per-patient classification accuracy by comparing our thresholded prediction against the patient’s original label.

As background to our approach, 2D convolutions work by sliding a number of “filters” (say, a 3x3-pixel “window”) across a 2D image, calculating a single value at each position of the filter’s output by taking the dot product of the filter weights and the pixel values lying under the filter at that position. In an intuitive sense, different filters can be thought of as learning different “concepts” or pixel patterns occurring in small patches of an image – whether simple geometric concepts like “corners” or more elaborate ideas like “faces” – that might be important to understand for purposes of identifying or classifying objects. Using convolutional layers on images is particularly effective because, unlike dense layers, they preserve the two-dimensional spatial relationships between pixels in the image. Our baseline network used a single 2D convolutional layer with a relatively large filter size (15x15 px) and stride (7x7 px), followed by a ReLU activation and a 4x4 max pooling layer. This was followed by two fully connected layers – one ReLU-activated, one tanh-activated – and a final projection layer to output the per-class logits. We used the standard softmax cross-entropy loss and an Adam optimizer to train the network.

Because this baseline model evaluates each slice of a patient’s 3D scan independently, it has several practical advantages: it is simple to build, and it normally less memory-intensive and far faster to train than a full 3D convolutional model (described next). However, it also comes with several disadvantages. One is that it fails to take advantage of any spatial relationships that may exist between pixels in neighboring slices. Perhaps the biggest concern is that the overall per-patient label may be misleading or effectively “wrong” when applied to individual slices of the patient’s scan. For example, if a patient only has a single nodule but we apply a label of “1” (cancerous) to a slice of that patient’s data where no nodule is visible, this might cause the model to learn an inaccurate representation of what a cancerous patient’s lungs actually look like (and how that may be different from a healthy patient). We were very aware of this potential drawback, but still wanted to explore how well such a model might be able to perform on this task.

3.2. 3D convolution on full patient scans

We believed that we could likely learn more from the Kaggle data if we preserved the original 3D structure of the CT images. Thus, our second strategy for predicting cancer incidence involved creating a 3D convolutional network and training the model on a full 3D pixel array for each patient. Convolutions in 3 dimensions are a logical extension of convolutions in 2 dimensions: the main difference is that filter sizes and strides are specified in 3 dimensions, because each filter moves over the entire 3D space of the training image in order to calculate the dot product between its own weights and the pixel values in each position. Our network used an initial ReLU-activated 3D convolution layer; two composite layers that include a 3D convolution, batch normalization, ReLU activation, and 3D max pooling, using decreasing filter sizes but more filters at deeper layers; and 3 fully connected layers with a combination of ReLU and tanh activations. Like the 2D convolutional model, this model uses softmax cross-entropy loss and an Adam optimizer.

Two key challenges with this model, described further in section 4 below, were the huge memory requirements for operating on 3D data and the non-uniformity of image dimensions across patients. (The latter issue meant that we had to scale, pad, and/or truncate patient data in some cases to achieve a uniform input size for the model.) These issues meant that our cycle time for experimenting with different model architectures and hyperparameters was relatively long, and we were never entirely sure whether the training data selected for a given patient actually included the visual elements that would help the model recognize cancer.

3.3. 3D convolution for nodule identification

One concern we had with the two previous methods was that training on a large CT image with just a single label per patient (cancer or not) might provide too little “signal” for a model to learn which image attributes were associated with cancer, and which were not. Therefore, we considered a second general strategy of trying to learn to predict cancer-related information using a more focused dataset with more specific labels.

As described in the introduction, radiologists who review CT imagery are trained to inspect the patient’s lungs for “nodules,” which are small tissue growths that are frequently an early indication of cancer. However, pre-cancerous nodules can be difficult to distinguish from other forms of lung tissue, including benign growths and normal structures such as blood vessels. At this stage of the project, we shifted our objective to classifying certain lung regions as “nodule” or “not nodule,” in the hope that if we could train a model to identify nodules with high precision, the output of this classifier could then be used to predict cancer incidence as well.

To pursue this strategy, we switched from the original Kaggle dataset to a similar dataset called LUNA 2016 [24], described in more detail in section 4 below. Like the Kaggle dataset, the LUNA data included 3D CT scans of patient lungs, but unlike the Kaggle data, its training data took the form of 3D coordinates of thousands of specific locations within the patient scans that were labeled as being nodules or not. We extracted “cubes” (3D pixel arrays of uniform dimension) around these candidate locations and set up a 3D convolutional model to try to classify the nodule status of each cube.

Our model architecture at this stage was similar but not identical to the architecture described by de Wit [17] for performing nodule identification on the LUNA dataset. Specifically, we included the following sequence of layers:

- Input has the same size along X/Y/Z dimensions (usually 64x64x64 px or 32x32x32 px).
- Average pooling with filter/stride size of 2 in the Z direction (effectively just downsampling data in this direction, since it typically has lower spatial resolution than the X/Y dimensions in LUNA data)
- 3D convolution [3x3x3 filter size, 1x1x1 stride, 64 filters] followed by 2x2 max pooling in the X/Y dimensions (brings overall dimensions along X/Y/Z axes back to the same value)
- 2-3 more (depending on input dimension) layers of 3D convolution (filter size of 3, stride of 1) followed by 2x2x2 max pooling, with filter count increasing at each layer to a maximum of 512
- Final 3D convolution layer with filter size of 2 and stride of 2, projecting filter count down to 64
- Final fully-connected layer projecting the remaining data down to un-normalized logits for the 2 output classes (nodule or non-nodule)
- Softmax cross-entropy loss on the output logits

3.4. Nodule identification via transfer learning

Our final approach employed transfer learning in an attempt to produce better results on the nodule identification task described in the previous step. The most robust pre-trained models that we found were all designed to operate on 2D RGB images, and had been trained over the past several years on the ImageNet dataset that is widely used in computer vision research. We chose to use Google’s “Inception v3” model [19], which forms a large network by stacking layers of “inception modules” performing parallel 5x5, 3x3, and 1x1 convolutions, as our starting point for transfer learning. We also followed the outlines of a tutorial [21] that the Tensorflow team prepared on how to retrain the final layer of the Inception network using arbitrary images, though we extended the code [22] from the tutorial to produce more detailed output about the

model’s performance on our data, as it originally produced only aggregate accuracy statistics over the dataset.

Since Inception and other networks trained on ImageNet expect 2D RGB images as inputs, we first created 2D slices of the 3D pixel “cubes” that we had previously extracted around the candidate locations identified in the LUNA data. For each cube, we extracted 3 different slices: one for each of the 3 different possible axes-aligned orientations of the image around the center of the cube. Although there was no concept of color or multi-channel pixels in the original CT data, we simulated RGB images by simply stacking 3 layers of the existing pixel values on top of each other. Finally, we produced each 2D image in a range of sizes, from 32 pixels up to 224 pixels wide -- the latter being the default input size for most ImageNet models, including Inception. (Note that the images of different sizes were not simply rescaled versions of each other, but included a varying number of pixels from the original patient data to provide more or less visual context around the center point of the image.) The goal of generating 2D images with multiple orientations and multiple sizes was to experiment with which combinations might produce better results. For example, we theorized that smaller images (i.e. those with fewer “context” pixels surrounding the center) might actually be easier to classify, because the model would be able to focus on the main body of the nodule, and not on (likely) irrelevant surrounding tissue.

Once the extracted 2D images were ready, we performed many rounds of model retraining on the final layer of the Inception network, exploring both the different image sizes/orientations that we had produced, but also a range of hyperparameters (learning rate, batch sizes, number of epochs, etc.). As in our previous models, we used softmax cross-entropy over the two classes as our loss metric. We had intended to use the classification outputs of this model to return to our original problem of predicting cancer development, but the data pipeline preparation for this task proved sufficiently time-consuming that we ultimately ran out of time to close this loop.

4. Datasets & data preparation

As described in the previous section, we used two different datasets of lung CT images in the course of our project.

The first dataset was published in early 2017 by Kaggle, the machine learning competition platform, and included 3D image data from approximately 1,600 patients that were known to be at a high risk of developing lung cancer. These images were published in the DICOM (Digital Imaging and Communications in Medicine) format, a well-established standard for encapsulating medical imagery with extensive metadata about patients and other context that may be relevant for medical professionals. Each

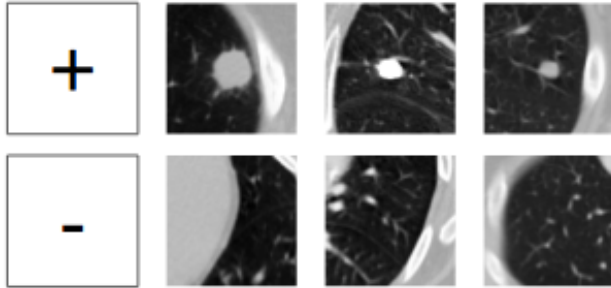


Figure 2. Examples of nodule candidates extracted from the LUNA 16 dataset. The top row shows examples of the positive class (“nodule”), while the bottom row shows examples of the negative class (“not nodule”).

“slices” of monochrome CT output. The slices were 512 by 512 pixels and oriented parallel to what radiologists refer to as the “axial” plane of the patient, meaning that they were horizontal and showed a top-down view of the patient’s lungs. The full 3D data for the patient could be reconstructed by reading in all the slices, extracting metadata that indicated their relative ordering along a vertical axis, and then “stacking” the images in the proper order. Finally, each patient’s 3D image came with a single binary label, indicating whether the patient was diagnosed with lung cancer within one year after their CT scan was taken. The class split in the dataset was moderately unbalanced, but not massively so, with about 71% of the patients falling into the “not cancer” class and the remainder in the “cancer” class.

The second dataset we used came from another machine learning challenge called LUNA (Lung Nodule Analysis) 2016, and consisted of 3D lung images for 888 patients. This data was represented in ITK format, another commonly-used standard for storing and manipulating medical imagery. The dimensionality of each patient’s data was similar to the Kaggle case: each patient had several hundred 2D slices of pixel data, with each slice containing 512x512 pixel values. The key difference between the LUNA and Kaggle datasets was in the label information: instead of a single per-patient label representing a cancer diagnosis, the LUNA data contained two lists of “annotations” over the patient data. The first list included the 3D coordinates of 1,186 nodules that had been conclusively identified and labeled by a team of radiologists over the images of the 888 patients. The second list contained a list of more than 750,000 nodule “candidates” that had been proposed by some recent algorithms for nodule detection in lung imagery. [2] Each candidate included a patient ID and 3D coordinates for the possible nodule location, along with a binary label indicating whether the candidate corresponded to one of the 1,186 nodules that had been positively identified in the dataset.

We spent significant time and effort on preparing the Kaggle and LUNA datasets for training. Some key steps:

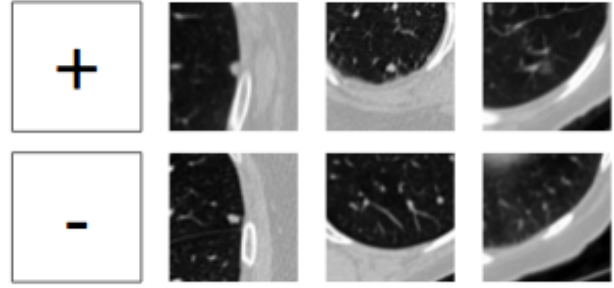


Figure 3. Additional examples of nodule candidates from the LUNA dataset. Unlike figure 2, the top-row and bottom-row candidates shown here are difficult to distinguish, though they belong to opposite classes.

- **Extracting pixels and metadata.** Both the DICOM and ITK formats required the use of specific Python code libraries and functions to retrieve the raw pixel data and other forms of metadata that we needed for things like 3D stacking and conversion to a format compatible with Numpy and Tensorflow.
- **Pixel value conversion and normalization.** The pixel values in the Kaggle dataset had to first be rescaled and translated from unsigned integers into “Hounsfield units” (HU), a unit of radio signal attenuation that is standard for capturing CT scan data. In both datasets, we also followed recommendations in the literature to truncate the data beyond certain HU values and normalize it to a [0.0, 1.0] scale.
- **Spatial normalization.** Although the 2D slices in each dataset had the same nominal dimensions – 512x512 pixels – the metadata in the files indicated that there was actually significant variation in the real-world distances represented by each pixel or slice. For example, the pixel density varied from roughly 0.5 mm per pixel to 1.0 mm per pixel in the X (patient’s left/right) and Y (patient’s front/back) directions, and from roughly 1.0mm to 2.5mm in the Z (patient’s head/foot) direction. In order to minimize distortions in the images due to these variations, we had to decimate and/or interpolate most images along all three dimensions in order to achieve a standard density of 1mm/pixel.
- **Constructing, padding, and truncating 3D arrays.** For both datasets, we used the metadata provided in the original images to “stack” the 2D slices into a 3D array in the proper vertical order. For the Kaggle dataset specifically, we also needed to choose a fixed input size for the 3D convolutional model that we tried to run over each patient’s 3D array, but the spatial normalization step meant that virtually every patient’s (normalized) 3D data had somewhat different sizes along the X, Y, and Z

dimensions. Therefore, after determining our model’s input size, we also had to zero-pad or truncate each patient’s data in each dimension to correspond to the selected size.

- **Extracting 2D and 3D arrays for nodules and candidates.** For the LUNA dataset, we had to pull out 2D and 3D images corresponding to each of the 1186 positive nodule annotations and a large number of the nodule “candidates.” This involved converting the 3D coordinates given in the annotation/candidate files to the coordinate system of our spatially-normalized 3D images and then extracting and saving 2D and 3D arrays of various pixel widths (32, 64, 128, 224) for use in our models.

Beyond the processing steps described above, we did not attempt to extract any explicit “features” from our images – all predictions in our models were made on the basis of preprocessed 2D or 3D pixel arrays. We did experiment with some forms of normalization and data augmentation – specifically, mean subtraction and 2D or 3D flips of training images – but they didn’t seem to affect our results in meaningful ways, so we did not prioritize further work along these lines.

Finally, from a practical standpoint, a key challenge in working with these datasets was simply the raw size of the image files. Even in compressed form, the Kaggle dataset was ~100GB and the LUNA data was another ~70GB. The largest 3D pixel arrays consisted of roughly 130 million pixels (512x512x500), meaning that just reading a single patient’s data required loading approximately 1GB of data from disk to memory. Though we used SSD storage and aggressive data preprocessing & caching strategies to minimize data loading times during our experiments, we continued to find that simply loading batches of patient data for each training step took far more time than actually running the training step – a significant obstacle to our speed of iteration.

5. Results and discussion

In this section, we describe the results from our experiments with the four different modeling approaches described in section 3.

5.1. Training 2D and 3D convolutional models from scratch

For each of our first three approaches – 2D convolution for cancer prediction, 3D convolution for cancer prediction, and 3D convolution for nodule identification – we built our own model architectures using Tensorflow, initialized their parameters using Xavier initialization, and attempted to train the model “from scratch” using the training data we prepared. In all cases, we experimented with a variety of

architecture configurations (conv filter sizes, number of layers, activation functions, batch norm and pooling vs. not) and a wide range of values for hyperparameters (learning rate, batch sizes, number of training epochs, and so on). Some representative parameter configurations included:

- Our 2D convolutional model for cancer prediction on the Kaggle dataset used batches of 100 slices for training and validation. We used a learning rate of 0.001 and decayed the learning rate gradually over the training process.
- Our 3D convolutional model for cancer prediction on the Kaggle dataset was trained on batches of 30 patients and ran validation on batches of 60 patients (even after reducing the input dimensionality of the 3D scans, anything batch size larger than ~60 was unable to fit in 12GB of GPU memory). We used fixed depths of 150-250 slices for each patient, padding or truncating data where necessary to fit that size. We primarily used learning rates of 1e-3 / 1e-4.
- Our 3D convolutional model for nodule identification on the LUNA dataset was trained on batches of 40 “cubes” around nodule candidates, and validated on batches of 100-200 such cubes. For this model specifically, because the class distribution in the candidate dataset was extremely skewed (roughly 1 positive candidate for every 500-600 negative candidates), we aggressively up-sampled members of the positive class to ratios of between 5% and 30% of the training and validation batches. For this model, we again used learning rates between roughly 1e-4 and 1e-2, though we also experimented with values well outside this range.

Unfortunately, despite extensive experimentation, we were never able to achieve good predictive performance with any of these 3 different models. Loss values did decline somewhat from the starting point of training, showing that the optimizer was doing its job on some level. However, in terms of classification accuracy, we commonly observed one of two conditions (depending on the hyperparameter values we chose):

- In the first scenario, the model would flounder with classification accuracies (on both training and validation sets) stuck in the range of 50-60% – not impressive for a binary classification task.
- In the second scenario, the model would simply decide to predict the same label for every element of the training or validation set – typically the majority class (e.g. “no cancer” or “non-nodule”), though sometimes the opposite if we tried to be more aggressive with the learning rate of up-sampling of positive examples in the training data.

Size of Image Input (pixels)			
Orientation	32	64	224
Side	78.7%	69.9%	62.6%
Front	76.8%	78.1%	63.0%
Top	83.9%	77.0%	74.8%

		Actual	
		Positive	Negative
Predicted	Positive	543	17
	Negative	107	107

Sensitivity	83.4%
Specificity	86.3%

Figure 4. (Top) Comparison of classification accuracy levels for different image sizes and orientations using transfer learning with default hyperparameters on the Inception v3 network. (Bottom) Confusion matrix for the specific training run highlighted in yellow.

Once in a while we would see classification accuracy begin to climb, as if the model had seen some particularly helpful training examples and was able to start learning. In these cases, we saw overall classification accuracy figures of 73-80% for the Kaggle data (vs. the baseline figure of ~71% of examples in the “not cancer” class), but usually this success was short-lived and the model soon fell back to a strategy of predicting the majority class.

Our background research on other attempts to apply deep neural networks to this type of imagery suggested that our (poor) results for these tasks were not unusual. For example, de Wit (one of the top performers on the actual Kaggle competition using the lung cancer dataset) noted that his and others’ initial attempts at applying end-to-end deep learning to the Kaggle dataset were also unsuccessful. [17] We suspect two basic reasons for this problem: first, the signal-to-noise ratio seemed extremely low in the lung cancer images that are only labeled with “cancer” or “not cancer,” and second, our volume of training data was quite small in a relative sense. In short, when looking at large regions of the lungs, it is extremely difficult even for a well-trained expert to identify whether the tissue structures that are visible are normal, abnormal, malignant, or benign – which means that it may not be reasonable to ask a neural network to learn to perform the same task with only a few hundred examples to work with. Even in the nodule identification scenario, the practitioners (de Wit and others) who found some success with running 3D conv models on the LUNA dataset appeared to rely heavily on augmented datasets with thousands of additional nodule annotations that we didn’t have time to add into our data pipeline.

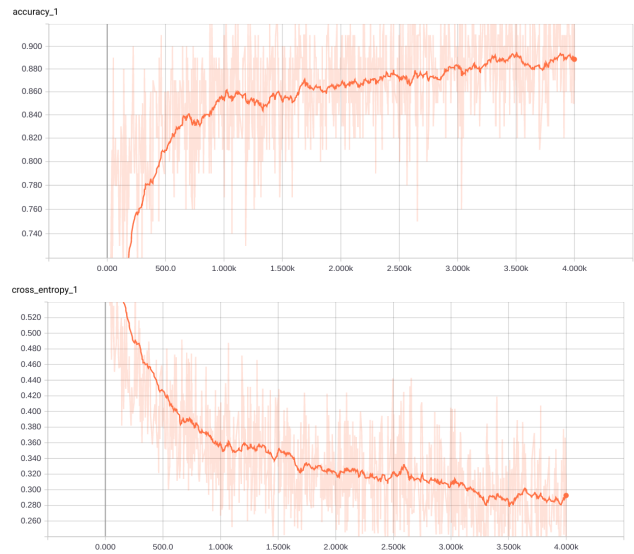


Figure 5. Training accuracy (top) and cross-entropy loss (bottom) graphs for one training run of our transfer learning model.

5.2. Retraining Inception v3 for nodule identification

Having failed to achieve very meaningful results from our attempts to build cancer and nodule classification networks from the ground up, we turned to a transfer learning strategy to see if a model originally trained on a much more extensive (though qualitatively very different) dataset might allow us to produce a model that would go beyond the simple and useless strategy of predicting the majority class.

Fortunately, our attempts to retrain Google’s Inception v3 model were much more successful than our previous strategies. Figure 4 shows a summary of some of our classification accuracy results for several different combinations of image sizes and orientations (without tuning hyperparameters). Figure 5 shows visualizations of training accuracy and cross-entropy loss over one training run, and Figure 6 highlights some variations in accuracy as we attempted to tune the learning rate for the model. Some highlights and discussion of the patterns that we saw:

- After hyperparameter tuning, the best-performing model we found had overall classification accuracy of 90.2% on the validation set that was automatically segregated from the rest of the data during the training process. We made a point of extracting the confusion matrix for the model’s predictions on the validation set to evaluate the sensitivity and specificity of the model, which are important metrics for comparing our performance with standards in the medical field. Although the average levels we achieved for these metrics (around 85% on most runs) are lower than the

Impact of Learning Rate vs Accuracy (32px image input)

	1	0.1	0.01	0.001	0.0001
Accuracy	83.7%	89.7%	83.9%	81.0%	78.2%

Figure 6. Variations in classification accuracy as learning rate changes.

current state of the art (in the low to mid-90s), we feel our results show promise for transfer learning-based approaches to nodule classification.

- In general, we found that predictive performance increased meaningfully as we progressed from larger to smaller images. Keeping in mind that the larger images are not simply rescaled versions of the smaller (or vice versa), but rather that different image sizes correspond to different extents of “context” in the pixel data, we believe that this is because the smaller images are much more “focused” on the actual nodule location (as given in the LUNA candidate list), and therefore the model is able to look closely at the pixels in and around the nodule area without being distracted by extraneous tissue that appears further away.
 - We also noted that training the nodule classifier on images in the “fixed z” (i.e. top-down) orientation tended to produce slightly higher accuracy levels than training on the “fixed y” (front-facing) or “fixed x” (side-facing) orientations of images – or even than training on a full set of images from all three orientations. Though we lack the medical expertise to comment on precisely why this might be the case, it seems plausible that something about the way that nodules typically form inside the lungs makes their shape or size more distinctive when viewed from a top-down orientation.
- Because the nodule candidate from the LUNA data was extremely skewed toward negative examples, we explicitly chose to balance the classes by choosing only roughly ~7 negative examples for each of the 1,186 positive examples contained in the dataset. We believe that this was an important factor in helping the transfer learning process to work well in recognizing both classes.

6. Conclusions and Future Work

Our work on this project has shown that although medical imagery may be amenable to productive analysis and prediction using computer vision techniques, the process of developing models that work well on medical data is not necessarily straightforward. We believe that there are several distinct challenges with medical imagery, including the lung cancer datasets we worked with, that make algorithm-based classification and detection particularly tricky. Perhaps the two most obvious challenges are:

- **The inherent difficulty of the task.** Even for a large and diverse dataset like ImageNet, with 14M+ images and 1000 categories, most humans do not have significant trouble distinguishing between “broccoli,” “sledding,” and other class labels. In our case, even highly skilled and experienced radiologists can have trouble distinguishing between “nodule” and “not nodule” when looking at CT images of the lungs, suggesting that this is a fundamentally hard vision task stemming from the ambiguity of the classes and the general lack of contextual clues in this type of imagery. The fact that the best existing algorithms for nodule detection produce an average of 500-1000 “candidates” for each actual nodule also provide some qualitative evidence for this point.
- **Small quantities of training data.** While datasets like ImageNet might be assembled by asking humans to label large volumes of data at relatively low cost through platforms like Mechanical Turk, collecting reasonable volumes of training data for vision tasks on medical imagery is orders of magnitude slower and more costly. Complex deep learning models with many layers and millions of parameters often work best when they are able to learn over large volumes of training data, so the lack of training data in our case - - only 1600 total patients (with one label each) in the Kaggle dataset, or 1,186 positive training examples of nodules in LUNA -- was a significant hindrance to our ability to train convolutional models from scratch.

On the positive side, it was heartening to find that a transfer learning strategy can be applied effectively even when the original problem domain (ImageNet) and the “new” domain to which the model is being transferred (lung nodule classification) are quite different from each other. Since we found some success with transfer learning for nodule identification using the Inception network, but lacked time to move beyond that approach, avenues that we would propose for future work include the following:

- Exploring whether transfer learning with other “base” models – for example, ResNets or networks pre-trained on medical images – might perform better for our task than Inception.
- Extending our transfer learning strategy to try retraining all layers of an existing model on our new data, instead of retraining only the final output layer.
- Experimenting with different loss functions that place asymmetric weights on false positive or false negative predictions, to encourage the model to bias toward eliminating the most costly or negative outcomes.
- Closing the loop with our original Kaggle dataset to see if we could build a two-stage cancer classifier that took the output of the nodule identification step and then attempted to predict cancer development.

References

- [1] Kaggle Data Science Bowl 2017. <https://www.kaggle.com/c/data-science-bowl-2017>
- [2] B. van Ginneken, C. Schaefer-Prokop, and M. Prokop. Computer-aided diagnosis: How to move from the laboratory to the clinic. *Radiology*, December, 2011.
- [3] K. Murphy, B. van Ginneken, A. M. Schilham, B. J. de Hoop, H. A. Gietema, M. Prokop. A large-scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbour classification. *Medical Image Analysis*. October, 2009.
- [4] E.J. Van Beek, B. Mullan, B. Thompson. Evaluation of a real-time interactive pulmonary nodule analysis system on chest digital radiographic images: a prospective study. *Academic Radiology*. May, 2008.
- [5] D.W. De Boo. Advances in digital chest radiography: impact on reader performance. UvA DARE. 2012.
- [6] M. Niemeijer, M. Loog, M D. Abramoff, M. A. Viergever, M. Prokof, B. V. Ginneken. On Combining Computer-Aided Systems. *IEEE Transactions on Medical Imaging*. February 2011.
- [7] S.B. Lo, S. Lou, J. S. Lin, M. T. Freedman, M. V. Chein, S. K. Mun. Artificial convolution neural network techniques and applications for lung nodule detection. *IEEE Transactions on Medical Imaging*. December 1995.
- [8] S. Hussein, R. Gillies, K. Cao, Q. Song, U. Bagci. TumorNet: Lung nodule characterization using multi-view convolutional neural network with Gaussian Process. *IEEE ISBI 2017*.
- [9] H. R. Roth HR, L. Lu, J. Liu, J. Yao, A. Seff, K. Cherry, L. Kim, R. M. Summers. Improving Computer-Aided Detection Using Convolutional Neural Networks and Random View Aggregation. *IEEE Transactions on Medical Imaging*. May 2016.
- [10] A. Esteva, B. Kuprel, R. Novoa, J. Ko, S. Swetter, H. Blau, S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*. February 2017
- [11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. Going Deeper with Convolutions. September 2014. arXiv:1409.4842 [cs.CV].
- [12] D. H. Murphree, C. Ngufor. Transfer Learning for Melanoma Detection: Participation in ISIC 2017 Skin Lesion Classification Challenge. March 2017. arXiv:1703.05235 [cs.CV].
- [13] B. V. Ginneken, A. A. A. Setio, C. Jacobs, F. Ciompi. Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans. *IEEE ISBI 2015*.
- [14] R. Anirudh, J J Thiagarajan, T. Bremer, H. Kim. Lung nodule detection using 3D convolutional neural networks trained on weakly labeled data. *Medical Imaging 2016: Computer Aided Diagnosis. Conference Volume 9785*.
- [15] K. Kuan, M. Ravaut, G. Manek, H. Chen, J. Lin, B. Nazir, C. Chen, T. C. Howe, Z. Zeng, V. Chandrasekhar. Deep Learning for Lung Cancer Detection: Tackling the Kaggle Data Science Bowl 2017 Challenge. arXiv:1705.09435 [cs.CV].
- [16] S. Ren, K. He, R. Girshick, J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems (NIPS) 2015*.
- [17] Julian de Wit, “2nd place solution for the 2017 national datascience bowl”, <http://juliandewit.github.io/kaggle-ndsb2017/>. Accessed 5/6/2017.
- [18] Daniel Hammack, “Forecasting Lung Cancer Diagnoses with Deep Learning,” https://github.com/dhammack/DSB2017/blob/master/dsb_2017_daniel_hammack.pdf. Accessed 4/22/2017.
- [19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. Rethinking the inception architecture for computer vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2016* (pp. 2818-2826).
- [20] Tensorflow deep learning framework. <https://www.tensorflow.org/>.
- [21] Tutorial: How to Retrain Inception’s Final Layer for New Categories. Accessible at https://www.tensorflow.org/tutorials/image_retraining.
- [22] Python code to retrain final layer of Inception code: https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/image_retraining/retrain.py.
- [23] Tutorial: Sequence-to-Sequence Models. (Used as model for overall code structure in our custom-built TF models.) <https://www.tensorflow.org/tutorials/seq2seq>
- [24] LUNA Grand Challenge 2016. <https://luna16.grand-challenge.org/>