Deep Convolutional Neural Networks for Lung Cancer Detection

Albert Chon
Department of Computer Science
Stanford University
achon@stanford.edu

Niranjan Balachandar
Department of Computer Science
Stanford University
niranja9@stanford.edu

Peter Lu
Department of Computer Science
Stanford University
peterlu6@stanford.edu

Abstract

Here we demonstrate a CAD system for lung cancer classification of CT scans with unmarked nodules, a dataset from the Kaggle Data Science Bowl 2017. Thresholding was used as an initial segmentation approach to to segment out lung tissue from the rest of the CT scan. Thresholding produced the next best lung segmentation. The initial approach was to directly feed in the segmented CT scans into 3D CNNs for classification, but this proved to be inadequate. Instead, a modified U-Net trained on LUNA16 data (CT scans with labeled nodules) was used to first detect nodule candidates in the Kaggle CT scans. The U-Net nodule detection produced many false positives, so regions of CTs with segmented lungs where the most likely nodule candidates were located as determined by the U-Net output were fed into 3D Convolutional Neural Networks (a vanilla 3D CNN and a Googlenet-based 3D CNN) to ultimately classify the CT scan as positive or negative for lung cancer. The vanilla 3D CNN produced a test set AUC of ??? and the Googlenet-based 3D CNN produced a test set AUC of ???. While performance of our CAD system is not state-of-the-art, current CAD systems in literature have several training and testing phases that each requires a lot of labeled data, while our CAD system has only three major phases (segmentation, nodule candidate detection, and malignancy classification), allowing more efficient training and detection and more generalizability to other cancers.

1. Introduction

Lung cancer is one of the most common cancers, accounting for over 225,000 cases, 150,000 deaths, and $12 billion in health care costs yearly in the U.S. [1]. It is also one of the deadliest cancers; overall, only 17% of people in the U.S. diagnosed with lung cancer survive five years after the diagnosis, and the survival rate is lower in developing countries [1]. The stage of a cancer refers to how extensively it has metastasized. Stages 1 and 2 refer to cancers localized to the lungs and latter stages refer to cancers that have spread to other organs. Current diagnostic methods include biopsies and imaging, such as CT scans [2]. Early detection of lung cancer (detection during the earlier stages) significantly improves the chances for survival, but it is also more difficult to detect early stages of lung cancer as there are fewer symptoms.

Our task is a binary classification problem to detect the presence of lung cancer in patient CT scans of lungs with and without early stage lung cancer. We aim to use use methods from computer vision and deep learning, particularly 2D and 3D convolutional neural networks, to build an accurate classifier. An accurate lung cancer classifier could speed up and reduce costs of lung cancer screening, allowing for more widespread early detection and improved survival. The goal is to construct a computer-aided diagnosis (CAD) system that takes as input patient chest CT scans and outputs whether or not the patient has lung cancer.

Though this task seems straightforward, it is actually a needle in the haystack problem. In order to determine whether or not a patient has early-stage cancer, the CAD system would have to detect the presence of a tiny nodule (< 10 mm in diameter for early stage cancers) from a large 3D lung CT scan (typically around 200 mm $\times$ 400 mm $\times$ 400 mm). An example of an early stage lung cancer nodule shown in within a 2D slice of a CT scan is given in Figure 1. Furthermore, a CT scan is filled with noise from surrounding tissues, bone, air, so for the CAD system's search to be efficient, this noise would first have to be preprocessed. Hence our classification pipeline is image preprocessing $\rightarrow$ nodule candidates detection $\rightarrow$ malignancy classification.

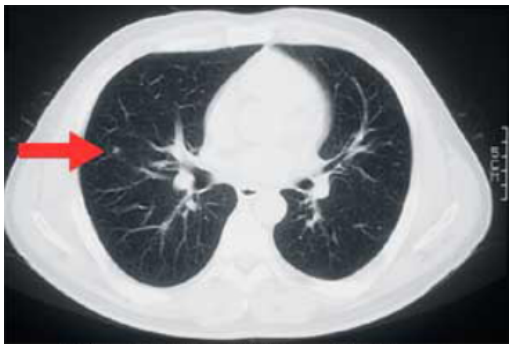


Figure 1: 2D CT scan slice containing a small (5mm) early stage lung cancer nodule [3].

2. Background

Typical CAD systems for lung cancer from literature have the following pipeline: image preprocessing → detection of cancerous nodule candidates → nodule candidate false positive reduction → malignancy prediction for each nodule candidate → malignancy prediction for overall CT scan [4]. These pipelines have many phases, each of which is computationally expensive and requires well-labeled data during training. For example, the false positive reduction phase requires a dataset of labeled true and false nodule candidates, and the nodule malignancy prediction phase requires a dataset with nodules labeled with malignancy.

True/False labels for nodule candidates and malignancy labels for nodules are sparse for lung cancer, and may be nonexistent for some other cancers, so CAD systems that rely on such data would not generalize to other cancers. In order to achieve greater computational efficiency and generalizability to other cancers, our CAD system will have shorter pipeline and will only require the following data during training: a dataset of CT scans with true nodules labeled, and a dataset of CT scans with an overall malignancy label. State-of-the-art CAD systems that predict malignancy from CT scans achieve AUC of up to 0.83 [5]. However, as mentioned above, these systems take as input various labeled data that we do not use. We aim for our system to reach close to this performance.

3. Data

Our primary dataset is the patient lung CT scan dataset from Kaggle's Data Science Bowl 2017 [6]. The dataset contains labeled data for 2101 patients, which we divide into training set of size 1261, validation set of size 420, and test set of size 420. For each patient the data consists of CT scan data and a label (0 for no cancer, 1 for cancer). Note that the Kaggle dataset does not have labeled nodules. For each patient, the CT scan data consists of a variable number of images (typically around 100-400, each image is an axial slice) of 512×512 pixels. The slices are provided in DICOM format. Around 70% of the provided labels in the Kaggle dataset are 0, so we use a weighted loss function in our malignancy classifier to address this imbalance.

Because the Kaggle dataset alone proved to be inadequate to accurately classify the validation set, we also use the patient lung CT scan dataset with labeled nodules from the LUng Nodule Analysis 2016 (LUNA16) Challenge [7] to train a U-Net for lung nodule detection. The LUNA16 dataset contains labeled data for 888 patients, which we divide into a training set of size 710 and a validation set of size 178. For each patient the data consists of CT scan data and a nodule label (list of nodule center coordinates and diameter). For each patient, the CT scan data consists of a variable number of images (typically around 100-400, each image is an axial slice) of 512×512 pixels.

LUNA16 data was used to train a U-Net for nodule detection, one of the phases in our classification pipeline. The problem is to accurately predict a patient's label ('cancer' or 'no cancer') based on the patient's Kaggle lung CT scan. We will use accuracy, sensitivity, specificity, and AUC of the ROC to evaluate our CAD system's performance on the Kaggle test set.

4. Methods

We preprocess the 3D CT scans using segmentation, normalization, downsampling, and zero-centering. Our initial approach was to simply input the preprocessed 3D CT scans into 3D CNNs, but the results were poor, so we needed additional preprocessing to input only regions of interests into 3D CNNs. To identify regions of interest, we train a U-net for nodule candidate detection. We then input regions around nodule candidates detected by the U-net into 3D CNNs to ultimately classify the CT scans as positive or negative for lung cancer. The structure of our code and weight-saving methods is from CS224N Winter 2016 Assignment 4 starter code.

4.1. Proprocessing and Segmentation

For each patient, we first convert the pixel values in each image to Hounsfield units (HU), a measurement of radiodensity, and we stack 2D slices into a single 3D image. Because tumors form on lung tissue, we use segmentation to mask out the bone, outside air, and other substances that would make our data noisy, and leave only lung tissue information for the classifier. A number of segmentation approaches were tried, including thresholding, clustering (K-means and Meanshift), and Watershed. K-means and Meanshift allow very little supervision and did not produce good qualitative results. Watershed produced the best qualitative results, but took too long to run to use by the deadline. Ultimately, thresholding was used in our pipeline. Thresholding and Watershed segmentation are described below.

After segmentation, we normalize the 3D image by applying the linear scaling to squeezed all pixels of the original unsegmented image to values between 0 and 1. Then we use spline interpolation to downsample each 3D image by a scale of 0.5 in each of the three dimensions. Finally, we zero-center the data be subtracting the mean of all the images from the training set.

4.1.1 Thresholding

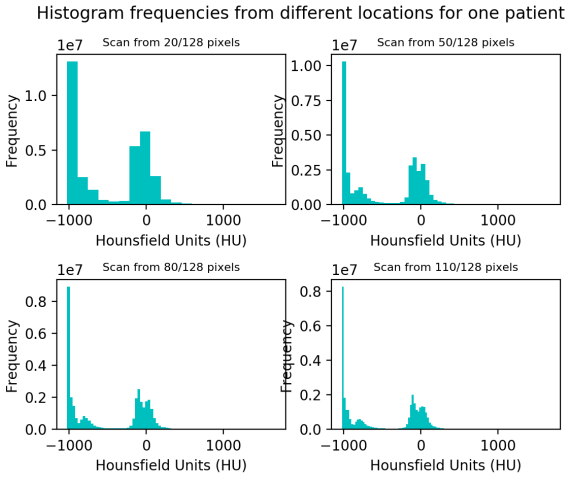
Typical radiodensities of various parts of a CT scan are shown in Table 1. Air is typically around -1000 HU, lung tissue is typically around -500, water, blood, and other tissues are around 0 HU, and bone is typically around 700 HU, so we mask out pixels that are close to -1000 or above -320 to leave lung tissue as the only segment. The distribution of pixel Hounsfield units at various axial slices for a sample patient are shown in Figure 2. Pixels thresholded at 400 HU are shown in Figure 3, and the mask is shown in Figure 4 However, to account for the possibility that some cancerous growth could occur within the bronchioles (air pathways) inside the lung, which are shown in Figure 5, we choose to include this air to create the finalized mask as shown in Figure 6.

Substance	Radiodensity (HU)
Air	-1000
Lung tissue	-500
water and blood	0
bone	700

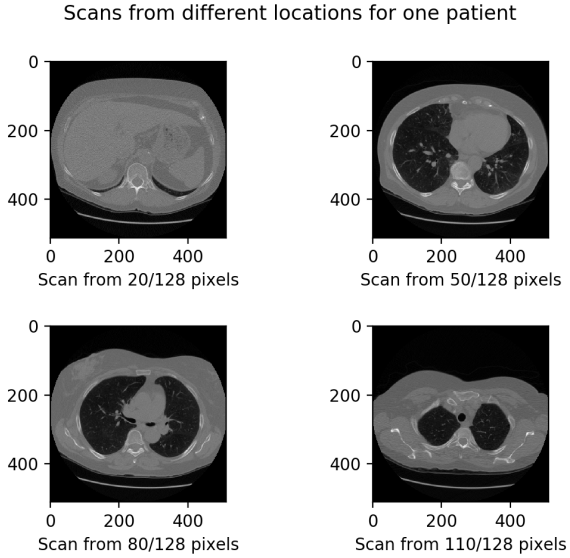
Table 1: typical radiodensities in HU of various substances in a CT scan [8]

4.1.2 Watershed

The segmentation obtained from thresholding has a lot of noise- many voxels that were part of lung tissue, especially voxels at the edge of the lung, tended to fall outside the range of lung tissue radiodensity due to CT scan noise. This means that our classifier will not be able to correctly classify images in which cancerous nodules are located at the edge of the lung. To filter noise and include voxels from the edges, we use Marker-driven watershed segmentation, as described in Al-Tarawneh et al. [9]. An original 2D CT slice of a sample patient is given in Figure 7. The resulting 2D slice of the lung segmentation mask created by thresholding is shown in Figure 8, and the resulting 2D slice of the lung segmentation mask created by Watershed is shown in Figure 10. Qualitatively, this produces a much better segmentation than thresholding. Missing voxels (black dots in Figure 8) are largely reincluded. However, this is much less efficient than basic thresholding, so due to time limitations,



(a) Histograms of pixel values in HU for sample patient's CT scan at various slices.



(b) Corresponding 2D axial slices

Figure 2: (a) Histogram of HU values at (b) corresponding axial slices for sample patient 3D image at various axial slice

we were unable to preprocess all CT scans using Watershed, so we used thresholding.

4.2. U-Net for Nodule Detection

We initially tried directly inputting the entire segmented lungs into malignancy classifiers, but the results were poor. It was likely the case that the entire image was too large a search space. Thus we need a way of inputting smaller regions of interest instead of the entire segmented 3D image. Our way of doing this is selecting small boxes containing top cancerous nodule candidates. To find these top nodule

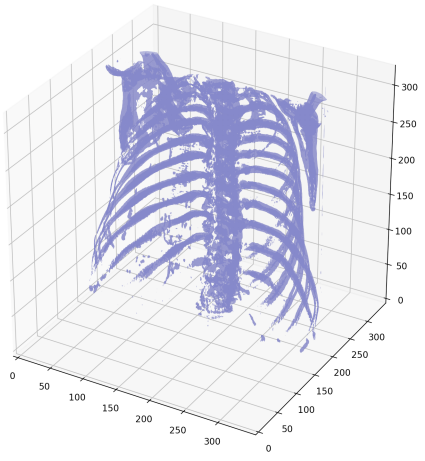


Figure 3: Sample patient 3D image with pixels values greater than 400 HU reveals the bone segment.

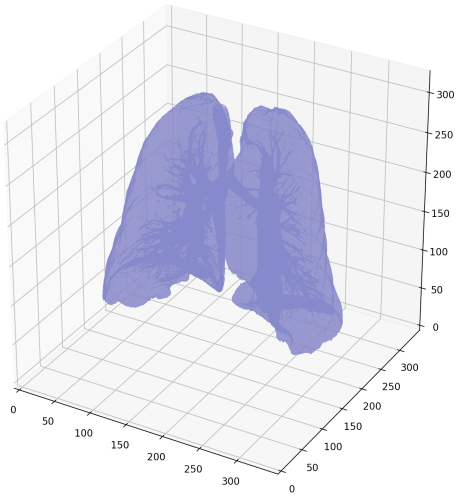


Figure 4: Sample patient initial mask with no air

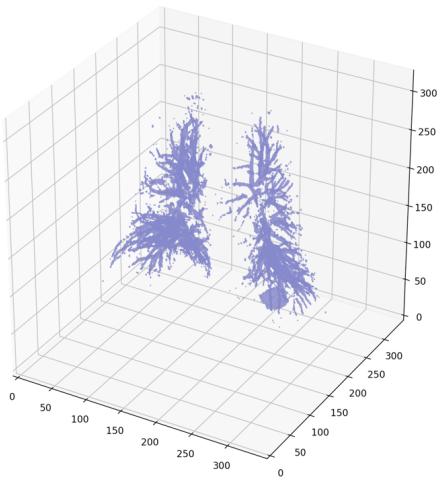


Figure 5: Sample patient bronchioles within lung

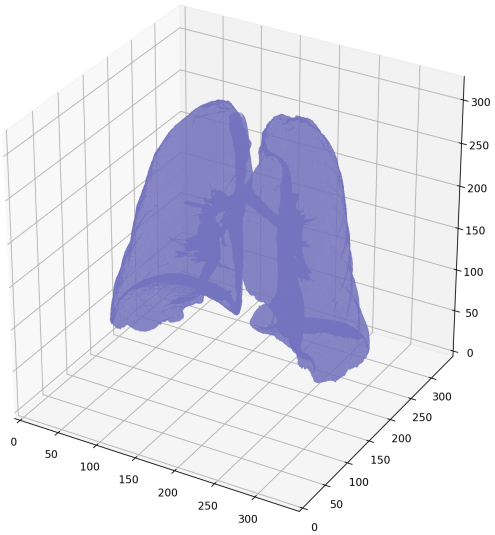


Figure 6: Sample patient final mask in which bronchioles are included

candidates, we train a modified version of the U-Net as described in Ronneberger et al. on LUNA16 data [10]. U-Net is a 2D CNN architecture that is popular for biomedical image segmentation. We designed a stripped-down version of the U-Net to limit memory expense. A visualization of our U-Net architecture is included in Figure 11 and is described in detail in Table 2. During training, our modified U-Net takes as input 256×256 2D CT slices, and labels are provided (256×256 mask where nodule pixels are 1, rest are 0). The model is trained to output images of shape 256×256 were each pixels of the output has a value between 0 and 1 indicating the probability the pixel belongs to a nodule. This is done by taking the slice corresponding to label one of the

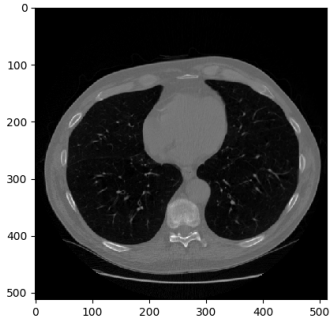


Figure 7: original 2D slice of sample patient

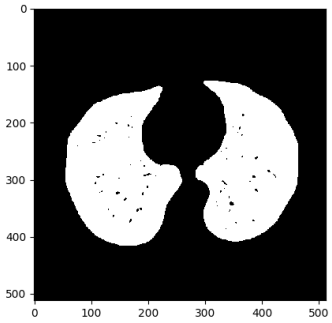


Figure 8: lung segmentation mask by thresholding of sample patient

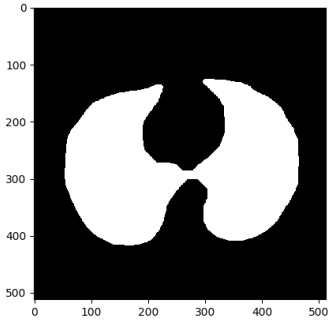


Figure 9: final watershed segmentation mask of sample patient

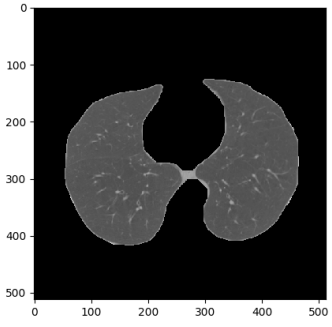


Figure 10: final watershed lung segmentation of sample patient

softmax of the final U-Net layer. Corresponding U-Net inputs, labels, and predictions on a patient from the LUNA16 validation set is shown in Figures 12, 13, and 14 respectively. most nodules are much smaller A weighted softmax cross-entropy loss calculated for each pixel, as a label of 0 is far more common in the mask than a label of 1. The trained U-Net is then applied to the segmented Kaggle CT scan slices to generate nodule candidates.

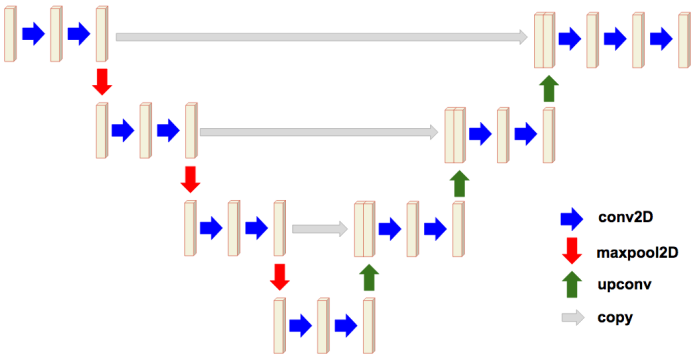


Figure 11: Modified U-Net architecture

Layer	Params	Activation	Output
Input			256 x 256 x 1
Conv1a	3x3x3	ReLu	256 x 256 x 32
Conv1b	3x3x3	ReLu	256 x 256 x 32
Max Pool	2x2, stride 2		128 x 128 x 32
Conv2a	3x3x3	ReLu	128 x 128 x 80
Conv2b	3x3x3	ReLu	128 x 128 x 80
Max Pool	2x2, stride 2		64 x 64 x 80
Conv3a	3x3x3	ReLu	64 x 64 x 160
Conv3b	3x3x3	ReLu	64 x 64 x 160
Max Pool	2x2, stride 2		32 x 32 x 160
Conv4a	3x3x3	ReLu	32 x 32 x 320
Conv4b	3x3x3	ReLu	32 x 32 x 320
Up Conv4b	2x2		64 x 64 x 320
Concat	Conv4b,Conv3b		64 x 64 x 480
Conv5a	3x3x3	ReLu	64 x 64 x 160
Conv5b	3x3x3	ReLu	64 x 64 x 160
Up Conv5b	2x2		128 x 128 x 160
Concat	Conv5b,Conv2b		128 x 128 x 240
Conv6a	3x3x3	ReLu	128 x 128 x 80
Conv6b	3x3x3	ReLu	128 x 128 x 80
Up Conv6b	2x2		256 x 256 x 80
Concat	Conv6b,Conv1b		256 x 256 x 112
Conv6a	3x3x3	ReLu	256 x 256 x 32
Conv6b	3x3x3	ReLu	256 x 256 x 32
Conv7	3x3x3		256 x 256 x 2

Table 2: U-Net architecture (Dropout with 0.2 probability after each 'a' conv layer during training, 'Up' indicates resizing of image via bilinear interpolation, Adam Optimizer, learning rate = 0.0001)

4.3. Malignancy Classifiers

Once we trained the U-Net on the LUNA16 data, we ran it on 2D slices of Kaggle data and stacked the

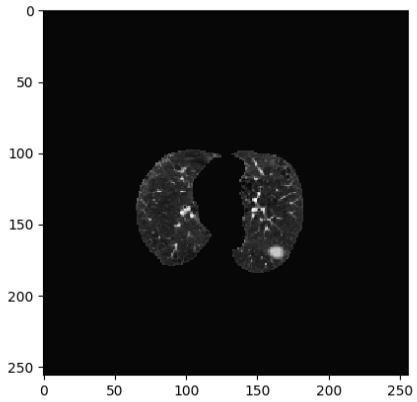


Figure 12: U-Net sample input from LUNA16 validation set. Note that the above image has the largest nodule from the LUNA16 validation set, which we chose for clarity-most nodules are significantly smaller than the largest one in this image.

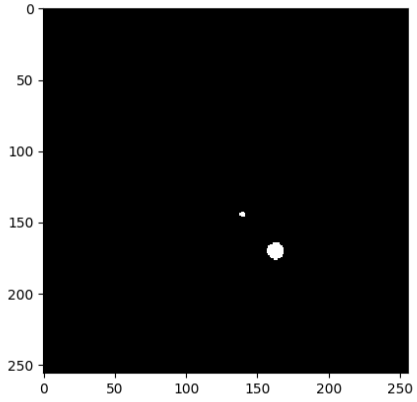


Figure 13: U-Net sample labels mask from LUNA16 validation set showing ground truth nodule location

2D slices back to generate nodule candidates (Preprocessing and reading of LUNA16 data code based on `https://www.kaggle.com/arnavkj95/candidate-generation-and-luna16-preprocessing`). Ideally the output of U-Net would give us the exact locations of all the nodules, and we would be able to say images with nodules as detected by U-Net are positive for lung cancer, and images without any nodules detected by U-Net are negative for lung cancer. However, as shown in Figure 14, U-Net produces a strong signal for the actual nodule, but also produces a lot of false positives, so we need an additional classifier that determines the malignancy.

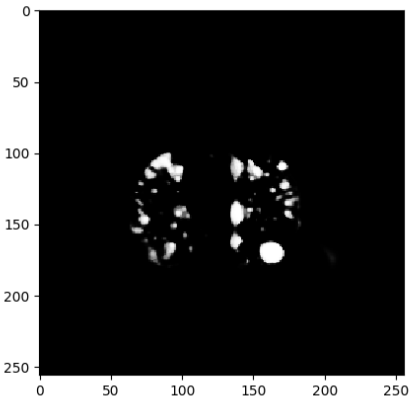


Figure 14: U-Net predicted output from LUNA16 validation set

Because our U-Net generates more suspicious regions than actual nodules, we located the top 8 nodule candidates ($32 \times 32 \times 32$ volumes) by sliding a window over the data and saving the locations of the 8 most activated (largest L2 norm) sectors. To prevent the top sectors from simply being clustered in the brightest region of the image, the 8 sectors we ultimately chose were not permitted to overlap with each other. We then combined these sectors into a single $64 \times 64 \times 64$ image, which will serve as the input to our classifiers, which assign a label to the image (cancer or not cancer).

We use a linear classifier as a baseline, a vanilla 3D CNN, and a Googlenet-based 3D CNN. Each of our classifiers uses weighted softmax cross entropy loss (weight for a label is the inverse of the frequency of the label in the training set) and Adam Optimizer, and the CNNs use ReLU activation and droupout after each convolutional layer during training. The vanilla 3D CNN is based on a 3D CNN designed for this task [11]. We shrunk the network to prevent parameter overload for the relatively small Kaggle dataset. A visualization of our vanilla 3D CNN architecture is included in Figure 15 and described in detail in Table 3

We also designed a 3D Googlenet-based model is based on the 2D model designed in Szegedy et al. for image classification [12]. A visualization of our 3D Googlenet is included in Figure 16 and described in detail in Table 4. Refer to Szegedy et al. for more information on the inception module [12].

5. Results

The results are shown in Table 5, and ROC curves for the Vanilla CNN and 3D Googlenet are shown in Figure 17.

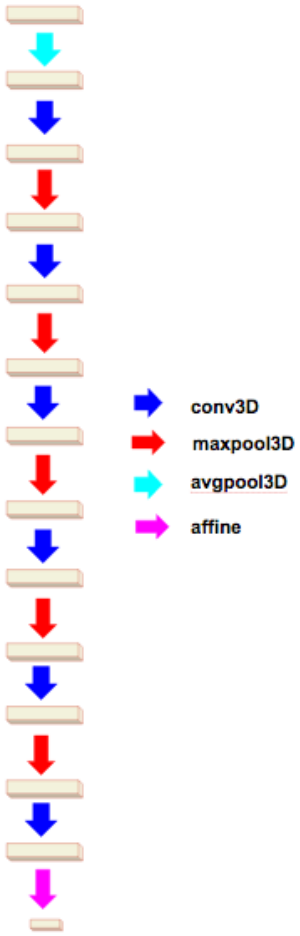


Figure 15: Vanilla 3D CNN architecture

Layer	Params	Activation	Output
Input			64 x 64 x 64 x 1
AvgPool	2x1x1		32 x 64 x 64 x 1
Conv1	3x3x3	ReLu	32 x 64 x 64 x 32
MaxPool	2x2x2		16 x 32 x 32 x 32
Conv2	3x3x3	ReLu	16 x 32 x 32 x 64
MaxPool	2x2x2		8 x 16 x 16 x 64
Conv3	3x3x3	ReLu	8 x 16 x 16 x 128
MaxPool	2x2x2		4 x 8 x 8 x 128
Conv4	3x3x3	ReLu	4 x 8 x 8 x 256
MaxPool	2x2x2		2 x 4 x 4 x 256
Conv5	3x3x3	ReLu	2 x 4 x 4 x 256
MaxPool	2x2x2		1 x 2 x 2 x 256
Conv6	3x3x3	ReLu	1 x 2 x 2 x 512
Dense			2

Table 3: Vanilla CNN architecture (dropout with 0.2 probability after each conv layer during training, Adam Optimizer with learning rate = 0.0003)

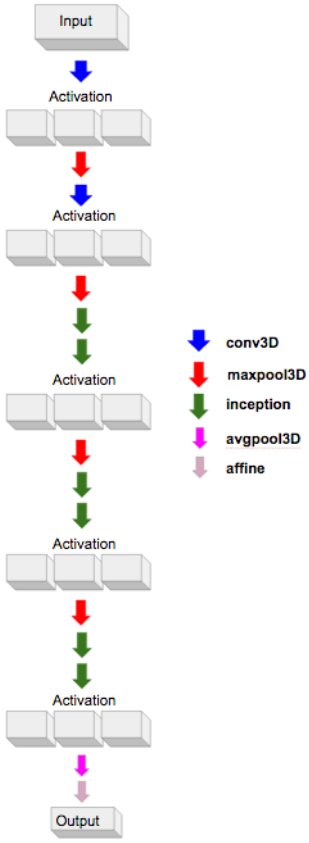


Figure 16: 3D googlenet architecture

Layer	Params	Activation	Output
Input			64 x 64 x 64 x 1
Conv1	7x7x7	ReLu	32 x 32 x 32 x 32
MaxPool	2x2x2		16 x 16 x 16 x 32
Conv2	3x3x3	ReLu	16 x 16 x 16 x 64
MaxPool	2x2x2		8 x 8 x 8 x 64
Inception1	1,3,5	ReLu	8 x 8 x 8 x 128
Inception2	1,3,5	ReLu	8 x 8 x 8 x 128
MaxPool	2x2x2		4 x 4 x 4 x 128
Inception3	1,3,5	ReLu	4 x 4 x 4 x 256
Inception4	1,3,5	ReLu	4 x 4 x 4 x 256
MaxPool	2x2x2		2 x 2 x 2 x 256
Inception5	1,3,5	ReLu	2 x 2 x 2 x 512
Inception6	1,3,5	ReLu	2 x 2 x 2 x 512
AvgPool	2x2x2	ReLu	1 x 1 x 1 x 512
Dense			2

Table 4: 3D Googlenet architecture (dropout with 0.3 probability after each conv and inception layer during training, Adam Optimizer with learning rate = 0.0001)

Model	Acc.	Sens.	Spec.	AUC
Linear	0.665	0.652	0.672	0.663
Vanilla 3D CNN	0.705	0.593	0.761	0.695
3D Googlenet	0.751	0.770	0.741	0.757

Table 5: Kaggle test set accuracy, sensitivity, specificity, and AUC of ROC (not shown for linear)

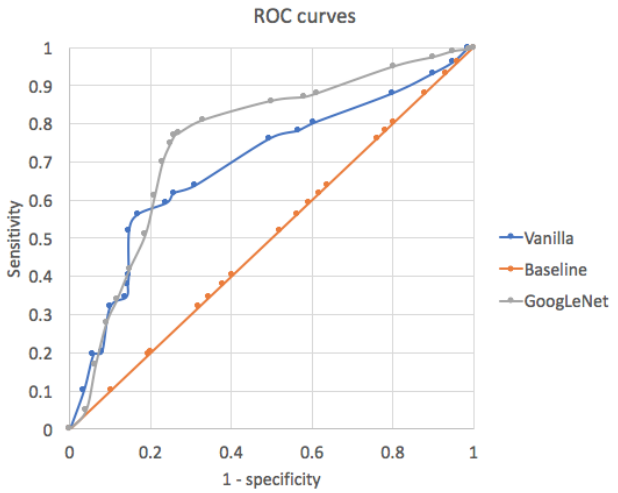


Figure 17: ROC for Vanilla 3D CNN and 3D Googlenet

6. Conclusions

The deep 3D CNN models, namely the Googlenet-based model, performed the best on the test set. While we do not achieve state-of-the-art performance AUC of 0.83, we perform well considering that we use less labeled data than most state-of-the-art CAD systems. As an interesting observation, the first activation layer of one of our older models (where we input the entire CT volume) for a validation example that was labeled as positive for cancer is shown in Figure 18. The bright pixels usually corresponded with the location of cancerous nodules, so it could be possible to extend our current model to not only determine whether or not the patient has cancer, but also determine the exact location of the cancerous nodules. The most immediate future work is to use Watershed segmentation as the initial lung segmentation. Other opportunities for improvement include making the network deeper, and more extensive hyperparameter tuning. Also, we saved our model parameters at best validation AUC, but perhaps we could have saved at other metrics, such as F1. Other future work include extending our models to 3D images for other cancers. The advantage of not requiring too much labeled data specific to our cancer is it could make it generalizable to other cancers.

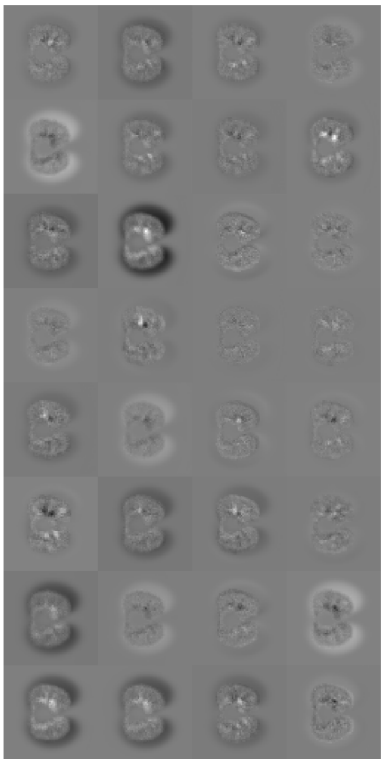


Figure 18: activations showing that cancerous nodule presence detected in some outputs

References

[1] Centers for Disease Control and Prevention, "Lung cancer statistics." https://www.cdc.gov/cancer/lung/statistics/, 2016.

[2] Mayo Clinic Staff, "Tests and diagnosis." http://www.mayoclinic.org/diseases-conditions/lung-cancer/basics/tests-diagnosis/con-20025531, 2015.

[3] Global Resource for Advancing Cancer Education, "Potential advantages, disadvantages and limitations of lung cancer screening." http://cancergrace.org/lung/2016/02/24/gcvl_lu_potential_advantages_disadvantages_limitations_lung_cancer_screening/, 2016.

[4] M. Firmino *et al.*, "Computer-aided detection system for lung cancer in computed tomography scans: Review and future prospects," *BioMedical Engineering OnLine*, vol. 13, p. 41, 2014.

[5] S. Hawkins *et al.*, "Predicting malignant nodules from screening ct scans," *Journal of Thoracic Oncology*, vol. 11, p. 21202128, 2016.

[6] Kaggle, "Data Science Bowl 2017." https://www.kaggle.com/c/data-science-bowl-2017.

[7] LUNA16, "Lung nodule analysis 2016." https://luna16.grand-challenge.org/.

[8] H. Lepor, *Prostatic Diseases*, vol. 2000. W.B. Saunders Company, 2000.

[9] M. S. Al-Tarawneh, "Lung cancer detection using image processing techniques," *Leonardo Electronic Journal of Practices and Technologies*, vol. 20, pp. 147–58, May 2012.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[11] J. d. Wit, "2nd place solution for the 2017 national data-science bowl."

[12] C. Szegedy *et al.*, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.