Classification of lung cancer in chest CT scans

Cláudia Rodrigues
up201508262@fe.up.pt
João Barbosa
up201406241@fe.up.pt
Manuel Curral
up201202445@fe.up.pt

Faculdade de Engenharia
Universidade do Porto
Porto, Portugal

Abstract

Lung cancer is still a very present and deadly type of cancer in our societies. A major issue of the disease is that its diagnosis usually happens in late stages of its development, making it harder to treat. Computed tomographies (CTs) have long been suggested as a potential tool for early detection.

In this paper, we describe solutions for nodule classification and segmentation, using data provided by the LNDb Challenge. We use Deep Learning for our tasks, testing multiple models for classification and using the UNet3D Neural Network for image segmentation.

1 Introduction

Lung cancer is one of the deadliest types of cancer worldwide, affecting both men and women. Contrary to other types of cancer, where the detection of its presence can be done in much earlier stages, the diagnosis of the lung cancer disease is often late. Low-dose computed tomography (CT) has long been suggested as a potential early screening tool and a 20% reduction in lung cancer mortality has been demonstrated for lung cancer risk groups [4]. However, bringing this type of technology to general population is still a challenge due to the difficulty of the task and the major costs associated. Thus, building an independent second-opinion would provide valuable insights as well as removing some of the burden from the medical staff.

This work uses the guidelines and data provided by the LNDb Challenge [4]. The main goal of this challenge is the automatic classification of chest CT scans according to the 2017 Fleischner society pulmonary nodule guidelines for patient follow-up recommendation. The challenge is divided into 4 sub-challenges. We will explore and describe the challenges B, in which our goal is to segment the nodule in the image, and C, whose objective is to classify the nodule's texture. Both of these sub-challenges are necessary for the main challenge (classification according to the Fleischner guidelines).

We use Deep Learning techniques in order to learn classification and segmentation models, training on data provided by the challenge. The LNDb dataset contains 294 CT scans collected retrospectively at the Centro Hospitalar e Universitário de São João (CHUSJ) in Porto, Portugal between 2016 and 2018. For each CT, it is also provided the xyz coordinates of the finding, whether it is a nodule or not, its volume and texture, as well as the segmentation result (according to the radiologists). More information is available on the challenge's website [5]. In all of our experiments, we use a 4-fold cross validation with the splits already provided by the challenge.

2 Challenge C - Nodule Texture Classification

The aim of this challenge is to automatically classify a nodule's texture, given a CT image. As previously mentioned, we have access to the CTs as well as their texture classification, according to the radiologists. The data provided classifies texture in the 0-5 range, which is converted to 0-3 range according to the challenge's specification (textures 1 and 2 as well as textures 4 and 5 are grouped together).

For this challenge, we've computed four different types of data input, which are further explained in each of this section's subsections.

2.1 Models

We tested three different type of networks: the first, named I3D, is a pre-built model made available by its developers, while the other networks were built by us. The full architecture for each model can be inspected in the source code.

The *I3D (Two-Stream Inflated 3D ConvNet)* [3] is a deep neural network classifier that receives 3D data as input. The neural network leverages pre-computed weights from deep networks used for ImageNet, such as VGG-16 and ResNet, inflating those values to a 3D architecture, followed by training on video data. Despite its original purpose of video classification, we can apply transfer learning to use it on our CT classification.

The second model, which doesn't have any pre-trained weights, is our own Convolution Neural Network that takes 3D images as input. It serves as a good baseline for the I3D model. From now on, this model is referenced as *CNN3D*.

The third network used, which also doesn't have any pre-trained weights, is a 3 layer deep neural network, composed of two 1024 dense layers followed by a 1D global average pooling layer. From now on, this model is referenced as *DenseNN*.

2.2 Data Pre-processing

In order to use the desired deep learning models, data provided by the challenge, which comes in the *.mhd/.raw* format, must be converted into the formats the neural networks expect.

The I3D neural network uses a 3-channel RGB image. Thus, we normalized each value to fit the 0-255 range, as well as replicate the value three times (one for each needed RGB channel). The same approach is used for the CNN3D.

The DenseNN neural network was purposefully built to take a matrix as input. The matrix is always 32-value wide, but may take a different length per batch.

2.3 Approaches

This section describes the four different approaches used in the classification challenge. The first three methods can use both I3D or CNN3D neural networks, while the last approach is fed to the DenseNN.

1. **Scan Cubes** - In this first approach, we feed scan cubes to the neural network. Scan cubes are computed as a 80x80x80 square around the nodule's center (as provided by the xyz coordinates in the dataset).

2. **Masked Scan Cubes** - This approach is similar to the previous, with the additional step of masking the computed scan cubes i.e. we only pass the sections of the image that the radiologists marked as being part of the nodule.

3. **Scan Cubes and Masks** - Our third proposal uses a multimodal learning approach. We provide both the scan cube as well as the mask, each to an individual neural network. Both networks are then joined into a final, cohesive architecture using late fusion [8].

4. **Descriptors** - Our final approach uses descriptors computed from OpenCV's ORB detector [6]. For each slice of the image (in the z axis), we computed the descriptors and appended all of them in a 2D matrix. When training, matrices are allowed to have a variable size between batches - however, in the same batch, the shape of the arrays must be the same. We used padding (filling missing rows with 0s) in each batch.

2.4 Experiments and Results

In this section, we describe all the experiments done in this work for the classification problem, as well as their results. We used *loss* as the main performance indicator, which is computed as the average from the 4 folds. We compute the average both for the training set (*train_loss*) and validation set (*val_loss*). Every experiment is run at least 10 epochs to analyze not only the models' capacity but also their proneness to overfitting.

2.4.1 I3D vs CNN3D

We started by comparing the performance of the I3D vs the CNN3D. Figure 1 shows the performance of both neural networks.

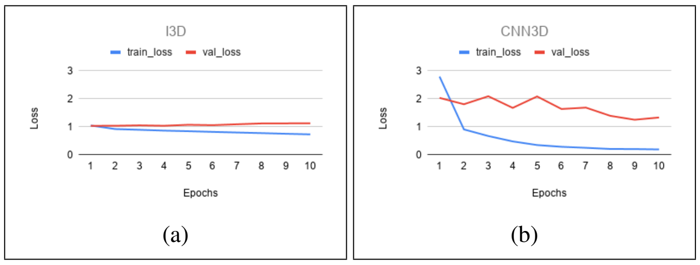

Figure 1: Performance comparison between I3D and CNN3D: (a) I3D; (b) CNN3D.

We expected to obtain better results with the I3D, which was then confirmed by this experiment: it consistently achieves lower loss for the validation set, compared to the CNN3D. Also, the values obtained for the CNN3D are much more prone to variation, while the I3D shows more robustness.

Additionally, notice that the *val_loss* for the I3D model is pretty much constant throughout the epochs in the training cycle. This means that one epoch is enough to learn the problem, which makes us believe that the model would definitely benefit from having more training data.

It is also worth comparing the *train_loss* for both models. After some epochs, the *train_loss* for the CNN3D starts to decrease to the point of almost being zero (in fact, it reaches the zero value with more epochs), while the I3D displays a much slower decrease. We reason that the CNN3D is overfitting the training data. On the other hand, the I3D fights the overfitting problem quite well.

Given that the I3D achieves considerably better performance than the CNN3D model, we used only the former for the remaining experiments.

2.4.2 Trainable Layers

The objective of this experiment is to study the impact of allowing the update of the weights in all layers of the I3D (including the pre-trained weights, as explained in Section 2.1), or only the ones closer to the output.

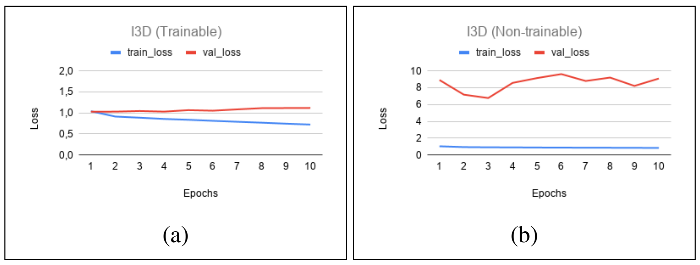

Figure 2: Impact of trainable layers: (a) Trainable; (b) Non-trainable.

The results displayed in Figure 2 show that allowing all layers to be trained greatly improves the performance of the model (notice the difference in the graph's vertical scale). It is worth noting, however, that the training phase will take much longer since it has significantly more parameters to train (approx. 13M vs 1M). Nonetheless, the tradeoff is worth it.

2.4.3 Optimizer: Adam vs SGD

We've also studied the impact of the neural network's optimizer in the training phase. In this experiment, we tested the *Adam* (Adaptive Moment Estimation) and the *SGD* (Stochastic Gradient Descent) optimizers. Results are shown in Figure 3.

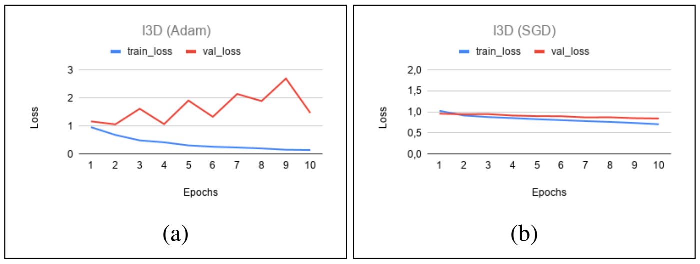

Figure 3: Optimizers' impact on performance: (a) Adam; (b) SGD.

The SGD provides significantly better results. Our experiment supports recent claims stating that the SGD optimizer performs better than Adam for several problems [1].

2.4.4 Classification Problem

Our final experiment for this challenge is about the classification problem itself. We compare the performance of the models, using the approaches described in Section 2.3. We used the I3D neural network (fully trainable, SGD optimizer) for the approaches 1, 2 and 3. The 4th approach uses the DenseNN neural network. See Figure 4 for the results.

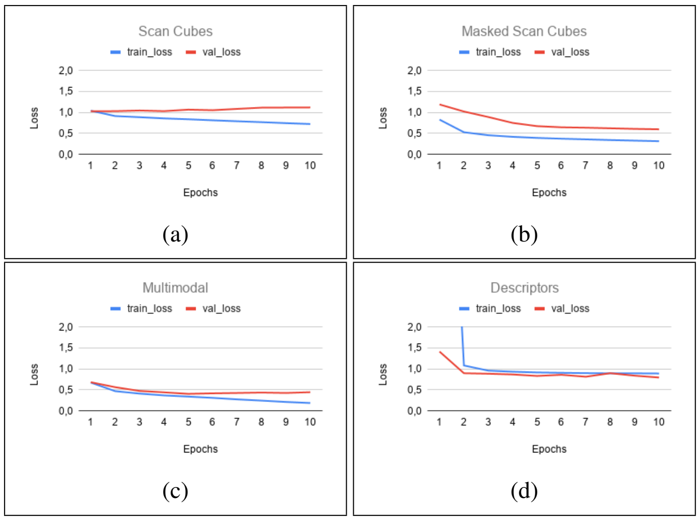

Figure 4: Performance comparison of proposed methods: (a) Scan Cubes; (b) Masked Scan Cubes; (c) Multimodal; (d) Descriptors.

Comparing the results from *Scan Cubes* and *Masked Scan Cubes*, we conclude that masking the image has a positive effect on the I3D's performance, since in the former the *val_loss* value is higher and constant throughout the epochs, while in the latter this value is reduced to almost half and also the loss follows the same behaviour on the training and validation data. Therefore, segmenting the nodule in the CT before its texture characterization should bring benefits to the medical procedure. This is further confirmed by the *Multimodal* approach, which got the best results in our experiments, that indicates that having both the CT around the nodule and its corresponding mask improves the overall performance of the task.

We've also found the results for the *Descriptors* approach to be quite positive and also surprising, reaching a performance superior to that of *Scan Cubes*. Additionally, this approach has strong resistance to overfitting, since the *train_loss* stays constant when the model has no more learning capacity, and *val_loss* follows the same behavior with very similar values. A definite advantage from this approach is its training speed, when compared to the others.

The following tables summarizes the results for this classification problem, showing the minimum loss and maximum accuracy obtained for each approach in the validation dataset. SC stands for *Scan Cubes*, MSC for *Masked Scan Cubes*, MM for *Multimodal* and finally DSC for *Descriptors*.

	SC	MSC	MM	DSC
min val_loss	1,03	0,60	0,40	0,79
max val_acc	63,9%	88,6%	88,5%	61,6%

The *Multimodal* approach displays the best performance among all methods, followed closely by *Masked Scan Cubes*, which means, as previously stated, that this classification task is improved by segmenting the image beforehand.

We also find that the *Descriptors* approach gives better results than *Scan Cubes*, is faster to train and also less prone to overfitting - which makes it the preferable choice between both.

3 Challenge B - Nodule Segmentation

This challenge consists in the automatic segmentation of pulmonary nodules in the CT scans. The goal is to compute a 80x80x80 cube centered on the nodule centroid with the predicted segmentation for each nodule.

3.1 Model

A *3D U-Net* [2] network was used in this challenge. It takes 3D volumes as input and learns to generate volumetric segmentations. It was designed to maximize brain tumor segmentation performance, extending the *U-Net* [7], a convolutional neural network developed for biomedical image segmentation.

3.2 Approach

The scan cubes are the input to the network and the mask cubes are the ground truth segmentation. A normalization was applied so that all values are within the range of 0 and 1. Thus, the network expects images with only one channel.

The network is build to learn one segmentation label. The predicted mask images from the model are the output of a sigmoid function, which will be values between 0 and 1. Each pixel activation p represents the probability of that pixel being a nodule. To obtain a binary image, a threshold of 0.5 was applied.

3.3 Experiments and Results

The network was initially trained with binary cross-entropy loss. This measure ensures equal learning to each pixel. Since we are dealing with class imbalance, the performance of the network was affected since it can minimize the training loss by predicting the background class more.

We opted for the dice coefficient since segmentation problems favor the use of overlap based metrics. As such, the loss function that the neural network aims to minimize is:

$$DiceLoss = 1 - DiceCoef$$

As in challenge C, we used 4-fold cross validation. In each fold, the model is trained for 15 epochs with a batch size of 3. The loss is then computed as the average from the 4 folds, shown next in Figure 5.

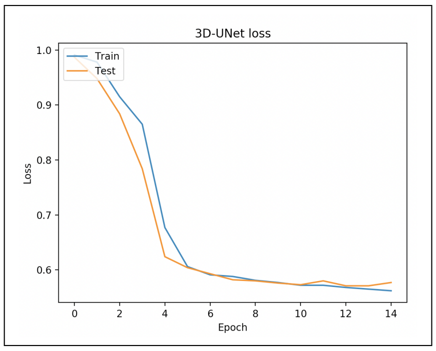

Figure 5: Performance of the 3D U-Net

It is possible to notice that the loss starts to decrease significantly until epoch 5, reaching a dice coefficient near 0.4, and stays almost constant through the remaining epochs in the training cycle. Additionally, the loss is very similar in the training and validation set.

As the result, the average dice coefficient is around 0.45. The final weights are the ones resulting in the lowest validation loss value. Figure 6 shows an example of the predicted nodule segmentation of a CT, in 3 different slices.

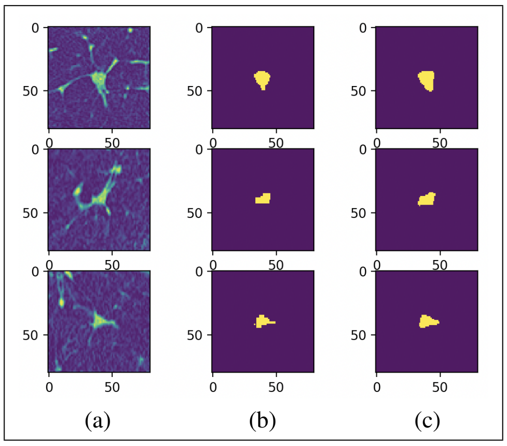

Figure 6: Prediction of a nodule segmentation: (a) CT scan; (b) Ground truth segmentation; (c) 3D U-Net segmentation.

4 Conclusions and Future Work

In this paper, we analyzed multiple solutions for classification and segmentation of lung cancer nodules (challenges B and C of the LNDb Challenge).

For the classification problem, the multimodal learner - where we feed to the neural network both the CT and the CT's mask - achieved the best results. In fact, our results lead to the conclusion that performance increases if we do the segmentation beforehand, as shown in the *Masked Scan Cubes* and *Multimodal* approaches.

Additionally, we find that both I3D and DenseNN networks are good models for the problem at hand, while the CNN3D quickly degrades in its quality. The methods with the best results were computed using the I3D network, but it is worth noting that the DenseNN network provides satisfactory results, requiring less computational power and is also less prone to overfitting.

We've also realized that the optimizer of the neural network plays a huge part in training. In this research, we've tried the Adam and SGD optimizers, opting for the latter. Future research should analyze more optimizers and their impact on performance.

For the segmentation problem, different modifications of the basic 3D U-Net have been introduced, designed to improve the performance for their specific tasks. Since the training of the network is restricted with memory and time constraints, it was not possible to test more complex models. Future work should analyse different combinations of metrics in the loss function or the performance of different types of networks.

As a final remark, we believe that this problem would benefit from more training data. Obtaining more TCs with the corresponding labels might be difficult, so a simple solution would be data augmentation. Due to the lack of a mature, trustworthy library for augmenting 3D images, and the infeasibility of doing our own (given the time constraints for this work), we weren't able to test this hypothesis. Therefore, we would like to see future research and experiments tackling this issue.

References

[1] Mitchell Stern Nathan Srebro Benjamin Recht Ashia C. Wilson, Rebecca Roelofs. The marginal value of adaptive gradient methods in machine learning. *ArXiv*, abs/1705.08292v2, 2018.

[2] Fabian Isensee, Philipp Kickingereder, Wolfgang Wick, Martin Bendszus, and Klaus Maier-Hein. Brain tumor segmentation and radiomics survival prediction: Contribution to the brats 2017 challenge. 02 2018.

[3] A. Zisserman J. Carreira. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset, 2018.

[4] LNDb. Grand Challenge on automatic lung cancer patient management. https://lndb.grand-challenge.org/, 2019. Accessed: 26-12-2019.

[5] LNDb. LNDb Challenge Data. https://lndb.grand-challenge.org/Data/, 2019. Accessed: 26-12-2019.

[6] OpenCV. OpenCV. cv::ORB Class Reference. https://docs.opencv.org/3.4/db/d95/classcv_1_1ORB.html, 2019. Accessed: 27-12-2019.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015.

[8] L. Morency T. Baltrusaitis, C. Ahuja. Multimodal Machine Learning: A Survey and Taxonomy, 2017.