DeepMedic and U-Net neuronal network architectures for lung segmentation in CT scans

Lionel Rajaona
University of Western Ontario
Email: lrajaona@uwo.ca

Rajat Balhotra
University of Western Ontario
Email: rbalhotr@uwo.ca

Daniel Allen
University of Western Ontario
Email: dallen44@uwo.ca

Steffen Bleher
University of Western Ontario
Email: sbleher@uwo.ca

***Abstract*—This work examines the DeepMedic and the U-Net architectures of neuronal nets for segmentation of lung structures in computed tomography scans. The application of the algorithms will be evaluated on the LUNA16 dataset.**

I. Introduction

Since hardware specifications and computational power increased dramatically in the last years machine learning approaches can be applied in various disciplines today. On top of that the progress in research on convolutional neuronal networks (CNN) made it a very powerful tool for image processing where information is gained from image data.
One challenging application is medical image computing (MIC). The main goal of MIC is to extract clinically relevant information or knowledge from medical images. Furthermore Segmentation is the process of partitioning an image into different meaningful segments (e. g. organs, bones, ...).
In this project the goal was to segment the lung of a human body from computed tomography (CT) scans of the LUNA16 dataset [1]. Every CT scan consist of a variable number of grey-scale image slices. Given these slices a 3D model of the lung needs to be created. To reach this goal every slice of the CT scan will be evaluated separately by a CNN architecture and a label map for the lung and the bronchus on these slices will be predicted. From these outputs a 3D model of the structure can be created.

Fig. 1. CT scan from the LUNA16 dataset

The segmentation of the lung from the rest of the picture is the first step for further image processing. In the LUNA16 dataset the final goal for example is to detect nodules of the lung indicating cancer. Machine learning approaches can be a powerful support for the doctors who treat patients with suspected cancer. The algorithms can reduce human errors, and it might even have the potential to outperform human capabilities or could automatize the process of cancer detection. This could have a positive effect on health care quality and costs.
In this work two different neuronal networks will be trained and tested to segment the lung on the LUNA16 dataset. Furthermore they will be examined and compared under different metrics.
First a short overview on the DeepMedic and the U-Net architecture will be given and will then be related to other approaches for medical image segmentation. After that the process and implementation will be explained and then the results of the two algorithms will be examined and compared. In the end a conclusion on the work will be drawn and further steps sketched.

II. Background

A. DeepMedic architecture

DeepMedic is a 3D neural network. It has been initially used for segmentations in biomedical 3D scans, especially for detecting brain anomalies such as injuries, tumors and lesions. In this project we will try this algorithm for lung segmentations.
A DeepMedic model consists of detecting a particular pattern in an image. This is achieved by multi-layer convolution in the network. We distinguish two components in the model. First there is a three-dimension convolutional neural network (CNN) model used for dense segmentation (figure 2 left part), then a three-dimension fully connected conditional random field (CRF) model which does post-processing and handles the hard segmentation as well (figure 2 right part). We can also notice a double-path network composed of a regular-resolution and a lower-resolution path. This second one mainly prevents from overfitting. Each layer of the CNN model contains channels called feature maps, i.e. a group of neurons identifying a feature in the previous layer. Then featurer are defined by associated kernel weights. Number of inputs, outputs, feature-maps and kernels are tunable.

B. U-Net architecture

Semantic segmentation is partition of an image in coherent parts. U-Net is mostly used for biomedical image segmentation. In figure 3 the structure of the U-Net is shown.

Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box.

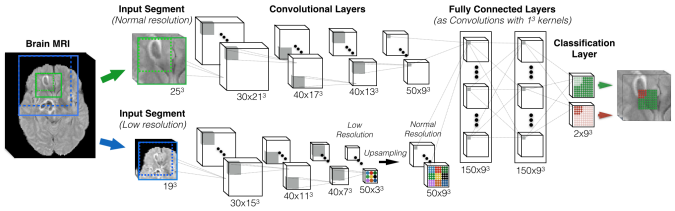

Fig. 2. Structure of DeepMedic architecture

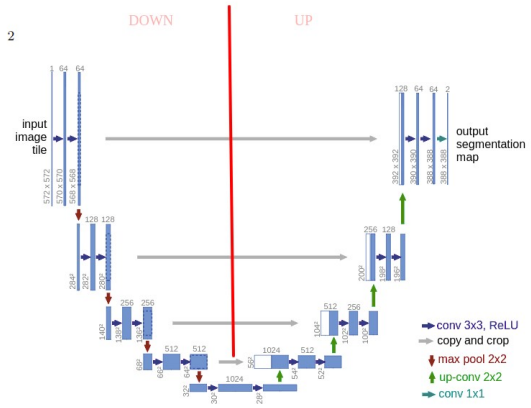

Fig. 3. Structure of U-Net architecture

X-Y-size is provided at the lower left edge. White boxes represent copied feature maps. Arrows denote the different operations.

First part is called down or encoder part. Convolutional blocks followed by maxpool downsampling layers are applied to encode the input image into feature representations at multiple different levels. The second part of the network consists of upsampling and concatenation layers followed by regular convolution operations. The dimensions from left are expanded to meet the original image size. The grey and green arrows indicate where to concatenate future maps together.

In comparison to other fully convolutional segmentation networks, the main feature of U-Net is that while upsampling and going deeper in the networks, it also concatenates the higher resolution features from the down sampling part with upsampled features in order to better localize and learn representation.

C. Metrics

1) Dice Loss: We need validation criteria to build our models. We use a Dice loss coefficient to compare the similarity of two samples. In the following equation, smooth is added and stands for backpropagation. Ours is defined by:

$$diceloss = 1 - \frac{2 * (Y_{true} \bigcap Y_{prediction}) + 1}{|Y_{true}| + |Y_{prediction}| + 1} \quad (1)$$

2) BCE Dice Loss: We also use binary cross-entropy (BCE) to measure the performance of our classification model. For labels y and predcted probabilities p, binary cros-entropy is:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \quad (2)$$

Then, our BCE Dice Loss is:

$$Loss = diceloss + BCE \quad (3)$$

3) Hausdorff distance: The Hausdorff distance is a metric to calculate the maximum of the difference between two sets of coordinates.

For the set of coordinates A and B it is defined as following

$$d_{hausdorff}(A, B) = \max_{a \in A}(\min_{b \in B} d(a, b)) \quad (4)$$

where $d(a, b)$ represents the Euclid distance between the two coordinates a and b. It can be interpreted as the maximum distance of set B to set A and is therefore a important metric for evaluating the difference between two label maps.

4) Mean distance: The mean distance is similar to the Hausdorff metric but is a measurement for the mean of the difference between two sets of coordinates.

It is therefore defined as following

$$d_{mean}(A, B) = \frac{1}{|A|}\sum_{a \in A} \min_{b \in B} d(a, b) \quad (5)$$

where again $d(a, b)$ represents the Euclid distance between the two coordinates a and b. It furthermore can be seen as the average difference between the two sets and will also be used for evaluation.

III. Related Work

text and subsections

IV. Process and occurring problems

In this section an detailed description of the dataset will be given. The process of data preprocessing and the training process for both architectures will explained.

A. LUNA16 dataset

The used dataset is from the LUNA16 challenge and each scan contains a number of slices. The algorithm creates a label map for each slice marking every pixel that is part of the lung. An example of one scan and the corresponding labeling is shown in figure 8.

The dataset consists of about 10 GB of mdh and raw image files. The files contain a varying number of slices of 512 x 512 pixel grey-scale images and corresponding 512 x 512 pixel label maps. Since computational resources were limited just a subset of the data was used for this work. Furthermore the more detailed label map was reduced to one label for the lungs and one label for the bronchus.

From the LUNA16 dataset 10 CT scans for training and 20 CT scans for evaluation were chosen randomly.

Fig. 4. CT scan of the lung and labeled lung (green) and the tube (yellow). The first picture represents one slice of the image, the second a 3D representation of the label and the lower two pictures are visualizations of the side and front view.

B. Preprocessing of image data

Each file was converted to a NIfTI image file for better processing and visualization. These files contain all grey-scale image slices of one scan. For more appropriate input values, each image was normalized following the standard normal distribution ($\mu = 0.0$ and $\sigma = 1.0$) separately.
Then, it was necessary to implement data importation to properly use them. Data from directory were converted into input and label matrices following the size 512x512 for each slice (on average 200 slices per image).

C. Training and validation

Both network architectures were trained on the 2145 slices of the 10 training CT scans over 35 epochs.
For the DeepMedic architecture a open-source implementation [2] was modified for the purpose of lung segmentation and the parameters were set up according to successful implementations for brain tumor detection. RMS propagation with a initial learning rate of 0.001, the Nesterov momentum (momentum rate of 0.06) and L2 regularization (regularization rate of 0.0001) were used as an optimizer. For the training a batch size of 10 samples was chosen and the network was evaluated with the binary cross entropy (BCE) as a loss function.
The U-Net architecture was built up according to the structure in figure 3. For the training process the Adam optimizer with the described loss function $Loss = diceloss + BCE$ from chapter II-C was used in combination with a batch size of 3 slices. To get an idea of the performance of the model during training 10 % of the slices were used for validation in each epoch.

D. Dealing with the big dataset

Although the dataset was reduced to 10 CT scans some adjustment in the training process still had to be made to handle the size of the dataset.
Therefore, a generator was defined to reduce the data size that will be fed to the training process at once. The generator feeds just a batch of images in real time to the training process instead of loading the whole dataset at once.

V. Evaluation

A. Setup

Python 3.x has been chosen to design this solution. NiBabel package was used to deal with specificity of medical images. TensorFlow and Keras are additional libraries required to build training models.

B. Evaluation on training model

graph loss on training evaluation (dice loss, bce dice loss, batch size (correlates to gpu), epochs, steps per epoch, optimizer) prediction image on label and prediction evaluating and examining models with hausdorff and mean distance on pictures (graphs or statistics, table) comparison of unet and deepmedic

explain high hausdorff and how to reduce it

Architecture	Dice Coefficient	Hausdorff distance (mm)	Mean Distance (mm)
DeepMedic	0.968	119.50	1.45
U-Net	0.976	103.21	0.33

TABLE I
Errors of the LSTM network compared to the forecasting approaches of assignment 1.

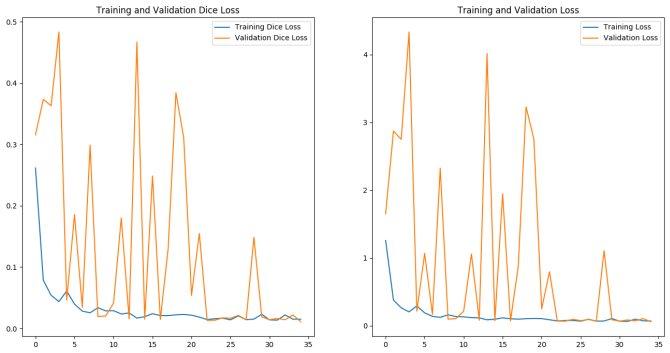

Fig. 5. xxx

VI. Conclusion

- conclusion on network comparison - complex and specific networks especially for this kind of task - difficulties with huge data - training time and allocation of computational resources - draw conclusion on how good ml is for MIC, issues, ... - further steps (nodules, ...)

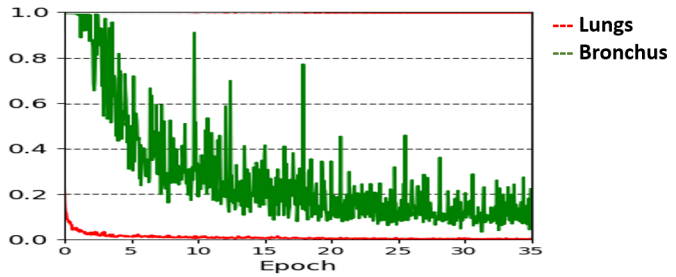

Fig. 6. xxx

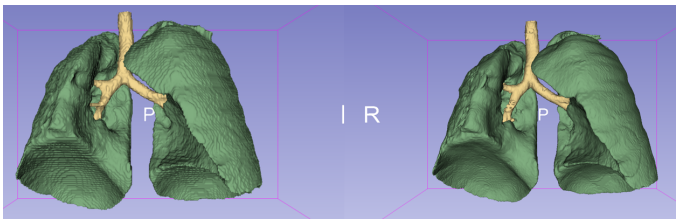

Fig. 7. xxx

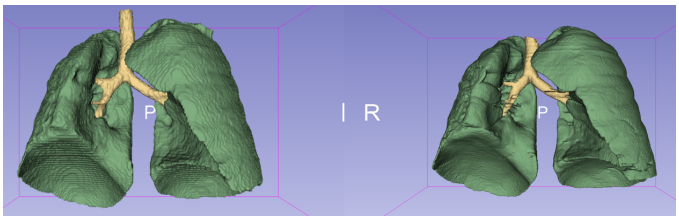

Fig. 8. xxx

REFERENCES

[1] "Lung nodule analysis 2016." [Online]. Available: https://luna16.grand-challenge.org/

[2] "Efficient multi-scale 3d convolutional neural network for brain lesion segmentation." [Online]. Available: https://github.com/Kamnitsask/deepmedic