lucanode: automatic lung cancer nodule detection

Octavi Font

June, 2018

Contents

Abstract

This Master's final thesis details the implementation of a computer-aided detection (CADe) system for lung cancer nodule detection (hence *lucanode*). Its aim is to provide assistance to radiologists for early diagnosis of lung cancer by detecting round abnormalities in the lung (the nodules). The thesis contextualizes the impact that such a system could have in the prognosis for this disease, analyzes the current state of the art and then dwells on the implementation of the system, trained on the publically available LUNA dataset [1].

lucanode is divided into a 4 step pipeline: scan preprocessing, lung segmentation, nodule segmentation and false positive reduction. For each step, there are multiple attempted approaches, which have been quantified and evaluated against one another. Finally, the system as a whole is compared against the state of the art following the approach established in the LUNA grand challenge.

On the discussion we comment on the possible improvements for each step of the pipeline, as well as the steps that would be required to integrate it into a clinical workflow. We conclude the thesis by pointing to future lines of research to expand on the topic.

Keywords: lung cancer; lung nodules; image segmentation; image recognition; deep learning; CADe

Acknowledgments

And this is where the acknowledgments section goes. I should write something nice.

Chapter 1

Introduction

1.1 Clinical context

1.1.1 Lung cancer

Mention 2 for the cancer statistics and initial paragraphs.

Lung cancer is the most deadly cancer in both men and women worldwide1. It is the second most common cancer in both men and women, trailing prostate cancer for men, and breast cancer for women1. In the Netherlands, more than 10,000 people die of lung cancer every year2. The Dutch Cancer Society estimates that lung cancer will account for 25% of all cancer-related deaths in 20152. In the United States, the American Cancer Society estimates a similar percentage for 2015: 27%1. Figure 1.1 shows the estimated number of new cancer cases and deaths in men and women in the United States in 2015. The 5-year relative survival rate for all stages combined is only 17%1. This low rate can be largely attributed to the fact that at present, only 15% of all lung cancers are diagnosed in an early stage1. The reason for this is that symptoms usually do not occur until the cancer is in an advanced stage. If lung cancer is detected in an early stage when the disease is still localized and more curative treatment options are available, the 5-year relative survival rate is 54%1. Therefore, early detection of lung cancer is of major importance to reduce lung cancer mortality. By far the most important risk factor for lung cancer is tobacco use. The risk in- creases both with quantity and duration of smoking. An estimated 87% of all lung cancer deaths are caused by cigarette smoking3. Therefore, complete banning of to- bacco use would be the best recipe to reduce lung cancer mortality. Although the risks of smoking are well-known, it remains a major cause of the increasing global burden of cancer. Other risk factors for lung cancer are exposure to asbestos, expo- sure to radon, and air pollution.

1.1.2 Computed Tomography

Basically talk about the technique and how it has been changing diagnosis recently.

Table 1.1: Houndsfield Units range of different body tissues and fluids.

Substance	HU
Air	-1000
Fat	-120 to -90
Soft Tissue, Contrast	+100 to +300
Water	0
Blood	+13 to +50
Lung parenchyma	-700 to -600
Muscle	+35 to +55

Substance	HU
Cancellouus bone	+700
Cortical bone	+3000

1.1.3 Lung cancer screening with CT

Talk about the NLST study and NELSON. Reduction of 20% in mortality if screened, so early detection is important to improve the outcomes.

1.1.4 Lung nodules

Explain nodule types. Solid and subsolid.

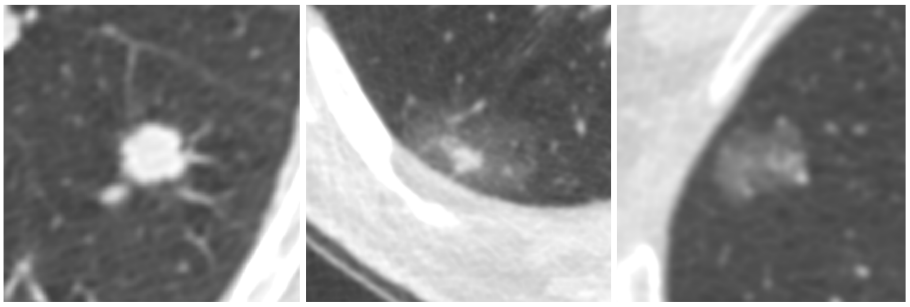

Figure 1.1: Different types of lung nodules. *left*: solid nodule, *middle*: part-solid nodule, *right*: non-solid nodule. Image from 3

1.2 Lung nodule CAD

1.2.1 Objectives

Explain why it would be useful (reduce workload, reduce intra-variability for radiologists). Also cheaper. Explain why historically they haven't worked (mention main problems a system like this faces) and why I think now is a good time to create a system that improves upon the existing state of the art.

1.2.2 Shortcomings

Explain what are the main things that fail

1.2.3 Pipeline

Even though there has been much effort in developing new techniques to improve the performance of CAD systems due to the availability of annotated datasets and challenges (NLST, ISBI, LUNA, DSB2017), the published systems tend to be brittle and very much focused on demonstrating good results on those specific challenges but useless as an integrated system. Also, what is not available tends to be proprietary systems, which might be good, but who knows really.

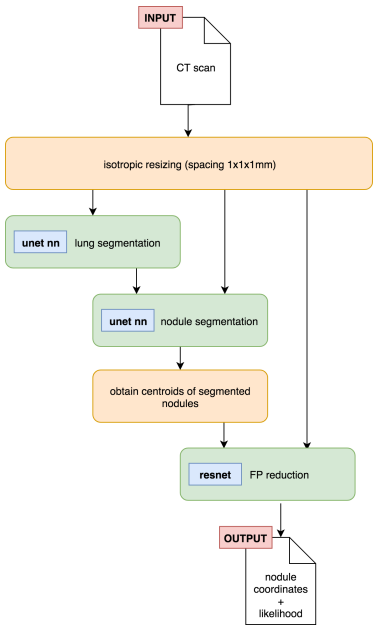

Figure 1.2: the lucanode pipeline

One of the improvements that I wanted to bring to the state of the art was to prepare a system which could be easily deployed in a real system. To achieve this I had to automate the scan preprocessing and prepare a full pipeline that could later on be integrated in a real system. In fact, this integration with a system has been performed by Albert , that has a queue which picks up the scan and returns a CSV with the annotated nodules to check for.

What do we need to do:

- preprocessing: basically reading the ct scan in a SimpleITK compatible format and rescale it to 1x1x1mm
- lung segmentation: using the input from before, segment the lung and get a segmentation mask
- nodule segmentation: using the isotropic scan and the lung segmentation, compute the segmentation mask for the nodules, then measure the centroids

of the labels and convert those coordinates to real world coordinates.
- fp reduction: Using the scan and the centroids in the previous step, apply the nodule classifier and retrieve a probability for each of the nodules. Once we have this probability per candidate, discard any that are below a required threshold. If instead of using a probability threshold what we are interested is in a false positive rate, use the numbers in the evaluation phase to basically determine how the probability maps to a specific FPR, and adjust the output candidates with that.

To run this basically I've packaed everything in a conda environment. This has allowed me to list all the necessary packages and provide an easy way to create environments with all the necessary dependencies, even stuff like CUDA libraries, which is not native python, can be easily installed using conda. This also makes it very easy to then create a Docker image that has all the necessary packages to run this stuff.

What else? Well, the docker image contains the weights of the different neural networks. I've basically just included the best network for each of the steps, based on the evaluation of the results. Both the code, dependencies and weights is included in a Docker image, which can also have GPU support (very much recommended) by using nvidia-docker.

Once that is built, we have a ready to go image, which only needs to mount 2 volumes (folders) for the input image and the output result. Then it's just a matter of running a command and all of this code can be easily run. Apart from the ease in reproducibility (not only the final script can be executed, but everything else, such as evaluation scripts and the like), we gain a very convenient way to distribute the results and an even better way to test our system in other datasets with minimum hassle, since the whole pipeline has been integrated.

Currently on an i7 7700, 32GB of RAM, GTX 1080Ti, evaluating a scan from start to finish requires around 2mins of processing time.

1.2.4 Metrics

Small section to introduce the metrics I'll use and what are they used for and what drawbacks they have:

```
- DICE
- FROC
- Average FROC
- AUC
- TP, FP, sensitivity and F1
```

1.2.5 Lung segmentation

I might just put this after nodule segmentation and false positive reduction, since it basically just an addendum on nodule segmentation that needs to be done for the pipeline to work in an end to end fashin. Interestingly, this chapter could serve to demonstrate the transferability of deep learning techniques to other domains, which is not a bad thing. Essentially the network and everything is exactly the same thing as the nodule segmentation, but using the lung masks as ground truth, instead of nodule masks, so the problem is actually simpler.

Not much really. Basically the idea is that, if the previous network works well for something as complicated as segmenting nodules, segmenting the lungs themselves should be easier, but basically the same concepts should apply.

1.2.6 Nodule detection

I could say that based on the work I did in the LUNA challenge chapter, best approach right now seems UNET based. Explain again that for this part of the system what we are interested in is basically something with very high sensitivity. And finally I guess say that I went for a 2D network cause the images are big, it is a very deep network, and I wanted to avoid as much technical trouble as possible, especially since it was a first for me.

1.2.7 FP reduction

Similarly to an object detection problem (4), we've divided our pipeline in two phases: candidate proposal and false positive reduction. As we have seen in the previous chapter, our UNET-based proposal network primed sensitivity above all else, but now we need a classifier with high precision so that the signal-to-noise ratio of the system will be high enough to prove useful to a radiologist.

One of the main benefits of performing a previous step to detect candidates is the fact that the search space is reduced and that makes it computationally feasible to run image recognition algorithms with high computational costs within a reasonable timeframe.

In this chapter we'll cover two different approaches to false positive reduction. The first one will be a classifier trained on features manually extracted from the previous segmentation phase of the pipeline. The second one is based on a volumetric ResNet (5). The original 2D version of this deep neural network (6) achieved a deeper architecture bypassing the vanishing/exploding gradients problem (7, 8) by using a combination of normalization techniques (9, 10, 11) and the use of residuals.

1.3 State of the art

1.3.1 The LUNA grand challenge

This chapter will serve as an introduction to what is the LUNA grand challenge, its dataset, competition tracks and metrics. After that is out of the way, I'll go over the current top 20 and do a survey of the different techniques that compound the state of the art for this kind of problem. This will serve as an introduction to what I amb about to do.

Basically talk about the technique and how it has been changing diagnosis recently. This could be a copy pasta of 1 and explain a bit on how they've reworked on the LIDC dataset to prepare the data, what it does and what is missing (malignancy!), which is actually available in LIDC.

What is this dataset and why is it useful to evaluate CAD systems

Talk about the tracks and metrics. Again, this appears in 1, so I don't know how much I want to add

Interesting to go over the top 20 of LUNA as it stands right now. Thankfully most of the systems are closed so I don't have to explain them, but for the open ones, it would be good to go over the methods they present, and basically argument why I chose what I did Talk about the top 20. Basically put a table with the methods, describe them slightly. Then divide method by groups and expand more on that.

1.3.2 Nodule detection track

Review of the top20. Cover here any deep learning content I might have to.

1.3.3 FP reduction

Review of the top20 (again paper and all). Basically cover here any deep learning content I might have to.

1.4 Outline

Talk about the chapters, and how the work is organized.

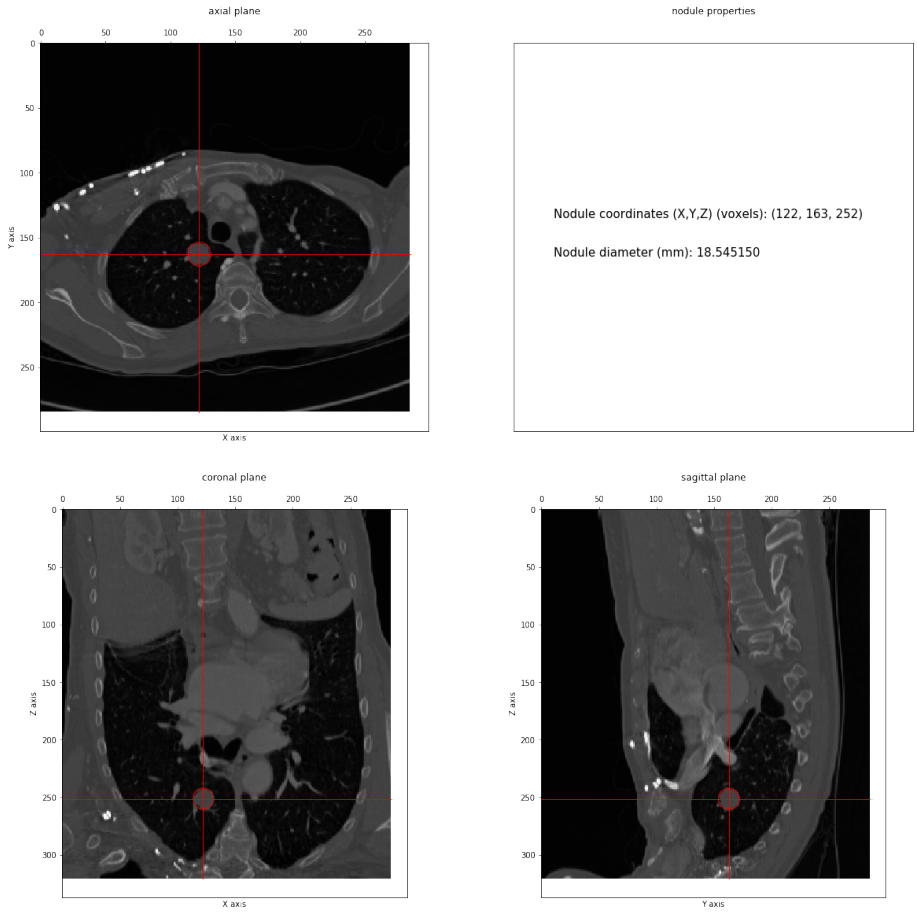

Figure 1.3: An annotated lung nodule in the LUNA dataset.

Chapter 2

Methods

2.1 Lung segmentation

To perform the automated lung segmentation of a CT scan we will use the same deep learning U-Net architecture as in the nodule segmentation. The U-Net will use a batch normalization and ReLU prior to any convolutional layer, as suggested in [5]. The training will be performed over 40 epochs. Subsets 0 to 7 of the LUNA dataset will be used for training, subset 8 for validation and subset 9 for testing. The learning rate is set to *1e-3*, with Adam [12] as our optimization algorithm, which will adaptatively adjust the learning rate. The batch size is 5 and the weights randomly initialized. Hardware wise, we employed an Intel i7 7700, 32GB of RAM and a Nvidia 1080Ti GPU. The network was implemented with Keras [13], using Tensorflow [14]as its backend. The inverse of the Dice coefficient was used as loss function:

$$loss = 1 - DSC = 1 - \frac{2|X \cap Y|}{|X| + |Y|}$$

As part of our preprocessing, the scans will be isotropically resampled to a voxel size of 1x1x1mm. To train this network there is no image augmentation process nor any image filtering applied beforehand. The input passed to the network consists of an axial slice of a CT scan sized 400x400. The ground truth corresponds to the corresponding binary mask slice. They've also been automatically segmented using [15], which combines both a fast pass applying thresholding and then a second finer phase that corrects erroneous areas with applying state of the art methods. The masks are also provided as part of the challenge.

To evaluate the results of our resulting network, we will calculate the Dice coefficient of each slice and then average it accross the whole scan.

2.2 Nodule detection

The basic setup for the nodule detection is the same segmentation network used on the lung segmentation. That is, a U-Net with batch normalization and ReLU, trained over 40 epochs. The dataset is split the same way: subsets 0 to 7 for training, subset 8 for validation and subset 9 for testing. We're also using a learning rate of *1e-3*, Adam, and the same hardware (Intel i7 7700, 32GB of RAM and a Nvidia 1080Ti GPU). This is where the similarities with the previous section end, as the preprocessing steps and the network variations are much different to train the same network to perform nodule segmentation.

The LUNA dataset only has annotations of the centroid and diameter of the annotated nodules, so we had to manually create segmentation masks. For this, we created spherical masks using the annotated diameter on the corresponding centroid. This masks are quite accurate on smaller nodules, but not so much

when the diameter increases (>15mm). This spherical masks have a voxel size of 1x1x1 and a resolution of 400x400 in their axial plane.

As input, we used the axial slices of the CT scans, clipped with their matching lung mask. Values outside the mask were set to -4000HU, which is below the values for air. This artificially low number was a way to tell the network that areas outside the lung were not of our interest. Also, the same lung mask is used to clip the nodule masks, as nodules around the parenchyma could appear otherwise outside the lung, which would confuse the network since it has been clipped.

Only slices with visible nodules were used to train this network. Slices without abnormalities were discarded. This was done to correct the class imbalance that we were otherwise facing. At the slice level we used the Dice coefficient to compute its score, but evaluating a scan requires to:

1. Apply the segmentation network over the whole scan
2. Label the predicted mask using connected components [16, 17, 18]
3. Extract the centroid of said labels
4. Convert the coordinates of the centroid to the real world coordinates of the original scan
5. Check whether the euclidean distance between a candidate and any of the scan annotations is within its radius. If it is, count that candidate as a True Positive. Otherwise it will be a False Positive.

The evaluation of the system will report two metrics: sensitivity and average false positives per scan. Our main goal is to achieve the highest possible sensitivity, but reducing false positives will simplify the task of our false positive reduction module, so it is worth keeping track of its score.

There is also a set of variations in the preprocessing that we've applied incrementally so that we could study their invividual impact in the performance analysis. A different network has been trained from scratch for each of those variations, using both a binary cross entropy and Dice as loss functions. The variations are:

- **normalization:** Train the network with and without the use of batch normalization in its convolutional layers.
- **augmentation:** Enable the use of randomized image augmentation. Full description of the parameters in Table 2.1.
- **3ch depth:** Use the 3 color channels of an image to pass the current slice along its two contiguous slices.
- **laplacian filter:** Apply a laplacian filter on the slice to increase its contrast.

Table 2.1: Range of transformations randomly applied to the axial slices used in the nodule segmentation training.

transformation	range
rotation	[-10º, +10º]

transformation	range
shearing	[-20%, +20%]
scaling	[-20%, +20%]
flip vertically	[True, False]
flip horizontally	[True, False]

Apart from studying the impact in performance of each of the variations, we will also analyze the diversity of the resulting candidates, to ponder the usefulness of ensembling them into a more complete model.

2.3 False positive reduction

2.3.1 Handpicked feature classifier

2.3.1.1 Selected features

As seen in the previous chapter, the probability map obtained by the segmented slices is not informative enough to calculate the likelihood of the predictions, but the shape of the labels themselves potentially hold information that can help us distinguish between real and false nodules. To explain this concept visually, we can compare the segmented nodules A and C in Figure 2.1. The first one is an example of a large nodule, mostly round, mostly contiguous in the Z-axis. Nodule C, on the contrary, while having a round segmentation in the axial plane, is almost flat, which typically translates to a false positive. Another frequent source of false positives are caused by the presence of airways in the lung. On a single slice they can be easily mistaken for a nodule, but if we pay attention to their coronal and sagittal projections we will appreciate large displacements, forming an elliptical shape. This effect can be observed to some degree in nodule B, and more agressively in nodule D.

Based on the visual inspection of the masks obtained by our segmentation, we engineered the following features to characterize the nodules:

diameter mesures diameter (in mm) of the bounding box in the axial plane.

layers measures number of contiguous layers of the bounding box in the z-axis.

squareness measures how similar the shape is between the axial and its ortogonal planes. Values range between 0 and 1. 0 means ratio between axial and the ortogonal planes (sagittal and coronal) is the same. 1 would mean that one side is completely square, while the other flat. Formulated as:

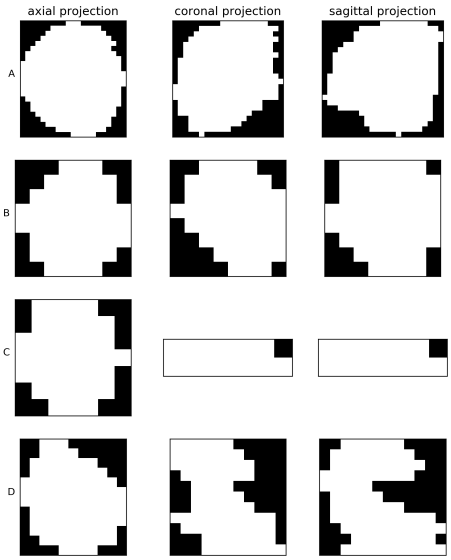

Figure 2.1: axial, coronal and sagittal projections of 4 nodule masks as segmented by our U-Net network. Even though the axial projection is similar in all the examples, the sagittal and coronal views offer a much larger degree of variance.

$$squareness(length, width, depth) = abs\left(\frac{min\{width, length\}}{max\{width, length\}} - \frac{min\{depth, \frac{width+length}{2}\}}{max\{depth, \frac{width+length}{2}\}}\right)$$

extent measures the ratio between masked and unmasked area in a labeled bounding box. Formulated as:

$$extent = \frac{num\ masked\ pixels\ of\ bbox}{num\ total\ pixels\ of\ bbox}$$

axial eccentricity measures the geometric eccentricity of the segmented nodule projected on the axial plane. 0 would indicate the projection is a perfect circle.

sagittal eccentricity measures the geometric eccentricity of the segmented nodule projected on the sagittal plane. 0 would indicate the projection is a perfect circle.

It should be noted that these features are only capturing basic information about the shape of the segmentations. This model ignores texture or other finer-grained features based on shape.

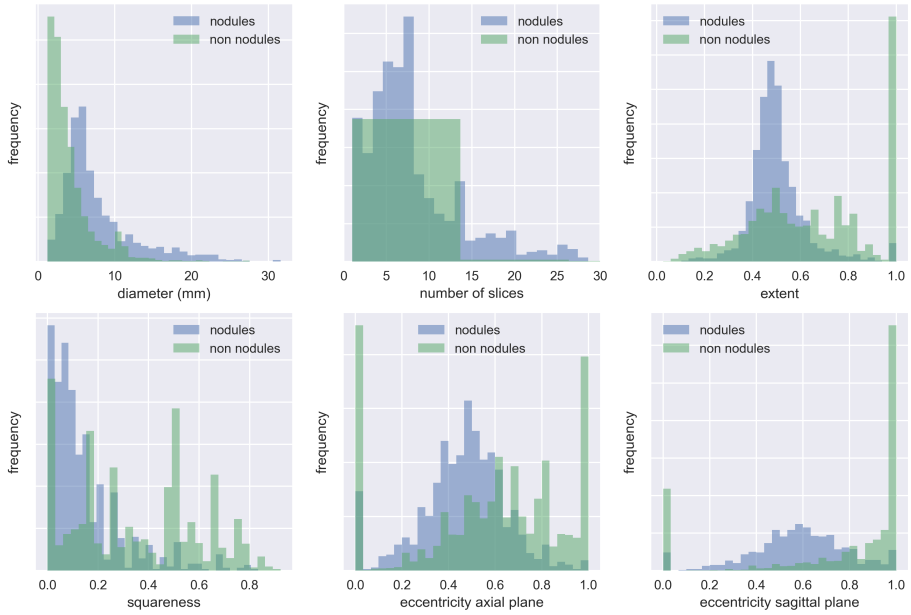

Figure 2.2: frequency distribution of the nodule candidates features, obtained by segmenting the entire LUNA dataset with the *augmented, 3ch, batch normalized, bce unet*. The histograms of TP and FP are overlapped and normalized.

2.3.1.2 Training the model

We're going to train multiple binary classifiers with the features presented above and compare their performance quantitatively employing the AUROC. We're also going to plot the entire ROC curve to qualitativaly assess the behaviour of the classifier as the false positive rate increases. The tests will be performed both on the training and test sets, so we can also compare the performance of both side-by-side and assess the tendency to overfit of each of the classifiers.

The training and testing will be performed on the candidates obtained by the segmentation network based on binary cross entropy plus all variations (see previous chapter). Candidates from subsets 0 to 8 will be used as training data, while candidates in subset 9 will serve as our test dataset. We're not going to tune hyperparameters on the classifiers, so no validation set will be employed. This basically leaves us a dataset with a 4 to 1 ratio in FP vs TP that we will not rebalance. More details about the dataset can be found in Table 2.2.

Table 2.2: Baseline from running the segmentation network. The classifier will be trained and evaluated on the features extracted form those candidates.

	Training (subsets 0 to 8)	**Test (subset 9)**
number of scans	776	84
number of candidates	5415	599
TP	1032	93
FP	4383	506
average FP per scan	5.6482	6.0238

We've selected a list of 5 classification algorithms (see Table 2.3), from simple logistic regression models to more advanced tree boosting classifiers, in an attempt to understand what sort of classification strategy works best both in terms of performance and generalization. We've used the `scikit-learn` [19] implementation of those algorithms, initialized with default parameters, for training and evaluation purposes.

Table 2.3: Types of classifiers trained on the candidates' dataset

Classifiers
Logistic regression
Decision tree
Random forest
AdaBoost
Gradient boosting

2.3.2 Radiomics based classifier

Expanding on the idea of the previous classifier, we wanted to further investigate the applicability of radiomics [20] to discriminate between nodules and false positives. In this classifier, instead of training it based on manually handpicked features, we've used the software package pyradiomics [21] to automatically extract 105 features from each nodule segmentation. Instead of comparing different classification algorithms, we've used the same AdaBoost with different subsets of radiomic features. We have tested its predictive performance with 105 features, 20 and 5 (the last two cases after applying a dimensionality reduction with PCA). This allows us to determine how much of a predictive advantage are we obtaining in comparison to the features we've manually engineered. A complete list of the extracted features can be reviewed in [22].

Methodology wise, the only difference in reference to the system trained on the handpicked features lies in the feature extracting process. We've transformed the scan segmentation masks to individual nodule masks, each of which is used in conjunction with the scan to automatically extract the features. The pyradiomics package also allows us to apply different filters on the original scan, and then extract them based on the filtered image, but we haven't used that functionality in this particular experiment.

2.3.3 ResNet based classifier

We're going to train multiple volumetric ResNet networks with different depths and compare their performance quantitatively emplying the AUROC. Similarly to what we've done in the manual feature classifier, we'll also plot the entire ROC curve of the classifier. As before, both training and testing curves will be plotted side by side, to assess the overfitting of the model.

Regarding the network architecture itself, we introduced the suggestions by [5] and added a batch normalization and ReLU layer before each convolutional layer on the residual module, to facilitate convergence and weight stability while training. The same network was trained on different layer depths: 34, 50, 101 and 152.

As training data we will use the annotations provided by LUNA for the false positive reduction track of the challenge. They contain the world coordinates of the candidate centroid and a label indicating whether or not it is a nodule. See Table 2.4 for details regarding the distribution of this dataset. We will evaluate the model against the candidates obtained by the same segmentation network as in the previous section, so that we can compare the performance between the two different methods.

Table 2.4: Number of entries per class in the candidate annotations dataset, divided by split. The class imbalance between the two categories is very prominent, which we'll have to take into account when training the network.

dataset split	FP	TP	ratio
training (subsets 0 to 7)	603345	1218	495 to 1
validation (subset 8)	74293	195	381 to 1
test (subset 9)	75780	144	526 to 1

Since we are not using an ensemble of multiple models, the volumetric patch we will use as input should capture the entirety of the nodule. Based on the data observed in Figure 2.2, the dataset does not contain diameters above 32mm, so we will fix the input resolution to be `32x32x32x1`. The scans have been rescaled to a spacing of 1x1x1mm and the images only have 1 color channel, with values corresponding to the Hounsfield value of the voxel (no normalization or clipping applied in the preprocessing).

The training is performed for a maximum of 50 epochs, only saving the weights in the iterations with better validation loss. We're using Adam as our method for stochastic optimization, initialized to a learning rate of *1e-3*. Early stopping is applied if the validation loss is not shown to improve in 10 consecutive epochs. The batch size for resnets {34, 50 and 101} was 64, while the batch size for resnet 152 was 32 due to memory constrains on the GPU side. Binary crossentropy was used as the loss function. The hardware employed during training consisted on an Intel i7 7700, 32GB of RAM and a Nvidia 1080Ti GPU.

To offset the data imbalance observed in the dataset (see Table 2.4) we will oversample the nodule annotations with replacement so the training and validation ratio is 2 to 1 (FP vs TP). This effectively means that a nodule annotation will be seen during training 250 times per each non-nodule one, which could very well induce the network to overfit. We mitigate this effect by using 3D image augmentation. As detailed in Table 2.5, affine transformations are randomly applied to the input cube before passing it to the neural network. Since this transformations would be lossy if applied to the actual cube of 32x32x32, we actually retrieve a larger cut of 46x46x46, apply the augmentation, and return a centered view of 32 pixels per side. The augmentation cube side needs to be larger than the diagonal of the input one for this to be valid. Also important, the augmentations are randomly applied to each sample each time and the dataset is shuffled on each epoch.

Table 2.5: Range of transformations randomly applied to both the axial and coronal planes of the input volume

transformation	range
rotation	[-90º, +90º]
shearing	[-20%, +20%]
scaling	[-10%, +10%]
flip vertically	[True, False]
flip horizontally	[True, False]
translation width	[-2px, +2px]
translation height	[-2px, +2px]

It should also be noted that the training and validation have been performed on a smaller fraction (35%) of the original data. This is the case purely due to hardware limitations when performing the experiment. Basically, extracting small patches of data from a much larger image is only fast if said image is already loaded, so we reduced the dataset size until it could fit in memory (32GB). Preloading the scans in-memory instead of reading them from disk supposed a speed-up larger than 2 orders of magnitude per epoch, so we considered the trade-off worthwhile.

2.4 LUNA performance comparative

Once the individual systems have been evaluated, we pick the best variations of nodule detector and false positive reduction and rank it according to the LUNA grand challenge rules. For this, we will have to plot the FROC curve at the average false positives rates between 0.125 and 8. Also, we will report the average sensitivity at the selected false positive rates of {0.125, 0.25, 0.5, 1, 2, 4, 8}.

It should be noted that there is a set of excluded annotations available in the LUNA dataset that neither count as false positives nor true nodules. Any candidate matching one of those annotations need to be ignored towards the results. Another caveat of our particular comparison is the fact that the challengers in the LUNA scoreboard train their models performing a 10-fold cross validation over the whole dataset, and then evaluate their results on all the annotations, using a 1000 bootstraps. The reported metrics for our system will only be calculated over the test split (subset 9 of LUNA), on a model that has not been cross validated. This is mostly due to the required time and resources that we would need for this training to finish. We still feel it is a fair assessment, on the understanding that it is not biased to favour our system.

Chapter 3

Results

The results chapter is divided in five sections. The first three report the metrics for each individual problem of the CAD pipeline, paying especial emphasis to the differential in performance between different approaches. The 4th section compares the metrics of the system against other competitors of the LUNA grand challenge to contextualize it within the state of the art. Finally, there is a 5th section that assesses qualitatively the efforts placed to integrate the system into a clinical context.

3.1 Lung segmentation

The results of using the U-Net for lung segmentation purposes show a network that achieves a Dice score over 98% in 40 epochs of training, although it only takes two epochs for the scores to be above 96%. As we can see in Table 3.1, there are no signs of overfitting. In fact, the Dice coeficients for both the validation and testing sets are 0.5% better than the results on the training dataset, although this could be explained due to the extra variability found in the larger training dataset (616 CT scans vs 88 each on the validation and test splits).

Table 3.1: mean Dice coefficient for each dataset split. Each axial slice in the dataset is evaluated individually and then the mean is calculated in two steps. First the mean over the whole scan and then the mean over the dataset .

dataset split	**Dice score**
training	0.977392
validation	0.983458
test	0.984037

The U-Net lung segmentation is especially accurate in the superior and middle lobe of the lung, as we can appreciate in the bottom left slice on Figure 3.1. Irregular areas with lower contrast, like the one in the left lung shown in the bottom right slice (same figure) can confuse the network. The trend is towards more expansive masks, which in our case is a valid trade-off, since this is only a preprocessing step done towards reducing the complexity of the nodule segmentation task, and we don't want to discard potential lung mass which may contain a nodule.

Lower lung lobes are generally those with lower Dice scores (see top left and right slices in Figure 3.1). It is also possible to observe holes inside a segmented lung (bottom right), which could be fixed by applying a morphological closing in the mask.

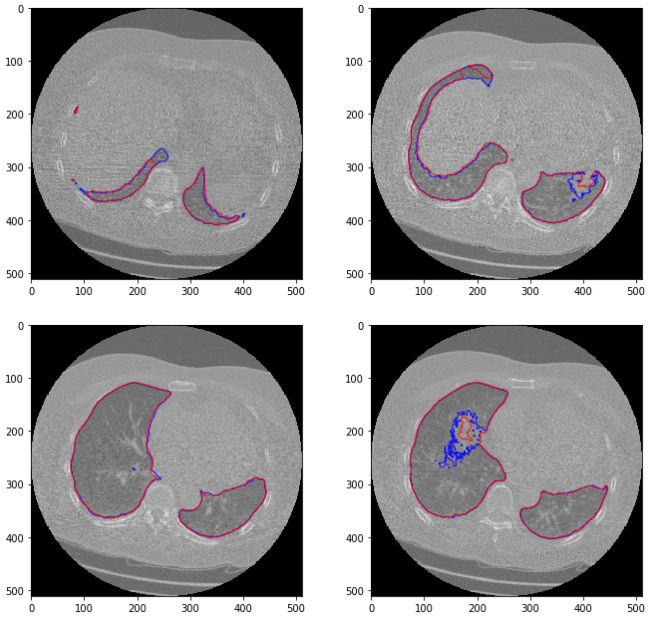

Figure 3.1: Axial slices of a lung with both masks superposed. The blue segmentation corresponds to the ground truth used to train the U-Net, while the red segmentation is the prediction returned by the model.

3.2 Nodule detection

Our main metric for the nodule detection module is its sensitivity. This will determine the upper limit performance of the system. We will also keep track of the number of false positives returned, since this directly affects the complexity of the false positive reduction module. More false positives will require a more complex model in order to be competitive with the state of the art.

In Table 3.2 we have both metrics divided by the loss function used during the training phase and the different image processing variations applied. We haven't included the figures of the network trained without batch normalization as they were not very telling themselves, but we still wanted to report those negative results, because they were the key that allowed the network to learn. As we can see in Figure 3.2, a U-Net without normalization, neither on the input image nor on its convolutional layers, is incapable of learning the true representation of a nodule. Basically the only information it can extract form the original image is a rough segmentation of the lung parenchyma, which happens to be the area of major contrast in the original slice (as a reminder, a lung mask is applied as a preprocessing step, fixing the value of any voxel outside the lung tissue to -4000HU). This was the key discovery that made this approach feasible.

Both loss functions achieve similar top sensitivity scores in Table 3.2. In general, binary cross entropy displays a more stable behaviour during training and it penalizes false positives more heavily, as we can see from looking at the false positive rates of both networks (3 to 1 ratio favoring binary cross entropy). The downside of this heavy penalization of false positives is a slight drop in sensitivity (*0.915* vs *0.930*) that caps the maximum performance of the system.

It is also worth mentioning the effect of applying augmentation to mitigate overfitting. If we take a look at the differences between the augmented binary cross entropy variation vs the non-augmented (Table 3.2) the differential in training sensitivity barely achieves a 0.5%, but the gap between the training and testing scores goes from a 19% to 11%, and down to 6% in its best performing variation.

We also analyzed whether the different variations introduced diversity in the nodules detected by the network. As we can see in Figure 3.3, the nature of each variation is purely additive. The network is able to detect more nodules while still discerning the previous subset. In practice, this would mean that we might need to develop an entirely different model to detect the nodules we are currently missing, so that an ensemble based on both models would beat their individual performance.

Finally, we compared the individual performance of our best segmentation networks against the results presented in the LUNA16 challenge survey (1) in Table 3.3. Our network is able to beat the individual systems described in the paper both in sensitivity and in number of false positives reported by scan. It is especially in this last metric where the differences are the most striking.

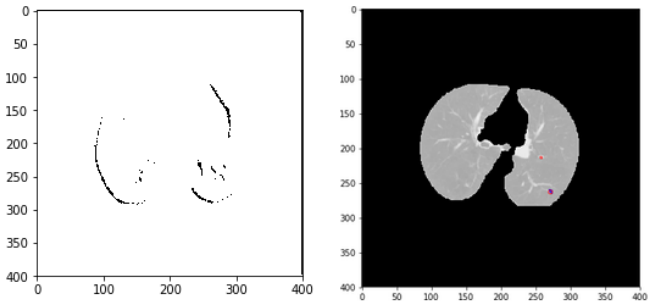

Figure 3.2: nodule segmentation network results when the U-Net is trained without applying batch normalization on each convolutional layer on the left. On the right we see the output of the same network after enabling batch normalization. The red contour corresponds to the predicted mask while the blue markings match the ground truth used for training.

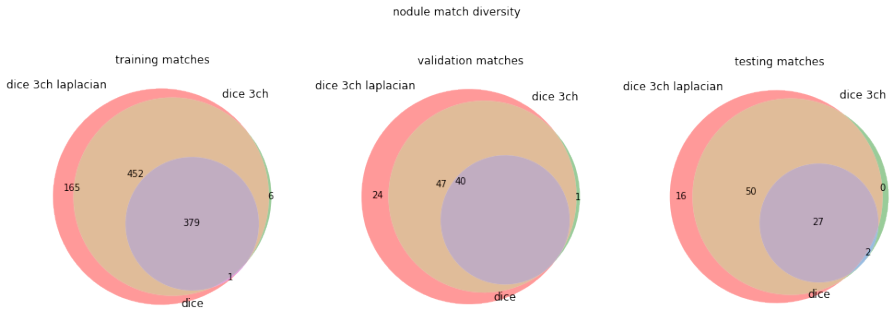

Figure 3.3: This venn diagram showcases the additive nature of the variations performed in the U-Net. The lack of diversity in its predictions discouraged the use of an ensemble to increase its overall performance.

Table 3.2: U-Net nodule segmentation variations along with their sensitivity and number of FP per slice. It should be noted that nodule candidates may refer to the same annotation, so it is possible for the sensitivity to take values over 1.

loss function	variation	set	sensitivity mean	FP mean
crossentropy	no augmentation, normalization	test	0.783051	7.329545
		train	0.977351	6.707865
		validation	0.796944	6.840909
crossentropy	augmentation, normalization	test	0.859275	6.011364
		train	0.972629	5.703652
		validation	0.922778	5.488636
crossentropy	augmentation, normalization, 3ch, laplacian	**test**	**0.915490**	**5.750000**
		train	0.974303	5.515449
		validation	0.940417	5.181818
dice	no augmentation, normalization	test	0.740254	7.125000
		train	0.828008	7.063202
		validation	0.795972	6.784091
dice	augmentation, normalization	test	0.339407	1.443182
		train	0.390669	2.252809
		validation	0.399306	1.750000
dice	augmentation, normalization, 3ch	test	0.803672	34.125000
		train	0.818526	34.228933
		validation	0.806389	33.715909
dice	augmentation, normalization, 3ch, laplacian	**test**	**0.930791**	**15.420455**
		train	0.944604	17.234551
		validation	1.044861	14.193182

The U-Net trained with binary cross entropy is able to match the sensitivity of ETROCAD (best reported) within a 1%, but it is able to do so with 47 times lesser amount of candidates. The levels of accuracy provided by our network will, in fact, enable us to develop false positive reduction methods based on the nodule segmentation themselves, and still be competitive in the general LUNA scoreboard.

Table 3.3: Candidate detection systems performance as reported by 1. Even though each individual system is offering worse performance than our custom U-Net, an ensemble combining them reported sensitivity rates up to 0.983.

system	sensitivity	avg num candidates / scan
ISICAD	0.856	335.9
SubsolidCAD	0.361	290.6
LargeCAD	0.318	47.6
M5L	0.768	22.2
ETROCAD	0.929	333.0
lucanode bce	0.915	7.0
lucanode dice	0.930	18.0

3.3 False positive reduction

In Figure 3.4 we have plotted the resulting ROC curve for the different classifiers trained on the handpicked features of the nodule segmentation. Both classifiers based on trees (decision tree and random forest) overfit, and while the logistic regression does not, it doesn't perform as well as the boosting classifiers.

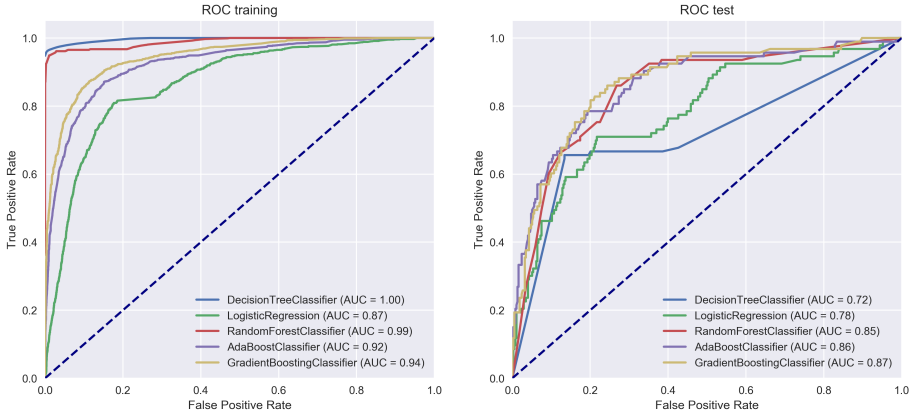

Figure 3.4: ROC curves and AUC of the handpicked feature classifiers.

In Figure 3.5 we have the probability histogram of both nodules and non-nodules. As expected, even though the distributions are different in all classifiers, only in the boosting algorithms there is a clear distinction between the two classes, which translates to its performance in the ROC curve.

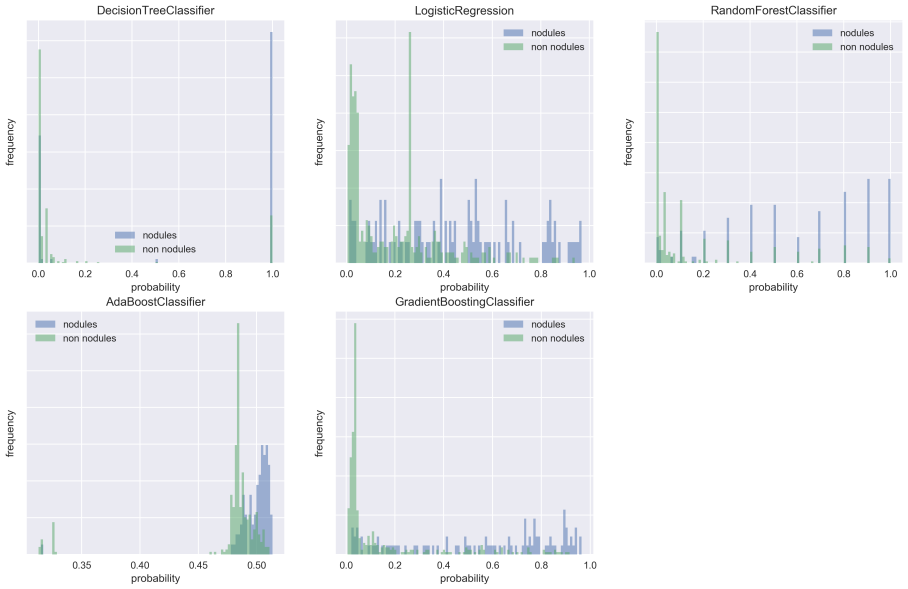

Figure 3.5: histogram pdf handpicked features.

We've performed the same tests on the radiomics based classifier, as seen in Figure 3.6. In this plot, instead of testing different classifiers, we've decided to plot the ROC of the same classifier trained on different subsets of radiomic features. As we can see, having more of them does not necessarily translate to better performance. In fact, there is a sweet spot around 20 (vs the original 105). Still, the same figure also shows that the radiomics classifier is lagging behind the one based on only 6 handpicked features (Figure 3.5).

On Figure 3.7 we have the results for a classifier based on a 3D ResNet trained at diferent depths. Apart from offering the best performance of the three systems, the AUC of both training and testing is almost identical, being the set of classifiers with the least amount of overfitting. Still, it should be noted that this classifier model has been trained on a dataset two orders of magnitude larger than the previous two approaches. The ResNet network has signs of saturation as its depth increases. Past 50 layers, AUC slightly decreases, and even during its training phase it was rare for the validation loss to reliably decrease on each epoch.

In Figure 3.8 we have the ResNet 50 and the handpicked based classifiers side by side. As expected, the ResNet has a better AUC and plateaus at a lower false

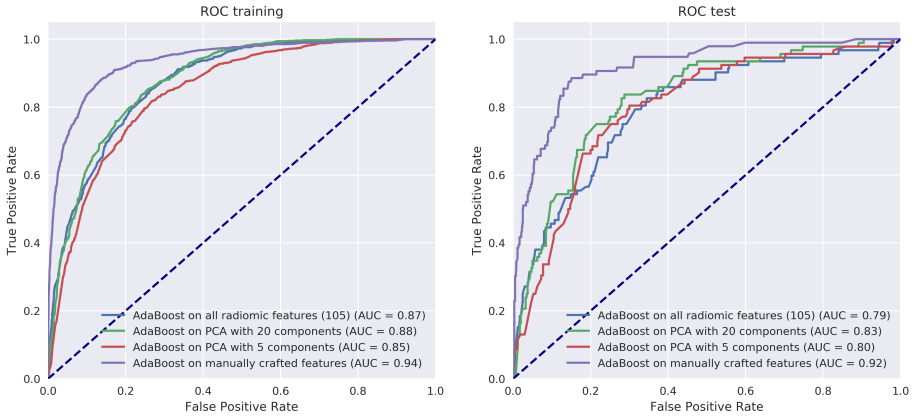

Figure 3.6: ROC curves and AUC of the radiomics based classifiers.

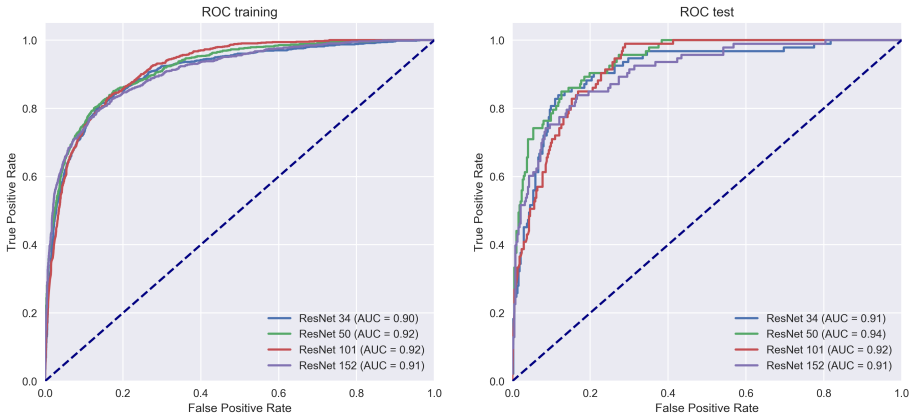

Figure 3.7: ROC curves and AUC of the ResNet based classifiers.

positive rate than the AdaBoost classifier, which is a very desirable property for this kinds of systems.

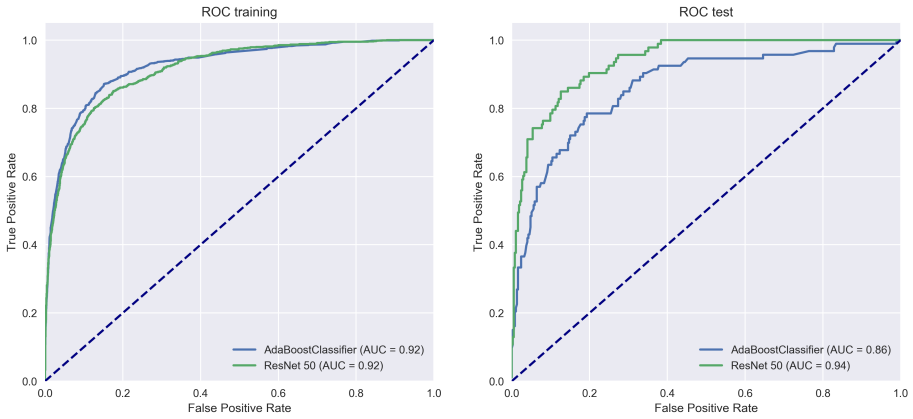

Figure 3.8: ROC curves and AUC comparing the best 2 variations of FP reduction method.

3.4 LUNA performance comparative

On Figure 3.9 we have plotted the FROC curves of the best two variations of FP reduction systems along with the U-Net nodule segmentation (binary cross entropy loss, normalization, augmentation, three channels and laplacian filters applied). The reported score is an average of the sensitivity at selected FP rates [0.125, 0.25, 0.5, 1, 2, 4 and 8]. Both average false positive ranges and metrics have been set by the LUNA grand challenge to enable fast and objective comparisons between CAD systems.

Our system, *lucanode*, achieves a top-18 performance in the general LUNA leaderboard (as of June 2018, see Figure 3.10). Even the model with handpicked features would be worthy of a top-20, which is commendable for a system essentially based on a single model. In our favour, we would like to remind that our results come exclusively from the testing split (subset 9 of LUNA), while the other systems are the results of a model trained on a 10-fold cross validation and evaluated over a 1000 nodules with bootstrapping. We would expect that training the existing models in this manner, plus using the higher sensitivity segmentation U-Net (with Dice as its loss function) would bring us closer to the top 14. Still, even if we could easily improve the upper bound of our system, the slope of our false positive reduction is too steep compared to *resnet's*, *ZNET's* or *PATech's*, so our performance at lower averages of false positives per scan would still be subpar.

All of these results have been obtained with the binary cross entropy variation of

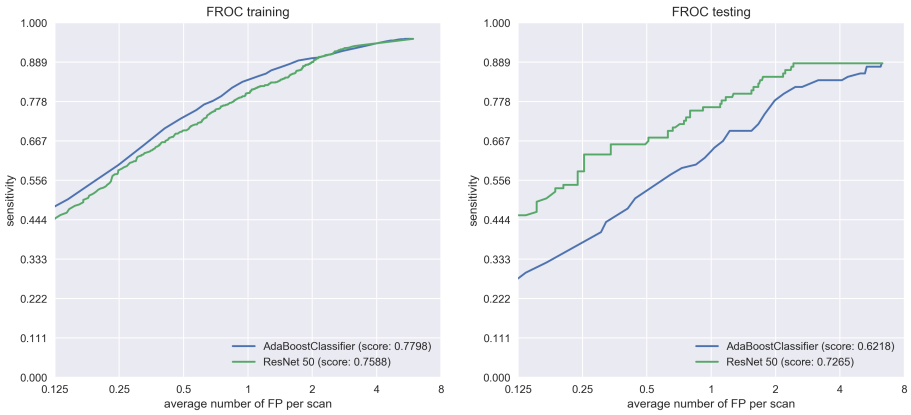

Figure 3.9: FROC curves and averaged sensitivity at selected FPR comparing the best 2 variations of false positive reduction.

the segmentation network. Even though its sensitivity rate was 1.5% lower it was 3 times as accurate, which at the end pushed the overall score slightly above the variation trained with Dice. This also demonstrates that a good segmentation step simplifies the false positive reduction problem.

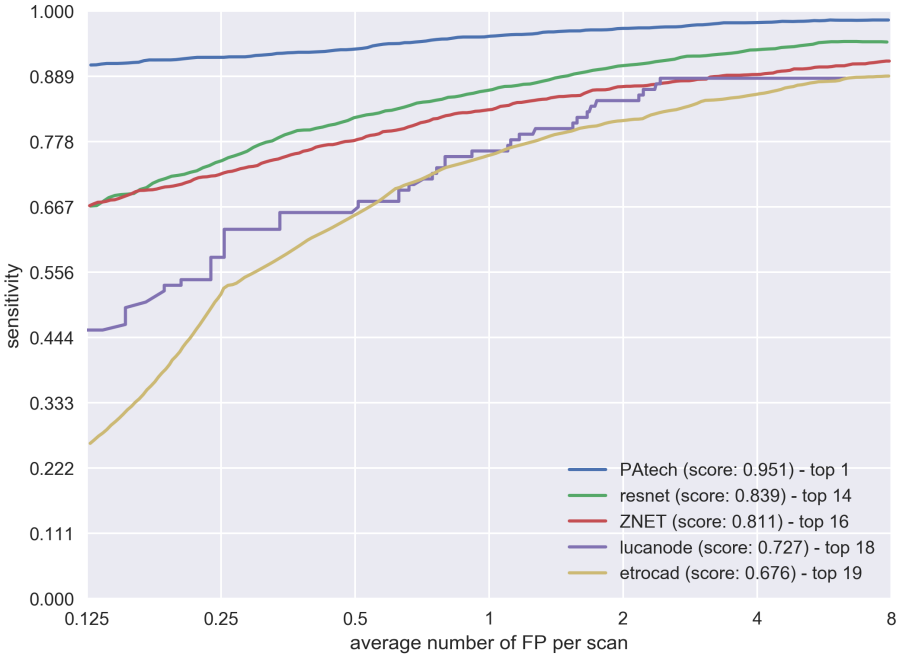

Figure 3.10: FROC comparison of lucanode with U-Net binary cross entropy + ResNet50 against other LUNA contenders.

Chapter 4

Discussion

4.1 Lung segmentation

The initial approach we wanted to follow for the lung segmentation was to create a multiatlas lung segmentation such as the ones described in [23, 15]. Still, due to the required computational time (registration libraries only operate on CPU) and the fact that lung segmentation has a relative low impact in the final score of the system, we chose to employ the same architecture we had been using to segment nodules for the lungs themselves.

Perhaps unsurprising, but still very interesting, to demonstrate the high degree of transferability of the network architecture to different tasks, just by adjusting the input masks. Even the issues the network was currently facing segmenting the lower lobule could very likely be fixed by rebalancing the dataset to oversample that part of the lung. Also, there are some other easily corrected errors, such as the removal of holes inside masks. Even after applying these fixes, we could still bring things further by applying some image preprocessing to increase the contrast of the image, or applying augmentation to safely oversample troublesome slices.

4.2 Nodule detection

The first thing we would have to do with the current results is to visualize the currenly undetected nodules and try to understand what similarities do they share. If they are mostly non-solid nodules, we could try to find another image filter that highly increases their contrast. If that does not help, it might be time to consider using a 3D U-Net architecture, or maybe an object detection network (based on an image recognition model) instead of a segmentation one. The intuition behind this affirmation, apart from our review of the state of the art, is also based in the resulting masks for the false positive candidates. They tend to be too flat, which is a sign that the network is not capturing depth properly. In this case it might also be worth it to train another 2D segmentation layer over another one of the CT planes (i.e. sagittal). Once that would be ready, we could use both models to create an ensemble that better captures the shape of a nodule.

This section of the thesis has also been a very good practical demonstration on how small details can make or break a network's performance. For a long time, we were trying to train the U-Net without a batch normalization layer, which didn't converge. That was one of the multiple times where the non-linear progress on the development of this project came into full display. Also, related to the batch normalization, there are other strategies, like the one presented in [24] which look promising for our case at hand, since the length of the batch is severily limited due to the size of the images.

4.3 False positive reduction

The classifier based on handpicked features was an interesting use case of the remarkably low number of false positives returned by our U-Net model. It was only thanks to the strength of the original segmentation network that such approach could ever prove useful. Still, an approach like this is not without drawbacks. It simplifies the system, but it introduced dependencies between layers. Can we affirm that the same model would equally work if the underlying U-Net had been trained with another loss function? Or maybe another set of masks? It probably wouldn't, since the results are very much dependent on the original network. This feedback also introduces some questions regarding the generalization of the model (performance between training and testing suffers a noticeable drop, especially compared to the model trained with residual networks). Finally, the fact that we are relying on the output of the segmentation network also limits the number of examples we have to train the classifier, which caps its performance. The usage of image augmentation might have been a worthwhile idea to explore, although this would involve quite a bit of image preprocessing. Probably not enough to justify the improvement in performance.

To our surprise, the classifier based on automatically extracted radiomic features actually performed worse than our manually handpicked classifier. A possible explanation to this fact could be that the segmentation itself we're parting from is already far from a proper representation of a nodule, and this handicaps its performance (radiomics are usually applied on manual segmentation of tumors to characterize them, not to classify lesions from non-lesions). It is also interesting noting that a PCA with lesser components is able to outperform the original 105 features. This suggests that the scattershot approach of radiomics might not actually deliver on its promises. Still, we would like to hold our judgment on the technique unless applied on a dataset with proper segmentations and maybe another kind of classification task. For example, applying radiomics to predict malignancy from a set of nodule segmentations would probably provide a fairer appraisal of the technique.

Finally, we prove again why image recognition methods based on deep learning are the current state of the art. A volumetric convolutional network, based on a residual architecture, beats previous systems by more than a 10% margin in the final LUNA FROC score. Even though the ResNet beats the other systems, we still face the problem that the deeper layered versions saturate (probably due to the small input size of the image). For this particular experiment we've also had to face a few technical limitations that prevented us from fully utilizing the whole available dataset, which surely didn't help on the final score.

In an improved version of the system, it would be interesting to explore the impact of using multiple ResNets trained on multiple input fields. Having multiple models trained at different scales would be best to capture the inherent nodule heterogeneity. If we only use one model we have to ensure that the size of the input image is going to be big enough to capture the vast majority of nodules

inside it, which might be too big a cube for the majority of nodules (below 8.5mm in diameter). Also relevant for the false positive reduction track, we are using an isotropic resize to 1x1x1mm, sacrificing resolution in the axial plane (originally around 0.6mm). If the resize had been performed at 0.6mm or 0.5mm for this particular part of the pipeline, we would also have more information from the original image, which might positively impact in the overall performance of the system.

4.4 LUNA performance comparative

We have already commented on possible improvements on both nodule detection and false positive reduction in the previous two sections, so we are not going to cover the topic. Instead, we would like to focus on the importance of the FROC metrics to determine where the development efforts are best spent.

Compared to our immediate competitors, we have a competitive upper bound but the FROC curve is not as flat. This readily suggests that our false positive reduction system is not as competitive and should be the first component of the system to be brought for assessment. This is just an example of how visualizations are key into gaining insights about the system, which has been a constant throughout the project.

It is also interesting to note how small gains in the ranking actually require an exponential amount of work. We can achieve top-20 results by employing a single U-Net model and engineering a few features from the resulting segmentations, but we require training a deep residual network for days to improve the ranking by 2 spots. Going further than that would surely demand longer training times, a more creative use of image preprocessing, fully switching to volumetric models and embracing ensembling, which would force us to deal with the vicissitudes of each one of its models, which just keeps raising the bar for such a system to be ever deployed in production.

4.5 Integration into a clinical workflow

One of the questions we haven't directly adressed in this thesis is how *lucanode* could be integrated into a clinical workflow. Ideally a CAD such as this one would be part of the hospital protocol, and automatically executed whenever a thoracic CT scan is performed. This would always provide a second opinion to the radiologist in charge of diagnosing the results and, hopefully, result in a lower *interobserver variability*, that is, the performance gap between an experienced radiologist and a resident, for example.

Usually systems participating in challenges such as LUNA are only worried about the raw performance of their algorithm, which puts the possibility of executing

it end to end as part of an integrated pipeline very far down the list of priorities. We didn't want to do this with lucanode, so we took steps towards making the results easier to reproduce and execute on an arbitrary scan.

The first step towards integrating the pipeline is ensuring the preprocessing of the scans can be performed on the fly. This is the reason why we've trained an automated lung segmentation model. Also, as part of this model, we're also performing the CT scan voxel resampling, so all the necessary inputs to execute the nodule segmentation and false positive reduction are in place.

Another important concern that needs to be tackled is the packaging of the software. A self-contained solution is much easier to setup and distribute, but it takes effort to prepare such a package. In our case, thanks to conda and Docker, we have readily available images that contain both the code and the model weights to execute the *lucanode* system as if it were a self-contained binary. There is even experimental support for nvidia docker [25], a containerization technology capable of using the underlying GPU resources of its host. In fact, even though we haven't properly benchmarked the system for this thesis, the results of our effort are already live and have been presented in Albert Moral's graduation final project [26].

For a future revision of the work, we would like to properly benchmark the system and also take advantage of it by analyzing its performance in other datasets than LUNA. This would also prove to be a very valuable addition to the literature, as we could check how well the model generalizes to different datasets.

4.6 Future work

As far as improving the performance of the current system, the logical step forward is to create an ensemble of multiple models to improve its overall performance. There are also other paths worth pursuing. For example, the candidate detection network could be changed from a segmentation network into an object detection one. This way we could use the same model we employ for false positive reduction to detect the nodules themselves. It would be interesting to test the usage of a R-CNN [27] or a YOLO [27]. Plus, we are not aware of such architectures having been used to detect objects in a volume, so that could be a worthy contribution in and of itself.

Apart from incremental improvements on the performance of the current system, there is an angle which hasn't been explored, and that is inferring the malignancy of the nodules. The LUNA dataset does not contain such information, but the dataset it is based on (LIDC-IDRI) contains annotations by radiologists with the estimated malignancy per nodule, a real handmade segmentation, and even the real diagnostic for a few of them. All this information could be used to transform the false positive reduction system into a malignancy predictor, so the system would not only detect abnormalities, it would also diagnose them.

4.7 General discussion

The most unexpected side effect of developing *lucanode* has been the paradigm shift from traditional software development, where you are iteratively coding an increasing set of rules to make the system smarter to training models, where you are repackaging your data the way your network understands it best. In a way, training a deep learning model is just about finding the metaphor that makes it click. Data scientists, like comedians, need to find the right kind of humour for their audience.

I come from an engineering background and I am more used to blacks and whites than shades of grey, so the change has been quite a shock. On the one hand, the performance of such systems is first-class, but they also make you wonder about their maintainability in the long term. How can you fix what you can't explain? In fact, you can check [28] for more on the topic.

Another aspect I would like to mention is the brittleness of these models, especially on systems with pipelines that feed one another. Any error, anywhere in the pipeline, can invalidate the whole experiment. And it is very typical to have them, especially since there are shared elements that need to be setup in slightly different, but very significant, ways. It gets especially bad when you want to train and evaluate different variations of the same model side by side. You need to do it to properly test your hypotheses, but that in itself increases the error rate of said experiment.

My best recommendation to the grievings above is to work on the visualizations. Whenever I've been stuck, whenever I wanted to confirm a result, I've always found solace in a good visualization. They are the best way to confirm your system works as intended and the best way to understand where it doesn't.

Bibliography

1. Arindra, A. *et al.* Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The LUNA16 challenge. (2017).

2. Siegel, R. L., Miller, K. D. & Jemal, A. Cancer Statistics , 2018. **68,** 7–30 (2018).

3. Jacobs, C. *Automatic detection and characterization of pulmonary nodules in thoracic CT scans.* (2015).

4. Hosang, J., Benenson, R., Dollar, P. & Schiele, B. What Makes for Effective Detection Proposals? *IEEE Transactions on Pattern Analysis and Machine Intelligence* **38,** 814–830 (2016).

5. Chen, H., Dou, Q., Yu, L., Qin, J. & Heng, P. A. VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images. *NeuroImage* **170,** 446–455 (2018).

6. Wu, S., Zhong, S. & Liu, Y. Deep residual learning for image recognition. 1–17 (2017). doi:10.1007/s11042-017-4440-4

7. Bengio, Y., Simard, P. & Frasconi, P. Learning Long Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks* **5,** 157–166 (1994).

8. Glorot, X. & Bengio, Y. Understanding the difficulty of training deep feedforward neural networks.

9. Ioffe, S. & Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. (2015). doi:10.1007/s13398-014-0173-7.2

10. LeCun, Y. A., Bottou, L., Orr, G. B. & Müller, K. R. Efficient backprop. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **7700 LECTU,** 9–48 (2012).

11. He, K. Delving Deep into Rectifiers : Surpassing Human-Level Performance on ImageNet Classification. (2014). doi:10.1.1.725.4861

12. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. 1–15 (2014). doi:http://doi.acm.org.ezproxy.lib.ucf.edu/10.1145/1830483.1830503

13. Chollet, F. & Others. Keras. (2015).

14. Agarwal, A. *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. (2015).

15. Van Rikxoort, E. M., De Hoop, B., Viergever, M. A., Prokop, M. & Van Ginneken, B. Automatic lung segmentation from thoracic computed tomography scans using a hybrid approach with error detection. *Medical Physics* **36,** 2934–2947 (2009).

16. Walt, S. van der *et al.* scikit-image: image processing in Python. *PeerJ* **2,** e453 (2014).

17. Fiorio, C. & Gustedt, J. Two linear time Union-Find strategies for image processing. *Theoretical Computer Science* **154,** 165–181 (1996).

18. Wu, K., Otoo, E. & Shoshani, A. Optimizing connected component labeling algorithms. 1965 (2005). doi:10.1117/12.596105

19. Nielsen, D. Tree Boosting With XGBoost Why Does XGBoost Win "Every" Machine Learning Competition? *NTNU Tech Report* 2016 (2016).

20. Gillies, R. J., Kinahan, P. E. & Hricak, H. Radiomics: Images Are More than Pictures, They Are Data. *Radiology* **278,** 563–577 (2016).

21. Griethuysen, J. J. van *et al.* Computational Radiomics System to Decode the Radiographic Phenotype. *Cancer Research* **77,** e104–e107 (2017).

22. Griethuysen, J. J. van. PyRadiomics documentation - features. (2018).

23. Rohlfing, T., Brandt, R., Menzel, R. & Maurer, C. R. Evaluation of atlas selection strategies for atlas-based image segmentation with application to confocal microscopy images of bee brains. *NeuroImage* **21,** 1428–1442 (2004).

24. Wu, Y. & He, K. Group Normalization. (2018).

25. NVIDIA. Nvidia docker. (2018).

26. Moral Lleo, A. (. Creacio d'un entorn tecnologic destinat al calcul de resultats de recerca mitjancant orquestracio al nuvol. (2018).

27. Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017). doi:10.1109/TPAMI.2016.2577031

28. Sculley, D. *et al.* Machine Learning : The High-Interest Credit Card of Technical Debt. *NIPS 2014 Workshop on Software Engineering for Machine Learning (SE4ML)* 1–9 (2014). doi:10.1007/s13398-014-0173-7.2