Lung Cancer Detection and Classification with 3D Convolutional Neural Network (3D-CNN)

Wafaa Alakwaa
Faculty of Computers & Info.
Cairo University, Egypt

Mohammad Nassef
Faculty of Computers & Info.
Cairo University, Egypt

Amr Badr
Faculty of Computers & Info.
Cairo University, Egypt

***Abstract*—This paper demonstrates a computer-aided diagnosis (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 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 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 (CNNs) to ultimately classify the CT scan as positive or negative for lung cancer. The 3D CNNs produced a test set Accuracy of 86.6%. The performance of our CAD system outperforms the current CAD systems in literature which 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.**

***Keywords*—Lung cancer; computed tomography; deep learning; convolutional neural networks; segmentation**

I. 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. 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. 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 [1].

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 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 [2].

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 Fig. 1. Furthermore, a CT scan is filled with noise from surrounding tissues, bone, air, so for the CAD systems search to be efficient, this noise would first have to be preprocessed. Hence our classification pipeline is image preprocessing, nodule candidates detection, malignancy classification.

In this paper, we apply an extensive preprocessing techniques to get the accurate nodules in order to enhance the accuracy of detection of lung cancer. Moreover, we perform an end-to-end training of CNN from scratch in order to realize the full potential of the neural network i.e. to learn discriminative features. Extensive experimental evaluations are performed on a dataset comprising lung nodules from more than 1390 low dose CT scans.

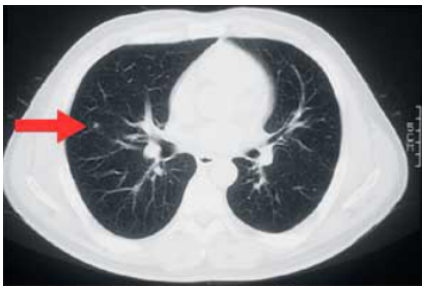


Figure 1: 2D CT scan slice containing a small (5mm) early stage lung cancer nodule.

The paper's arrangement is as follows: Related work is summarized briefly in Section II. Dataset for this paper is described in Section III. The methods for segmentation are presented in section IV. The nodule segmentation is introduced in Section V based on U-Net architecture. Section VI presents 3D Convolutional Neural Network for nodule classification and

patient classification. Our discussion and results are described in details in Section VII. Section VIII concludes the paper.

II. Related Work

Recently, deep artificial neural networks have been applied in many applications in pattern recognition and machine learning, especially, Convolutional neural networks (CNNs) which is one class of models [3]. Another approach of CNNs was applied on ImageNet Classification in 2012 is called an ensemble CNNs which outperformed the best results which were popular in the computer vision community [4]. There has also been popular latest research in the area of medical imaging using deep learning with promising results.

Suk et al. [5] suggested a new latent and shared feature representation of neuro-imaging data of brain using Deep Boltzmann Machine (DBM) for AD/MCI diagnosis. Wu et al. [6] developed deep feature learning for deformable registration of brain MR images to improve image registration by using deep features. Xu et al. [7] presented the effectiveness of using deep neural networks (DNNs) for feature extraction in medical image analysis as a supervised approach. Kumar et al. [8] proposed a CAD system which uses deep features extracted from an autoencoder to classify lung nodules as either malignant or benign on LIDC database. In [9], Yaniv et al. presented a system for medical application of chest pathology detection in x-rays which uses convolutional neural networks that are learned from a non-medical archive. that work showed a combination of deep learning (Decaf) and PiCodes features achieves the best performance. The proposed combination presented the feasibility of detecting pathology in chest x-ray using deep learning approaches based on non-medical learning. The used database was composed of 93 images. They obtained an area under curve (AUC) of 0.93 for Right Pleural Effusion detection, 0.89 for Enlarged heart detection and 0.79 for classification between healthy and abnormal chest x-ray.

In [10], Suna W. et al., implemented three different deep learning algorithms, Convolutional Neural Network (CNN), Deep Belief Networks (DBNs), Stacked Denoising Autoencoder (SDAE), and compared them with the traditional image feature based CAD system. The CNN architecture contains eight layers of convolutional and pooling layers, interchangeably. For the traditional compared to algorithm, there were about 35 extracted texture and morphological features. These features were fed to the kernel based support vector machine (SVM) for training and classification. The resulted accuracy for the CNN approach reached 0.7976 which was little higher than the traditional SVM, with 0.7940. They used the Lung Image Database Consortium and Image Database Resource Initiative (LIDC/IDRI) public databases, with about 1018 lung cases.

In [11], J. Tan et al. designed a framework that detected lung nodules, then reduced the false positive for the detected nodules based on Deep neural network and Convolutional Neural Network. The CNN has four convolutional layers and four pooling layers. The filter was of depth 32 and size 3,5. The used dataset was acquired from the LIDC-IDRI for about 85 patients. The resulted sensitivity was of 0.82. The False positive reduction gotten by DNN was 0.329.

In [12], R. Golan proposed a framework that train the weights of the CNN by a back propagation to detect lung nodules in the CT image sub-volumes. This system achieved sensitivity of 78.9% with 20 false positives, while 71.2% with 10 FPs per scan, on lung nodules that have been annotated by all four radiologists

Convolutional neural networks have achieved better than Deep Belief Networks in current studies on benchmark computer vision datasets. The CNNs have attracted considerable interest in machine learning since they have strong representation ability in learning useful features from input data in recent years.

III. Data

Our primary dataset is the patient lung CT scan dataset from Kaggles Data Science Bowl (DSB) 2017 [13]. The dataset contains labeled data for 1397 patients, which we divide into training set of size 978, and test set of size 419. 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 $\times$ 512 pixels. The slices are provided in DICOM format. Around 70% of the provided labels in the Kaggle dataset are 0, so we used 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 used the patient lung CT scan dataset with labeled nodules from the Lung Nodule Analysis 2016 (LUNA16) Challenge [14] to train a U-Net for lung nodule detection. The LUNA16 dataset contains labeled data for 888 patients, which we divided 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 $\times$ 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.

IV. Methods

Typical CAD systems for lung cancer have the following pipeline: image preprocessing, detection of cancerous nodule candidates, nodule candidate false positive reduction, malignancy prediction for each nodule candidate, and malignancy prediction for overall CT scan [15]. 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.

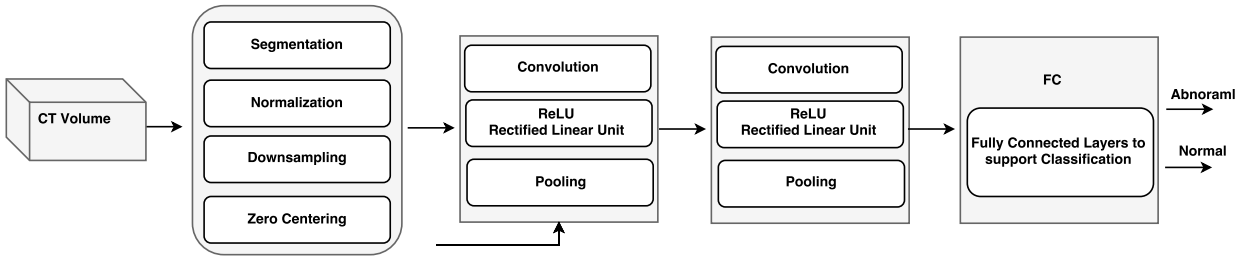


Figure 2: 3D convolutional neural networks architecture.

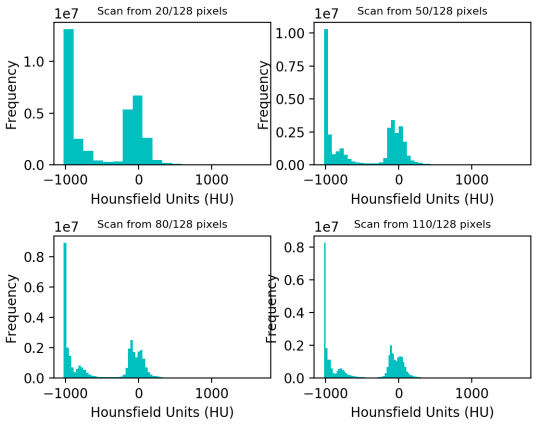
True/False labels for nodule candidates and malignancy labels for nodules are sparse for lung cancer, and may be nonexistent for some other cancers, so CAD systems that rely on such data would not generalize to other cancers. In order to achieve greater computational efficiency and generalizability to other cancers, the proposed CAD system has shorter pipeline and only requires 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 [16]. However, as mentioned above, these systems take as input various labeled data that is not used in this framework. The main goal of the proposed system is to reach close to this performance.

The proposed CAD system starts with preprocessing the 3D CT scans using segmentation, normalization, downsampling, and zero-centering. The initial approach was to simply input the preprocessed 3D CT scans into 3D CNNs, but the results were poor. So an additional preprocessing was performed to input only regions of interests into the 3D CNNs. To identify regions of interest, a U-Net was trained for nodule candidate detection. Then input regions around nodule candidates detected by the U-Net was fed into 3D CNNs to ultimately classify the CT scans as positive or negative for lung cancer. The overall architecture is shown in Fig. 2, all details of layers will be described in the next sections.

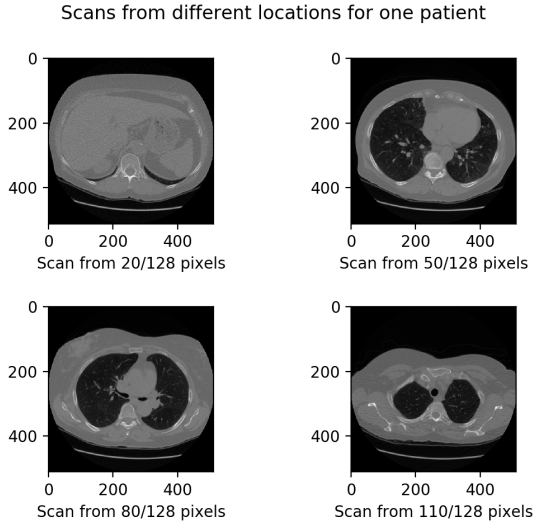
A. Proprocessing and Segmentation

For each patient, pixel values was first converted in each image to Hounsfield units (HU), a measurement of radiodensity, and 2D slices are stacked into a single 3D image. Because tumors form on lung tissue, segmentation is used to mask out the bone, outside air, and other substances that would make data noisy, and leave only lung tissue information for the classifier. A number of segmentation approaches were tried, including thresholding, clustering (Kmeans 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.

After segmentation, the 3D image is normalized by applying the linear scaling to squeezed all pixels of the original unsegmented image to values between 0 and 1. Spline interpolation downsamples each 3D image by a scale of 0.5 in each of the three dimensions. Finally, zero-centering is performed on data by subtracting the mean of all the images from the training set.



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



(b) Corresponding 2D axial slices.

Figure 3: 3a Histogram of HU values at 3b corresponding axial slices for sample patient 3D image at various axial.

1) Thresholding: Typical radiodensities of various parts of a CT scan are shown in Table I. 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 pixels that are close to -1000 or above -320 are masked

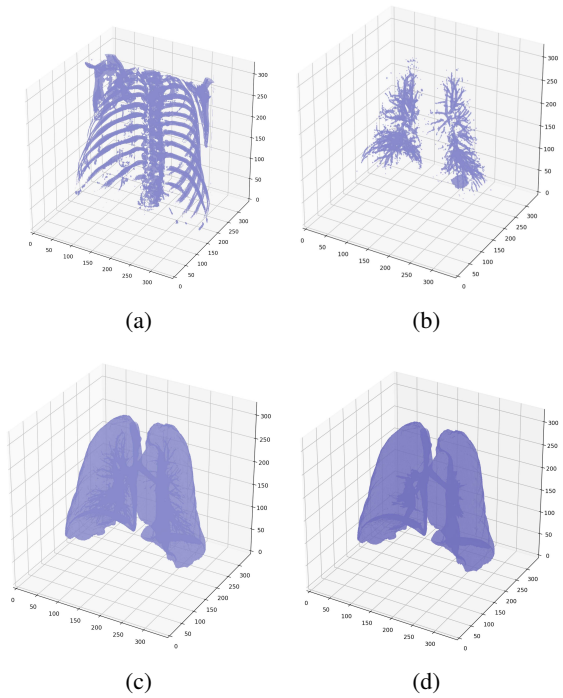


Figure 4: (4a) Sample patient 3D image with pixels values greater than 400 HU reveals the bone segment, (4b) Sample patient bronchioles within lung, (4c) Sample patient initial mask with no air, and (4d) Sample patient final mask in which bronchioles are included.

out 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 Fig. 3. Pixels thresholded at 400 HU are shown in Fig. 3a, and the mask is shown in Fig. 3b. However, to account for the possibility that some cancerous growth could occur within the bronchioles (air pathways) inside the lung, which are shown in Fig. 4c, this air is included to create the finalized mask as shown in Fig. 4d.

Table I: Typical Radiodensities in HU of Various Substances in a CT Scan

Substance	Radiodensity (HU)
Air	-1000
Lung tissue	-500
Water and Blood	0
Bone	700

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. [17]. An original 2D CT slice of a sample patient is given in Fig. 5a. The resulting 2D slice of the lung segmentation mask created by thresholding is shown in Fig. 5b, and the resulting 2D slice of the lung segmentation mask created by Watershed is shown in Fig. 5d. Qualitatively, this produces a much better segmentation than thresholding. Missing voxels (black dots in Fig. 5b) are largely re-included. However, this is much less efficient than basic thresholding, so due to time limitations, it was not possible to preprocess all CT scans using Watershed, so thresholding is used instead.

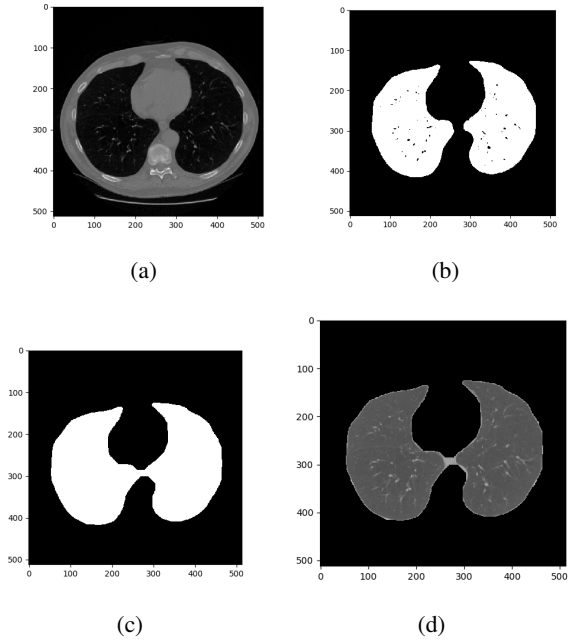


Figure 5: (5a) Original 2D slice of sample patient, (5b) Lung segmentation mask by thresholding of sample patient, (5c) Final watershed segmentation mask of sample patient, and (5d) Final watershed lung segmentation of sample patient.

V. U-Net for Nodule Detection

Feeding the entire segmented lungs into malignancy classifiers made results very poor. It was likely the case that the entire image was too large search space. Thus feeding smaller regions of interest instead of the entire segmented 3D image is more convenient. This was achieved by selecting small boxes containing top cancerous nodule candidates. To find these top nodule candidates, a modified version of the U-Net was trained as described in Ronneberger et al. on LUNA16 data [18]. U-Net is a 2D CNN architecture that is popular for biomedical image segmentation. A stripped-down version of the U-Net is designed to limit memory expense. A visualization of the U-Net architecture is included in Fig. 6 and is described in detail in Table II. During training, the 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 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 Fig. 7a, 7b, and 7c, respectively.

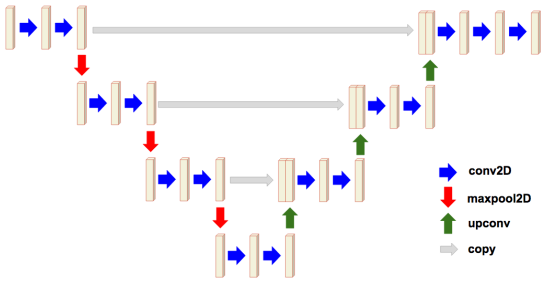
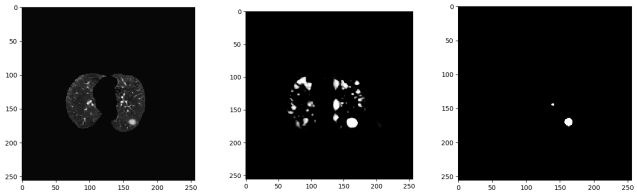


Figure 6: Modified U-Net architecture.



(a) (b) (c)

Figure 7: (7a) 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, (7b) U-Net predicted output from LUNA16 validation set, (7c) U-Net sample labels mask from LUNA16 validation set showing ground truth nodule location.

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.

VI. Malignancy 3D CNN Classifiers

Once the U-Net was trained on the LUNA16 data, it is ran on 2D slices of Kaggle data and stacked the 2D slices back to generate nodule candidates [1]. Ideally the output of U-Net would give the exact locations of all the nodules, and it would be able to declare 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 Fig. 7c, 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.

Because U-Net generates more suspicious regions than actual nodules, the top 8 nodule candidates are located ($32 \times 32 \times 32$ $volumes$) 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 were not permitted to overlap with each other. Then these sectors are combined into a single $64 \times 64 \times 64$ image, which will serve as the input to classifiers, which assign a label to the image (cancer or not cancer).

[1]Preprocessing and reading of LUNA16 data code based on https://www.kaggle.com/arnavkj95/ candidate-generation-and-luna16-preprocessing

Table II: 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)

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

A 3D CNN is used as linear classifier. It 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 network is shrunk to prevent parameter overload for the relatively small Kaggle dataset. The 3D CNN architecture is described in detail in Table III.

Convolutional neural network consists of some number of convolutional layers, followed by one or more fully connected layers and finally an output layer. An example of this architecture is illustrated in Fig. 8.

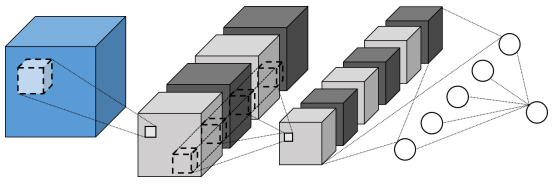


Figure 8: An example architecture of a 3D convolutional neural network used here. On the left is the input 3D volume, followed by two convolutional layers, a fully connected layers and an output layer. In the convolutional layers, each filter (or channel) is represented by a volume.

Formally, we denote the input to layer m of the network by $I^{(m)}$. The input to a 3D convolutional layer m of a neural network is a $n_1^{(m-1)} \times n_2^{(m-1)} \times n_3^{(m-1)}$ 3D object with $n_c^{(m-1)}$

so $I^{(m-1)} \in (\mathbb{R}^{n_1^{(m-1)} \times n_2^{(m-1)} \times n_3^{(m-1)}}$ and its elements are denoted by $I_{i,j,k}^{(m,\ell)}$ where i, j, and k index the 3D volume and ℓ selects the channel. The output of a convolutional layer m is defined by its dimensions, i.e., $n_1^{(m)} \times n_2^{(m)} \times n_3^{(m)}$ as well as the number of filters or channels it produces $n_c^{(m)}$. The output of layer m is a convolution of its input with a filter and is computed as

$$I_{i,j,k}^{(m,\ell)} = f_{tanh}(b^{(m,\ell)} + \sum_{\tilde{i},\tilde{j},\tilde{k},\tilde{\ell}} I_{\tilde{i},\tilde{j},\tilde{k}}^{(m-1,\tilde{\ell})} W_{i-\tilde{i},j-\tilde{j},k-\tilde{k},\tilde{\ell}}^{(m,\ell)}) \quad (1)$$

where, $W^{(m,\ell)}$ and $b^{(m,\ell)}$ are the parameters which define the ℓth filter in layer m The locations where the filters are evaluated (i.e., the values of i, j, k for which $I_{i,j,k}^{(m,\ell)}$ is computed) and the size of the filters (i.e., the values of $W^{(m,\ell)}$) which are non-zero) are parameters of the network architecture. Finally, we use a hyperbolic tangent activation function with $f_{tanh}(a) = tanh(a)$.

Convolutional layers preserve the spatial structure of the inputs, and as more layers are used, build up more and more complex representations of the input. The output of the convolutional layers is then used as input to a fully connected network layer. To do this, the spatial and channel structure is ignored and the output of the convolutional layer is treated as a single vector. The output of a fully connected is a 1D vector $I^{(m)}$ whose dimension is a parameter of the network architecture. The output of neuron i in layer m is given by

$$I_i^{(m)} = f_{ReLU}\left(b^{(m,i)} + \sum_j I_j^{(m-1)} W_j^{(m,i)}\right) \quad (2)$$

where, $W^{(m,i)}$ and $b^{(m,i)}$ are the parameters of neuron i in layer m and the sum over j is a sum over all dimensions of the input. The activation function $f_{ReLU}(.)$ here is chosen to be a Rectified Linear Unit (ReLU) with $f_{ReLU}(a) = max(0, a)$. This activation function has been widely used in a number of domains [19], [20] and is believed to be particularly helpful in classification tasks as the sparsity it induces in the outputs helps create separation between classes during learning.

The last fully connected layer is used as input to the output layer. The structure and form of the output layer depends on the particular task. Here we consider two different types of output functions. In classification problems with K classes, a common output function is the softmax function:

$$f_i = \frac{exp(I_i^{(o)})}{\sum_j exp(I_j^{(o)})} \quad (3)$$

$$I_i^{(o)} = b^{(o,i)} + \sum_{k=1}^{K} W_k^{(o,i)} I_k^{(N)} \quad (4)$$

where, N is the index of the last fully connected layer, $b^{(o,i)}$ and $W^{(o,i)}$ are the parameters of the ith output unit and $f_i \in [0, 1]$ is the output for class i which can be interpreted as the probability of that class given the inputs. We also consider a variation on the logistic output function:

$$f = a + (b - a)\left(1 + exp(b^{(o)} + \sum_j W_j^{(o)} I_j^{(N)}\right)^{-1} \quad (5)$$

which provides a continuous output f which is restricted to lie in the range (a, b) with parameters $b^{(o)}$ and $W^{(o)}$. We call this the *scaled logistic* output function. We note that when considering a ranking-type multi-class classification problem like predicting the malignancy level this output function might be expected to perform better.

Table III: 3D CNN Architecture (Dropout with 0.2, Adam Optimizer, Learning Rate = 0.0001)

Layer	Params	Activation	Output
Input			28 × 28 × 28
Conv1	5 × 5× 5	ReLu	28 × 28 × 28× 7
Max Pool	1× 1× 1, stride 2× 2× 4		14 × 14 × 7 × 7
Conv2	5 × 5× 3	ReLu	14 × 14 × 7 × 17
Max Pool	2× 2 × 2, stride 1× 1 × 0		6 × 6 × 3 × 17
Dense			256
Dense			2

A. Training

Given a collection of data and a network architecture, the main goal is to fit the parameters of the network to that data. To do this we will define an objective function and use gradient based optimization to search for the network parameters which minimize the objective function. Let $D = n_i, y_i{}_{i=1}^{D}$ be the set of D (potentially augmented) training examples where n is an input (a portion of a CT scan) and y is the output (the malignancy level or a binary class indicating benign or malignant) and Θ denote the collection of all weights W and biases b for all layers of the network. The objective function has the form

$$E(\Theta) = \sum_{i=1}^{D} L(y_i, f(n_i, \Theta)) + \lambda E_{prior}(\Theta) \quad (6)$$

where, $f(n_i, \Theta))$ is the output of the network evaluated on input n with parameters $\Theta, L(y_i, f(n_i, \Theta))$ is a loss function which penalizes differences between the desired output of the network y and the prediction of the network $\hat{y}$. The function $E_{prior}(\Theta) = \|W\|^2$ is a weight decay prior which helps prevent over-fitting by penalizing the norm of the weights and λ controls the strength of the prior.

We consider two different objective functions in this paper depending on the choice of output function. For the softmax output function we use the standard cross-entropy loss function $L(y_i, \hat{y}) = -\sum_{k=1}^{K} y_k log(\hat{y}_k)$ where y is assumed to be a binary indicator vector and $\hat{y}$ is assumed to be a vector of probabilities for each of the K classes. A limitation of a cross-entropy loss is that all class errors are considered equal, hence mislabeling a malignancy level 1 as a level 2 is considered just as bad as mislabeling it a 5. This is clearly problematic, hence for the scaled logistic function we use the squared error loss function to capture this. Formally, $L(y_i, \hat{y}) = (y - \hat{y})^2$ where we assume y and $\hat{y}$ to be real valued.

Given the objective function $E(\Theta)$, the parameters Θ are learned using stochastic gradient descent (SGD) [21]. SGD

operates by randomly selecting a subset of training examples and updating the values of the parameters using the gradient of the objective function evaluated on the selected examples. To accelerate progress and reduce noise due to the random sampling of training examples we use a variant of SGD with momentum [22]. Specifically, at iteration t, the parameters are updated as

$$\Theta_{t+1} = \Theta_t + \triangle\Theta_{t+1} \tag{7}$$

$$\triangle\Theta_{t+1} = \rho\triangle\Theta_t - \epsilon\nabla E_t(\Theta_t) \tag{8}$$

where, $\rho = 0.9$ is the momentum parameter, $\triangle\Theta_{t+1}$ is the momentum vector, ϵ_t is the learning rate and $\nabla E_t(\Theta_t)$ is the gradient of the objective function evaluated using only the training examples selected at iteration t. At iteration 0, all biases are set to 0 and the values of the filters and weights are initialized by uniformly sampling from the interval $[-\sqrt{\frac{6}{f_{an\_in}+f_{an\_out}}}, \sqrt{\frac{6}{f_{an\_in}+f_{an\_out}}}]$ as suggested by [23] where f_{an_in} and f_{an_out} respectively denote the number of nodes in the previous hidden layer and in the current layer. Given this initialization and setting $\epsilon_t = 0.01$, SGD is running for 2000 epochs, during which ϵ_t is decreased by 10% every 25 epochs to ensure convergence.

VII. Simulation Results

The experiments are conducted using DSB dataset. In this dataset, a thousand low-dose CT images from high-risk patients in DICOM format is given. The DSB database consists of 1397 CT scans and 248580 slices. Each scan contains a series with multiple axial slices of the chest cavity. Each scan has a variable number of 2D slices (Fig. 9), which can vary based on the machine taking the scan and patient. The DICOM files have a header that contains the necessary information about the patient id, as well as scan parameters such as the slice thickness. It is publicly available in the Kaggle [13]. Dicom is the de-facto file standard in medical imaging. This pixel size/coarseness of the scan differs from scan to scan (e.g. the distance between slices may differ), which can hurt performance of our model.

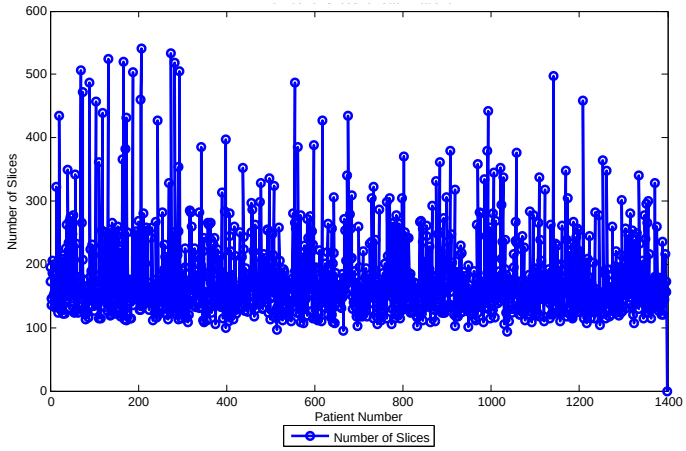


Figure 9: Number of slices per patient in data science bowl dataset.

The experiments are implemented on computer with CPU i7, 2.6 GHz, 16 RAM, Matlab 2013b, R-Studio, and Python. Initially speaking, the nodules in DSB dataset are detected and segmented using thresholding and U-Net Convolutional Neural Network. The diameters of the nodules range from 3 to 30 mm. Each slice has 512×512 pixels and 4096 gray level values in Hounsfield Unit (HU), which is a measure of radiodensity.

In the screening setting, one of the most difficult decisions is whether CT or another investigation is needed before the next annual low-dose CT study. Current clinical guidelines are complex and vary according to the size and appearance of the nodule. The majority of nodules were solid in appearance. For pulmonary nodule detection using CT imaging, CNNs have recently been used as a feature extractor within a larger CAD system.

For simplicity in training and testing we selected the ratings of a single radiologist. All experiments were done using 50% training set, 20% validation set and 30% testing set. To evaluate the results we considered a variety of testing metrics. The accuracy metric is the used metric in our evaluations. In our first set of experiments we considered a range of CNN architectures for the binary classification task. Early experimentation suggested that the number of filters and neurons per layer were less significant than the number of layers. Thus, to simplify analysis the first convolutional layer used seven filters with size $5\times5\times5$, the second convolutional layer used 17 filters with $5\times5\times3$ and all fully connected layers used 256 neurons. These were found to generally perform well and we considered the impact of one or two convolutional layers followed by one or two fully connected layers. The networks were trained as described above and the results of these experiments can be found in Table I. Our results suggest that two convolutional layers followed by a single hidden layer is one of the optimal network architecture for this dataset. The average error for training is described in Fig. 10.

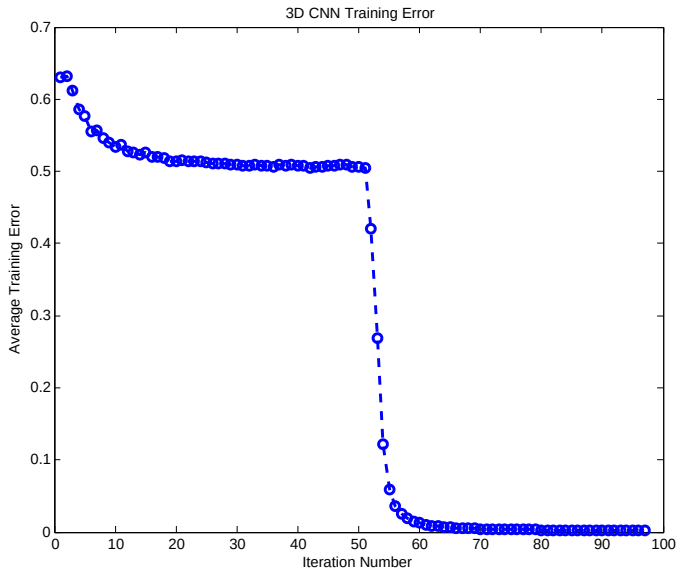


Figure 10: Average training error in 3D CNN.

Another important parameter in the training of neural networks is the number of observations that are sampled

at each iteration, the size of the so-called minibatch. The use of minibatches is often driven in part by computational considerations but can impact the ability of SGD to find a good solution. Indeed, we found that choosing the proper minibatch size was critical for learning to be effective. We tried minibatches of size 1, 10, 50 and 100. While the nature of SGD suggests that larger batch sizes should produce better gradient estimates and therefor work better, our results here show that the opposite is true. Smaller batch sizes, even as small as 1, produce the best results. We suspect that the added noise of smaller batch sizes allows SGD to better escape poor local optima and thus perform better overall.

The recognition results are shown by confusion matrix achieved on the DSB dataset with 3D CNN as shown in Table IV. As shown from the Table IV, Accuracy of model is 86.6%, Mis-classification rate is 13.4%, False positive rate is 11.9%, and False Negative is 14.7%. Almost all patients are classified correctly. Additionally, there is an enhancement on accuracy due to efficient U-Net architecture and segmentation.

Table IV: Confusion Matrix of 3D CNN using 30% Testing

	Predicted	
	Abnormal	Normal
Actual		
Abnormal	**0.853**	0.147
Normal	0.119	**0.881**

VIII. Conclusion

In this paper we developed a deep convolutional neural network (CNN) architecture to detect nodules in patients of lung cancer and detect the interest points using U-Net architecture. This step is a preprocessing step for 3D CNN. The deep 3D CNN models performed the best on the test set. While we 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 layer is a preprocessing layer for segmentation using different techniques. Threshold, Watershed, and U-Net are used to identify the nodules of patients.

The network can be trained end-to-end from raw image patches. Its main requirement is the availability of training database, but otherwise no assumptions are made about the objects of interest or underlying image modality.

In the future, 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 hyper parameter tuning. Also, we saved our model parameters at best accuracy, 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.

References

[1] W.-J. Choi and T.-S. Choi, "Automated pulmonary nodule detection system in computed tomography images: A hierarchical block classification approach," *Entropy*, vol. 15, no. 2, pp. 507–523, 2013.

[2] A. Chon, N. Balachandar, and P. Lu, "Deep convolutional neural networks for lung cancer detection," tech. rep., Stanford University, 2017.

[3] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision.," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 253–256, IEEE, 2010.

[4] K. Alex, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25 (NIPS 2012)* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, 2012.

[5] H. Suk, S. Lee, and D. Shen, "Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis," *NeuroImage*, vol. 101, pp. 569–582, 2014.

[6] G. Wu, M. Kim, Q. Wang, Y. Gao, S. Liao, and D. Shen, "Unsupervised deep feature learning for deformable registration of mr brain images.," *Medical Image Computing and Computer-Assisted Intervention*, vol. 16, no. Pt 2, pp. 649–656, 2013.

[7] Y. Xu, T. Mo, Q. Feng, P. Zhong, M. Lai, and E. I. Chang, "Deep learning of feature representation with multiple instance learning for medical image analysis," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pp. 1626–1630, 2014.

[8] D. Kumar, A. Wong, and D. A. Clausi, "Lung nodule classification using deep features in ct images," in *2015 12th Conference on Computer and Robot Vision*, pp. 133–138, June 2015.

[9] Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, and H. Greenspan, "Chest pathology detection using deep learning with non-medical training," *Proceedings - International Symposium on Biomedical Imaging*, vol. 2015-July, pp. 294–297, 2015.

[10] W. Sun, B. Zheng, and W. Qian, "Computer aided lung cancer diagnosis with deep learning algorithms," in *SPIE Medical Imaging*, vol. 9785, pp. 97850Z–97850Z, International Society for Optics and Photonics, 2016.

[11] J. Tan, Y. Huo, Z. Liang, and L. Li, "A comparison study on the effect of false positive reduction in deep learning based detection for juxtapleural lung nodules: Cnn vs dnn," in *Proceedings of the Symposium on Modeling and Simulation in Medicine*, MSM '17, (San Diego, CA, USA), pp. 8:1–8:8, Society for Computer Simulation International, 2017.

[12] R. Golan, C. Jacob, and J. Denzinger, "Lung nodule detection in ct images using deep convolutional neural networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 243–250, July 2016.

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

[14] LUNA16, "Lung nodule analysis 2016." https://luna16.grand-challenge.org/, 2017.

[15] M. Firmino, A. Morais, R. Mendoa, M. Dantas, H. Hekis, and R. Valentim, "Computer-aided detection system for lung cancer in computed tomography scans: Review and future prospects," *BioMedical Engineering OnLine*, vol. 13, p. 41, 2014.

[16] S. Hawkins, H. Wang, Y. Liu, A. Garcia, O. Stringfield, H. Krewer, Q. Li, D. Cherezov, R. A. Gatenby, Y. Balagurunathan, D. Goldgof, M. B. Schabath, L. Hall, and R. J. Gillies, "Predicting malignant nodules from screening ct scans," *Journal of Thoracic Oncology*, vol. 11, no. 12, pp. 2120–2128, 2016.

[17] M. S. AL-TARAWNEH, "Lung cancer detection using image processing techniques," *Leonardo Electronic Journal of Practices and Technologies*, pp. 147–158, June 2012.

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

[19] M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, and G. E. Hinton, "On rectified linear units for speech processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3517–3521, May 2013.

[20] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, 2013.

[21] L. Bottou, *Large-Scale Machine Learning with Stochastic Gradient Descent*, pp. 177–186. Augest 2010.

[22] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on International Conference on Machine Learning*, ICML'13, pp. 1139–1147, JMLR.org, 2013.

[23] H. Han, L. Li, H. Wang, H. Zhang, W. Moore, and Z. Liang, "A novel computer-aided detection system for pulmonary nodule identification in ct images," in *Proceedings of SPIE Medical Imaging Conferenc*, vol. 9035, 2014.