Deep neural networks models comparison for intestines scans segmentation with transfer learning

Mathilde Larchevêque
CentraleSupélec Student

Alexis-Raja Brachet
CentraleSupélec Student

Abstract

This project consists in the semantic segmentation of intestines medical scans. The performances of three Deep Learning architectures – Unet, Unet++, and a combined model - are compared. Two experiments are made : one training is done with a fixed number of epochs and another one with a fixed batch size and a fixed training duration. On both experiments Unet model achieved the best Dice coefficients score (respectively 75.1% and 73.1%). A failed attempt to train and evaluate a Mask R-CNN model has also been made. This document contains the description of the experiments and commentary of the results.

1. Introduction

The task of semantic segmentation - also called pixel-wise classification - consists in clustering and classifying each pixel of the input image. It has multiple applications in medical imaging, such as detecting brain and tumors (see Moon et al. 2002 [8]). Before deep neural networks, the methods used for the tasks were feature extraction using pixel colours, Histogram of Oriented Gradient (HOG) along with machine learning classifiers [2]. Fully Convolutional Networks (FCN) then were state-of-the-art models with the introduction of U-net architecture by Ronneberger et al. [9]. Taking advantage of nested, dense skipped pathways, Unet++ was also proposed as an improvement of Unet model.

The application this project focuses on is the semantic segmentation of stomach, large bowel and small bowel on MRI (Magnetic Resonance Imaging) scans. It consists in the comparison of U-net, U-net++ and a combined model of both architecture with respect to the task. Indeed , this project is based on the UW-Madison GI Tract Image Segmentation Kaggle challenge, where most competitors used Unet-based models. The idea is to empirically compare Unet model with other semantic segmentation models.

1.1. Motivation

The identification process of stomach and intestines is a mandatory step prior to X-ray exposure. It can take up to one hour, as the organs move with respect to time (as shown in fig. 1). Saving this time by automating the identification would allow more people to be cured.

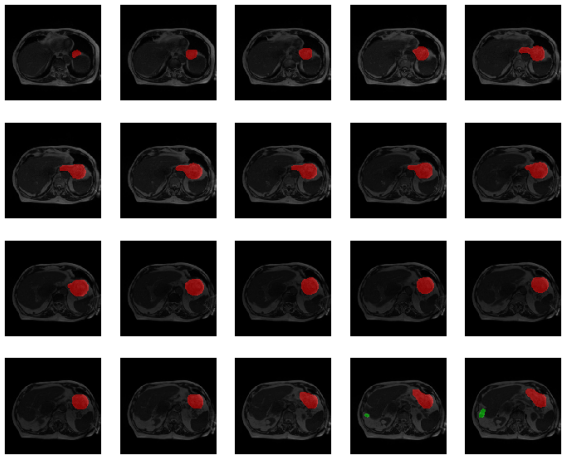

Figure 1. Intestines position evolution according to time

Neural networks applied to medical image semantic segmentation have been proved to be very efficient. This motivates the use of such algorithm for stomach and intestines segmentation.

1.2. Objectives

There are two objectives for this project:

- Use transfer learning with efficient pre-trained segmentation models to segment the scans.
- Compare Unet, Unet++ and a combined model performances with respect to the task and by using mostly similar training parameters.

2. Problem Definition

Original Problem Let $(X^{(n)})_{n\in\{1,...N\}}$ be the scan images, $([L_n^{(s)}, L_n^{(lb)}, L_n^{(sb)}])_{n\in\{1,...N\}}$ the corresponding lists of pixels for the stomach (s), large bowel (lb) and small bowel (sb).

Let Φ_1, Φ_2 and Φ_3 be 3 neural networks with different architectures. The objective of the problem is to find:

$$\underset{i\in 1,2,3}{min} \sum_{n=1}^{N_{test}} ||\Phi_i(X^{(n)}), f_{ohe}(L_n^{(i)}, L_n^{(lb)}, L_n^{(sb)})||_{DICE} \tag{1}$$

Where f_{ohe} function is the one hot encoding transform and $DICE$ the region-based coefficient function defined as:

$$||\cdot||_{DICE} : (y, y_{pred}) \mapsto 1 - \sum_{i\in\{(s),(lb),(sb)\}} 2\times\frac{y^{(i)} \cap y_{pred}^{(i)}}{y^{(i)} + y_{pred}^{(i)}} \tag{2}$$

where $y^{(i)} \cap y_{pred}^{(i)}$ is the pixel-wise comparison of mask $y^{(i)}$ and $y_{pred}^{(i)}$ for the intestine class (i).

The one hot encoding function transforms the list of pixels for each intestines into 4 binary masks - one for each pixel class: None, stomach, large bowel, small bowel.

$$f_{ohe} : \mathbb{R}^{\mathbb{N}} \times \mathbb{R}^{\mathbb{N}} \times \mathbb{R}^{\mathbb{N}} \to \{0,1\}^{w\times h\times 4}$$
$$(L_n^{(i)}, L_n^{(lb)}, L_n^{(sb)}) \mapsto (BMask^{(i)})_{i\in(\emptyset),(s),(lb),(sb)}$$

where w and h are the weight and height of the corresponding scan image.

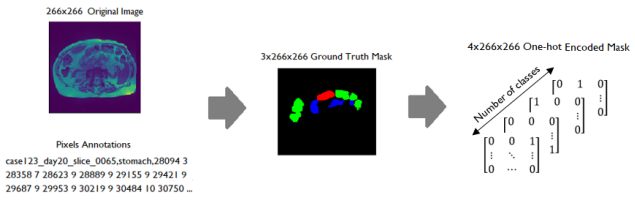

Figure 2. From annotations to one-hot-encoded mask.

3. Related Work

Semantic segmentation Semantic segmentation is a highly studied task and literature is well furnished. It has been especially applied in medical imaging, in the segmentation of neuronal structures with Unet architecture by Ronneberger et al. 2015 [9], but also in brain tumor segmentation by Moon et al. ([8]) and crypts detection in colons [1]. The high generalisability of the models (using transfer learning) and the results achieved allows real-world applications.

The Kaggle challenge UW-Madison GI Tract Image Segmentation aims at studying the result of such models on tract intestines scans. The particularity of the study is the evolution with respect to time of the intestine scans. To the best of our knowledge no paper has been published on the exact task of stomach, small bowel and large bowel segmentation using deep learning.

The architectures of Unet, Unet++ and Mask R-CNN - widely used for semantic segmentation - are explained in the following sub sections.

3.1. Unet and Unet++ algorithm

Unet Unet architecture (fig. 3) - developed by Olaf Ronneberger et al. for Bio Medical Image Segmentation - is composed of an encoder followed by a decoder. The encoder aims at contracting the image in order to capture its context. It is composed of convolutional, Max pooling and up-sampling layers. The size of the image gradually decreases while the depth gradually increases. The decoder has a symmetric architecture which enables precise localisation using transposed convolutions. Thus it is an end-to-end fully convolutional network (FCN) (it has no dense layer). This architecture is widely used for semantic segmentation: the encoder aims at classifying the objects of the image, while the decoder precisely locates the object on the pixels.

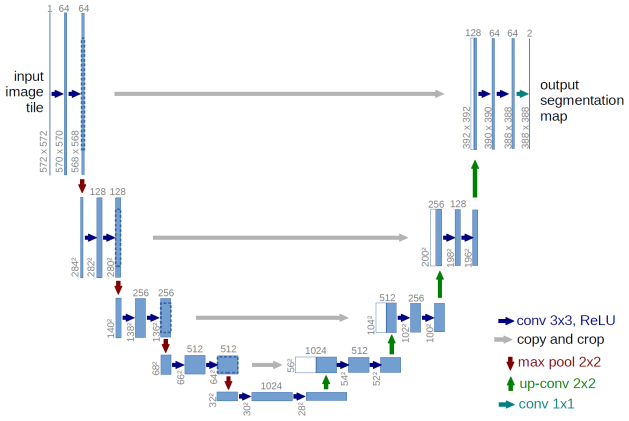

Figure 3. Famous Unet architecture schema from [9] paper

Unet ++ Unet++ (fig. 4) - developped by Zhou et al, is a modification of the Unet architecture. It has the same encoder-decoder architecture but differs in the skip pathways between the two. In the Unet model, these skipped connections are simple copy and crop transforms whereas in Unet++ these connections are dense and convolution lay-

ers. It also has deep supervision. It is also widely used for semantic segmentation for the same reasons as Unet.

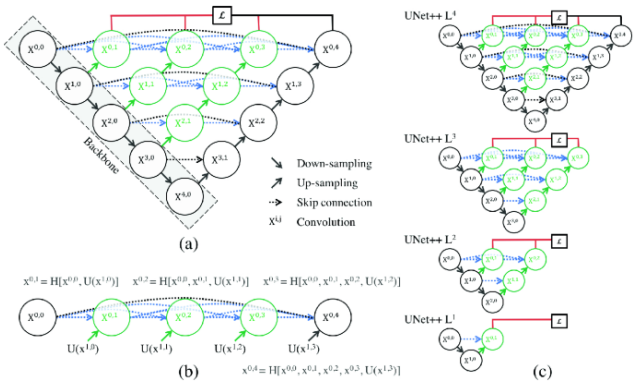

Figure 4. Unet++ architecture schema from [12] paper

3.2. Mask R-CNN algorithm

Mask R-CNN (*Mask Regional Convolutional Neural Network*) - introduced in 2017 [3] - is a modification of the Fast R-CNN model [4] with the addition of 'masks'.

Basic principles The R-CNN part extracts features from the images and proposes boxes where the objects should be (it is the first segmentation). It then classifies the object within the boxes. Finally, the mask part applies masks to the pooled regions. They are scaled up in order to fit the original images.

One particular aspect is that the classification, the bounding boxes and the masks are computed in parallel, whereas in other architectures, the classification depends on the masks.

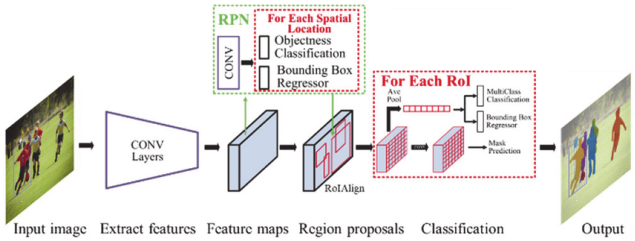

Figure 5. Mask R-CNN architecture [11]

Architecture The backbone is a ResNet50 or ResNet101 that serves as a feature extractor to first detect features as edges and corners then higher level features ones such as car, person, sky.

While the backbone described above works great, a Feature Pyramid Network (FPN) is added on the features extracted in order to enhance its performances, especially for multiple scale objects [6].

Then, a Region Proposal Network (RPN) scans the images over the previously extracted features in order to identify areas being likely to contain an object. Those areas are then processed in order to keep those which are the most likely to contain foreground objects : these are the so-called Regions of Interest (RoI).

However, because pooling has been performed, some misalignment might be produced due to quantisation [3]. Thus, a RoI alignement is performed in order to tackle this issue and to have masks that are well aligned with the original image.

Finally, the following 3 tasks are performed in parallel :

- Classification : the aligned RoIs go through a deep network to be classified
- Bounding Box Refinement : it refines the previous proposals made by the RoI
- Mask : the aligned RoIs, again, go through a fully connected network (FCN), then the masks are predicted. Finally, the masks are scaled up in order to fit the original image shape [7]

The network is trained with respect to the following losses :

$$L = L_{\text{classification}} + L_{\text{box}} + L_{\text{mask}}$$

The algorithm we used has been based on the akTwelve Mask R-CNN GitHub repository here.

4. Methodology

Objective As described in section 2., the objective of this project is to compare the performance of three different architectures: Unet, Unet++ and a combined model of these architectures (described in subsection 4.3). (As written before, an attempt has also been made to compare Mask R-CNN model but no result could be exploited for performance comparison.) The following methodology has been used:

Dataset We split the dataset between 85% training set and 15% test set at *patient-level*, in order to avoid data leakage.

Training *For all models*, we used the following training configuration :

- Optimizer: Adam optimizer with a learning rate $lr = 0.001$
- Loss: Dice loss

Two different training methods were used for all three models:

1. Various batch sizes; fixed number of epoch (= 2)

2. Fixed batch size (= 6); fixed training duration (= 3 hours); various number of epoch

Unet and Unet++ have been trained using transfer learning. The pre-trained weights are the following:

- Encoder backbone: ResNet50
- Weights: ImageNet pre-training

This results in 6 different trainings.

Comparison metrics The metrics used for the performance comparison is the average Dice coefficients **on the intestine classes**.

Limitations of this methodology (in particular the absence of cross-validation) are described section 4.4.

4.1. Dataset

The dataset is from the UW-Madison GI Tract Image Segmentation Kaggle Challenge. It contains 16'589 annotated scans of approximate size of 266x266.

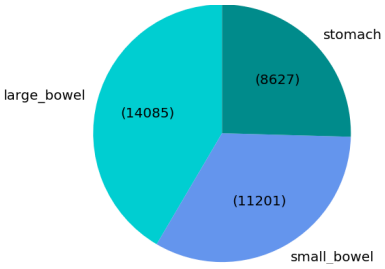

Figure 6. Number of images containing stomach, small bowel and large bowel annotations in the dataset

The dataset is not strongly imbalanced, as a result average Dice coefficients is used for performance comparison (on intestines class only). Finally, the number of images in the dataset (16'500) in relation to their sizes (70'000) plus the limited computational capacities makes transfer learning particularly pertinent for the models' training.

4.2. Image Pre-processing

Image annotations to one-hot encoded mask for Unet The scan images have 16-bits pixel debts. Therefor, the pixels are first divided by 2^{16} to get values between 0 and 1 (i). Then, they are normalized using mean and standard deviation values of the training set (ii). Finally, they are centre cropped to 288x288 shape images, in order to get fixed size images (iii).

As described in the Problem Definition section, the labels are transformed from pixel list to one-hot-encoded mask.

Image annotations to COCO format for Mask R-CNN In order to perform segmentation and classification with Mask R-CNN, the raw annotations need to be converted into the following format : `dataset_annotation = { 'filename': 'image.png','regions': {'0': { 'region_attributes': {name:'stomach'},'shape_attributes': {'all_points_x': [...],'all_points_y': [...],'name': 'polygon'}}, ... more regions ...},'size': 100202 } .`

In addition, the organs should be separated in order to have their own bounding boxes. To do so, we have performed a clustering among the pixels with a threshold of 1 pixel, as each pixel among the same organ are 1 pixel away from each other as shown in figure 7.

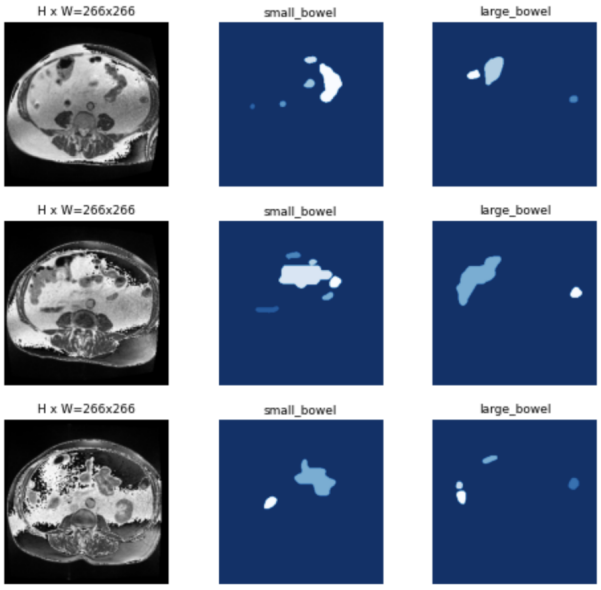

Figure 7. Display of processed masks after performing clustering on 3 scans

Superposing the image, the bounding boxes, and the masks for the last image of figure 7, we get figure 8.

Difficulties encountered Unfortunately, due to computational limitations and time remaining to train the Mask R-CNN algorithm, we haven't been able to train this architecture. We have only trained the heads of the model with 200 images (<1 epoch) and got the following results in figure 9.

In addition, the weights of the Mask R-CNN were trained on the COCO dataset. It is composed by common photos and doesn't contain any medical images. We should have performed a long training through several epochs in order to compute suitable weights and obtain suitable predictions.

As a result, Mask R-CNN hasn't been used for evaluation.

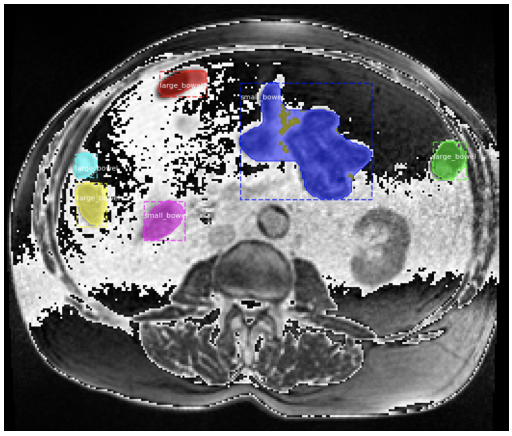

Figure 8. Superposition of an image, the masks and their bounding boxes.

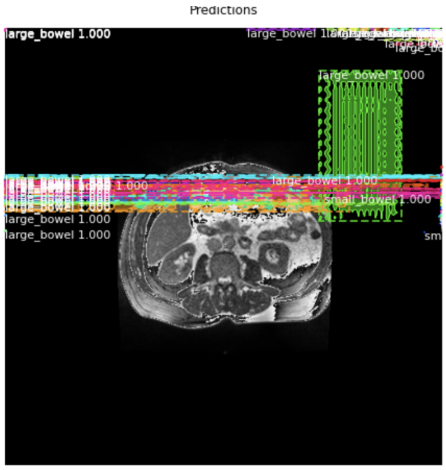

Figure 9. Poor predictions of the Mask R-CNN

4.3. Combined model

In the first training results (using 15% of the training dataset to test if the training procedure was correct), it has been observed that Unet model performed well on the segmentation task, while Unet++ classified more correctly the pixels (see fig.11). Therefor, we choose to build a sequential model made up of a pretrained Unet model and a randomly initialised Unet++. The Unet model first performs semantic segmentation on 2 classes: background and intestines, then the concatenation of its output and the raw image is given as input of a Unet++ model, which outputs the four classes segmentation mask. This architecture is illustrated fig. 10.

5. Evaluation

5.1. Model performance

The experiment results for the three models can be found in table 1 and the training loss plots in figure 15 and 16. The best Dice score of 75.1% is achieved by Unet model on the fixed epoch number experiment and with maximum batch size - maximum in the sense of the memory capacity available for the experiments-. We can observe that the batch size has an influence on the model performance. Indeed, Unet model trained on batch size = 16 and 2 epochs outperforms the Unet model trained on batch size = 6, even though this model was also trained on 2 epochs and for a longer duration. We can also observe that the training parameters had no influence on the performance ranking of the models. Faster-to-train models also achieved better Dice score. This can be partially explained by the small number of epochs used for the training and the fact that the training parameters (optimiser and learning rate) were not fine-tuned for each model.

Comparison between Unet and Unet++ In general, Unet++ has been shown to perform better than Unet on certain datasets, particularly where the objects of interest are small and complex ([12]). The additional context provided by the nested skip connections in Unet++ can help the network better distinguish between the objects of interest and the background. However, in this case, Unet performs better than Unet++. This could be explain by the simplicity of Unet which makes it easier to train compared to Unet++, which has a more complex architecture.

Comparison with combined model There are several factors that could explain the performance of the combined model compared to the other model. Firstly the model training: the combined model contains a pre-trained Unet and a randomly initialised Unet++. Thus the models are not trained on the same data, which could explain that they are not able to work together effectively. An attempt could be made to train a combined model with both non pre-trained weights. Secondly the model complexity: the combined model makes the overall model more complex and difficult to train. We can observe in figure 15 that the combined model training loss decreases less rapidly on the first epoch compared to Unet and Unet++. The combination of models does not lead in these experiments to a significant improvement in performance and simply increases the complexity of the model for little gain. Yet it is interesting to see that this model outperforms Unet++ on the two experiments.

Batch size influence The batch size is a hyperparameter that determines the number of samples processed by the model in one forward and backward pass during training. A

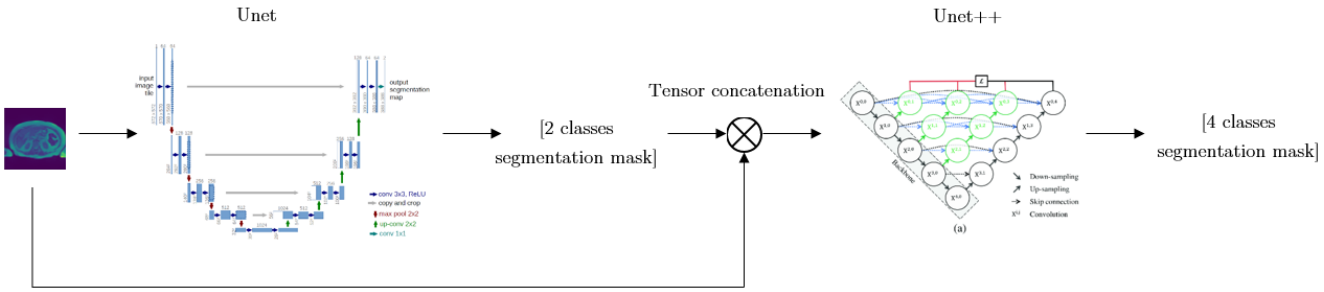

Figure 10. The combined model is a sequential model of pre-trained Unet for background and intestine semantic segmentation and a non-pretrained Unet++ with a skip connection.

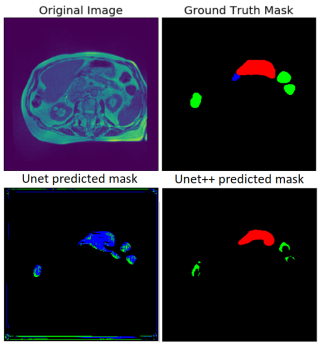

Figure 11. Predictions of pre-trained Unet and Unet++ after training on 15% of the training set. These results encouraged the choice of a combined model of the two architectures.

Figure 12. Training loss of Unet model for two batch sizes.

Figure 13. Training loss of Unet++ model for two batch sizes.

larger batch size results in fewer updates but with a higher variance in the update direction than for smaller batch size. With a smaller batch size, the model is updated more frequently, which can help prevent overfitting by allowing the model to learn a more general representation of the data.

The model considered are large models and we observe that a larger batch size results in better scores - as we can see in figures 12 and 13-. The loss actually also decreases faster for larger batch size than smaller (whereas usually larger batch size results in a slower convergence). The batch sizes difference (16 and 6 for Unet, 7 and 6 for Unet++) are probably too small to encapture the advantages of a small batch size over a large one. However due to memory capacity no higher batch size could be tested. In conclusion, larger batch size results in the experiments to better Dice score and faster convergence for both Unet and Unet++ model.

5.2. Comparison with Kaggle challenge results

The evaluation for the Kaggle challenge is the following weighted sum of Dice score and Haussdorff distance (surface-distance measure):

$$\begin{aligned}\text{score} = 0.4 \times ||\text{model}(X), \text{Ground truth}||_{DICE} \\ + 0.6 \times ||\text{model}(X), \text{Ground truth}||_{HAUSS}\end{aligned}$$

The winners of the competition (see code here) were able to reach a score of 0.89. Their architecture was based

Model	Training parameters			DICE Score
	Number of epochs	Training duration	Batch size	
Unet	2	2h24	16	**75.1**
Unet ++	2	10h06	7	63.1
Combined model	2	4h56	6	73.4
Unet	2	3h	6	73.1
Unet ++	0.49	3h	6	59.2
Combined model	1 (+ 0.15)	3h	6	69.9

Table 1. Model Dice score (on intestine classes only) on the test set for the two training

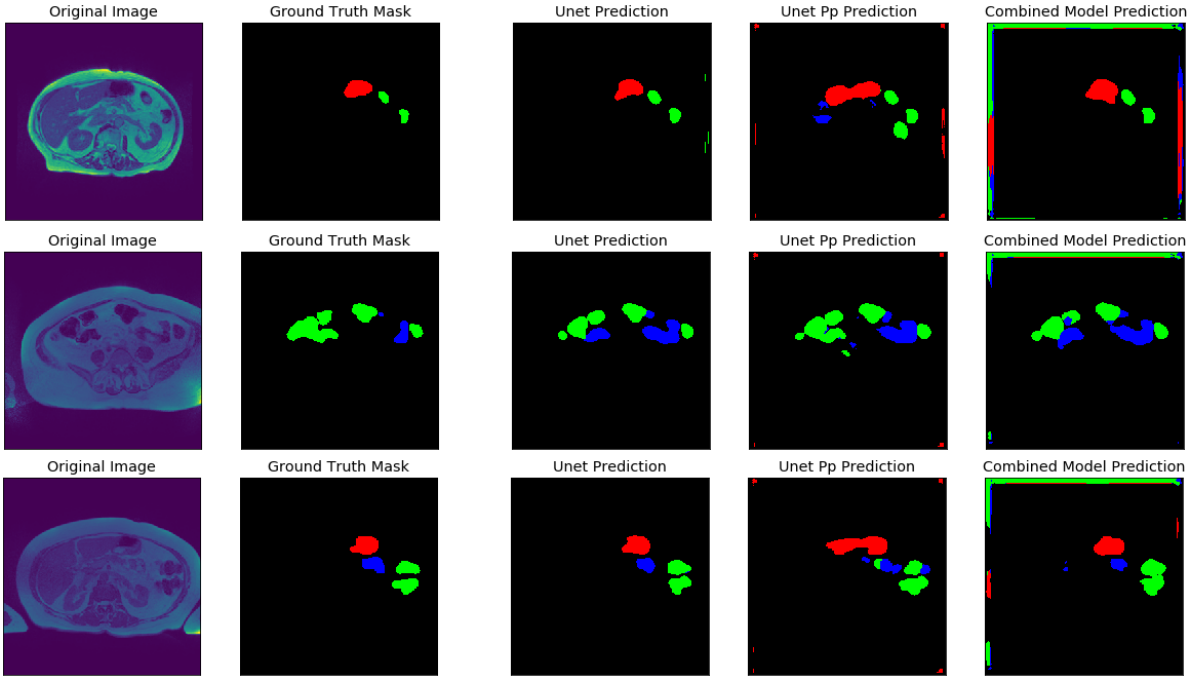

Figure 14. Models prediction example. Stomach is colored in red, large bowel in green and small bowel in blue.

on Unet with an efficientnet backbone, pre-trained on ImageNet and trained on the Cross-Entropy Loss. They worked in 2.5D framework.

The second team (see code here) also used a Unet model and both a 2.5D and 3D framework. In addition, they performed a two stage pipeline : the first stage determines whether the scans contains the seek organs (positives) or not (negatives) and the second stage performs segmentation and classification only on the positive scans. In addition, their loss during training was a combination of the Dice loss and the cross entropy.

5.3. Limitations

Statistical limitations In order to correctly compare the model performances, a cross-validation protocol should have been performed. The training dataset composed of over 14 thousands images is big enough to apply it. However, due to computational cost, we did not compute it.

Training configuration limitations The training configurations (fixed learning rate, batch sizes, optimiser) have an influence on the model performances (as we can see for example with Unet model between the two experiments). As a result these performance comparison are weakly generalisable.

6. Conclusions

6.1. About our work

The comparison between Unet (resnet50 encoder), Unet++ (efficiennet encoder) and the combined model shows that with a small number of epoch and under the same specific training parameters Unet outperforms both Unet++ and the combined model in both experiments.

	Haussdorff-DICE score	DICE score
(Kaggle model) Unet using 2.5D framework	0.89	-
(Kaggle model) Unet using 2.5 and 3D frameworks, DICE + CE loss and two stages pipeline	0.88	0.83
(Our model) Unet trained on 2 epochs and batch size of 16.	-	0.75

Table 2. Scores of the Kaggle two best models compared to our best model.

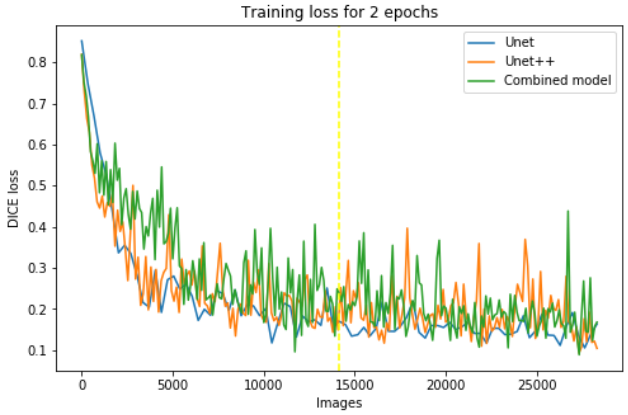

Figure 15. Training loss of the three models for the 2 epochs training experiment. The yellow vertical line represents the delimitation of one epoch training.

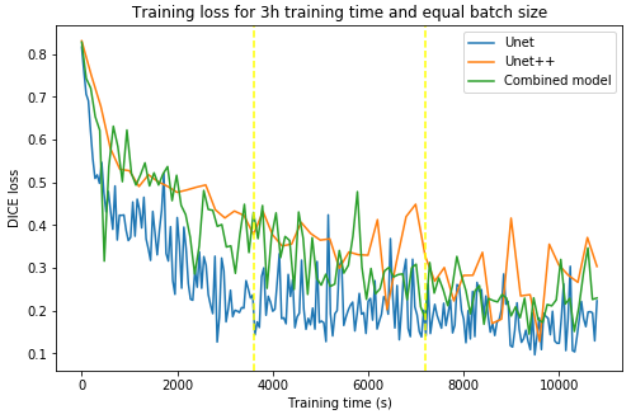

Figure 16. Training loss of the three models for the 3h training experiment. The yellow vertical lines represents the delimitation for 1h of training.

Combining Unet and Unet++ models seemed to be interesting to improve the metrics but ultimately only succeed to benefit from Unet efficiency. This project empirically illustrates the efficiency of Unet in the way of its simplicity of training resulting in better performance with classic training parameters and small amount of training data. The Dice score achieved by the best model is 75%.

6.2. Future steps

Pre-processing: Cropping the images Some scans have noisy information such as arm presence and others have large background. Cropping the image would maximise the area of interest around the intestines and would make training faster and more accurate.

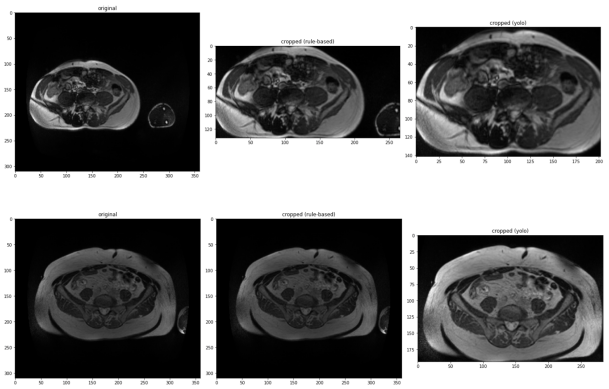

Figure 17. Cropped scans using 2 different methods (source here)

Profit from 2.5D and / or 3D framework The dataset is specific in the way that the scans are temporally linked. As mentioned above, all the top teams of the Kaggle challenge took advantage of this 2.5D or 3D data, the third dimension being time. The 3D framework is implemented as follows : the scans are concatenated on top of each other, the first slice being at the bottom, and the last one at the top; and a 3D Unet model (developed by Çiçek et al. [10]) is used. Taking advantage of the dataset structure results in better scores. It would be interesting to compare its performance with simpler Unet model with respect to non fine tuned training parameters.

Implement an ensemble model and use nnUnet Instead of implementing a combined model; it would be interesting to implement an ensemble model using Unet, Unet++ and Mask R-CNN (after training it). We expect this model to outperform Unet. Also, to improve Unet score we could use nnU-Net (developped by Isensee et. al. 2021 [5]) which is a segmentation method that automatically configures a

Unet model (including preprocessing, network architecture, training and post-processing).

References

[1] Assaf Cohen, Ehud Rivlin, Ilan Shimshoni, and Edmond Sabo. Memory based active contour algorithm using pixel-level classified images for colon crypt segmentation. *Computerized Medical Imaging and Graphics*, 43:150–164, July 2015. 2

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005, IEEE, Computer Society Conference on Computer Vision and Pattern Recognition, CVPR, 05*. IEEE. 1

[3] Kaiming He et al. Mask r-cnn. 2017. 3

[4] Ross Girshick. Fast r-cnn. 2015. 3

[5] Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. nnU-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, Dec. 2020. 8

[6] KT.-Y. Lin, P. Doll, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. 2017. 3

[7] Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. 2015. 3

[8] Nathan Moon, Elizabeth Bullitt, Koen van Leemput, and Guido Gerig. Automatic brain and tumor segmentation. In *Medical Image Computing and Computer-Assisted Intervention , MICCAI, 2002*, pages 372–379. Springer Berlin Heidelberg, 2002. 1, 2

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing, 2015. 1, 2

[10] Özgün undefinediçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016. 8

[11] Zhou Yu-Cheng, Hu Zhen-Zhong, Yan Ke-Xiao, and Lin Jia-Rui. Deep learning-based instance segmentation for indoor fire load recognition. 2021. 3

[12] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. UNet: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE Transactions on Medical Imaging*, 39(6):1856–1867, June 2020. 3, 5