Synthetic Lung Nodule 3D Image Generation Using Autoencoders

Steve Kommrusch[a] and Louis-Noel Pouchet[a]
[a]Dept. of Computer Science, Colorado State University, Fort Collins, CO, USA

November 8, 2018

Technical Report CS-18-101

CONTACT Steve Kommrusch. Email: steve.kommrusch@gmail.com
CONTACT Louis-Noel Pouchet. Email: pouchet@colostate.edu

Synthetic Lung Nodule 3D Image Generation Using Autoencoders

Steve Kommrusch[a] and Louis-Noel Pouchet[a]

[a]Dept. of Computer Science, Colorado State University, Fort Collins, CO, USA

ARTICLE HISTORY
Compiled November 8, 2018

ABSTRACT

One of the challenges of using machine learning techniques with medical data is the frequent dearth of source image data on which to train. A representative example is automated lung cancer diagnosis, where nodule images need to be classified as suspicious or benign. In this work we propose an automatic synthetic lung nodule image generator. Our 3D shape generator is designed to augment the variety of 3D images. Our proposed system takes root in autoencoder techniques, and we provide extensive experimental characterization that demonstrates its ability to produce quality synthetic images.

KEYWORDS
Lung nodules; CT scan; machine learning; 3D image; image generation; autoencoder

1. Introduction

Worldwide in 2017, lung cancer remained the leading cause of cancer deaths (Siegel, Miller, & Jemal, 2017). Computer aided diagnosis, where a software tool analyzes the patient's medical imaging results to suggest a possible diagnosis, is a promising direction: from an input low-resolution 3D CT scan, image processing techniques can be used to classify nodules in the lung scan as potentially cancerous or benign. But such systems require quality 3D training images to ensure the classifiers are adequately trained with sufficient generality. Cancerous lung nodule detection still suffers from a dearth of training images which hampers the ability to effectively automate and improve the analysis of CT scans for cancer risks (Valente et al., 2016). In this work, we propose to address this problem by automatically generating synthetic 3D images of nodules, to augment the training dataset of such systems with meaningful (yet computer-generated) lung nodules images. This is the full length paper for work originally presented at the 3rd International Workshop on Biomedical Informatics with Optimization and Machine Learning in conjuction with International Joint Conference on Artificial Intelligence (IJCAI) (Kommrusch & Pouchet, 2018).

Li et al. showed how to analyze nodules using computed features from the 3D images (such as volume, degree of compactness and irregularity, etc.) (Q. Li, Li, & Doi, 2008). These computed features are then used as inputs to a nodule classification algorithm.

CONTACT Steve Kommrusch. Email: steve.kommrusch@gmail.com
CONTACT Louis-Noel Pouchet. Email: pouchet@colostate.edu

2D lung nodule image generation has been investigated using generative adversarial networks (GANs) (Chuquicusma, Hussein, Burt, & Bagci, 2017), reaching sufficient quality to be classified by radiologists as actual CT scan images. In our work, we aim to generate 3D lung nodule images which match the feature statistics of actual nodules as determined by an analysis program. We propose a new system inspired from autoencoders, and extensively evaluate its generative capabilities. Precisely, we introduce LuNG: a synthetic lung nodule generator, which is a neural network trained to generate new examples of 3D shapes that fit within a broad learned category.

Our work is aimed at creating synthetic images in cases where input images are difficult to get. For example, the Adaptive Lung Nodule Screening Benchmark (ALNSB) from the NSF Center for Domain-Specific Computing (CDSC, 2018) uses a flow that leverages compressive sensing to reconstruct images from low-dose CT scans. These images are slightly different than those built from filtered backprojection, a technique which has more samples readily available (such as LIDC/IDRI (Rong et al., 2017)). To evaluate our results, we integrate our work with the ALNSB system (Shen, Rawat, Pouchet, & Hsu, 2015) that automatically processes a low-dose 3D CT scan, reconstructs a higher-resolution image, isolates all nodules in the 3D image, computes features on them and classifies each nodule as benign or suspicious. We use original patient data to train LuNG, and then use it to generate synthetic nodules that are processed by ALNSB. We create a network which optimizes 3 metrics: (1) increase the percentage of generated images accepted by the nodule analyzer; (2) increase the variation of the generated output images relative to the limited seed images; and (3) decrease the error of the seed images with themselves when input to the autoencoder. We make the following contributions.

- A new 3D image generation system, that can create synthetic images that resemble (in terms of features) the training images. The system is fully implemented and automated.
- Novel metrics which allow for numerical evaluation of 3D image generation aligned with qualitative goals related to lung nodule generation.
- An extensive evaluation of this system to generate 3D images of lung nodules, and its use within an existing computer-aided diagnosis benchmark application.
- The evaluation of iterative training techniques coupled with the ALNSB nodule classifier software, to further refine the quality of the image generator.

The rest of the paper is organized as follows. Sec. 2 briefly motivates our work and design choices. Sec. 3 describes the LuNG system. Extensive experimental evaluation is presented in Sec. 4. Related work is discussed in Sec. 5 before concluding.

2. Motivation

To improve early detection and reduce lung cancer mortality rates, the research community needs to improve lung nodule detection even given low resolution images and a small number of sample images for training. The images have low resolution because low radiation dosages allow for screening to be performed more frequently to aid in early detection, but the low radiation dosage limits the spatial resolution of the image. The number of training samples is small due in part to patient privacy concerns, but is also related to the rate at which new medical technology is being created which generates a need for new training data on the new technology. Our primary goal is to create 3D voxel images that are within the broad class of legal nodule shapes that may be generated from a CT scan.

With the goal of creating improved images for training, we evaluate nodules generated from our trained network using the same software that analyzes the CT scans for lung nodules. Given the availability of 'accepted' new nodules, we test augmenting the training set with these nodules to improve generality of the network. The feedback process we explore includes a nodule reconnection step (to insure final nodules are fully connected in 3D space) followed by a pass through the analyzer which will prune the generated set to keep 3D nodule feature averages close to the original limited training set. The need to avoid overfitting the network for a small set of example images, as well as learning a 3D image category by examples, guided many of the network architecture decisions presented below.

Another goal of our work is to demonstrate the possibility to create a family of images which have *computed characteristics* (e.g., elongation, volume) that fit within a particular distribution range, and in particular which are similar to an observed input nodule image. Hence, in addition to creating a generator network, we shall to create a network that can find the latent feature inputs to the generator aligned with a given known shape. The goal of generating images related to a given input image motivates our inclusion of the reconnection algorithm. Other generative networks will prune illegal outputs as part of their use model (Cummins, Petoumenos, Wang, & Leather, 2017), but we wanted to provide more guarantee of valid images when exploring the feature space near a given sample input. The goal of finding latent feature values for existing images leads naturally to an autoencoder architecture for our neural network.

Generative adversarial networks (GANs) (J. Li et al., 2017) and variational autoencoders (VAEs) (Doersch, 2016) are two sensible approaches to generate synthetic images from a training set, and could intuitively be applied to our problem. However, traditional GANs do not provide a direct mapping from source images into the generator input feature space ((Skymind, 2017)), which limits the ability to generate images similar to a specific input sample or require possibly heavy filtering of "invalid" images produced by the network. In contrast, using an autoencoder-based approach as we develop below allows to better explore the feature space near the known/input images. A system that combines the training of an autoencoder with a discriminator network such as GRASS (J. Li et al., 2017) would allow some of the benefits of GAN to be explored relative to our goals. However, our primary goal is not to create images that emulate the provided training set as determined by a network training loss function. As we show in section 3.4, our goal can be summarized as creating novel images that are within a category acceptable to an automated nodule analyzer. As such, we strive to generate images that are not identical to the source images but fit within a broad category learned by the network.

A similar line of reasoning can be applied to VAEs relative to our goals. Variational autoencoders map the distribution of the input to a generator network in order to allow for exploration of images within a distributional space. In our work, we tightly constrain our latent feature space so that our training images map into the space but the space itself may not match the seed distribution exactly to aid in the production of novel images. Like GANs, there are ways to incorporate VAEs into our framework, and to some extent our proposed approach is a form of variational autoencoder, although with clear differences in both the training and evaluation loop, as developed below. Our work demonstrates one sensible approach for a full end-to-end system to create synthetic 3D images that can effectively cover the feature space of 3D lung nodules reconstructed via compressive sensing.

3. The LuNG System

The LuNG project is based on a neural network trained to produce realistic 3D lung nodules from a small set of seed examples to help improve automated cancer screening. To provide a broader range of legal images to train the autoencoder, guided training is used in which each nodule is modified to create 15 additional training samples. We call the initial nodule set, of which we were provided 51 samples, the 'seed' nodules. The 'base' nodules include 15 modified samples per seed nodule for a total of 816 samples. The base nodules are used to train an autoencoder neural network with 3 latent feature neurons in the bottleneck layer. The output of the autoencoder goes through a cleanup algorithm to increase the likelihood that viable fully connected nodules are being generated. A nodule analyzer program then extracts relevant 3D features from the nodules and prunes away nodules outside the range of interesting feature values. We use the ALNSB (Shen et al., 2015) nodule analyzer and classifier code for the LuNG project, but similar analyzers compute similar 3D features to aid in classification. The accepted nodules are the final output of LuNG for use in classifier training or medical evaluation. Given this set of generated images which have been accepted by the analyzer, we explore adding them to the autoencoder training set to improve the generality of the generator. We explore having a single generator network or 2 networks that exchange training samples which have been validated by the analyzer as legal nodule examples.

The code and dataset for the LuNG project is available to reviewers upon request.

3.1. *Input images*

The input dataset comes from the Automatic Lung Screening Benchmark (ALNSB) (Shen et al., 2015), produced by the NSF Center for Domain-Specific Computing (CDSC, 2018). This pipeline, shown in figure 1, is targeting the automatic reconstruction of 3D CT scans obtained with reduced (low-dose) radiation, i.e., reducing the number of samples taken by the machine. Compressive sensing is used to reconstruct the initial 3D image of a lung, including all artifacts such as airways, etc. A series of image processing steps are performed to isolate all 3D nodules that could lie along the tissue in lungs. Then, each of the candidate nodules is quantified to obtain a series of domain-specific metrics, such as elongation, volume, 2D projection size, minimal width/length in any dimension, etc. An initial filtering is done based on static criteria (e.g., volume greater than 4mm^3) to prune nodules that are not suspicious for potential cancerous origin. Eventually, the remaining nodules are fed to a trained SVM-based classifier, which classifies the nodules as potentially cancerous or not. The end goal of this process is to trigger a high-resolution scan for only the regions containing suspicious nodules, while the patient is still on the table.

This processing pipeline is extremely specific regarding both the method used to obtain and reconstruct the image, via compressive sensing, and the filtering imposed by radiologists regarding the nodule metrics and their values about potentially cancerous nodules. In this paper, we operate with a single-patient data as input, that is, a single 3D lung scan. About 2000 nodules are extracted from this single scan, but out of which only 51 are true candidates for the classifier. Our objective is to create a family of nodules that are also acceptable inputs to the classifier (i.e., which have not been dismissed early on based on simple thresholds on the nodule metrics), from these 51 images as a starting point. We believe this represents a worst-case scenario where the lack of input images is not even sufficient to adequately train a simple classifier, and is therefore a sensible scenario to demonstrate our approach to generating images within a specific acceptable feature distribution.

These 51 seed images seed the network with the general nodules shape that we wish to generate new nodules from. Based on the largest of these 51 images, we set our input image size to 25mm by 28mm by 28mm, which also aligns with other nodule studies (Q. Li et al., 2008). The voxel size from the image processing pipeline is 1.25mm by 0.7mm by 0.7mm, so our input nodules are 20x40x40 voxels. This results in an input to the autoencoder with 32,000 voxel values which can range from 0 to 1.

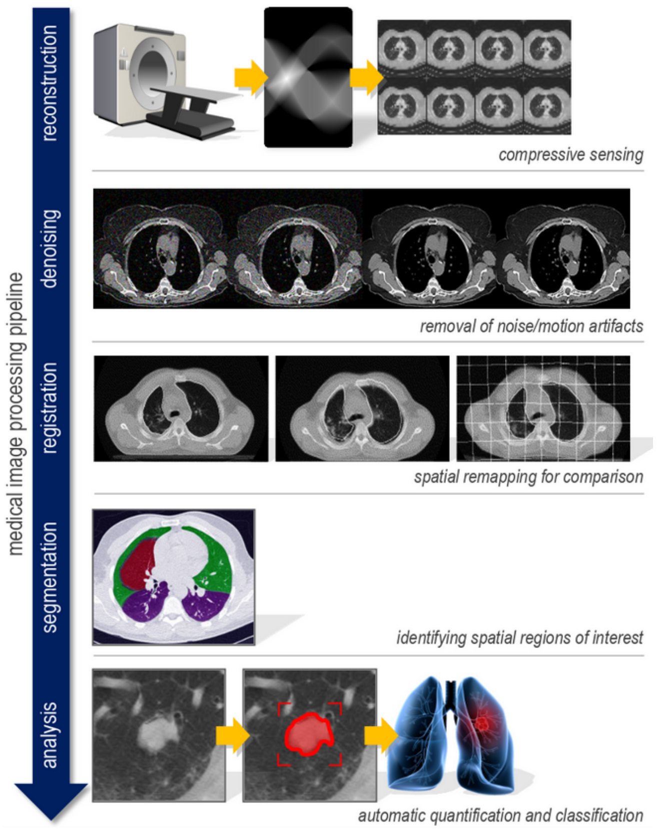

medical image processing pipeline
reconstruction
denoising
registration
segmentation
analysis
compressive sensing
removal of noise/motion artifacts
spatial remapping for comparison
identifying spatial regions of interest
automatic quantification and classification

Figure 1. Medical image processing pipeline

Figure 2 shows 6 of the 51 seed images from the CT scan. Each of the 51 seed images is centered in the 20x40x40 space although when an image has a distinct curve it is possible for the center voxels to be off (no object detected at that location). One of our nodules was slightly too wide and 21 out of 1290 total voxels were clipped; all other nodules fit fully within the training size. From an original set of 51 images, 816 are generated: 8 copies of each nodule are the 8 possible reflections in X,Y, and Z of the original; and 8 copies are the X,Y, and Z reflections of the original shifted by 0.5 pixels in X and Y. The reflections are still representative of legal nodule shapes (and the analyzer code recognizes them as legal), so it improves the generality of the autoencoder to have them included. The 0.5-pixel shift also aids generalization of the network by training it to tolerate fuzzy edges and less precise pixel values. We do not do any resizing of the images as we found through early testing that providing the full voxel data from the image pipeline resulted in better generated images than downsampling images to a smaller autoencoder and upsampling the final image for

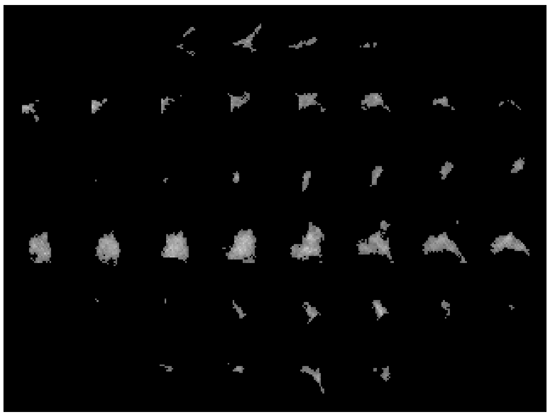


Figure 2. Six of the 51 seed nodules showing the middle 8 2D slices of the 3D image from the CT scan

use with the analyzer.

Our initial 51 seed images include 2 nodules that are classified as suspicious. These 2 nodules become 32 nodules in our base training set after reflections and offset, but still provide us with a limited example of potentially cancerous nodules. A primary goal of the LuNG system is to create a wider variety of images for use in classification based on learning a nodule feature space from the full set of 51 input images.

3.2. *Autoencoder network*

Figure 3 shows the autoencoder structure as well as the feature and generator networks that are derived from it. All internal layers use tanh for non-linearity, which results in a range of -1 to 1 for our latent feature space. The final layer of the autoencoder uses a sigmoid function to keep the output within the 0 to 1 range that we are targeting for voxel values.

We experimented with various sizes for our network and various methods of providing image feedback from the analyzer with results shown in sections 4.1 and 4.2. The network shown in figure 3 had the best overall score.

Our autoencoder is trained initially with the 816 images in our base set. We use Adam (Kingma & Ba, 2014) for stochastic optimization to minimize the mean squared error of the generated 32,000 voxel 3D images. After creating a well-trained autoencoder, the network can be split into feature and generator networks. The feature network can be used to map actual nodules into a latent feature space so that feature space steps from one nodule to another can create novel nodule images using the generator network. If this stepping is done between a nodule suspected as cancerous and another suspected to be non-cancerous, a skilled neurologist could identify the shape at which the original suspicious nodule would not be considered suspicious to help train and improve an automated classifier. The generator network can also be used to generate fully random images for improving a classifier. For our random generation experiments we use uniform values from -1 to 1 as inputs for the 3 latent feature dimensions. We explore the reasons and benefits of this random distribution in section 4.3.

The autoencoder structure which yielded best results is not symmetric in that there are fewer neurons and layers before the bottleneck layer than after. The benefit of a small input network is based on the goal of avoiding overfitting as well as reference to Haykin's discussion of using a single layer to extract principle components (Haykin,

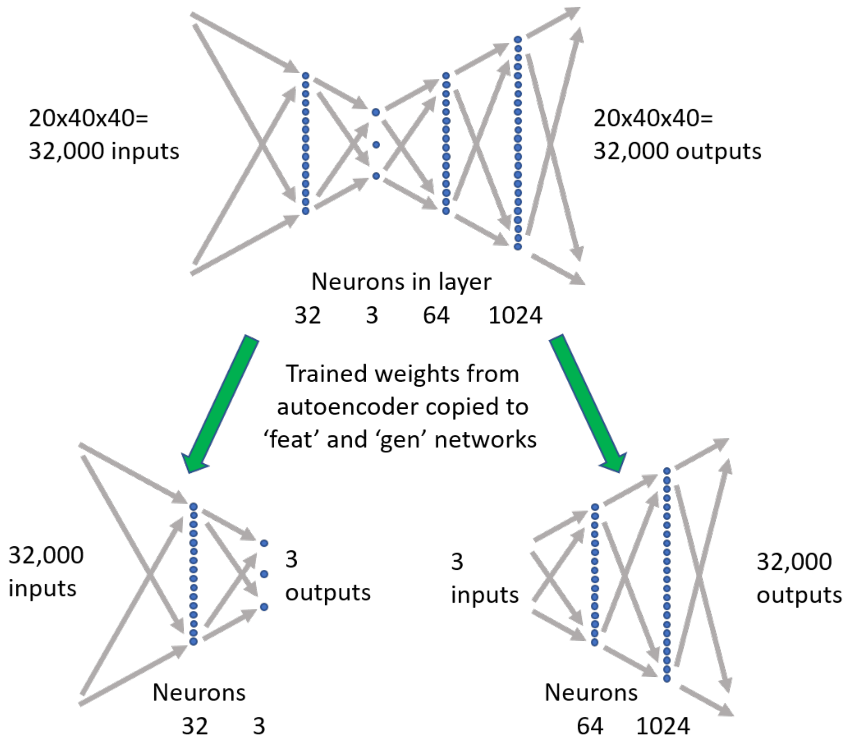


Figure 3. Autoencoder and derived feature/generator networks for nodules

2009). Our feature network is 2 layers, not one, so that it is capable of finding a non-linear mapping from the input shape into the 3 latent feature neurons.

3.3. *Reconnection algorithm*

The autoencoder was trained on single component nodules in that all the 'on' voxels for the nodule were connected in a single 3D shape. The variation produced by trained generator networks did not always result in a single component, and it is common for generative networks that have a technical constraint to discard output which fails to meet the requirements (Cummins et al., 2017). However, for the use case of exploring the feature space near a known image, we chose to add a reconnection algorithm to our output nodules to minimize illegal outputs. This algorithm insures that for any input to the generative network, a fully-connected nodule is generated.

When the generator network creates an image, most of the time a single fully-connected component is generated and the reconnection algorithm does not need to be invoked. In the rare case that no voxels are detected as 'on' the threshold for turning on a voxel is lowered until some voxels are set. In the case where multiple components are detected, the algorithm will find unset voxels to connect the nodules. First the algorithm will search through all empty voxels to check if setting that single voxel would connect 2 components, if so it will set at least one such voxel. If there are still multiple components, the network will find the voxels mid-way along a short path between 2 components, set them, and then restart the algorithm. These new voxels mid-way between components are sometimes new components themselves, which may be connectable with the first pass voxel setting or may require yet another iteration to set midpoint voxels. The process is iterated until only 1 component remains.

3.4. *Metrics for nodule analyzer acceptance and results scoring*

The nodule analyzer and classifier computes twelve 3D feature values for each nodule (features such as 3D volume, surface-to-volume ratio, and other data useful for classification). Our statistical approach to this data is related to Mahalanobis distances (Mahalanobis, 1936), hence we compute the average and standard deviations on these 12 features. Random nodules from the generator are fed into the classifier code and accepted to produce similar average feature values. This accepted set of images is the most useful image set for further analysis or use in classifier training. We test augmenting the network training set by using these same accepted nodules and running more training iterations on the network.

Metrics for analyzer acceptance of the images: Using the average and standard deviation values we create a distance metric d based on concepts similar to the Mahalanobis distance. Given S is the set of 51 seed nodules and i is the index for one of 12 features, μ_{Si} is the average value of feature i and σ_{Si} is the standard deviation. Given Y is the set of output nodules from LuNG, the running average for feature i of the nodules being analyzed is $\bar{Y_i}$. Given feature i of a nodule y is y_i then if either ($y_i \geq \mu_{Si}$ and $\bar{Y_i} \leq \mu_{Si}$) or ($y_i \leq \mu_{Si}$ and $\bar{Y_i} \geq \mu_{Si}$), then the nodule is accepted as it helps $\bar{Y_i}$ trend towards μ_{Si}. In cases where the nodule's y_i moves $\bar{Y_i}$ away from μ_{Si}, we compute a weighted distance d from μ_{Si} in multiples of σ_{Si} using:

$$d = |\frac{y_i + 3 * \bar{Y_i} - 4 * \mu_{Si}}{\sigma_{Si}}|$$

Note that d is zero if $y_i = \mu_{Si}$ and $\bar{Y_i} = \mu_{Si}$ and increases as y_i and $\bar{Y_i}$ move in the same direction away from μ_{Si}. We compute the probability of keeping a nodule y as P_{keep} which drops as d increases:

$$P_{keep} = \begin{cases} 1 & \text{if } y_i = \mu_{Si} \\ 1 & \text{if } y_i > \mu_{Si} \text{ and } (\bar{Y_i} \leq \mu_{Si} \text{ or } d \leq 3) \\ 0.7 + \frac{0.9}{d} & \text{if } y_i > \mu_{Si} \text{ and } \bar{Y_i} > \mu_{Si} \text{ and } d > 3 \\ 1 & \text{if } y_i < \mu_{Si} \text{ and } (\bar{Y_i} \geq \mu_{Si} \text{ or } d \leq 3) \\ 0.7 + \frac{0.9}{d} & \text{if } y_i < \mu_{Si} \text{ and } \bar{Y_i} < \mu_{Si} \text{ and } d > 3 \end{cases}$$

The specific numerical values used for computing d and P_{keep} were chosen to maximize the number of the original dataset which are accepted by this process while limiting the deviation from the seed features allowed by the generator. When selecting from the 816 base nodules derived from the original 51, 95% were accepted. Acceptance results from the nodules generated by a trained network are provided in section 4.

Metrics for scoring the accepted image set: Our goals for LuNG are to generate images that have a high acceptance rate for the analyzer and a high variation relative to the seed images while minimizing the error of the network when a seed image is reproduced. We track the acceptance rate simply as the percentage of randomly generated nodules that are accepted by the analyzer. For a metric of variation, we compute a feature distance *FtDist* based on the 12 3D image features used in the analyzer. To track how well the distribution of output images matches the seed image variation, we compute a *FtMMSE* based on the distribution means. The ability of the network to reproduce a given seed image is tracked with the mean squared error of the image output voxels, as is typical for autoencoder image training.

FtDist has some similarity to Mahalanobis distance, but finds the average over all the accepted images of the distance from the image to the closest seed image in the 12-dimensional analyzer feature space. As *FtDist* increases, the network is generating images that are less similar to specific samples in the seed images, hence it is a metric we want to increase with LuNG. Given an accepted set of n images Y and a set of 51 seed images S, and given y_i denotes the value of feature i for an image and σ_{Si} denotes the standard deviation of feature i within S:

$$FtDist = 1/n \sum_{y \in Y} \min_{s \in S} \sqrt{\sum_{i=1}^{12} (\frac{y_i - s_i}{\sigma_{Si}})^2}$$

FtMMSE tracks how much the 12 3D features have the same mean between the accepted set of images X and the seed images S. As *FtMMSE* increases, the network is generating average images that are increasing outside the typical seed image distribution, hence it is a metric we want to decrease with LuNG. Given μ_{Si} is the mean of feature i in the set of seed images and μ_{Yi} is the mean of feature i in the final set of accepted images:

$$FtMMSE = 1/12 \sum_{i=1}^{12} (\mu_{Yi} - \mu_{Si})^2$$

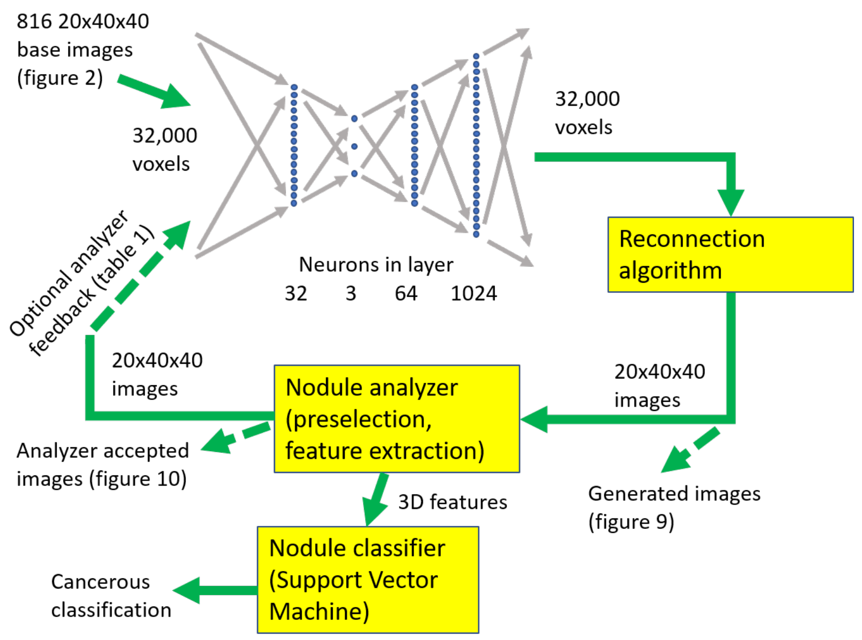
Score is our composite network scoring metric used to compare different networks, hyperparameters, feedback options, and reconnection options. In addition to *FtDist* and *FtMMSE*, we use *AC*, which is the fraction of generated images which the analyzer accepted, and *MSE* which is the traditional mean squared error which results when the autoencoder is used to regenerate the 51 seed nodule images.

$$Score = \frac{FtDist - 1}{(FtMMSE + 0.1) * (MSE + 0.1) * (1 - AC)}$$

Score increases as *FtDist* or *AC* increase and decreases when *FtMMSE* or *MSE* increase. The constants in the equation are based on qualitative assessments of network results; for example, using *MSE* + 0.1 means that *MSE* values below 0.1 don't override the contribution of other components and mathematically aligns with the qualitative statement that an *MSE* of 0.1 yielded acceptable images visually in comparison with the seed images.

Results using *FtDist*, *FtMMSE*, and *Score* to evaluate networks and LuNG interface features are shown in tables 1 and 2 and figures 5 and 6, and discussed further is section 4. Our use of *Score* to evaluate the entire nodule generation process rates the quality of the random input distribution, the generator network, the reconnection algorithm, and the analyzer acceptance and the interaction of these components into a system. Our use of the analyzer acceptance rate is similar in some functional respects to the discriminator network in a GAN as both techniques are used to identify network outputs as being inside or outside an acceptable distribution.

3.5. *Updating the training set*



816 20x40x40 base images (figure 2)
32,000 voxels
Optional analyzer feedback (table 1)
Neurons in layer
32 3 64 1024
32,000 voxels
Reconnection algorithm
20x40x40 images
20x40x40 images
Nodule analyzer (preselection, feature extraction)
Analyzer accepted images (figure 10)
Generated images (figure 9)
3D features
Nodule classifier (Support Vector Machine)
Cancerous classification

Figure 4. Interaction between trained autoencoder and nodule analyzer. The images from figure 2 are always part of the training set to the autoencoder. The reconnected images after the network can be seen in figure 9. The analyzer accepted output of LuNG can be seen in figure 10.

After a trained generator network produces images which are reconnected and validated by the nodule analyzer, a new training set may optionally be created for the next round of training iterations of the autoencoder. We explored training performance with and without this feedback during training rounds. Figure 4 diagrams the data loop between the autoencoder and the nodule analyzer. We explored 4 approaches for creating the augmented training set, but ultimately found that our best results came from proper autoencoder sizing and training with the 816 base images created by adding guided training examples to the original 51 seed images. Although the image feedback into the training set did not improve the LuNG system, we include the details and results from it to explain the drawbacks of the approach.

We were motivated to explore these 4 approaches because we wanted to learn if generated nodules for training set augmentation could improve network results. One of the approaches used images from one trained network to augment a second trained network, similar to the multi-adversarial networks discussed in (Durugkar, Gemp, & Mahadevan, 2016). The intent of this approach is to improve the breadth of the feature space used to represent all legal nodules by importing images that were not generated by the network being trained.

The feedback approach we considered the best of the 4 is analyzed in some detail in the results section. In this approach, we train the network for 50,000 iterations then generate 400 output images. We pass those images through the analyzer to insure they are considered legal and chose a single random reflection of each image to add to the training set. The image can be reflected in one or more of X, Y, and Z and is never fed unreflected into the training. In this way, the number of novel images is limited to the group accepted, but the images are likely not an output that the network parameters have already settled on for a legal output. Because we are feeding in the image reflection, we did not find added value in having 2 networks trade generated images - the network itself was generating images whose reflections were novel to both its own feature network as input and its generator network as output. This training feedback is only done for 2 rounds of 25,000 training iterations and then a final 50,000 training iterations train only on the 816 base images to improve original image alignment.

These 4 approaches explored whether having some training image variation helps fill the bottleneck neuron latent feature space with more variation on legal images. The intent of testing 4 approaches is to learn if an analyzer feedback behavior can be found that improves the criteria LuNG is trying to achieve: novel images accepted by the analyzer.

4. Experimental results

Using the *FtDist*, *FtMMSE*, and *Score* metrics introduced in section 3.4, we evaluated various network sizes, the analyzer feedback approaches discussed in section 3.5, and the reconnection algorithm discussed in section 3.3. Our naming for networks is given with underscores separating neuron counts between the fully connected layers, so 32_3_64_1024 is a network with 4 hidden layers that have 32, 3, 64, and 1024 neurons respectively. As seen by referencing tables 1 and 2, depending on the metric which is rated as most important, different architectures would be recommended.

4.1. *Results for reconnection, feedback options, and network depth*

Table 1 shows training results for some of the feedback options discussed in section 3.5. This table includes networks that allowed 'illegal' outputs (the reconnection method discussed in section 3.3 was not used), which limited the number of images accepted by the analyzer. The MSE column shows 1000 times the mean squared error per voxel for the autoencoder when given the 51 original seed nodules as inputs and targets. Instead of using solely the MSE column to evaluate our network quality, we were interested in all 4 columns because they each contain information relevant to creating quality artificial nodules. The "AC%" column shows what percentage of 400 images randomly generated by the generator network were accepted by the analyzer. The "FtDist" column shows the average minimum distance in analyzer feature space from any image to a seed image. The "FtMMSE" column shows the average mean squared error of all 12 analyzer features between the images and the 51 seed images. "No reflect" is one of our feedback options referring to using the accepted images from the analyzer directly in the training set. "multi" feedback refers to using 2 autoencoders and having each autoencoder use the accepted images that were output by the other. Using this table and other early results, we observed that the network with no analyzer feedback had overall good metrics, although the FtDist column indicating the novelty of images generated was lower than we would prefer, so we weighed *FtDist* heavier in our final scoring of networks as we explored network sizing.

Table 2 shows experiments in which we always reconnect the nodules for network output (see section 3.3). Using this approach, the analyzer always has a full set of

Table 1. Key metrics for networks which allowed 'illegal' output patterns as input to the analyzer.

Network parameter testing (2 run average after 6 rounds)				
Parameters	AC%	MSE	FtDist	FtMMSE
16_4_64_256_1024 No Feedback	54	0.03	1.75	0.07
16_4_64_256_1024 FB: no reflect	44	0.03	2.23	0.21
16_4_64_256_1024 FB: no reflect, multi	36	0.05	2.43	0.26
16_4_64_256_1024 FB: 4 reflects, multi	29	0.04	2.68	0.45

Table 2. Key metrics for networks which always produced legal inputs to the analyzer.

Network parameter testing (2 run average after 6 rounds)					
Parameters	AC%	MSE	FtDist	Clean	Invert
64_4_64_1024 No Feedback	85	0.08	1.78	109	0
64_4_64_1024 FB: 1 reflect	64	0.06	3.13	63	0
64_4_64_256_1024 No Feedback	80	0.02	1.96	117	2
64_4_64_256_1024 FB: 1 reflect	61	0.03	4.11	77	6.5

400 images to consider for acceptance, leading to higher acceptance rates. This table includes data on the number of raw generator output images which were clean when generated (one fully connected component) and the number that were inverted (white background with black nodule shape). The fact that deeper generation networks sometimes resulted in inverted output images is an indication that they have too many degrees of freedom and contributed to the decision to limit the depth of our autoencoder. The "1 reflect" feedback label refers to having a single reflected copy of each accepted image used to train the autoencoder for 2 of the 6 rounds. This "1 reflect" feedback was our most promising approach as described in section 3.5.

From the results in these 2 tables and other similar experiments, we concluded that the approach in section 3.5, which used analyzer feedback for 2 of the 6 training rounds, had the best general results of the 4 feedback approaches considered. Also, the approach in section 3.3, which will reconnect and repair generator network outputs, yielded 3D images preferable to the legal subset left when the algorithm was not applied. The results of these explorations informed the final constants that we used to create the *Score* metric for rating networks as described in section 3.4.

4.2. *Results for tuning network sizes*

Given the *Score* equation, and the observation that small networks yielded preferable results, we found that an output layer of 1024 was optimal compared to larger or smaller values. We analyzed neuron counts in other layers as summarized in figures 5 and 6. As can be seen, from the networks we studied, the network which yielded the highest score of 176 was 32_3_64_1024, which is the network used to generate the nodule images shown in section 4.4.

Our final network can train on our data in an acceptable amount of time. Even though our experiments gathered significant intermediate data to allow for image feedback during training, the final 32_3_64_1024 network can be trained in approximately 2 hours. Our system for training has 6 Intel Xeon E5-1650 CPUs (12 threads) at 3.6GHz,

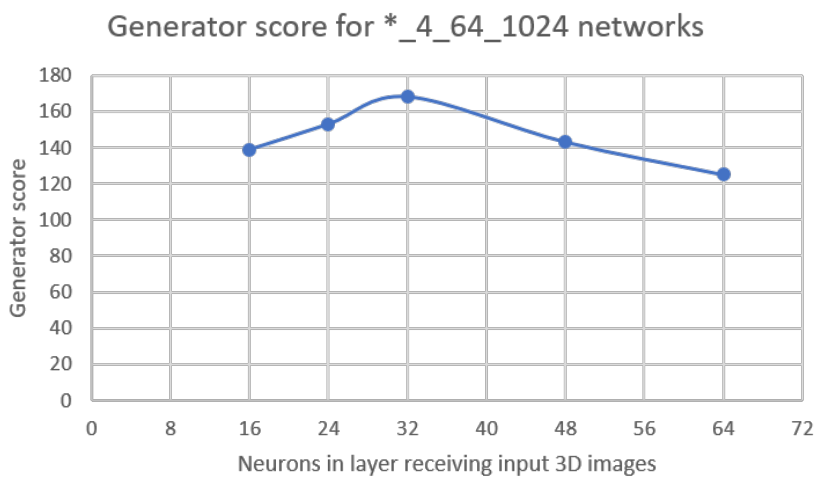


Figure 5. Scoring autoencoder networks of varying input neuron counts.

an Nvidia GeForce GTX 1060 6GB GPU (1280 CUDA cores at 1.86GHz), and 16GB of RAM. Those 2 hours (120 minutes) break down as: 10 minutes for creation of 816 base images from 51 seed images, 80 minutes to train for 150,000 epochs on the images, 20 minutes to generate and connect 400 nodules, and 10 minutes to run the analyzer on the nodules. Code tuning would be able to improve the image processing parts of that time, but the training was done using PyTorch (Paszke, Gross, Chintala, & Chanan, 2017) on the GPU and is already highly optimized. When used for generating images for practical use, we would recommend training 2 or more networks and using the results from the network that achieved the higher score.

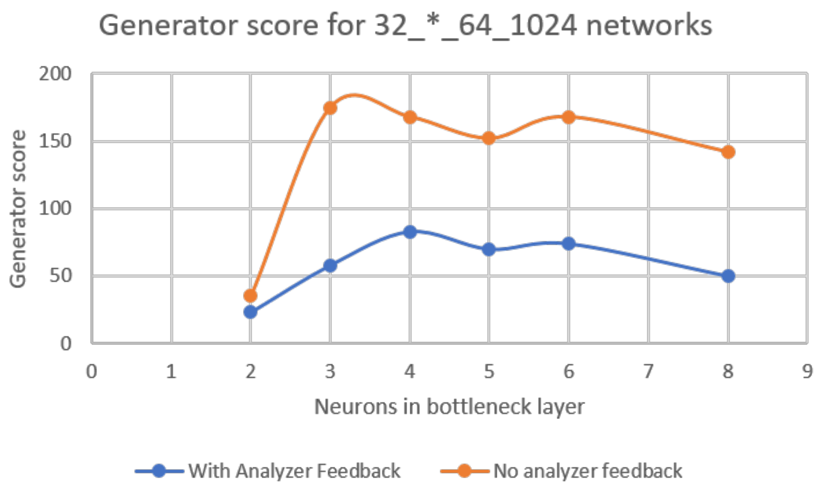


Figure 6. This figure compares results between a network that used 816 base images with no analyzer feedback for 150,000 iterations of training and a network that trained for 25,000 iterations on the base images, then added 302 generated nodules to train for 25,000 iterations, then added a different 199 generated nodules to train for 25,000 iterations, and then trained for a final 75,000 iterations with no feedback.

Figure 7 shows the components of the score for the final parameter analysis we did on the network. Note that the MSE metric (mean squared error of the network on training set) continues to decrease with larger networks, but the score we are optimizing occurs with 3 bottleneck latent feature neurons. As shown in the top and bottom images in figure 9, our network is able to learn and reproduce images from the training set. Our intuition is that limiting our network the 3 bottleneck neurons results in most of the avaible degrees of freedom being required for proper image encoding which results in our -1 to 1 uniform random distrubition creating a variety of acceptable images. As such, the prior distribution for our generative network is this uniform distribution. The *Score* metric helps us to tune the system such that we do not require VAE techniques to constrain our random image generation process, although such techniques may be a valuable path for future research.

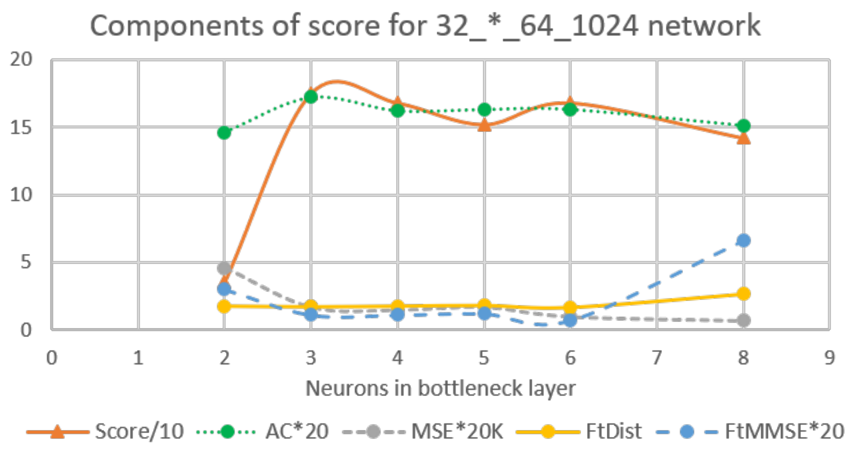


Figure 7. There are 4 components used to compute the network score. The component values are scaled as shown so that they can all be plotted on the same scale.

4.3. *Latent Feature Results*

To visualize the representation of the seed nodules within the autoencoder network, we plot the 51 seed nodule positions in the latent feature space represented by the neurons in the bottleneck layer in figure 8. To save space, we only present 2 of the 3 neurons for the final trained network, but we compare their positions to 2 of the 8 neurons from a trained network with 8 bottleneck neurons.

For the network with 3 bottleneck neurons, the plot shows that the 51 seed nodules are relatively well distributed in the 4 quadrants of the plot and the full range of both neurons is used to represent all the input images. For the network with 8 bottleneck neurons, most of the seed nodules map to the left half-plane in the plot and the full range of the 2 neurons are not used. This is a symptom of having a network with more degrees of freedom than needed to represent the nodule training space and this plot helps visualize the weaknesses of too many bottleneck neurons. Such weaknesses contribute to the high *FtMMSE* measurements shown in figure 7 for a network with 8 bottleneck neurons. The images also show how a random -1 to 1 uniform range for our generator interacts with the bottleneck layer. We want the full range of bottleneck layer values to be viable images and so our acceptance rate tracked as part of the score will correlate with the fraction of the bottleneck layer that creates acceptable images.

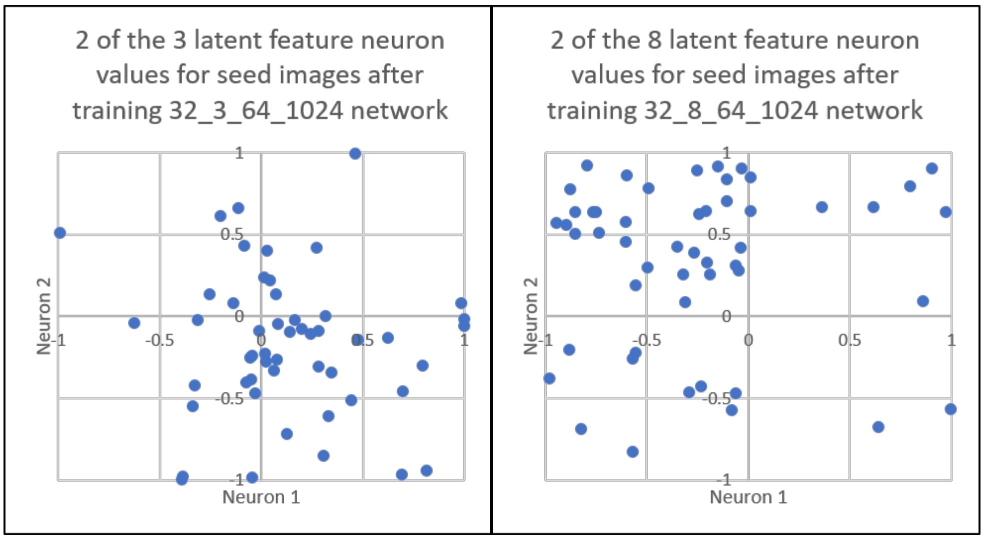


Figure 8. Distribution of images in feature space after training with no feedback

4.4. *Image results*

The images shown in this section are from a network with 4 hidden layers with 32, 3, 64, and 1024 neurons. The network was trained without analyzer feedback and the output is processed to guarantee fully connected nodules.

The quality of 3D nodules our network can produce is shown in figure 9 with 6 steps through the 3D bottleneck neuron latent feature space starting at the 2nd nodule from figure 2 and ending at the 4th nodule. First, note that the learned images for the start and end nodules are very similar to the 2nd and 4th input images, validating the MSE data that the network is correctly learning the seed nodules. The 4 internal step images have some relation to the start and end images, but depending on the distance between the 2 nodules in latent feature space a variety of shapes may be involved in the steps.

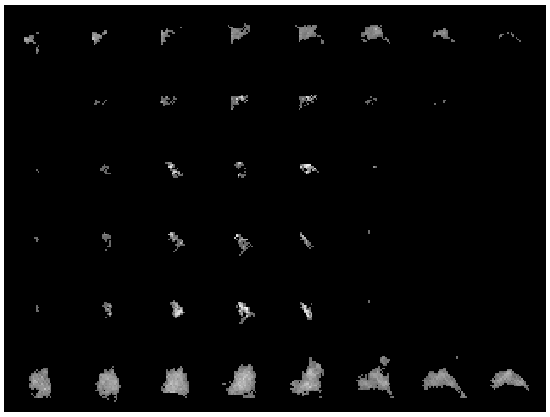


Figure 9. 6 steps through 3D latent feature space between original nodules 2 and 4 from figure 2

Figure 10 shows 6 images generated by randomly selecting 3 values between -1 and 1 for the latent feature inputs to the generator network and then being processed by the analyzer to determine acceptance. When using the network to randomly generate nodules (for classification by a trained specialist or training automated classifiers), this is an example of quality final results.

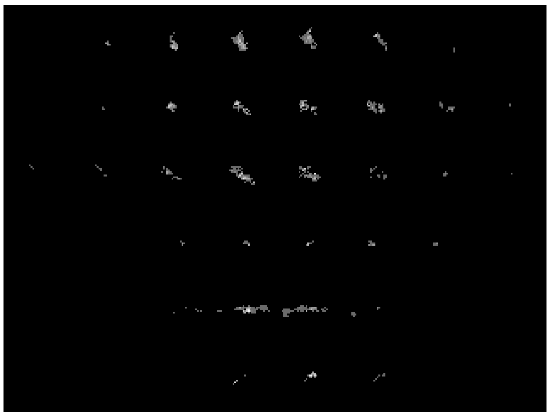


Figure 10. 6 images generated using uniform distribution from 3D feature space after passing nodule analyzer

4.5. *Image quality for training*

Figure 11 shows the 12 features that are used by the nodule analyzer and demonstrates another key success of our full approach. Characteristics like volume, surface area, Euler number (a metric related to concavity of the shape) and other values are used. We normalized the mean and standard deviation of each feature to 1.0 and the chart shows that the average and standard deviation of the generated nodules for all 12 features stays relatively close to 1 for the network with no analyzer image feedback. When no feedback is used, as we propose, the average of all 12 features shows that in aggregate the features from the generated nodules are close to the true seed nodules. However, when feedback is used, one can see that the nodule features which have some deviation from the mean get amplified (even though the analyzer tries to accept nodules in a way that maintains the same mean). For example, "surface area^3/volume^2" is a measure of the compactness of a shape; the generated images from the network with no feedback tended to have higher surface area to volume than the seed images, and when these images were used for further training the generated images had a mean that was about 2.6 times higher than the seed images and a much higher standard deviation.

The ALNSB classifier we interact with uses a support vector machine to map images onto a 2D space representing distances to positive (suspicious) or negative (non-cancerous) centroids. Figure 12 shows the positive and negative centroid distances for the seed data and 1000 samples of analyzer accepted generated data. Nodules that have a smaller distance to the positive centroid than to the negative centroid are classified as likely cancerous. The general distribution of generated images fits the shape of the seed images rather well, and there is a collection of nodules being generated near the decision line between cancerous and non-cancerous, allowing for improved training based on operator classification of the nodules. Even though the original seed dataset only included 2 nodules eventually classified as potentially cancerous, our approach is able to use shape data from all 51 nodules to create novel images that can be useful for improving an automated screening system. Note that these centroid distances themselves are not part of the 12 features that are used to filter the nodules, so this figure validates our approach to create images usable for further automated classification work.

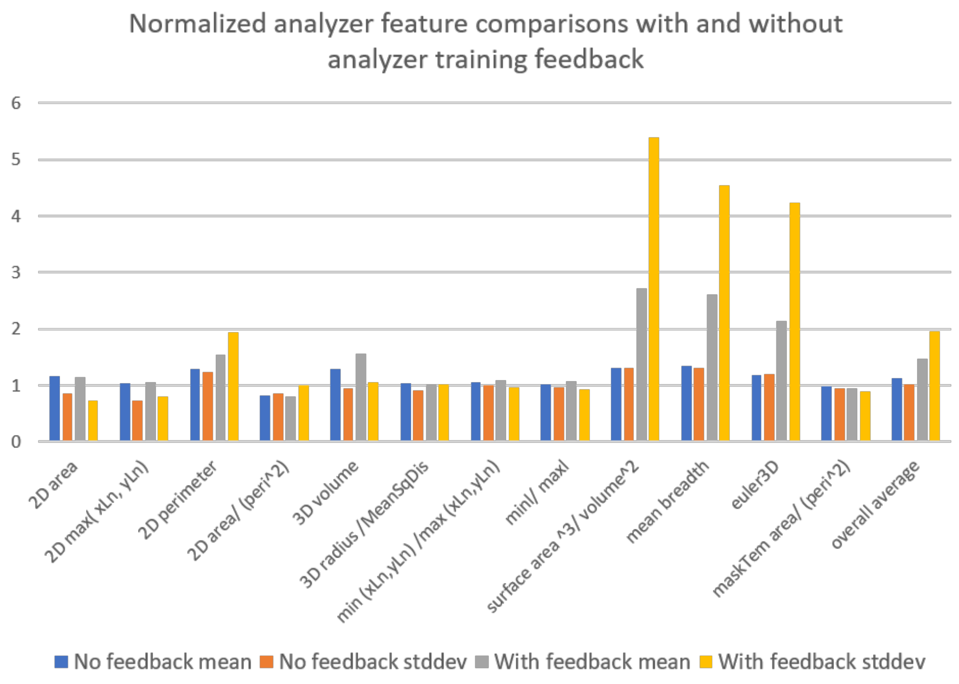


Figure 11. Computed 3D features of nodules analyzed for input to classifier. Our proposed method avoids the deviations shown by the grey and yellow bars.

5. Related work

Improving automated CT lung nodule classification techniques and 3D image generation are areas that are receiving significant research attention.

Recently, Valente et al. provided a good overview of the requirements for CADe (Computer Aided Detection) systems in medical radiology and they evaluate the status of recent approaches (Valente et al., 2016). Our aim is to provide a tool which can be used to improve the results of such CADe systems by both increasing the true positive rate (sensitivity) and decreasing the false positive rate of CADe classifiers through the use of an increase in nodules for analysis and training. Their survey paper discusses in detail the preprocessing, segmentation, and nodule detection steps similar to those used in the ALNSB nodule analyzer/classifier which we used in this project.

Li et. al provide an excellent overview of recent approaches to 3D shape generation in their paper ”GRASS: Generative Recursive Autoencoders for Shape Structures” (J. Li et al., 2017). While we do not explore the design of an autoencoder with convolutional and deconvolutional layers, the same image generation quality metrics that we teach could be used to evaluate such designs. Similar tradeoffs between overfitting and low error rates with seed images would have to be considered when setting the depth of the network and number of feature maps in the convolutional layers.

Durugkar et al. describe the challenges of training GANs well and discuss the advantages of multiple generative networks trained with multiple adversaries to improve the quality of images generated (Durugkar et al., 2016). LuNG explored using multiple networks during image feedback experiments. Larsen et al. (Boesen Lindbo Larsen, Kaae Sønderby, Larochelle, & Winther, 2015) teach a system which combines an autoencoder with a GAN which could be a basis for future work introducing GAN methodologies into the LuNG system by preserving our goal of generating shapes

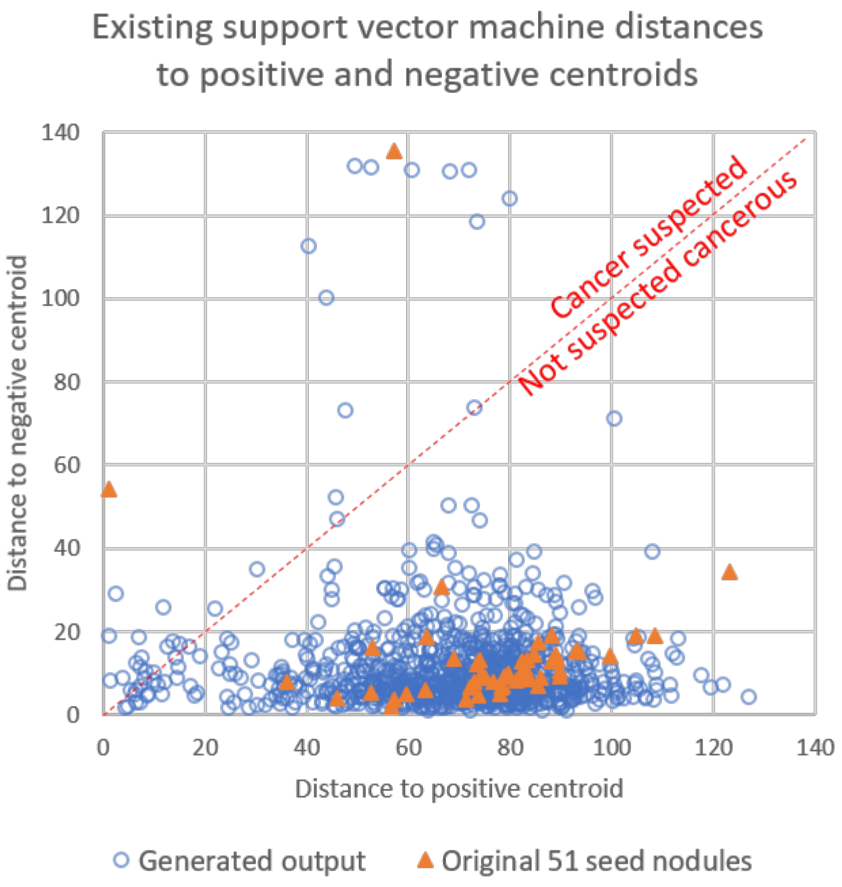


Figure 12. Classifier distances to positive and negative centroids of SVM for 1000 analyzer accepted network generated samples. (Nodules closer to positive than negative centroid after support vectors are applied are more likely cancerous).

similar to existing seed shapes.

6. Conclusion

To produce quality image classifiers, machine learning requires a large set of training images. This poses a challenge for application areas where such training sets are rare if they exist, such as for computer-aided diagnosis of cancerous lung nodules.

In this work we developed LuNG, a lung nodule image generator, allowing us to augment the training dataset of image classifiers with meaningful (yet computer-generated) lung nodule images. Specifically, we have developed an autoencoder-based system that learns to produce 3D images that resembles the original training set, while covering adequately the feature space. Our tool, LuNG, was developed using PyTorch and is fully implemented. We have shown that the 3D nodules generated by this process visually and numerically align well with the general image space presented by the limited set of seed images.

References

Boesen Lindbo Larsen, A., Kaae Sønderby, S., Larochelle, H., & Winther, O. (2015, December). Autoencoding beyond pixels using a learned similarity metric. *ArXiv e-prints*.

CDSC. (2018). *NSF Center for Domain-Specific Computing.* (https://cdsc.ucla.edu)

Chuquicusma, M. J. M., Hussein, S., Burt, J., & Bagci, U. (2017, October). How to Fool Radiologists with Generative Adversarial Networks? A Visual Turing Test for Lung Cancer Diagnosis. *ArXiv e-prints*.

Cummins, C., Petoumenos, P., Wang, Z., & Leather, H. (2017). Synthesizing benchmarks for predictive modeling. In *Proceedings of the 2017 international symposium on code generation and optimization* (pp. 86–99). Piscataway, NJ, USA: IEEE Press. Retrieved from `http://dl.acm.org/citation.cfm?id=3049832.3049843`

Doersch, C. (2016, June). "Tutorial on Variational Autoencoders". *ArXiv e-prints*.

Durugkar, I., Gemp, I., & Mahadevan, S. (2016, November). Generative Multi-Adversarial Networks. *ArXiv e-prints*.

Haykin, S. S. (2009). *Neural networks and learning machines* (Third ed.). Upper Saddle River, NJ: Pearson Education.

Kingma, D. P., & Ba, J. (2014, December). Adam: A method for stochastic optimization. *ArXiv e-prints*.

Kommrusch, S., & Pouchet, L.-N. (2018). Synthetic lung nodule 3d image generation using autoencoders. *3rd International Workshop on Biomedical Informatics with Optimization and Machine Learning in conjuction with IJCAI*. Retrieved from `https://www.ijcai-boom.org/proceeding.html`

Li, J., Xu, K., Chaudhuri, S., Yumer, E., Zhang, H., & Guibas, L. (2017). GRASS: Generative Recursive Autoencoders for Shape Structures. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2017)*, *36*(4), to appear.

Li, Q., Li, F., & Doi, K. (2008, Feb). Computerized detection of lung nodules in thin-section CT images by use of selective enhancement filters and an automated rule-based classifier. *Acad Radiol*, *15*(2), 165–175.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. In *Proceedings of the National Institute of Sciences of India* (Vol. 2, p. 49-55).

Paszke, A., Gross, S., Chintala, S., & Chanan, G. (2017). *PyTorch.* `https://github.com/pytorch/pytorch`. GitHub.

Rong, J., Gao, P., Liu, W., Zhang, Y., Liu, T., & Lu, H. (2017, March). Computer simulation of low-dose CT with clinical lung image database: a preliminary study. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (Vol. 10132, p. 101322U).

Shen, S., Rawat, P., Pouchet, L.-N., & Hsu, W. (2015). *Lung nodule detection c benchmark.* `https://github.com/cdsc-github/Lung-Nodule-Detection-C-Benchmark`. GitHub.

Siegel, R. L., Miller, K. D., & Jemal, A. (2017). Cancer statistics, 2017. *CA: A Cancer Journal for Clinicians*, *67*(1), 7–30. Retrieved from `http://dx.doi.org/10.3322/caac.21387`

Skymind. (2017). *GAN: A beginner's guide to generative adversarial networks.* `https://deeplearning4j.org/generative-adversarial-network`.

Valente, I. R. S., Cortez, P. C., Neto, E. C., Soares, J. M., de Albuquerque, V. H. C., & Tavares, J. a. M. R. (2016, February). Automatic 3D Pulmonary Nodule Detection in CT Images. *Comput. Methods Prog. Biomed.*, *124*(C), 91–107. Retrieved from `http://dx.doi.org/10.1016/j.cmpb.2015.10.006`