Skin Lesion Segmentation with U-Nets

Ramzi Charradi - Alaeddine Sabbagh

April 2019

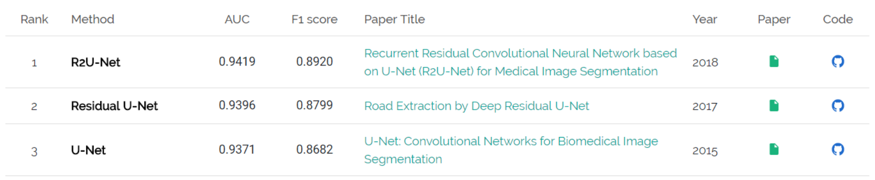
Contents

1 Introduction and objectif

The goal of this project is to implement the U-net deeplearning architecture based on the research paper:” Skin Lesion Segmentation: U-Nets versus Clustering” [1]. From a training set of 700 images of skin lesions we wish to train our U-net architecture to be able to extract binary masks that describe the lesion. Being the state of the art in many fields, U-net is used in our project to segment skin lesions. After many trials, We have eventually setteled for our own implementation.

2 State of the Art

Recently, fully convolutional networks (FCNs) have achieved great success in object detection and segmentation related problems [2]. The paper searches that yielded the best results in skin lesion competitions are the following [3]:



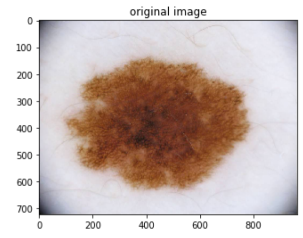
Rank	Method	AUC	F1 score	Paper Title	Year	Paper	Code
1	**R2U-Net**	0.9419	0.8920	Recurrent Residual Convolutional Neural Network based on U-Net (R2U-Net) for Medical Image Segmentation	2018	📄	○
2	**Residual U-Net**	0.9396	0.8799	Road Extraction by Deep Residual U-Net	2017	📄	○
3	**U-Net**	0.9371	0.8682	U-Net: Convolutional Networks for Biomedical Image Segmentation	2015	📄	○

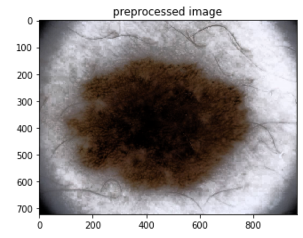
These state of the art methods rely all on the basic U-net netwok which we will explain in this report. The inspiration for those architectures are all taken from the same research paper: ”U-Net: Convolutional Networks for Biomedical Image Segmentation” [4] with some modifications. This research paper is also the premise of our work and We will further explore it.

3 Preprocessing

- We separated the images and their masks in two different folders.
- In order to increase the amount of contrast in the image, the image is converted to HSI color space, and histogram equalization is applied to the intensity (I) channel. After this, the image is converted back to RGB format. The reason for this equalization is that the contrast between the skin lesions (usually darker) is more noticeable compared to the surrounding skin (usually brighter).
- Final size of the input image should be resized to 128 $\times$ 128 pixels.

Here is an example of a preprocessed image:





4 U-Net Architecture

The U-NET was developed by Olaf Ronneberger et al. [5] for Bio Medical Image Segmentation. The architecture contains two paths. First path is the contraction path (also called as the encoder) which is used to capture the context in the image. The encoder is just a traditional stack of convolutional and max pooling layers. The second path is the symmetric expanding path (also called as the decoder) which is used to enable precise localization using transposed convolutions. Thus it is an end-to-end fully convolutional network (FCN), i.e. it only contains Convolutional layers and does not contain any Dense layer because of which it can accept image of any size.

In the original paper, the UNET is described as follows:

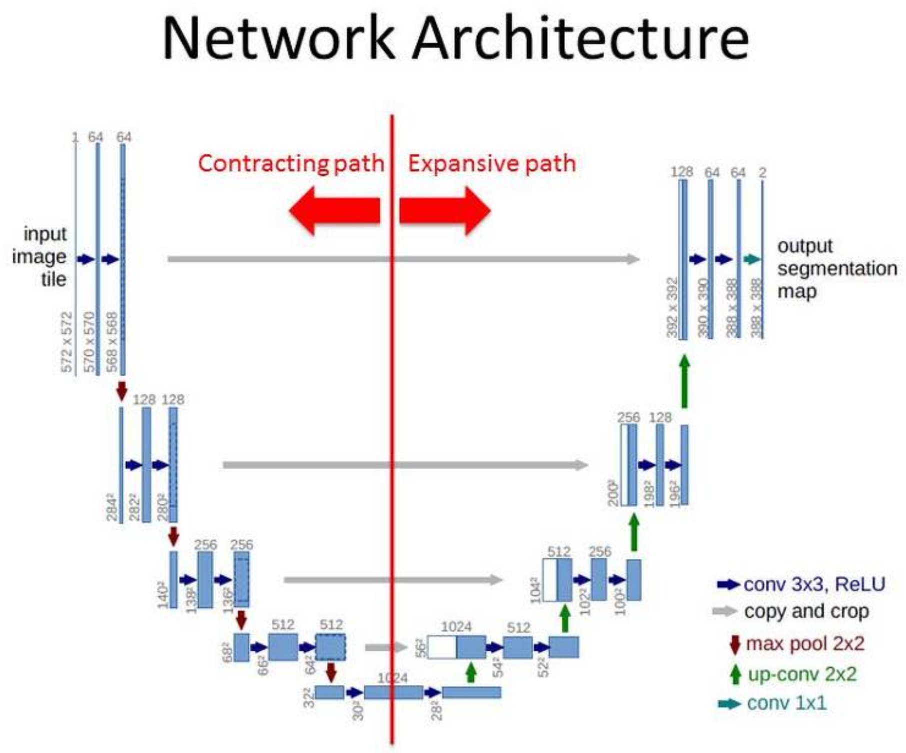


Figure 1: U-net architecture

Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on the top of the box. the x-y-size is provided at the lower edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

- The left hand side is the contraction path (Encoder) where we apply regular convolutions and max pooling layers. In the Encoder, the size of the image gradually reduces while the depth gradually increases. Starting from 128x128x3 to 8x8x256.
- This basically means the network learns the "WHAT" information in the image, however it has lost the "WHERE" information.
- The right hand side is the expansion path (Decoder) where we apply transposed convolutions along with regular convolutions.
- In the decoder, the size of the image gradually increases and the depth gradually decreases. Starting from $8{\times}8 \times 256$ to $128 \times 128 \times 1$.
- Intuitively, the Decoder recovers the "WHERE" information (precise localization) by gradually applying up-sampling
- To get better precise locations, at every step of the decoder we use skip connections by concatenating the output of the transposed convolution layers with the feature maps from the Encoder at the same level.

This is what gives the architecture a symmetric U-shape, hence the name UNET. On a high level, we have the following relationship: Input (128x128x1) $\Rightarrow$ Encoder $\Rightarrow$ $(8 \times 8 \times 256)$ $\Rightarrow$ Decoder $\Rightarrow$ Ouput $(128{\times}\ 128 \times 1)$

5 Our work strategy

5.1 Implementations found on Github

Our first strategy consisted of founding an implementation that works, then modifying it to our own use.

We started working separately, each one of us was tried 4 different implementations. Here are some examples:

- Unet with tensorflow [6]
- Unet using keras [7]
- Skin lesion CNN [8]

, but we encountred a lot of difficulties since we are not used to working with Tensorflow or Pytorch.
Difficulties mostly consisted in interpreting the code.

5.2 Our own implementation

After gettting a taste of deeplearning, Keras and Tensorflow, and after the Lab session, we decided to build our own implementation using Keras for the sake of simplicity. Our implementation used the same parameters and structure mentioned in the paper " Skin Lesion Segmentation: U-Nets versus Clustering" [1]. This implementation became eventually a success.

6 Results

6.1 Used Metrics

The Jaccard index, also known as Intersection over Union (originally given the French name coefficient de communauté by Paul Jaccard), is a statistic used for calculating the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between two finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

In our case,A is the predicted mask and B is the true mask. J(A,B) is the number of pixels on which they agree over the total number of pixels.

As a loss metric we have used ***The dice loss***, which is mathematically the opposite of the ***The dice coefficient***.
The discrete dice coefficient is :

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

In other words:

$$DSC = \frac{2TP}{2TP + FP + FN}.$$

6.2 Results

6.2.1 U-net with Batch Normalization

Training the basic U-net CNN is so slow. That's why we have thought of using some Batch Normalizatio layers. Indeed, training got faster, and we were able to try different parameters. Overall our results ressemble to the true masks but are kind of smooth. Here are some examples where:

After training on 700 images we got a Jaccard value of 0.67.

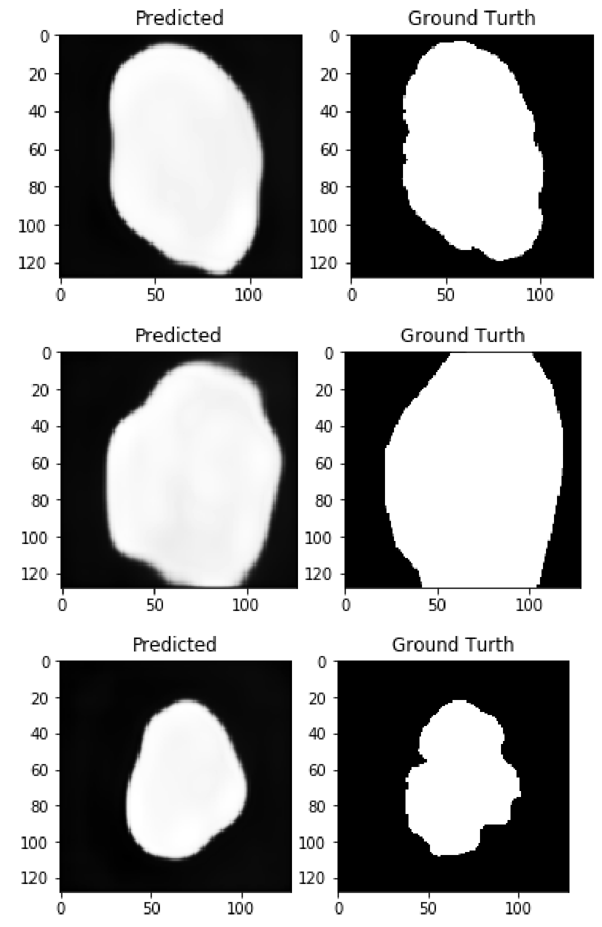
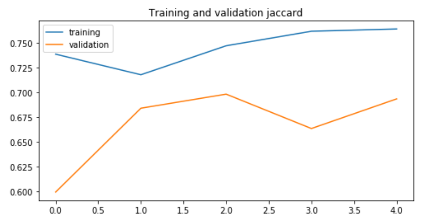
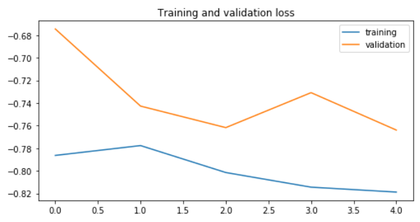


Figure 2: 3 examples with U-net that contains Batch Normalized layers

Here is the evolution of our metrics in function of the number of epochs:





6.2.2 Basic U-net

We were also able to test U-net version as it is written in the paper [1], but we were not very flexible with running it many times to try different parameters. Here are some examples:

Results are sharper and more precise when it comes to the contours. Our Jaccard index gave us 0.62.

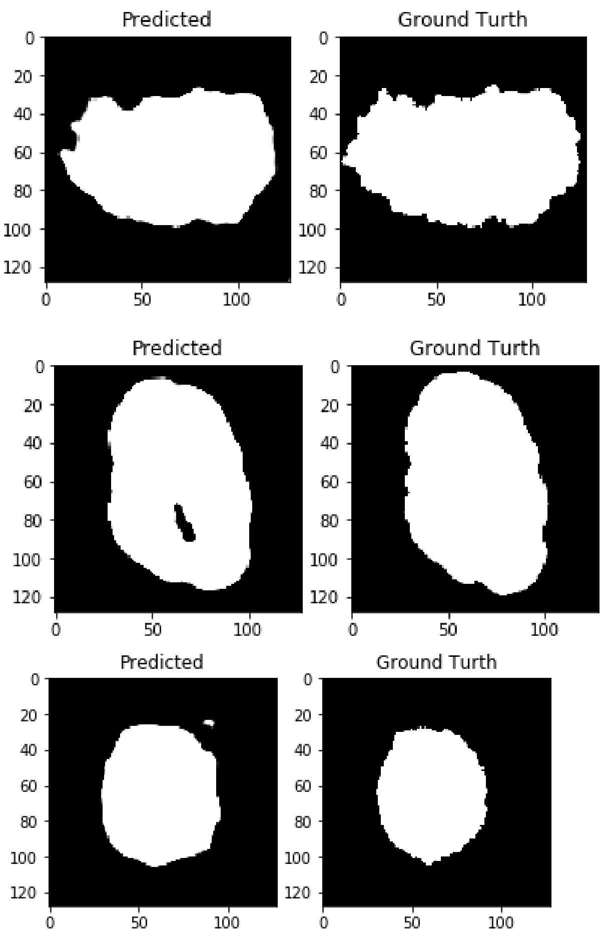


Figure 3: 3 examples with basic U-net

7 Conclusion and perspective

We have managed to implement the U-net architecture. Our final results are visually acceptable: it is exactly the true shape, with some smoothing along the contours. Our explaination is that, when doing Batch Normalization we loose some information which caused the loss of precision along the contours. To improve the final results, we can merge results obtained by our both U-net architectures to get a more precise result: One describes the overall shape, and the other better describes the contours.
Another perspective would be to adapt the U-net architecture to perform clustering. We can replace the last convolution by k (number of classes) filters with size (1,1) each.

References

[1] https://perso.telecom-paristech.fr/ytendero/ima_206_project_papers/lin2017.pdf.

[2] https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf.

[3] https://paperswithcode.com/sota/skin-cancer-segmentation-on-kaggle-skin.

[4] https://arxiv.org/pdf/1505.04597v1.pdf.

[5] https://lmb.informatik.uni-freiburg.de/people/ronneber/.

[6] https://github.com/jakeret/tf_unet.

[7] https://github.com/zhixuhao/unet.

[8] https://github.com/adriaromero/BSc_Thesis_Skin_Lesion_CNN.