

# UE PRAT

## **U-Net : Réseau convolutionnel pour la segmentation d'images biomédicales**

Yanis Chemli UPMC/Telecom ParisTech

**Résumé** Ce document vise à exposer une technique de segmentation décrite dans [1]. Les réseaux de neurones profonds ont montré leur efficacité dans de nombreuses taches de vision et traitement d'images. Elles ont permis de surpasser les résultats de nombreuses techniques, notamment pour la segmentation d'images. Mais, pour être efficaces, il leur faut une importante base d'apprentissage. Récemment une nouvelle approche a été présentée reposant sur une nouvelle architecture d'un réseau convolutionnel pour segmenter des images biologiques. Cette architecture en forme de U, se compose d'une première phase de contraction de l'image pour capturer le contexte et dans une seconde phase symétrique d'expansion pour permettre une bonne localisation. Nous introduisons dans cet écrit certains aspects de la segmentation et de la segmentation d'images biomédicales. Nous présentons ensuite les enjeux et certaines techniques d'apprentissage profond. Nous discuterons de l'intérêt et la pertinence de cette nouvelle architecture de réseau. Enfin, nous exposerons et commenterons des résultats de segmentations sur différentes bases d'images à l'aide du réseau U-Net.

**Abstract** Deep neural networks have proven their efficiency in many fields of computer vision and image processing. They have been used for image segmentation and outperformed state of the art technics. But to be effective, they need a large learning data base. Recently a new approach have been presented based on data augmentation to enable effective use a low volume of training data (annotated data available) and on a new network architecture for segmenting biological images. This U-shaped architecture, consists of a first image contraction phase to capture the context and in a symmetrical second phase of expansion to allow good localization. We introduce in this writing some issues of segmentation especially for biomedical images using conventional and advanced methods. Then, we present the issues and some deep learning techniques and discuss the contribution and the relevance of this new network architecture. Finnaly we present and comment some segmentation results on different databases done with the U-Net architecture.

## Table des matières

1	La segmentation.....	7
1.1	Introduction sur la segmentation d'images .....	7
1.2	La segmentation d'images biomédicales.....	8
2	Les réseaux convolutionnels .....	12
2.1	Généralités sur l'apprentissage profond ou deep learning.....	12
2.2	Intérêt du deep learning .....	13
2.3	Segmentation d'images à l'aide de réseaux convolutionnels .....	14
3	Une architecture nouvelle: U-net pour la segmentation d'images biomédicales.....	15
3.1	L'architecture en U .....	17
3.1.1	Convolution.....	18
3.1.2	Max-Pooling .....	19
3.1.3	ReLU.....	19
3.2	Augmentation de données .....	20
4	L'apprentissage des réseaux convolutionnel : optimisation d'une fonction de cout par descente de gradient .....	21
5	Mise en œuvre.....	23
5.1	Environnement de développement .....	23
5.2	Une architecture modifiée.....	24
5.3	Traitements des données d'entrée et régularisations .....	27
5.4	Matériel.....	29
6	Expériences et testes.....	30
6.1	Expérimentation 1 : Base d'image DRIVE .....	30
6.2	Expérimentation 2 : base Cellules.....	36

6.2.1	Synthèse des résultats de précision.....	43
	Conclusion.....	44
	Références .....	45
	Annexe.....	48

<i>Figure 1 Exemple de segmentation d'image (Ligne de partage des eaux)</i>	
<i>http://www.image-net.com/WebHelp6/Default.htm#PnPMethods/Watershed_Segmentation.htm.....</i>	7
<i>Figure 2 Perceptron à une couche cachée [8].....</i>	10
<i>Figure 3 Exemple possible d'une image segmenté à l'aide d'un réseau de neurone https://blog.altoros.com/experimenting-with-deep-neural-networks-for-x-ray-image-segmentation.html.....</i>	11
<i>Figure 4 Les premières étapes d'un réseau convolutionnel (https://upload.wikimedia.org/wikipedia/commons/6/63/Typical_cnn.png) ..</i>	13
<i>Figure 5 Approche de segmentation par approche CNN [14] .....</i>	14
<i>Figure 6 Sortie différente entre un "Fully connected" et un "Fully convolutional".</i>	
<i>https://wwwf.imperial.ac.uk/~gmontana/talks/crfasrnns_presentation.pdf.....</i>	15
<i>Figure 7 Cascaded hierarchical model (CHM) [21].....</i>	16
<i>Figure 8 Architecture U-net [1].....</i>	18
<i>Figure</i>	9
<i>https://upload.wikimedia.org/wikipedia/commons/e/e9/Max_pooling.png.....</i>	19
<i>Figure</i>	10
<i>https://upload.wikimedia.org/wikipedia/en/6/6c/Rectifier_and_softplus_functions.svg .....</i>	19
<i>Figure 11 Exemple de transformations pour l'augmentation de données.</i>	
<i>(https://moodle.technion.ac.il/mod/resource/view.php?id=39094&amp;lang=en) 20</i>	
<i>Figure 12 Image histologique (http://codexvirtualis.fr/) .....</i>	21
<i>Figure 13 Vue schématique de la backpropagation sur un réseau de neurones.</i>	
<i>On commence par calculer la dérivée du coût <math>L</math> par rapport à la sortie (<math>y_2</math> sur le schéma) puis on remonte dans le réseau en réutilisant les dérivées calculées précédemment lors du calcul des nouvelles dérivées. ....</i>	22
<i>Figure 14 Exemple d'images à segmenter par U-Net .....</i>	25
<i>Figure 15 Exemple d'image à segmenter : vaisseaux sanguins .....</i>	25
<i>Figure 16 Un autre type de donnée : image de cellules .....</i>	26
<i>Figure 17 Un autre type de donnée : Segmentation des noyaux marqués en marron.....</i>	26

<i>Figure 18 Architecture simplifiée utilisée.....</i>	27
<i>Figure 19 Image prétraitée, carte de segmentation vérité associée, masque de champ de vue, exemple de patch en orange.....</i>	28
<i>Figure 20 Exemples de patches à donner au réseau.....</i>	29
<i>Figure 21 Courbe roc et aire en dessous de la courbe.....</i>	32
<i>Figure 22 Synthèse des résultats optimaux des travaux d'orobix.....</i>	32
<i>Figure 23 Evolution de la précision et fonction de cout (139 000 patches, 60 epochs).....</i>	33
<i>Figure 24 Synthèse des résultats en teste pour 139 000 patches.....</i>	34
<i>Figure 25 Matrice de confusion 139 000 patches.....</i>	34
<i>Figure 26 Exemple de résultats de segmentation issu de la base de test (l'entrée à gauche, la vérité au centre et le résultat à droite) .....</i>	34
<i>Figure 27 Filtres, couche de convolution 1.....</i>	35
<i>Figure 28 Filtres, couche de convolution 5.....</i>	36
<i>Figure 29 Exemple d'une image de la base "Cellules".....</i>	37
<i>Figure 30 Exemples de patches pour la base "Cellules" .....</i>	37
<i>Figure 31 Evolution de la fonction de coût, 80 epochs.....</i>	38
<i>Figure 32 Sythsèse des résultats 190 000 patches 80 epochs .....</i>	38
<i>Figure 33 Matrice de confusion, 190 000 patches 80 epochs.....</i>	39
<i>Figure 34 exemple de segmentation sur la base cellules avec le paramétrage initial .....</i>	39
<i>Figure 35 Résultats du fine tunning.....</i>	40
<i>Figure 36 Visualisation du résultat de segmentation, les trous dans les cellules sont presque comblés(ancien en bas à gauche, nouveau en bas à droite) .....</i>	41
<i>Figure 37 Filtres, couche de convolution 3 base "Cellules" .....</i>	42
<i>Figure 38 Filtres, couche de convolution 5 base "Cellules" .....</i>	42
<i>Figure 41 Projection des lignes épipolaires après estimation de F...Erreur !</i>	
<i>Signet non défini.</i>	

# 1 La segmentation

## 1.1 Introduction sur la segmentation d'images

La segmentation d'images est l'un des problèmes les plus courant en traitement d'images. Son but consiste à partitionner l'image en un ensemble de régions connexes. La segmentation est alors utilisée pour isoler des régions du fond afin de faciliter certaines opérations de traitement ou d'analyse d'image comme la détection, l'identification, le suivi et beaucoup d'autres tâches de haut niveau.

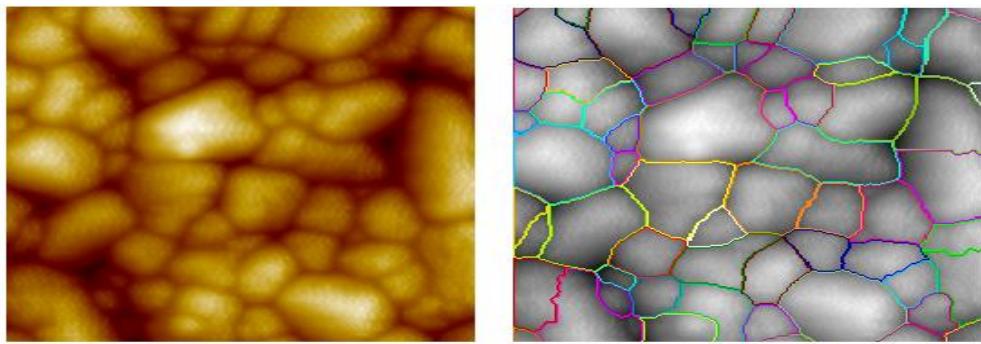


Figure 1 Exemple de segmentation d'image (Ligne de partage des eaux)  
[http://www.imagemet.com/WebHelp6/Default.htm#PnPMethods/Watershed\\_Segmentation.htm](http://www.imagemet.com/WebHelp6/Default.htm#PnPMethods/Watershed_Segmentation.htm)

Segmenter une image signifie donc trouver toutes les régions homogènes ainsi que ses contours. Ces régions et contours sont supposées être pertinentes. Les régions doivent correspondre aux parties significatives des objets et les contours à leurs contours apparents.

Une définition formelle de la segmentation a été donnée par Horowitz et Pavlidis [2] en 1975 :

*Soit  $X$  le domaine de l'image et  $f$  la fonction qui associe à chaque pixel une valeur  $f(x,y)$ . Si nous définissons un prédictat  $P$  sur l'ensemble des parties de  $X$ , la segmentation de  $X$  est définie comme une partition de  $X$  en  $n$  sous-ensemble  $\{R_1, \dots, R_n\}$  tels que :*

- 1-  $X = \bigcup_{i=1}^n R_i$ ,
- 2-  $\forall i \in \{1, \dots, n\} R_i$  est connexe,
- 3-  $\forall i \in \{1, \dots, n\} P(R_i) = \text{vrai}$ ,
- 4-  $\forall i, j \in \{1, \dots, n\} {}^2 R_i$  est adjacent à  $R_j$  et  $i \neq j \Rightarrow P(R_i \cup R_j) = \text{faux}$ .

C'est un problème mal posé dans la mesure où, dans le cas général, il n'existe pas de segmentation optimale.

La segmentation est généralement basée sur deux propriétés propres aux images :

- La discontinuité,
- La similarité.

La propriété de discontinuité implique une segmentation basée sur les changements abruptes d'intensité dans l'image. La seconde propriété permet une segmentation de l'image en différentes régions, selon une similarité liée à intensités de pixels. Il existe de très nombreuse méthode et approche de segmentation. Dans le cadre de notre étude, nous allons nous restreindre aux méthodes les plus utilisés dans le domaine du biomédical.

## 1.2 La segmentation d'images biomédicales

Etant l'une des étapes les plus importantes dans l'imagerie biomédicale, la segmentation va permettre de fournir une base pour des études quantitatives et de diagnostics. Une grande variété de techniques existe au sein même de ce domaine. Elles sont souvent spécifiques à la modalité et dépendent du principe physique et des caractéristiques des systèmes d'acquisition.

Les classes les plus courantes de nos jours regroupent:

- les approches morphologiques,
- les approches par régions (histogrammes, croissance de régions...),
- les approches statistiques (bayésiennes, EM, classification, champs de Markov...),
- les approches par modèles déformables,
- les approches par modèles (de forme, atlas...).

Chacune de ces classes d'approche tente de répondre à certaines problématiques telles que la robustesse aux différents types de bruits, à la présence d'artefacts, d'inhomogénéités en intensités ou d'occlusions.

On peut noter des méthodes couramment utilisées telles que les méthodes basées contours, les méthodes basées régions, les modèles déformables (Snakes, active contour) [4], les méthodes utilisant des modèles d'Atlas [4] etc.

Beaucoup de ces méthodes nécessitent une intervention extérieure (initialiser un contour grossier, placer un modèle etc.) ou n'arrivent pas à faire face à certaines des problématiques précédemment énoncées.

Des méthodes plus récentes tentent d'automatiser et de rendre plus robuste le processus de segmentation en utilisant des techniques d'intelligence artificielle. Basée sur des procédés d'apprentissage, on peut les classer en méthode non supervisés et supervisés.

La plupart des algorithmes non supervisés sont basés sur des méthodes par clustering type K-Means ou C-Means [5]. Pour ces techniques, le but est d'établir des frontières de décision reposants sur des données d'entraînement non labellisées. On cherche alors à déterminer des groupes dans l'espace des classes. La segmentation d'image peut être considérée comme un processus de clustering dans lequel les pixels sont classés dans des classes. Ces classes sont déterminées à l'aide de vecteurs caractéristiques (SIFT par exemple) calculés sur des pixels des images d'entraînement. Le clustering flou (Fuzzy C-Means) est une méthode pour classer les données en considérant la probabilité d'appartenance d'un pixel à une classe [5].

Ces algorithmes sont cependant des heuristiques, ils permettent de répondre à un problème difficile, mais ne donne pas de solution optimale. Ils dépendent en plus de conditions aléatoires initiales.

Les méthodes supervisées, qui utilisent des données labélisées, peuvent aussi être utilisées pour des problèmes de segmentation. On peut noter ces trois classificateurs supervisés très populaires qui ont fait leurs preuves et qui sont largement utilisées : la méthode des k plus proches voisins, les réseaux de neurones artificiels et les machines à vecteurs supports (SVM) [6].

Nous nous concentrerons ici sur les méthodes à base de réseaux de neurones qui sont la base du sujet d'étude.

Les réseaux de neurones artificiels constituent une classe d'outils de modélisation inspirés du fonctionnement du cerveau. Introduits dans [7], le principe est d'interconnecter de nombreuses unités de calculs simples, appelées neurones artificiels, afin de permettre la résolution de problèmes difficiles, non linéaire et multi-classes.

Les principaux avantages des réseaux de neurones artificiels sont :

- Leur aptitude à apprendre de façon adaptative, en utilisant des données d'entraînement pour résoudre des problèmes complexes,
- Leur capacité « d'auto-organisation » ; Ils peuvent mettre à jour leur organisation en fonction de l'information reçue lors de l'apprentissage,

- Leur capacité de performance en temps réel (de par leur nature ces architectures sont facilement parallélisables).

Le perceptron est un réseau de neurone largement utilisé pour la classification et la segmentation. C'est la structure la plus simple d'un réseau de neurone. Elle est composée d'une couche d'entrée, d'une couche cachée et d'une couche de sortie. [8]

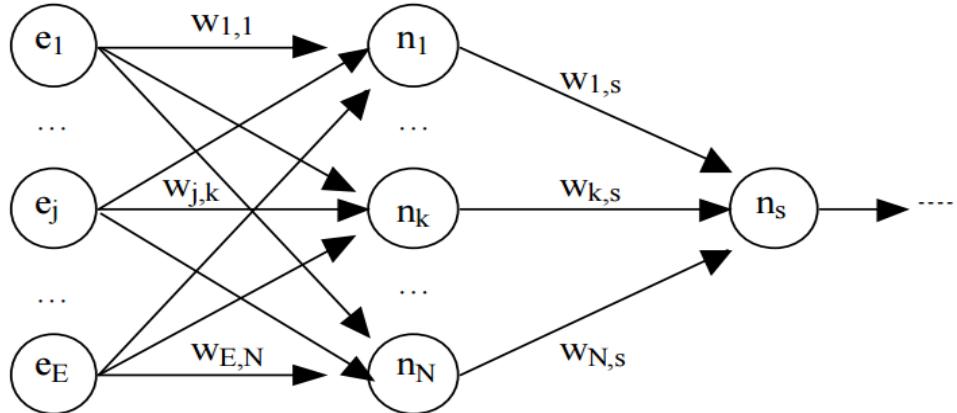
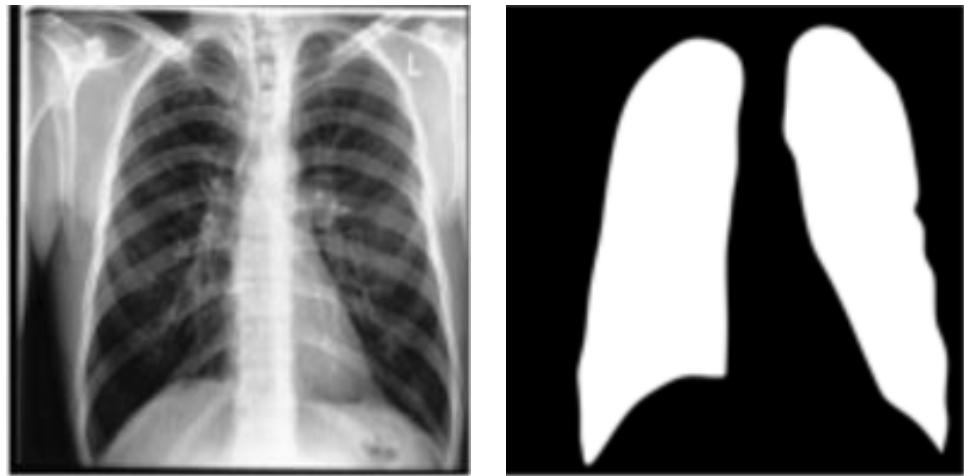


Figure 2 Perceptron à une couche cachée [8]

Chaque sortie est alors pondérée par un poids  $w$ . Usuellement un biais intervient et l'on propage les informations de l'entrée vers la sortie par une fonction d'activation (souvent non linéaire, type sigmoïde, ReLu ou tanh).

Pour un problème de classification, nous disposons de  $N$  exemples d'apprentissage, chacun étant une paire  $(x, y)$  constituée d'une donnée d'entrée  $x$  et de la classe vérité  $y$  associée. C'est à partir de ces données que le réseau de neurone peut être appris (c'est-à-dire que ses paramètres, poids et biais, sont ajustés aux données d'entraînement dans l'espoir d'une capacité de généralisation). La méthode de rétro propagation du gradient (ou back propagation) permet de réaliser cette opération.

Concrètement, la segmentation est réalisée par classification des pixels d'une image. Pour effectuer l'apprentissage, on présente au réseau une image à segmenter (donc classer les pixels). A l'issue de ce premier apprentissage, les poids et les biais du réseau de neurone sont fixés, et l'image est présentée au réseau pour être segmentée. Le résultat est comparé avec une segmentation manuelle, pour en tirer une image d'erreur de segmentation. Le processus est itéré jusqu'à ce que l'erreur globale de segmentation se stabilise. [8]



*Figure 3 Exemple possible d'une image segmenté à l'aide d'un réseau de neurone <https://blog.altoros.com/experimenting-with-deep-neural-networks-for-x-ray-image-segmentation.html>*

Les réseaux de neurones à une couche cachée ont fait leurs preuves en segmentation ces dernières années. Cependant ils demandent une base d'apprentissage conséquente et donc un temps de travail humain considérable.

De plus, l'algorithme de rétro-propagation souffre de deux inconvénients majeurs :

- L'expérience montre que le temps d'entraînement d'un RN croît rapidement lorsque le nombre de couches augmente. C'est d'ailleurs l'une des raisons pour lesquelles à partir des années 1990 les RN ont été remisés au profit d'autres algorithmes non-linéaires moins gourmand en ressources comme les SVM,
- Les RN n'échappent pas au problème central du Machine Learning : le sur-apprentissage (overfitting).

Le challenge est donc de réduire au maximum le nombre d'exemples d'apprentissage tout en évitant le sur-apprentissage. Des méthodes récentes sur lesquels se basent notre étude tentent de réaliser cette opération.

Dans la suite nous allons plus loin dans les méthodes à base d'apprentissage. Nous présentons des méthodes basées sur les réseaux de neurones artificiels appelé réseaux de neurones convolutionnels (CNN). Avant d'aborder le sujet des CNN, une introduction sur l'apprentissage profond ou deep learning est

nécessaire. Nous verrons en quoi l'utilisation de réseaux profonds est un avantage face aux réseaux classiques type perceptrons.

## 2 Les réseaux convolutionnels

### 2.1 Généralités sur l'apprentissage profond ou deep learning

Nous avons présenté précédemment les structures de réseaux de neurones pour la classification et la segmentation. On appelle réseau de neurones profonds un réseau qui possède plus d'une couche cachée. Le but théorique est alors d'approximer n'importe quelle fonction.

Cette famille d'algorithmes a permis de faire des progrès importants dans les domaines de la reconnaissance d'images et du traitement des images en général, notamment en segmentation.

Les modèles de réseaux de neurones profonds sont bâtis sur le même modèle que les perceptrons précédemment décrits. Cependant, on fait intervenir des couches intermédiaires plus nombreuses.

Chacune des couches intermédiaires cachées va être subdivisée en sous partie, traitant un sous problème plus simple et fournissant le résultat à la couche suivante et ainsi de suite.

Cette manière d'ordonner les déductions amènent à ajouter au fur et à mesure un contexte de plus en plus précis au sujet sur lequel le modèle opère. [9]

Il existe différents algorithmes de deep learning. Nous pouvons ainsi citer :

- Les réseaux de neurones convolutionnels (CNN ou Convolutional Neural Networks). Le problème est divisé en sous parties et pour chaque partie, un « cluster » de neurones sera créé afin d'étudier une portion spécifique. Par exemple, pour une image en couleur, il est possible de diviser l'image sur la largeur, la hauteur et la profondeur (les couleurs). [10]
- La machine de Boltzmann profonde (Deep Belief Network, G. Hinton) : Ces algorithmes fonctionnent suivant une première phase non supervisée, suivi de l'entraînement classique supervisé. Cette étape d'apprentissage non-supervisée, permet, en outre, de faciliter l'apprentissage supervisé. [11]

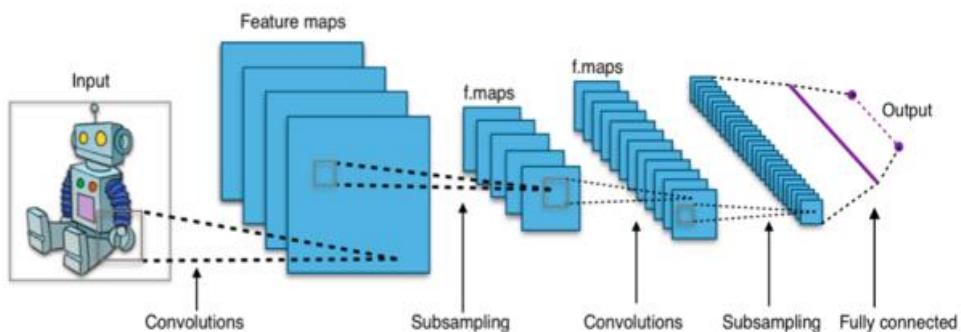
Nous détaillons dans la suite les réseaux de neurones convolutionnels sur lequel se base l'étude (il existe cependant une différence au niveau de l'architecture par rapport au CNN classique que nous mettrons en avant).

Les réseaux convolutionnels (ou ConvNet) sont des types de réseaux de neurones artificiels dans lesquels le motif de connexion entre les neurones est inspiré par le cortex visuel des animaux (du chat particulièrement). Leur fonctionnement est

inspiré par les processus biologiques, ils consistent en un empilage multicouche de perceptrons, dont le but est de prétraiter de petites quantités d'informations. [12]

Un des intérêts particuliers des réseaux convolutifs est leur capacité à apprendre les représentations (features) par elles-mêmes à partir des données.

Dans les architectures classiques de réseaux de neurones convolutionnels, les couches de convolution sont suivies par des couches de sous-échantillonnage. Une couche de sous-échantillonnage réduit la taille des entrées convolées et introduit de l'invariance aux faibles rotations et translations pouvant apparaître en entrée. Une couche de convolution est paramétrée par son nombre  $N$  de cartes de convolution, la taille des noyaux de convolution (souvent carrée), et le schéma de connexion à la couche précédente. Chaque carte de convolution est le résultat d'une somme de convolution des cartes de la couche précédente par son noyau de convolution respectif. Un biais  $b$  est ensuite ajouté et le résultat est passé à une fonction d'activation. [13]



*Figure 4 Les premières étapes d'un réseau convolutionnel  
[https://upload.wikimedia.org/wikipedia/commons/6/63/Typical\\_cnn.png](https://upload.wikimedia.org/wikipedia/commons/6/63/Typical_cnn.png)*

Le début de l'architecture consiste donc en une succession de couches de convolution, sous échantillonnage (subsampling) et fonctions d'activation. Ces enchaînements sont dédiés à l'extraction automatique de caractéristiques, tandis que la seconde partie, composée de couches de neurones complètement connectés (fully connected), est dédiée à la classification.

## 2.2 Intérêt du deep learning

L'approche deep learning est pertinente dans la mesure où elle permet d'approximer des fonctions complexes avec la même précision qu'un réseau de neurone classique en utilisant plus de couche mais finalement beaucoup moins de noeuds neuronaux. On fait donc intervenir moins de paramètres donc moins de

degrés de libertés. Outre les avantages computationnels, un modèle avec moins de degrés de libertés nécessite moins de données d'apprentissage. Or on a vu que le nombre d'élément d'une base d'entraînement peut être un facteur limitant. [15] La raison pour laquelle un réseau moins large en termes de nombre de neurones par couche, mais plus profond est plus efficace qu'un réseau moins profond à taille égale (en nombre de neurones) est qu'un réseau profond réduit la quantité de travail redondant effectué.

Pour une image contenant par exemple des formes géométriques dont nous aimions détecter et identifier le type, les premières couches d'un réseau profond effectueront des tâches de bas niveaux dont l'extraction de caractéristiques tel que les gradients, les formes élémentaires etc. pour la transformer en une image plus « concrète » et exploitable au regard du réseau et de la prise de décision. On fait donc un apprentissage des caractéristiques à différents niveaux d'abstractions. Les couches supérieures peuvent alors effectuer la détection et la classification à partir de la représentation plus simple de l'image.

Pour un réseau classique, type perceptron, les tâches de bas niveau devront être effectuées à plusieurs reprises, car il y a peu de propagation de résultats intermédiaires. Cela entraîne de la redondance. [15]

### 2.3 Segmentation d'images à l'aide de réseaux convolutionnels

On peut grâce aux réseaux convolutionnels segmenter des images biomédicales. Dans [14] Ciresan et al utilisent une architecture de deep learning comme classifieur de pixels. La sortie du réseau indique la probabilité d'appartenance d'un pixel à une classe. L'image est alors segmentée en classant tous ses pixels. On a vu précédemment la nécessité de l'intervention humaine pour l'établissement de vérités terrains lors de l'utilisation de méthodes supervisées. C'est le cas dans la méthode proposée où le réseau profond peut être entraîné avec des données manuellement annotées.

Le réseau prend en entrée les données brutes de l'image ou des données pré traitées. La plupart des méthodes de segmentation basées sur les CNN prédise la classe d'un pixel en donnant en entrée au réseau un patch centré en ce pixel.

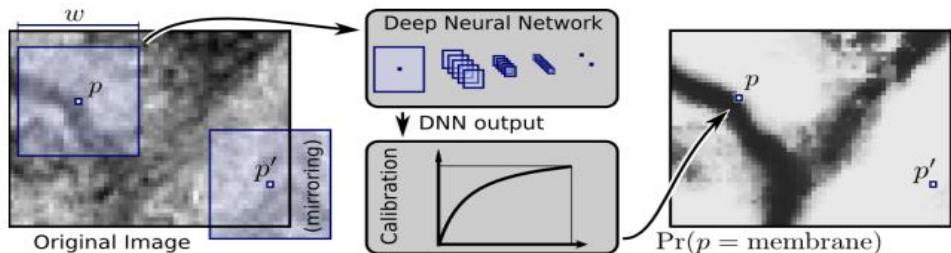
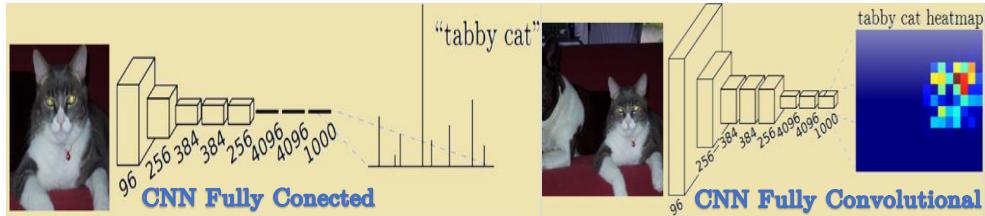


Figure 5 Approche de segmentation par approche CNN [14]

Plusieurs méthodes de segmentation utilisant des approches à base de réseaux profonds ont été récemment développées, particulièrement dans le domaine du biomédicale. On trouve dans la littérature des articles mettant en avant les atouts de ce type de réseau pour la segmentation de tumeur cérébrale [16], pour la classification cellulaire [17] ou la segmentation anatomique du cerveau [18]. D'autres méthodes sont proposées en segmentation, notamment pour la segmentation sémantique. Le but est alors d'attribuer un label aux éléments segmentés de l'image [19].

Ces méthodes, par rapport à ce qui se fait usuellement avec les CNN, c'est-à-dire un enchainement de convolutions/sous-échantillonnage/Fonction d'activation puis une connexion avec un « fully connected » propose une approche différente. En effet, la sortie d'un réseau donne classiquement un score, une probabilité, un nom de classe etc. Or ici, on cherche à récupérer une information structurelle. Les articles mentionnés [13, 15, 16, 17] font abstraction de la phase de connexion à un fully connected et utilise alors l'approche « fully convolutionnal network » [22]. En procédant ainsi, la sortie du réseau peut être vue comme une carte de chaleur indiquant la position du pixel et sa classe (ou la probabilité d'appartenance à la classe).



*Figure 6 Sortie différente entre un "Fully connected" et un "Fully convolutional".*

<https://wwwf.imperial.ac.uk/~gmontana/talks/crfasrnnpresentation.pdf>

### 3 Une architecture nouvelle: U-net pour la segmentation d'images biomédicales.

Basé sur les réseaux convolutionnels O. Ronneberger et al proposent une nouvelle architecture (2015) permettant l'utilisation d'un nombre réduit de données d'entraînement.

La qualité principale de l'approche de Ciresan, présenté ci-dessus, est de fournir à la sortie une information spatiale et non pas qu'un label de la classe à laquelle

appartient l'image. On associe un label à chacun des pixels de l'image. Pour cela, ce n'est pas l'image entière qui est passée comme entrée du réseau, mais des patches centrés en chacun des pixels à classer.

Cela a pour effet de multiplier la taille des données d'apprentissage, mais simultanément d'augmenter le temps de calcul. Parcourant chacun des pixels avec des patches qui se chevauchent, on fait intervenir en plus des données redondantes. De plus, en procédant localement, on perd les informations contextuelles pouvant être utile à la segmentation finale.

Il y a donc un compromis à faire entre : définir une taille élevée des patchs pour garder une information de la structure spatiale (on perd alors en précision) ou définir une taille réduite des patchs pour gagner en précision (on perd ici de l'information contextuelle).

Seyedhosseini & et al dans [21] pallient ce problème par leur méthode « cascaded hierarchical model (CHM) » qui apprend l'information contextuelle de l'image de façon hiérarchique. A chaque niveau de la hiérarchie, un classifieur (quelconque) est entraîné sur des images sous-échantillonées ainsi que sur les résultats des max-pooling (agrégation) des étapes précédentes.

Les sorties des classificateurs sont alors sur-échantillonées à la résolution originale de l'image d'entrée et sont utilisées pour entraîner un nouveau classificateur.

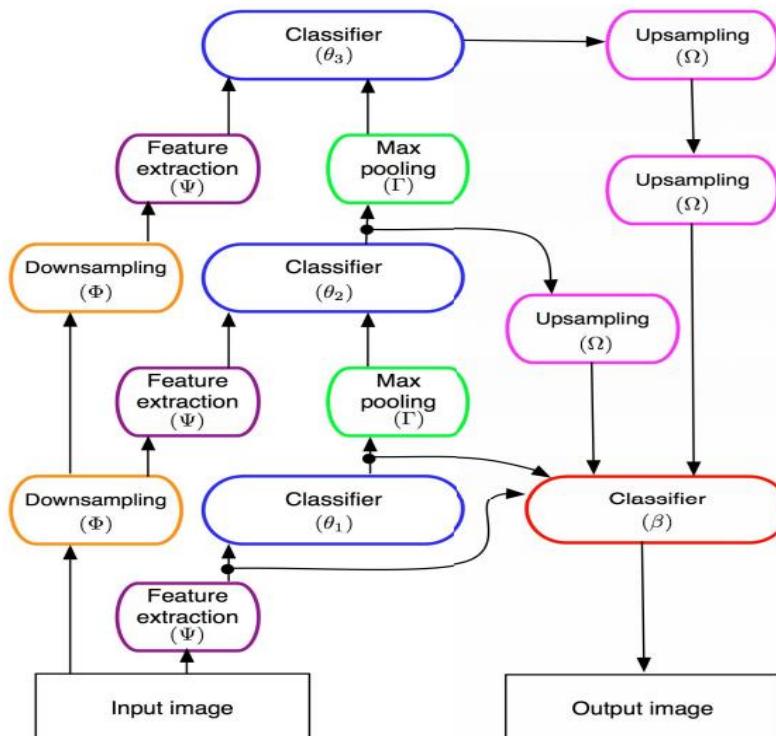


Figure 7 Cascaded hierarchical model (CHM) [20]

Appliquée à la segmentation, cette technique permet une localisation précise et permet de garder le contexte spatial de l'image. Cependant, elle demande d'entraîner plusieurs classificateurs. Pour un nombre de 328 images (en moyenne de taille 250 x 250), il faut 35 heures d'entraînement.

O. Ronneberger & al [1] propose alors d'adapter cette approche par l'utilisation d'un réseau convolutionnel.

### 3.1 L'architecture en U

L'idée est de reprendre l'architecture de Seyedhosseini & et al où chaque classifieur peut être approché par une couche d'un réseau convolutionnel.

Ainsi, en entré du réseau l'image à segmenter est convolée par 64 filtres avec un stride (ou pas) de 1, suivit d'un passage à la fonction d'activation ReLu. Cette étape est répétée à deux reprises. Il est à noter que les bords des images et des cartes générées ne sont pas gérés.

Après les premières étapes de convolution et de passage à la ReLu, un max-pooling est réalisé avec un stride de deux. Cela a pour effet de sous-échantillonner la donnée précédente par un facteur deux. On procède ainsi jusqu'à l'obtention d'une carte de taille 32x32 (taille arbitraire).

On a vu que dans le cas d'un réseau CNN classique, on connecterait cette couche à un « fully connected », cependant, ici, O. Ronneberger & al sont dans le cadre d'une approche « fully convolutionnal ».

À la façon du modèle hiérarchique en cascade, la donnée est ensuite sur-échantillonnée. On procède alors à des itérations de convolution, ReLu et sur-échantillonnage en propageant les données contextuelles jusqu'à obtenir une carte de segmentation. Cette carte de profondeur 2 indique la position des pixels et la classe associée.

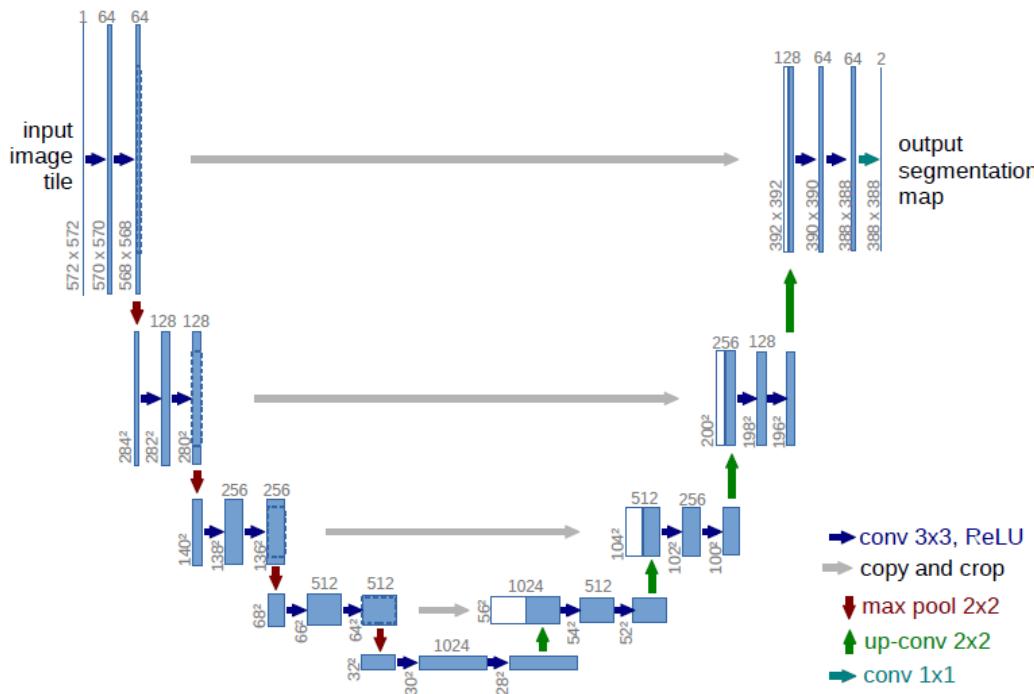


Figure 8 Architecture U-net [1]

On obtient donc cette architecture en U composée d'une phase de contraction puis d'une phase d'expansion.

On décrit ci-dessous les étapes de convolution, d'activation ReLu et de max-pooling.

### 3.1.1 Convolution

Une couche de convolution est paramétrée par son nombre N de cartes de convolution, la taille des noyaux de convolution (souvent carrée), et le schéma de connexion à la couche précédente. Chaque carte de convolution est le résultat d'une somme de convolution des cartes de la couche précédente par son noyau de convolution respectif. Les convolutions du réseau en U sont au nombre de 64 par couche et ont une taille fixe de 3x3.

### 3.1.2 Max-Pooling

On a vu précédemment que dans les architectures classiques, les couches convolées sont suivies par une étape de sous échantillonnage. Dans la méthode en U, une autre approche est proposée. C'est une technique existante qui a fait ses preuves [20]. Une couche de sous échantillonnage est alors remplacé par la couche de max-pooling. La sortie d'une couche de max-pooling est donnée par la valeur maximale d'activation au sein de la couche d'entrée pour différentes régions.

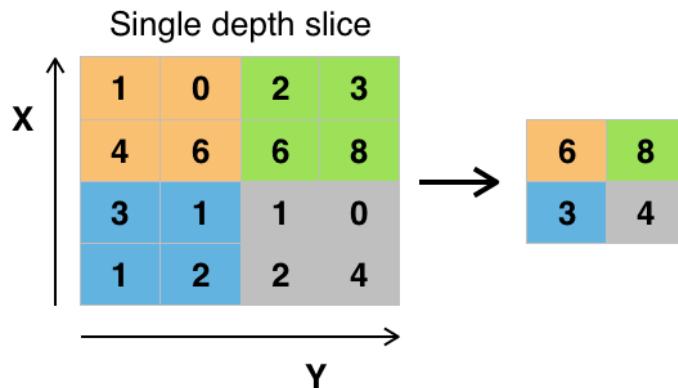


Figure 9

[https://upload.wikimedia.org/wikipedia/commons/e/e9/Max\\_pooling.png](https://upload.wikimedia.org/wikipedia/commons/e/e9/Max_pooling.png)

### 3.1.3 ReLu

La fonction ReLu est la fonction d'activation utilisée après chaque convolution. Elle est définie par :

$$f(x) = \max(0, x)$$

Avec  $x$ , l'entrée d'un neurone. Elle a pour représentation graphique :

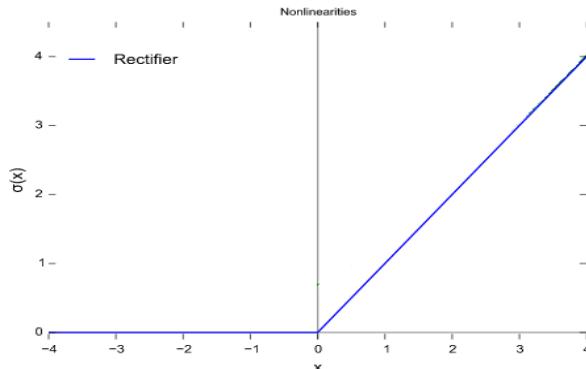


Figure 10

[https://upload.wikimedia.org/wikipedia/en/6/6c/Rectifier\\_and\\_softplus\\_functions.svg](https://upload.wikimedia.org/wikipedia/en/6/6c/Rectifier_and_softplus_functions.svg)

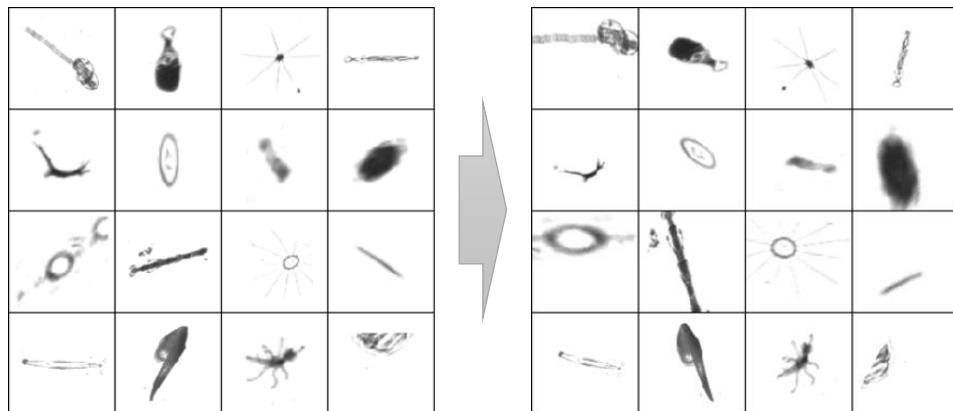
Cette fonction, étant aussi appelée "fonction d'activation non saturante", augmente les propriétés non linéaires de la fonction de décision et de l'ensemble du réseau sans affecter les champs récepteurs de la couche de convolution.

Une étape primordiale et préliminaire de la méthode U-net est la phase d'augmentation de données. Entrainer un réseau profond demande un nombre conséquent de données d'apprentissage (de l'ordre du millier) pour obtenir de bons résultats de généralisation et éviter un sur apprentissage (overfitting). O. Ronneberger et al on a disposions qu'une trentaine d'échantillons et un réseau à 26 couches.

### 3.2 Augmentation de données

La formation d'un grand réseau nécessite beaucoup d'échantillons. Pour augmenter nos données, l'idée est de procédé à des transformations sur les échantillons disponibles. On peut lister les techniques de transformations suivantes, qui sont largement utilisées :

- Les translations/rotations
- Les changements d'échelle
- Les réflexions miroirs
- Les distorsions photométriques
- Les transformations élastiques.



*Figure 11 Exemple de transformations pour l'augmentation de données.  
(<https://moodle.technion.ac.il/mod/resource/view.php?id=390948&lang=en>)*

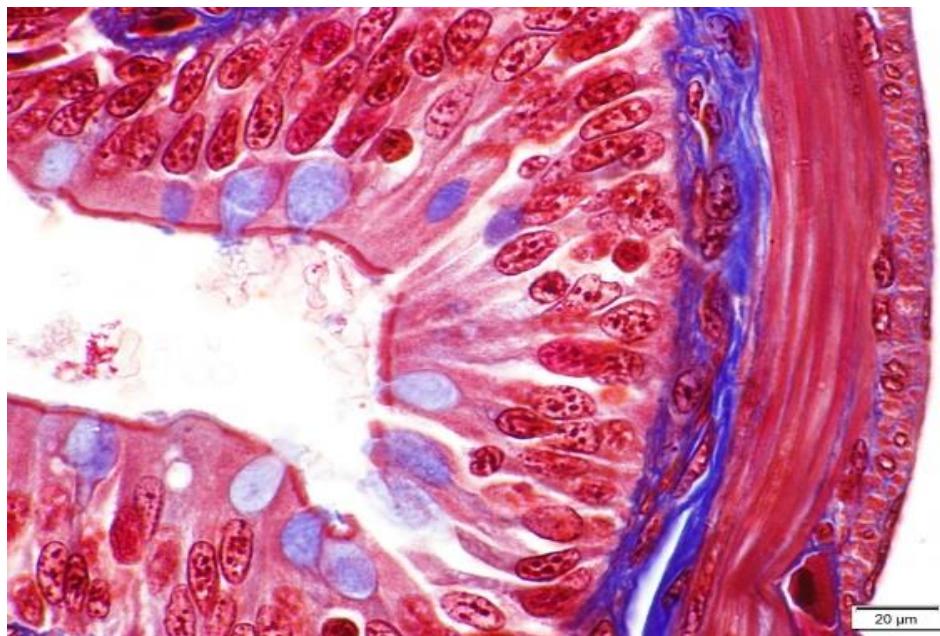


Figure 12 Image histologique (<http://codexvirtualis.fr/>)

On peut augmenter par d'autres façons en ajoutant du bruit ou par rognage par exemple, mais il va être important de choisir des méthodes d'augmentation qui sont pertinentes par rapport au type d'images que l'on a à disposition. Dans le cadre de leurs travaux, O. Ronneberger et al traitent des images biomédicales plus particulièrement des images cellulaires. Ils ont donc appliqué des translations, rotations, réflexions, de faibles distorsions photométriques et abusé de transformations élastiques qui sont naturels pour des images de cellules.

Il existe de nouvelles méthodes intéressantes de transformations des images « à la volée » directement pendant l'apprentissage.

#### 4 L'apprentissage des réseaux convolutionnel : optimisation d'une fonction de cout par descente de gradient

L'apprentissage d'un réseau de neurone au sens large revient à optimiser une fonction de cout. On cherche le minimum d'une fonctionnelle en fonction des paramètres du réseau. Plusieurs fonctions de cout sont adaptées aux réseaux de neurones et leurs applications. Ainsi, on optimisera une entropie croisée rendant compte de l'erreur entre le modèle réel et une estimation.

Pour un exemple seul de vecteur de vérité terrain  $\mathbf{y}$  et de vecteur de prédiction  $\hat{\mathbf{y}}$ , on a le coût unitaire :

$$l(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_i y_i \log \hat{y}_i$$

On calcule le coût pour un ensemble de données  $(\mathbf{X}, \mathbf{Y})$  comme la somme des coûts unitaires :

$$L(\mathbf{X}, \mathbf{Y}) = \sum_i l(\mathbf{y}^{(i)}, \hat{\mathbf{y}}^{(i)})$$

Où  $\hat{\mathbf{y}}^{(i)}$  correspond à la sortie du réseau pour l'entrée  $x^{(i)}$ .

Pour rappel, le principe de backpropagation consiste à appliquer le théorème de dérivation des fonctions composées (chain rule) :

$$\frac{\partial c}{\partial a} = \frac{\partial c}{\partial b} \frac{\partial b}{\partial a}$$

On visualise schématiquement l'application de la backpropagation sur un réseau de neurones sur la figure 13.

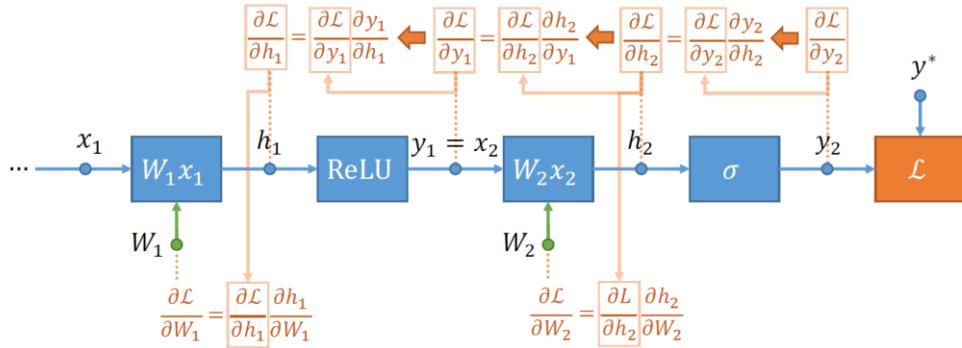


Figure 13 Vue schématique de la backpropagation sur un réseau de neurones. On commence par calculer la dérivée du coût  $L$  par rapport à la sortie ( $y_2$  sur le schéma) puis on remonte dans le réseau en réutilisant les dérivées calculées précédemment lors du calcul des nouvelles dérivées.

La descente en gradient est l'un des algorithmes les plus populaires pour effectuer l'optimisation et de loin le moyen le plus courant utilisé pour les réseaux neuronaux. Elle consiste à calculer la dérivée de la fonction de coût par rapport à un paramètre  $w$ , et à faire un pas (aussi appelé learning rate)  $\eta$  dans la direction du gradient, c'est à dire à modifier la valeur de  $w$  de  $\eta$  dans la direction du gradient :

$$w \leftarrow w - \eta \frac{\partial L(X, Y)}{\partial w}$$

Le gradient  $\frac{\partial L(X, Y)}{\partial w}$  peut être calculé sur différents ensembles, ce qui correspond à différentes variantes de l'algorithme de descente de gradient.

- Si l'on considère l'ensemble d'apprentissage (train) dans son ensemble, on reste dans le cadre d'une descente de gradient classique.
- Si l'on considère qu'un petit sous ensemble ou batch d'exemple d'apprentissage tirées aléatoirement, nous sommes dans le cadre de la descente de gradient stochastique par mini-batch. C'est en fait cette méthode que nous utilisons dans nos expériences.
- On peut aussi considérer une seule paire d'exemple d'apprentissage  $(x^{(i)}, y^{(i)})$ , on remplace alors la loss (fonction de coût) globale  $L$  par le coût unitaire  $l$ . C'est la descente de gradient stochastique « online ».
- 

L'entraînement par batch consiste à rétro propager l'erreur de classification par groupes d'images. Cette méthode est plus rapide qu'en calculant l'erreur sur tout le jeu d'entraînement à chaque itération. Elle est plus stable qu'en travaillant image par image, car les gradients d'erreurs ont moins de variance.

## 5 Mise en œuvre

### 5.1 Environnement de développement

Afin d'expérimenter sur l'architecture U-Net, plusieurs pistes se sont présentés à nous. Plusieurs frameworks et bibliothèques de deep learning sont mise à disposition sur internet. On peut citer Caffe, Torch, Thenao, Tensor Flow, Lasagne, MatConvNet etc. Ils présentent des possibilités et des limites qui leurs sont propres. On peut ainsi évaluer leur capacité à modéliser des modèles complexes, leur interface, leurs performances de calcul, leur scalabilité...

Les auteurs de l'article U-Net ont construit appris et testé leur réseau sous le framework Caffe. Caffe est une bibliothèque conçue à l'initiative de l'université de Berkeley qui propose des performances louables et très adaptée à des réseaux convolutionnels de grande taille. Basée sur le langage C++, elle permet entre autre d'être compilé sur une multitude de plateforme. Cependant l'interface n'est pas simple pour des débutants et demande certains prérequis.

D'autres chercheurs ont chercher à exploiter le U-Net pour résoudre leur propre problème de segmentation. C'est une architecture qui a été reprise sur plusieurs plateformes et frameworks. Nous avons ainsi décidé de baser notre travail sur les expériences de l'équipe d'Orobix (<https://github.com/orobix/retina-unet>).

Ils reprennent l'architecture sous le framework Keras afin de segmenter les vaisseaux sanguins d'images de rétine.

Keras est une bibliothèque open source de réseaux neuronaux écrite en Python. Elle a la particularité d'utiliser les bibliothèques Tensorflow ou Theano en tant que backend avec un niveau d'abstraction supérieur. Elle est donc conçue pour permettre une expérimentation rapide et simplifier avec des réseaux de neurones profonds. Modulaire, intuitive et extensible elle a été développée dans le cadre du projet de recherche ONEIRO (Open-end Neuro-Electronic Intelligent Robot Operating System) et son principal auteur et responsable est François Chollet.

Pour un souci de performance, il a fallu travailler sur une machine suffisamment puissante et disposant d'une puce graphique nVidia compatible Cuda.  
Nous avons donc mis en œuvre la méthode proposée par l'équipe d'Orobix sous Keras.

## 5.2 Une architecture modifiée

Dans leurs travaux, cette équipe a repris l'architecture des auteurs de l'article U-Net en modifiant certains aspects notamment dans l'exploitation des données. O. Ronneberger et al, on l'a vu, propose comme donnée d'entrée l'image en entier. Cela est pertinent de la cas où l'image présente peut de structures à segmenter (figure 13). Or, dans notre étude on cherche à segmenter des structures présentes en quantité dans nos images (figure 14,15,16) : Les vaisseaux sanguin (figure 14) dans les rétines, des cellules (figure 15), des noyaux dans des images histologiques (figure 16).

Ce qui a été proposé est d'entrainer le réseau sur des patches issus des images de taille adapté à un élément à segmenter. Cela a pour avantage d'entrainer sur un nombre beaucoup élevé de données. Il n'est plus forcément nécessaire de faire de l'augmentation de donnée par transformations comme l'on fait O. Ronneberger et son équipe.

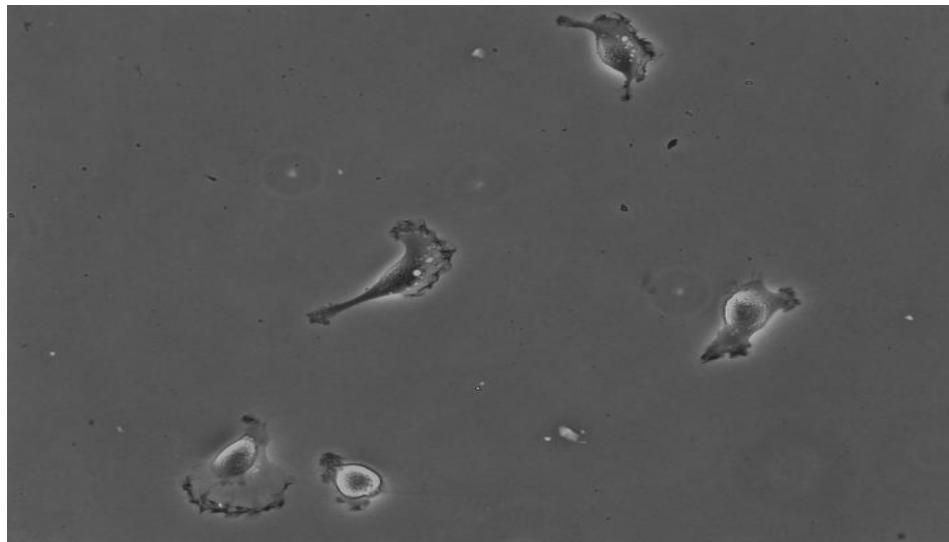


Figure 14 Exemple d'images à segmenter par U-Net

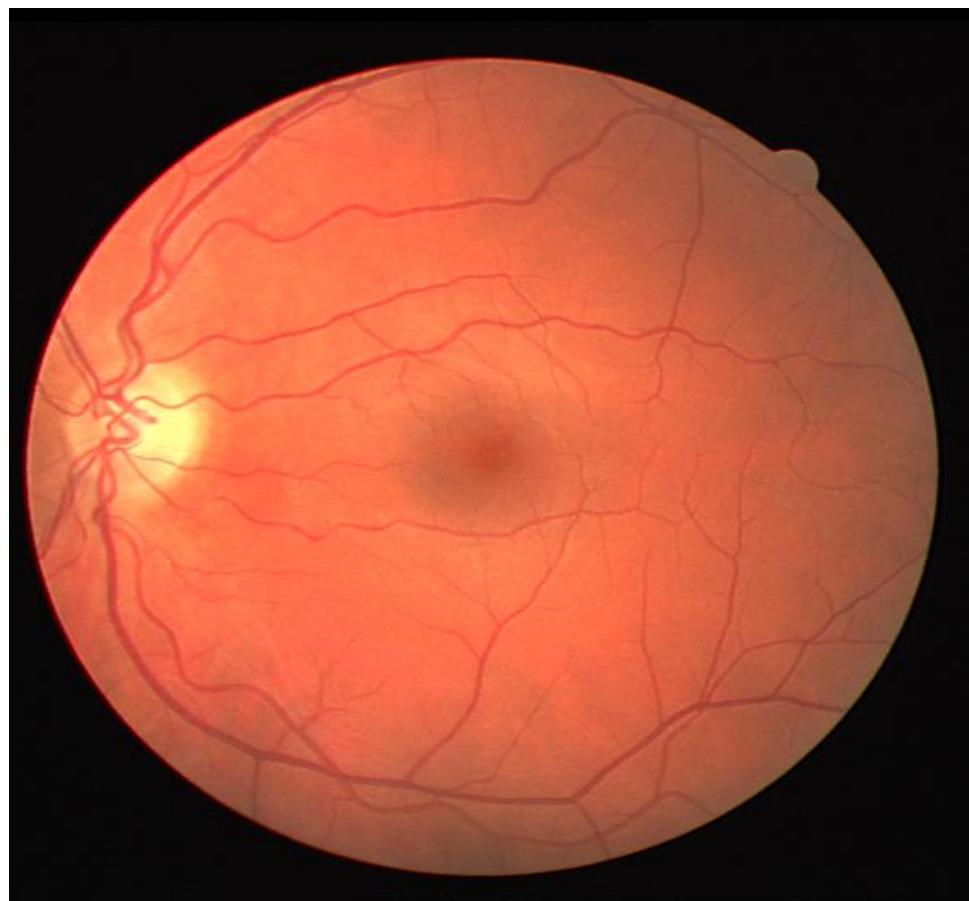


Figure 15 Exemple d'image à segmenter : vaisseaux sanguins

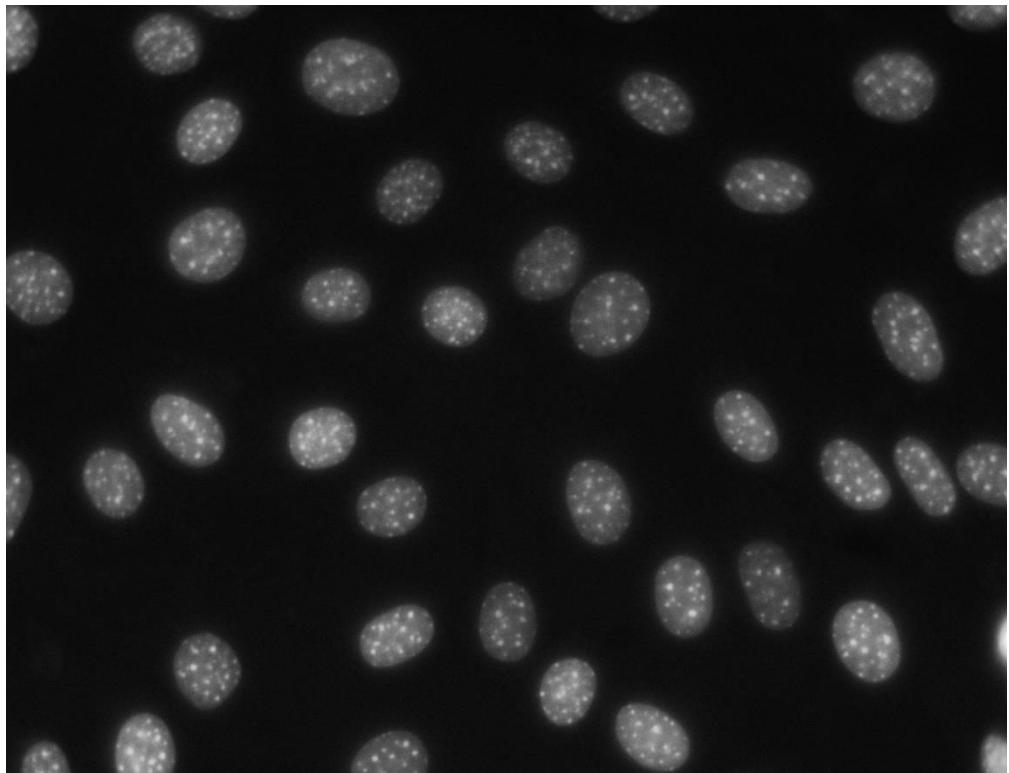


Figure 16 Un autre type de donnée : image de cellules

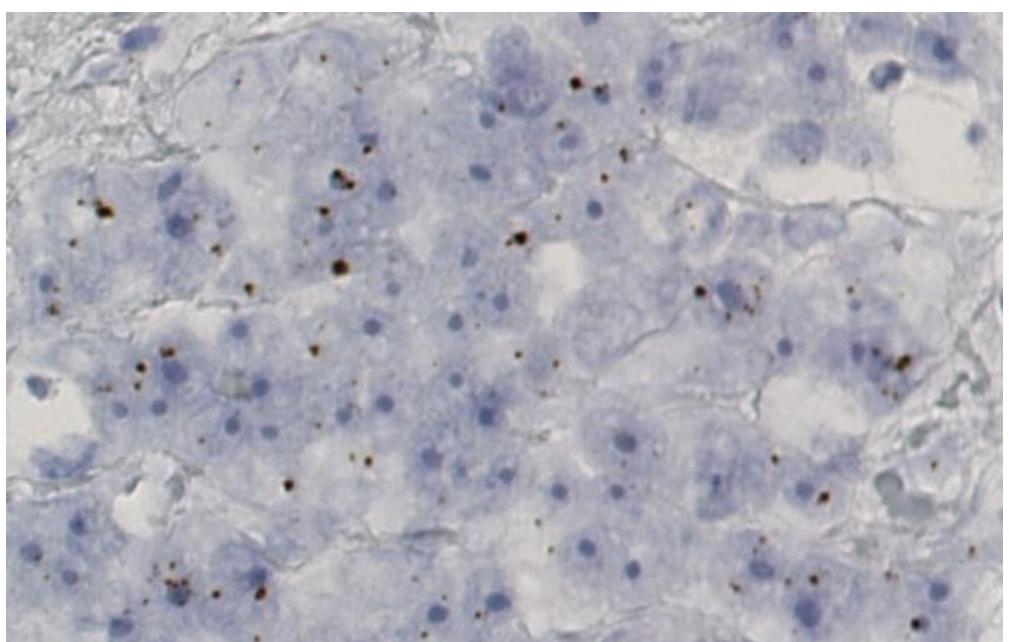


Figure 17 Un autre type de donnée : Segmentation des noyaux marqués en marron

Le réseau a aussi été légèrement simplifier en faisant abstraction de couches de convolutions, on le verra cela n'a pas de conséquences notables sur les résultats avec nos données. De plus nous avons modifier la couche d'entrée afin qu'elle puisse traiter des images couleurs.

L'architecture simplifiée est la suivante :

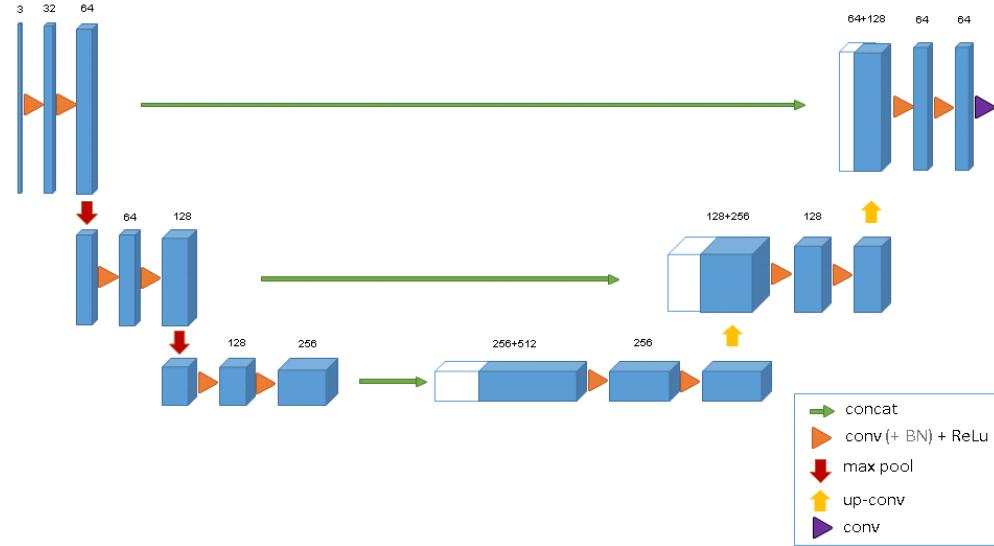


Figure 18 Architecture simplifiée utilisée

### 5.3 Traitement des données d'entrée et régularisations

Une première étape avant d'entrainer le réseau est le traitement des données afin d'intégrer de la régularisation.

Ainsi, avant l'entraînement, les images d'entraînement de nos bases sont prétraitées avec les transformations suivantes :

- Conversion en niveaux de gris,
- Normalisation,
- Egalisation d'histogramme,
- Corrections gamma.

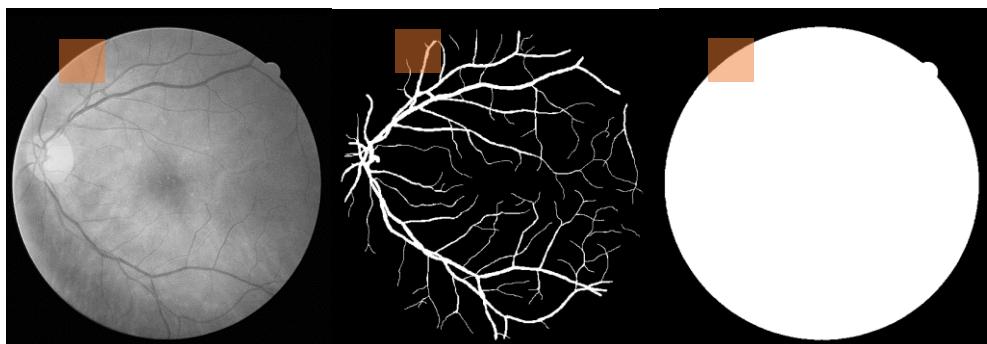
C'est transformations et normalisations permettent de mieux conditionner l'apprentissage.

Une autre technique de régularisation a été mise en place. Il s'agit du dropout. C'est une technique de régularisation qui permet de réduire la sur apprentissage des réseaux neuronaux en empêchant des coadaptations complexes sur les données d'entraînement. C'est un moyen efficace permettant d'effectuer la moyenne des modèles avec des réseaux de neurones. La technique repose sur la désactivation de certains neurones. Cela force l'apprentissage du réseau sur différents "chemins".

Cette régularisation permet d'éviter le sur-apprentissage et permet un gain de temps d'exécution. Cela est vrai car on a moins de paramètres à apprendre pour un nombre constant de donnée d'apprentissage.

On désactive ici 20% des neurones et obtient les résultats suivant

Comme expliqué précédemment, le réseau de neurone est appris sur des sous parties (des patches) des images prétraitées. Pour les image de rétine les éléments a segmenté sont fins et suivent une structure en arbre. Les auteurs ont choisi une taille de patch de 48x48. Ces derniers sont sélectionnés aléatoirement dans les images. Il a été défini au préalable un champ de vue servant au réseau à discriminer le bord d'une image de test des vaisseaux sanguins (on apprend au réseau que le bord d'une image de test ne peut pas être considéré comme un objet à segmenter. Pour cela les patches d'apprentissage sont sélectionnés aussi autour de ce champ de vision.



*Figure 19 Image prétraitée, carte de segmentation vérité associée, masque de champ de vue, exemple de patch en orange*

Il faudra spécifier par la suite les paramètres d'apprentissage dont :

- Le nombre total de patches à extraire,
- La taille des patches,
- Le nombre d'epoch (ou passe),
- La taille du batch.

On donnera donc en entrée au réseau un nombre prédéfinis de patches ainsi que la vérité associée :

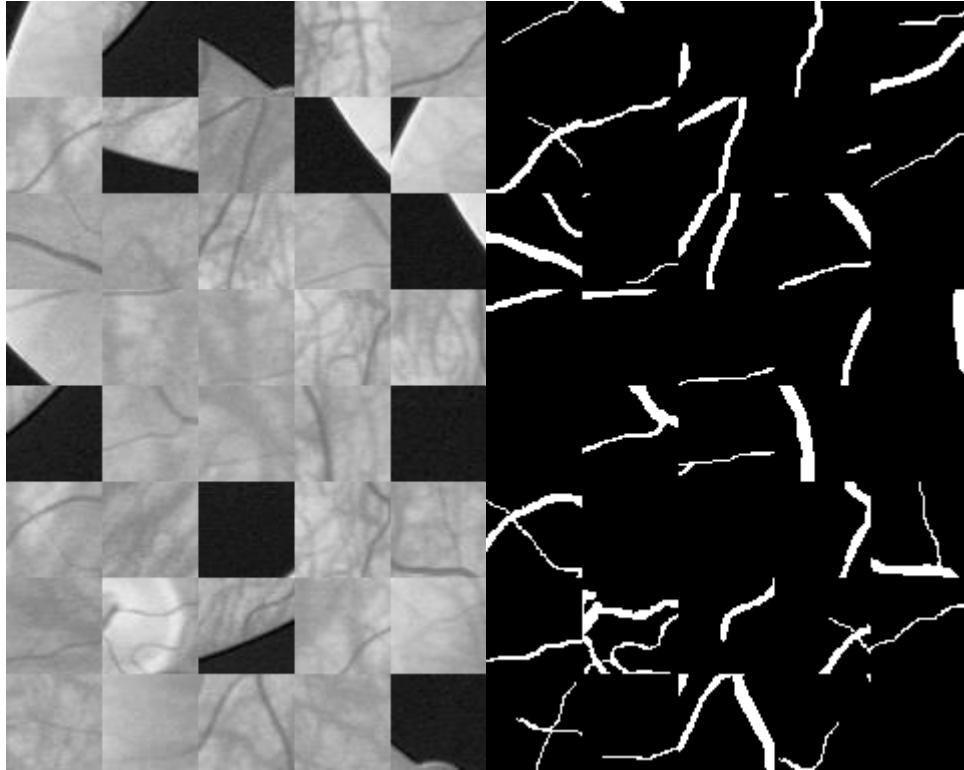


Figure 20 Exemples de patches à donner au réseau

#### 5.4 Matériel

Les algorithmes de deep learning, notamment les réseaux très profonds comme le U-Net nécessite une puissance de calcul. L'équipe Orobix arrive à surpasser les méthodes les plus récente de segmentation de la base DRIVE. Ils ont cependant à disposition une configuration matériel conséquente :

- GPU : GTX Titan X 12 gigas (3584 coeurs Cuda)
- RAM : 16 gigas

Avec cette configuration, il entraîne le réseau sur 190 000 patchs en 20 heures. Nous avons nous à disposition la configuration suivante :

- GPU : GTX 1060 6 gigas (1280 coeurs Cuda)
- RAM : 8 gigas

On rappelle que ce travail fait partie d'une étude. On cherche à savoir si l'on peut segmenter nos propres images à l'aide de ce type de réseau, on ne cherche pas forcément les meilleurs résultats sur les bases à disposition.

## 6 Expériences et tests

### 6.1 Expérimentation 1 : Base d'image DRIVE

Nous avons à disposition, dans un premier temps, la base d'image DRIVE. A chaque image est fournie la carte de segmentation vérité associée. Ces cartes sont utilisées pour entraîner le réseau ainsi que pour les phases de validation et de tests. Cette base est composée de 20 images d'entraînement et 20 images de tests.

Nous utiliserons pour mesurer les performances les métriques suivantes :

- Courbe ROC (Receiving Operating Characteristics)

La courbe ROC représente l'évolution de la sensibilité en fonction de la spécificité quand on fait varier un seuil  $t$ . Plus l'aire sous la courbe de ROC d'un modèle est grande, plus ce modèle est bon.

Spécificité et sensibilité :

La sensibilité est définie par :  $TP / (TP + FN) = TP / P$ ,  
La spécificité est définie par:  $TN / (TN + FP) = TN / N$ .

Avec :

TP (true positives) : les prédicts positifs qui le sont vraiment.

FP (false positives) : les prédicts positifs qui sont en fait négatifs.

TN (true negatives) : les prédicts négatifs qui le sont vraiment.

FN (false negatives) : les prédicts négatifs qui sont en fait positifs.

P (positives) : tous les positifs quel que soit l'état de leur prédiction.  $P = TP + FN$ .

N (negatives) : tous les négatifs quel que soit l'état de leur prédiction.  $N = TN + FP$ .

- Matrice de confusion

La matrice de confusion est un outil servant à mesurer la qualité d'un système de classification.

Chaque colonne de la matrice représente le nombre d'occurrences de la classe estimée, tandis que chaque ligne représente le nombre d'occurrences de la classe réelle.

Un des intérêts de la matrice de confusion est qu'elle montre rapidement si le système parvient à classifier correctement.

Dans notre étude est définie comme suit :

$$\begin{pmatrix} \text{Nombre de pixel } TP & \text{Nombre de pixel } FN \\ \text{Nombre de pixel } FP & \text{Nombre de pixel } TN \end{pmatrix}$$

Avec :

- Nombre de pixel TP : Nombre de pixel prédict appartenant bien à l'objet,
- Nombre de pixel TN : Nombre de pixel prédict appartenant bien au fond,
- Nombre de pixel FP : Nombre de pixels prédict comme appartenant à l'objet mais qui sont en réalité du fond,
- Nombre de pixel FN : Nombre de pixels prédict comme appartenant au fond mais qui en réalité appartiennent à l'objet.

- Précision (accuracy)

La précision d'un model mesure sa capacité (en %) à prédire de façon correct. Elle est définie par :

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

Nous rappelons dans un premier temps les résultats des auteurs. Nous rappelons qu'ils ont pu entraîner leur réseau avec 190 000 patches et 150 epochs. Leur testes indique les résultats suivants :

- Courbe ROC :

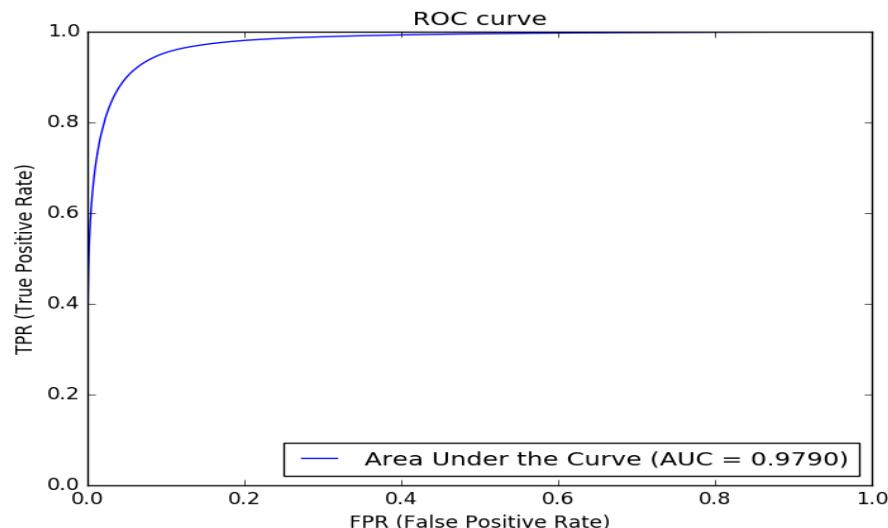


Figure 21 Courbe roc et aire en dessous de la courbe

- Matrice de confusion :

	3895256	65238	
	134513	443136	

- Précision : 96%

- Synthèse :

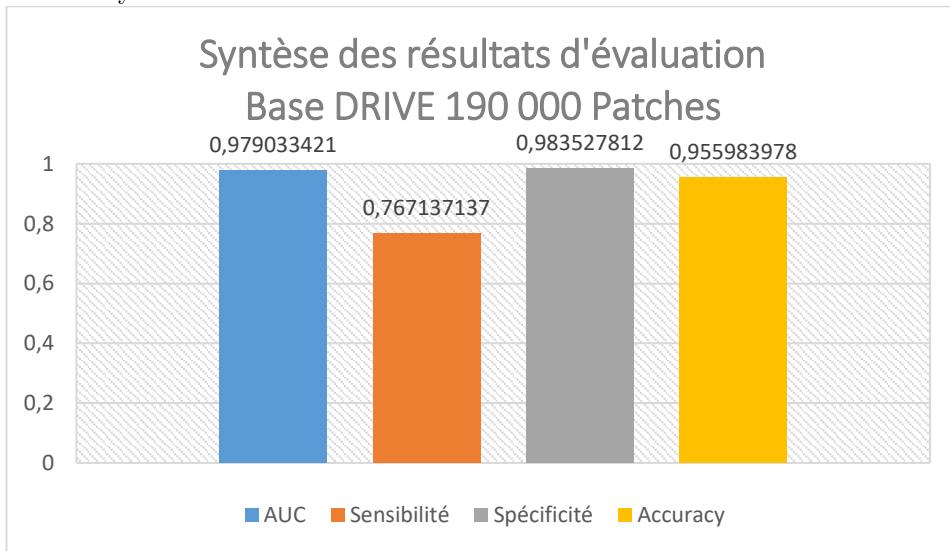


Figure 22 Synthèse des résultats optimaux des travaux d'Orobix

Ces résultats surpassent la plupart des techniques à base de réseaux de neurones (au 15 décembre 2016).

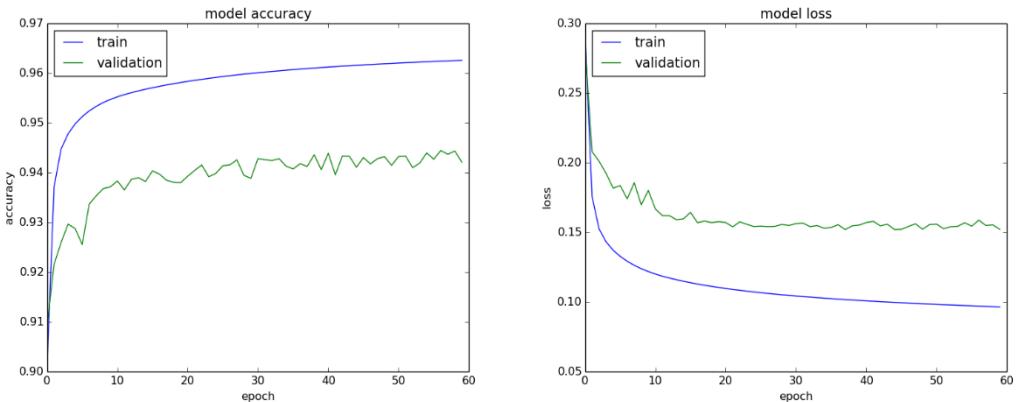
Nous avons donc tenté de réapprendre le réseau. D'une part avec la même base d'image DRIVE puis avec une nouvelle base d'image de cellules.

Pour cette première expérimentation nous avons suivi les recommandations des auteurs. Cependant il a fallu réadapter certains paramètres car la machine de test est limitée en puissance et en mémoire.

Pour un patch entré dans le réseau, celui-ci subit comme on l'a vu des opérations de convolution et maxpooling. Celles-ci en particulier augmentent la dimension de la donnée et donc la quantité de mémoire RAM nécessaire. Les paramètres critique sont ainsi naturellement le nombre de patches ainsi que leur taille. Or ces paramètres jouent sur le résultat de segmentation.

Nous avons effectué différentes expériences sur la base drive avec différent paramètres. Nous les synthétisons dans la suite et nous présentons les résultats pour le meilleur paramétrage.

Pour 139 000 patches de taille 48 x 48 et 60 epochs (la taille de batch est fixée à 32 dans tous les tests), nous atteignons les limites de notre machine. Les résultats d'apprentissage sont exposés dans la figure 23.



*Figure 23 Evolution de la précision et fonction de coût (139 000 patches, 60 epochs)*

L'apprentissage reste correct dans la mesure où le réseau ne sur-apprends pas et atteint des fortes valeurs de précision et en fonction de coût. On atteint en validation environ 94% de précision.

Comme précédemment, nous avons effectué une phase de tests sur les 20 images de la base utile à cet effet. Nous obtenons les résultats quantitatifs suivants :

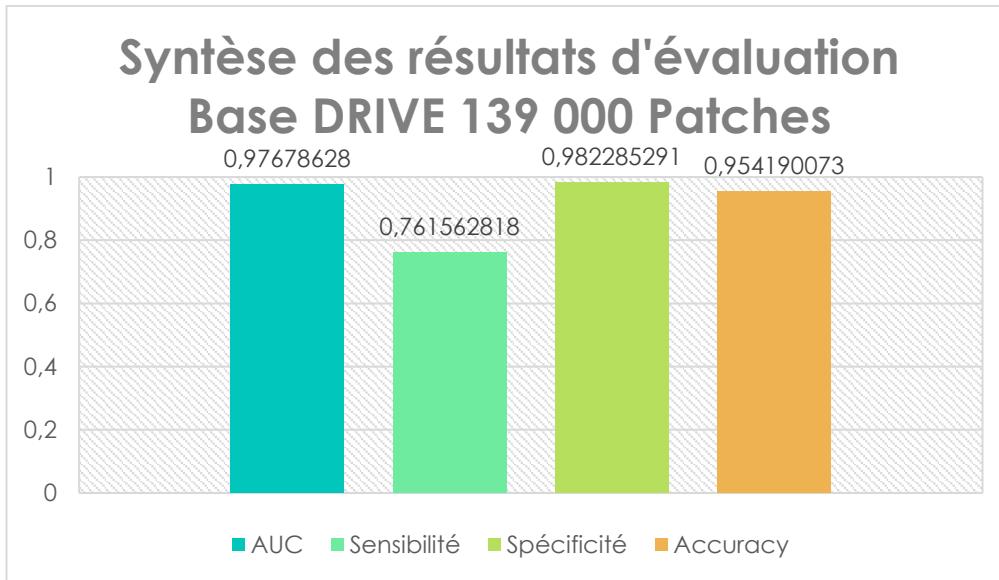


Figure 24 Synthèse des résultats en teste pour 139 000 patches

3890335	70159	_____
137733	439916	_____

Figure 25 Matrice de confusion 139 000 patches

On obtient des résultats visuels très proche du paramétrage optimal :

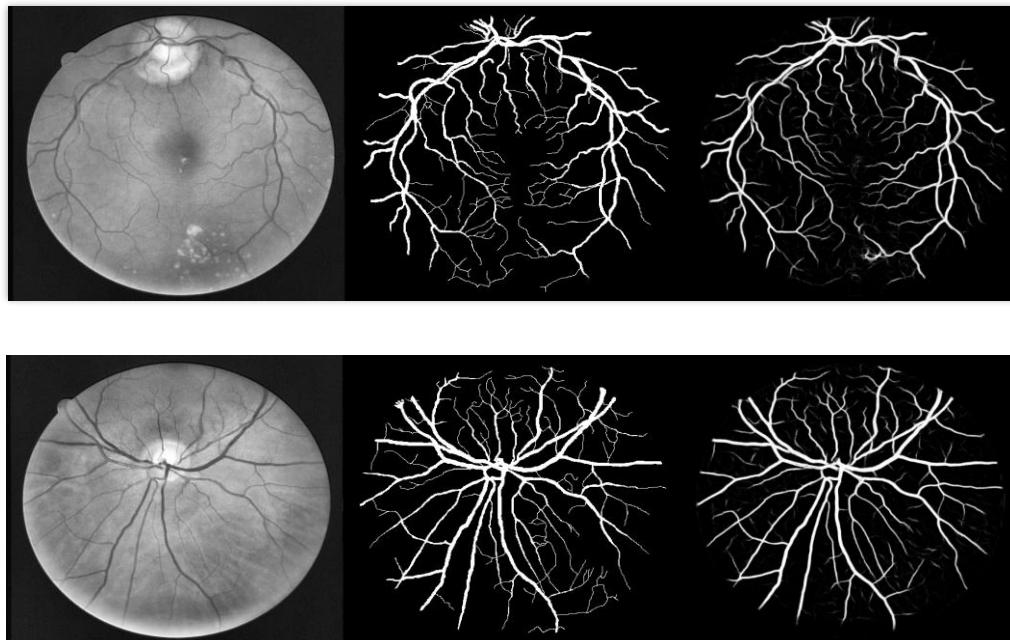


Figure 26 Exemple de résultats de segmentation issu de la base de test (l'entrée à gauche, la vérité au centre et le résultat à droite)

Le résultat retourné est en fait une moyenne des patches prédit ici, avec un stride de 5. Cela permet d'obtenir un résultat lissé et plus fin.

Nous avons mis en place un script permettant d'extraire les caractéristiques qui ont été apprises afin d'observé leur forme. Nous avons cherché à afficher les entrées qui maximisent l'activation des filtres dans différentes couches du réseau. Voici quelques résultats sur différentes couches :

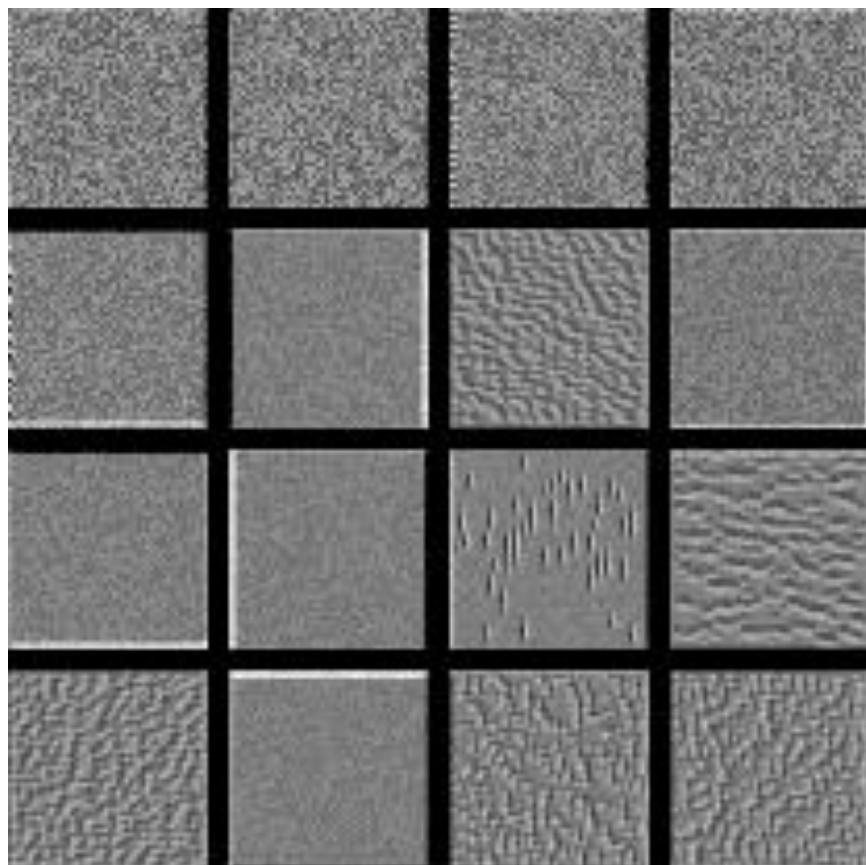


Figure 27 Filtres, couche de convolution 1

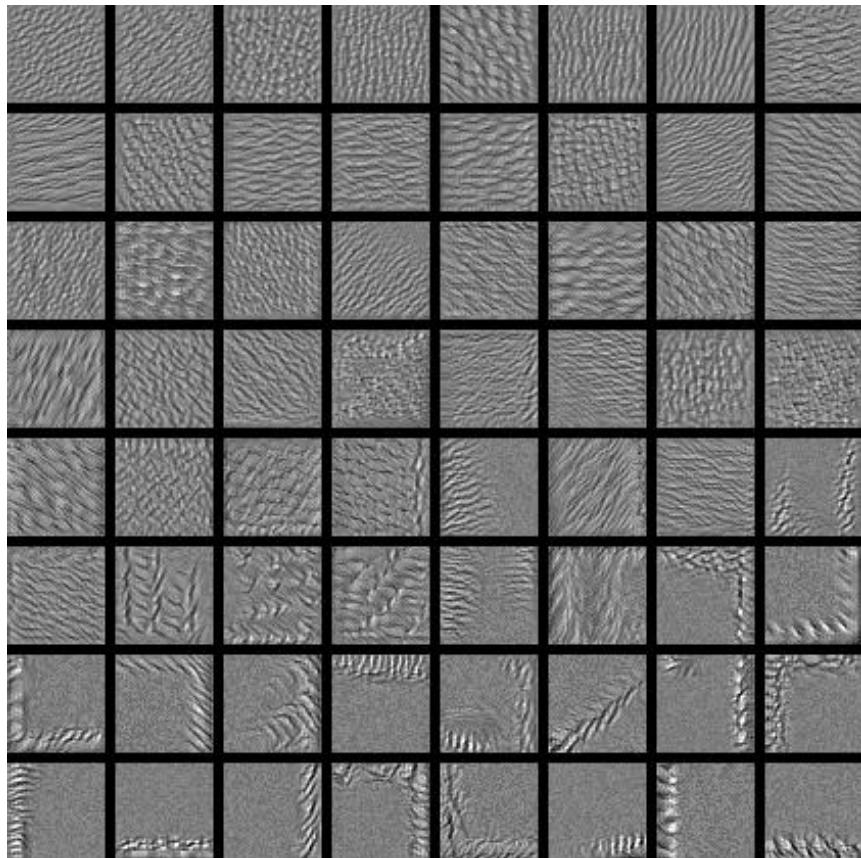
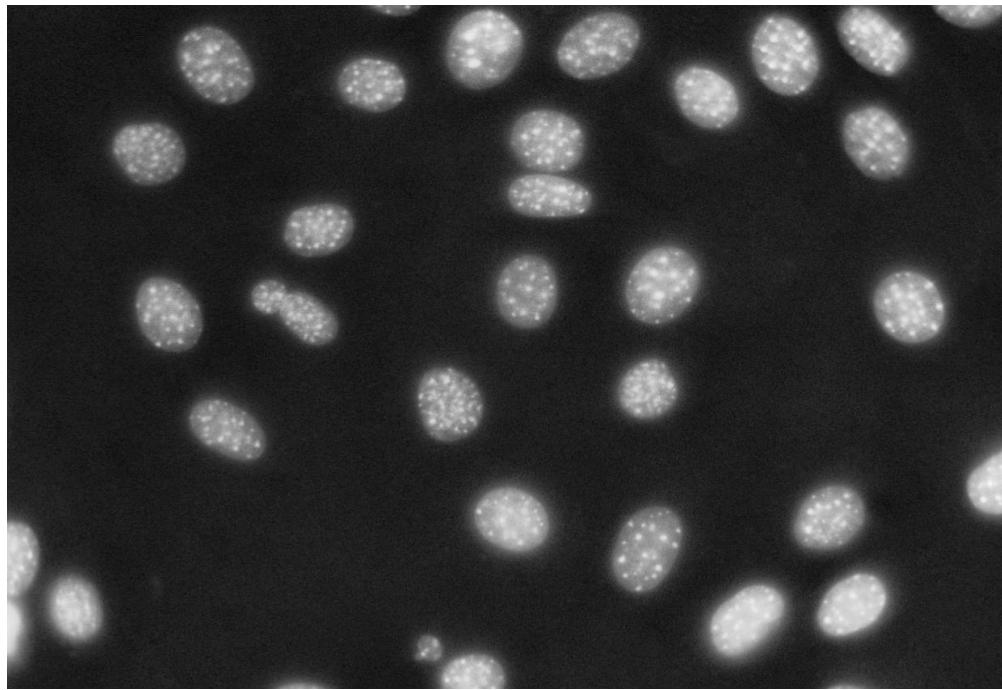


Figure 28 Filtres, couche de convolution 5

On observe ainsi que le réseau cherche à encodée certaines directions, formes, textures et bords. Il cherche notamment à détecter la forme particulière en arbre des vaisseaux de la rétine.

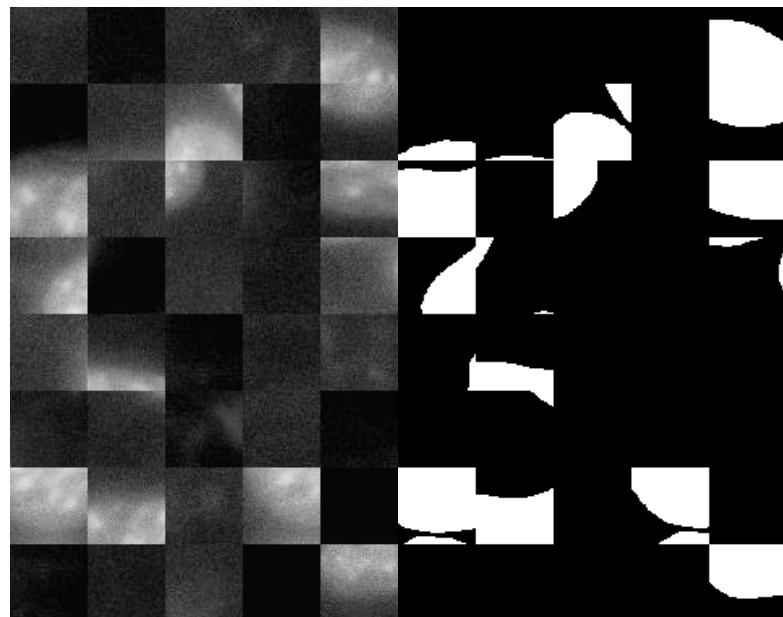
## 6.2 Expérimentation 2 : base Cellules

Dans la suit nous avons tenté d'utiliser ce même réseau sur la base s'apprentissage « Cellules ». Celle-ci est composée de 40 images de taille 1024x1344. Elles présentent des cellules dont la taille varie autour de 100 pixels en longueur et largeur.



*Figure 29 Exemple d'une image de la base "Cellules"*

Nous avons arbitrairement découpé cette base en 20 images d'entraînement et 20 images de test. L'idéal aurait été de faire une validation croisée, mais cela a été inenvisageable vu la durée nécessaire d'apprentissage. Les mêmes prétraitements sont appliqués que pour la base DRIVE et de la même façon, ce seront des patches qui seront utilisé en tant qu'entrée du réseau.



*Figure 30 Exemples de patches pour la base "Cellules"*

Nous avons utilisé le même matériel et les mêmes métriques pour l'évaluation de résultats. Pour le même paramétrage que précédemment (139 000 patches de taille 48 x 48 et 80 epochs), on a les résultats suivant en apprentissage :

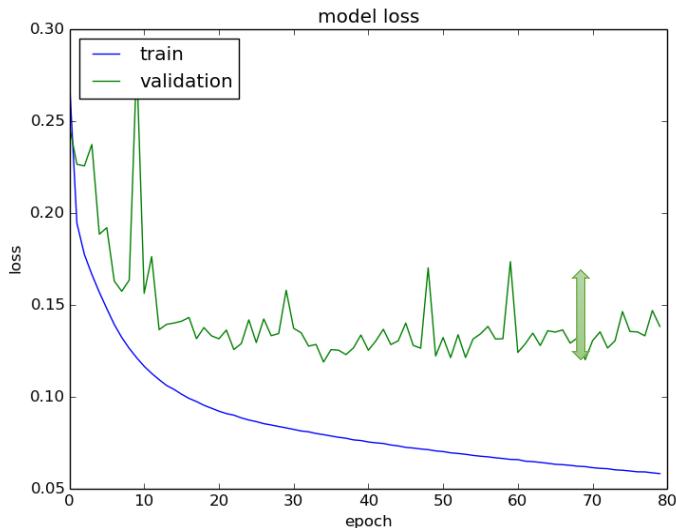


Figure 31 Evolution de la fonction de coût, 80 epochs

On observe alors que la fonction de coût en validation se stabilise mais qu'en entraînement elle continue à diminuer. C'est un signe de sur apprentissage.

Comme précédemment, nous avons effectué une phase de tests sur les 20 images de la base utile à cet effet. Nous obtenons les résultats quantitatifs suivants :

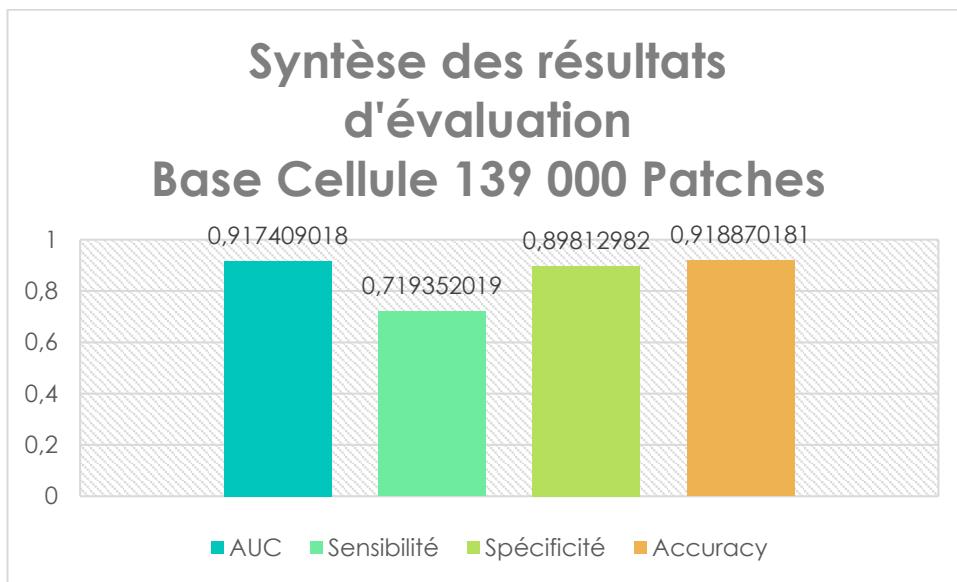


Figure 32 Synthèse des résultats 190 000 patches 80 epochs

2513511	45459	
248381	473001	

Figure 33 Matrice de confusion, 190 000 patches 80 epochs

On observe que l'on perde beaucoup en sensibilité, c'est-à-dire que les pixels de la classe objet sont moins bien classé. On peut l'observer sur des exemples de segmentation ou on peut voir des trous dans les cellules (figure 32 à droite) :

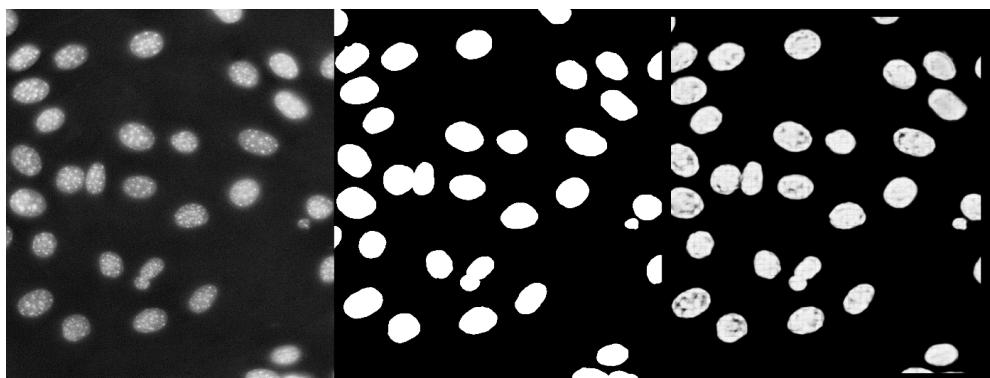


Figure 34 exemple de segmentation sur la base cellules avec le paramétrage initial

Vu la nature différente entre les image de cette base et des images de la base DRIVE, on s'attendait à ce que ce paramétrage ne soit pas forcément été le plus adapté. En effet, nous traitons des structures de taille, de forme, de texture et de nature différente. Plusieurs stratégies ont été pensées et mis en œuvre afin d'améliorer les résultats pour cette base.

Ne pouvant pas augmenter le nombre de patches (ce qui nous aurait permis d'éviter l'over fitting), nous avons pensé dans un premier lieu à augmenter la taille des patches de 48x48 à 128x128 pour couvrir une cellule en entier. Cette idée est envisageable mais la taille des patches est aussi un des paramètres critiques en ce qui concerne la mémoire. Le temps de calcul nécessaire pour une taille de patch de 128 x 128 explose, il est alors impossible d'expérimenter avec un nombre suffisant (plus de 100 000 patches). Nous avons tout de même pu mesurer un gain de 5% en spécificité sur un nombre réduit de patches (10 000, 15 epochs).

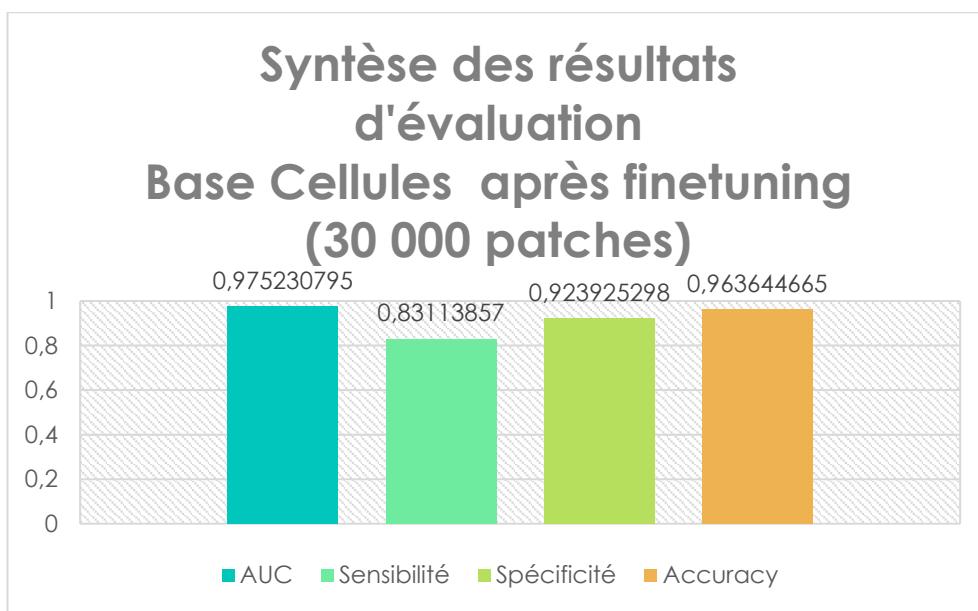
Une autre stratégie a été de faire ce qu'on appelle un fine tuning. Dans le cas général le fine tuning consiste à se servir d'un modèle pré-entraîné comme initialisation pour l'entraînement sur un nouveau problème.

L'intérêt est double : on utilise une architecture optimisée et l'on profite des capacités d'extraction de caractéristiques apprises sur un jeu de données conséquent. Le fine tuning sur des images consiste en quelques sortes à prendre un système visuel déjà bien entraîné sur une tâche pour le raffiner sur une tâche similaire.

Le réseau étant « fully convolutionnal » la seule modification nécessaire du réseau consiste à désactiver les premières couches et apprendre avec une évolution du pas d'apprentissage lente. Le réseau pré-entraîné possède, en effet, déjà des coefficients optimisés sur un grand jeu de données. Il s'agit donc de les modifier faiblement à chaque itération, pour s'adapter en douceur au nouveau problème, sans écraser agressivement la connaissance déjà acquise.

Il est à noter que l'utilisation du fine tuning a été pertinente dans notre cas seulement car nous n'avons pas eu assez de ressource en mémoire et calcule. Elle a en tout cas servi à démontré un intérêt sur son utilisation dans le cas où l'on souhaite entraîner un réseau rapidement avec peu de données d'entraînement [23].

Nous avons donc redécoupé notre base de test en deux afin de créer une nouvelle d'apprentissage pour le fin tuning. Nous reprenons notre réseau appris précédemment sur les 139 000 patches et ré entraînons les 10 dernières couches avec 30 000 patches. Le processus d'apprentissage a pris 5 minutes et nous a permis d'augmenter les résultats globaux de 6,5 pourcents.



*Figure 35 Résultats du fine tuning*

Visuellement on a aussi pu observer un gain, le résultat de segmentation bien que pas parfait est amélioré. Des trous sont comblés dans les cellules.

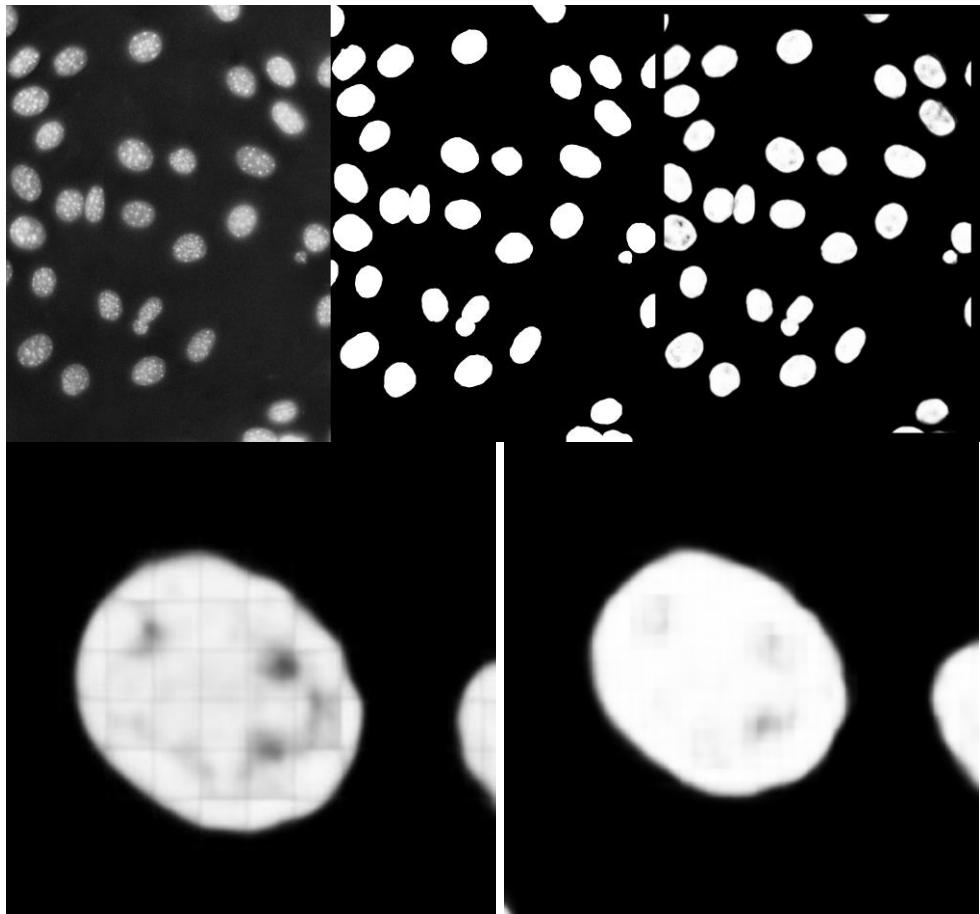


Figure 36 Visualisation du résultat de segmentation, les trous dans les cellules sont presque comblés(ancien en bas à gauche, nouveau en bas à droite)

Nous avons dans le cas des cellules afficher les caractéristiques apprises sur différentes couches. On peut alors observer les différences de filtre sur les mêmes couches 3 et 5 par exemple :

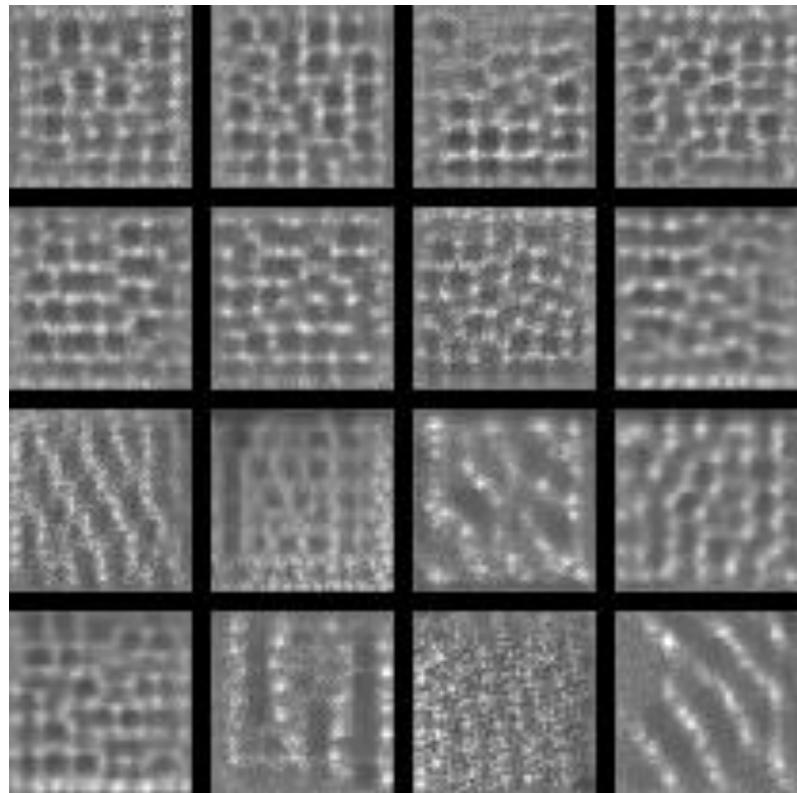


Figure 37 Filtres, couche de convolution 3 base "Cellules"

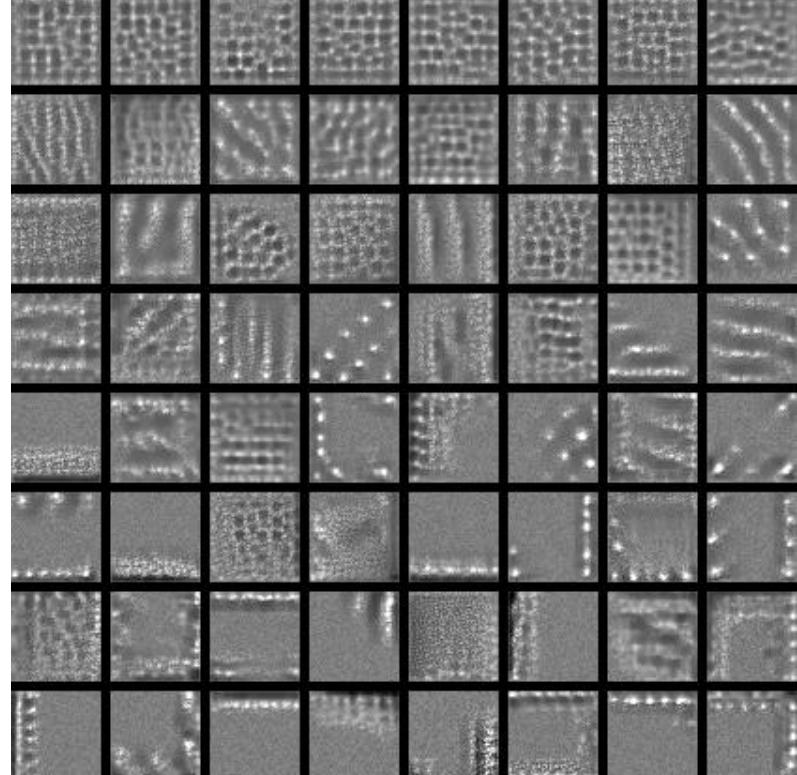


Figure 38 Filtres, couche de convolution 5 base "Cellules"

La différence est très notable, on voit que le réseau cherche des réponses à des formes circulaires correspondant aux formes des cellules.

### 6.2.1 Synthèse des résultats de précision

Nous avons synthétisé dans la figure l'ensemble des tests et optimisation effectués sur la base cellule :

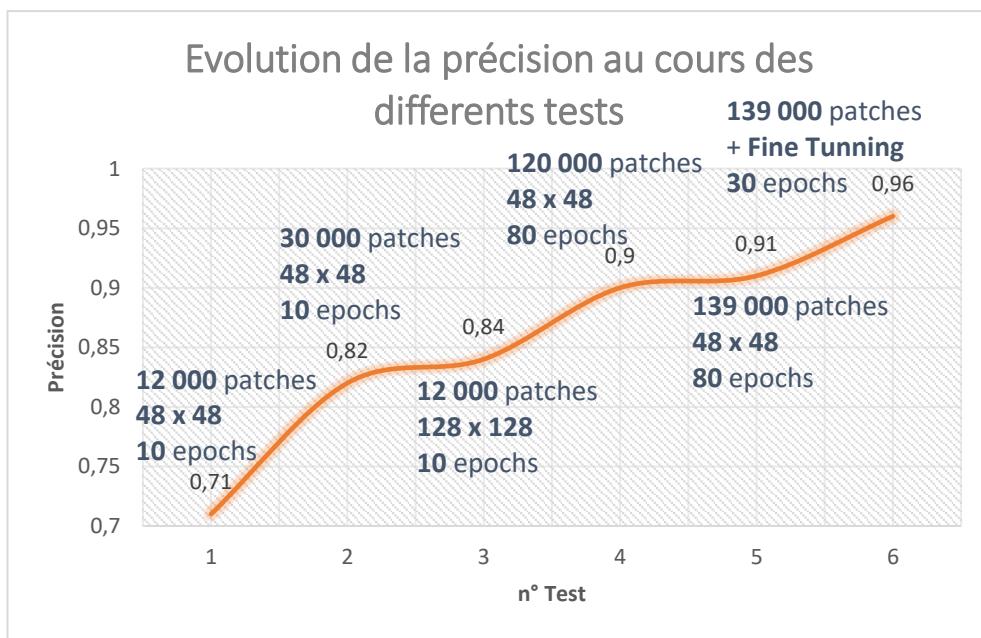


Figure 39 Synthèse de l'évolution de la précision au cours des tests

## Conclusion

Dans cet écrit, nous avons listé différentes méthodes de segmentations d'images. Nous avons évoqué certaines méthodes classiques qui sont encore très largement utilisés tel que les méthodes par modèles déformables, les méthodes morphologiques, mais aussi des méthodes plus modernes à base d'apprentissage. Nous avons listé des techniques non supervisées tel que K-means ou Fuzzy C-means et des méthodes supervisées comme par exemple le SVM, l'algorithme des K plus proches voisins ou bien les réseaux de Neurones.

Beaucoup de progrès ont été réalisés en segmentation et en reconnaissance visuelle en général avec la réapparition des réseaux de neurones profonds et plus particulièrement des réseaux de neurones convolutionnels. Ils permettent une nouvelle approche basée sur l'extraction hiérarchique de caractéristique de hauts niveaux des images. Ainsi, naissent plusieurs techniques prometteuses basée sur ces architectures. Certaines tentent de répondre à des problématiques intrinsèques aux réseaux profonds, tel que l'utilisation d'un faible volume de données d'entraînement. Un problème persistant est aussi de pouvoir garder avec une bonne précision pour les opérations tout en gardant le contexte spatial de l'image d'entrée.

On a alors présenté l'architecture en U de O. Ronneberger et al qui permet de réaliser une segmentation efficace en entraînant rapidement un réseau de neurones profond « fully convolutionnal » à l'aide d'un faible volume de données. Nous avons mis en œuvre une architecture adaptée avec laquelle nous avons pu expérimenté sur plusieurs base d'images. Les résultats sont très encourageant, cependant nous nous sommes heurtés à certaines limites matériels et de temps. Nous avons observé des phénomènes de sur apprentissage sur la base Cellules que nous avons tenté de corrigé avec certaines optimisations. D'autres de ces optimisations et tests seront mis en place, nous tenterons rapidement d'autres tailles d'images, nous tenterons de modifier l'architecture afin de la rendre plus profonde pour se rapprocher de l'originale (des premiers résultats montre un gain sur la base Cellules) et nous essayerons de segmenter des images histologiques en exploitant la couleur.

## Références

- [1] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in MICCAI, pp. 234–241, Springer, 2015.
- [2] S. Horowitz et T. Pavlidis : Picture segmentation by a directed split-and-merge procedure. Rapport de Recherche, Departement of Electrical Engineering, Princeton University, 1975.
- [3] M. Kass, A. Witkin and D. Terzopoulos, ”Snakes : active contour models,” Int. J. of Comp. Vis., vol.1(4), pp.321-331, 1988.
- [4] D. L. Pham, C. Xu, and J. L. Prince, “A Survey of Current Methods in Medical Image Segmentation,” in Annual Review of Biomedical Engineering, 2000, vol. 2, pp. 315–338.
- [5] Bezdek JC, Hall LO, Clarke LP. Review of MR image segmentation techniques using patternrecognition. Med Phys. 1993;20:1033–48.
- [6] Cyrille Faucheuex. Segmentation supervisée d’images texturées par régularisation de graphes. Image Processing. Université François-Rabelais de Tours, France, 2013. French. <tel-01131729>
- [7] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, Vol 65(6), Nov 1958, 386-408.
- [8] P. foucher, P. revollon, P. vigouroux. Segmentation d’images en couleurs par réseau de neurones : application au domaine végétal.
- [9] <https://blogs.msdn.microsoft.com/mlfrance/2016/04/28/une-premiere-introduction-au-deep-learning/>

- [10] Y. LeCun and Y. Bengio: Convolutional Networks for Images, Speech, and Time-Series, in Arbib, M. A. (Eds), The Handbook of Brain Theory and Neural Networks, MIT Press, 1995,
- [11] Teh, Y. and Hinton, G. E. (2001). Rate-coded restricted Boltzmann machines for face recognition. In Advances in Neural Information Processing Systems, volume 13.
- [12] Yann LeCun, « LeNet-5, convolutional neural networks »
- [13] Pierre Buyssens, Marinette Revenu, Olivier Lepetit. Réseau de Neurones Convolutionnels pour la Reconnaissance Faciale Infrarouge. GRETSI'09, Sep 2009, Dijon, France. 4 p., 2009. <hal-00812603>
- [14] Ciresan, D.C., Gambardella, L.M., Giusti, A., Schmidhuber, J.: Deep neural networks segment neuronal membranes in electron microscopy images. In: NIPS. pp. 2852–2860 (2012)
- [15] Matthew Lai, Daniel. Deep Learning for Medical Image Segmentation. Rueckert Apr 29, 2015
- [16] Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., Pal, C., Jodoin, P.-M. and Larochelle, H. (2016) ‘Brain tumor segmentation with deep neural networks’, Medical Image Analysis, 35, pp. 18–31. doi: 10.1016/j.media.2016.05.004.
- [17] Pierre Buyssens, Abderrahim Elmoataz. Réseaux de neurones convolutionnels multi-échelle pour la classification cellulaire. RFIA 2016, Jun 2016, Clermont-Ferrand, France.
- [18] de Brébisson, A. and Montana, G. (2015) Deep neural networks for anatomical brain segmentation, 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), . doi: 10.1109/cvprw.2015.7301312.
- [19] Noh, H., Hong, S. and Han, B. (2015) ‘Learning Deconvolution network for semantic segmentation’, 2015 IEEE International Conference on Computer Vision (ICCV), . doi: 10.1109/iccv.2015.178.

- [20] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In ICANN, pages 92–101, 2010.
- [21] Seyedhosseini M., Sajjadi M., Tasdizen T. Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: Computer Vision (ICCV), 2013 IEEE International Conference on. pp. 2168{2175 (2013)
- [22] Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation (2014), arXiv:1411.4038 [cs.CV]
- [23] <http://blog.octo.com/classification-dimages-les-reseaux-de-neurones-convolutifs-en-toute-simplicite/>

## Annexe

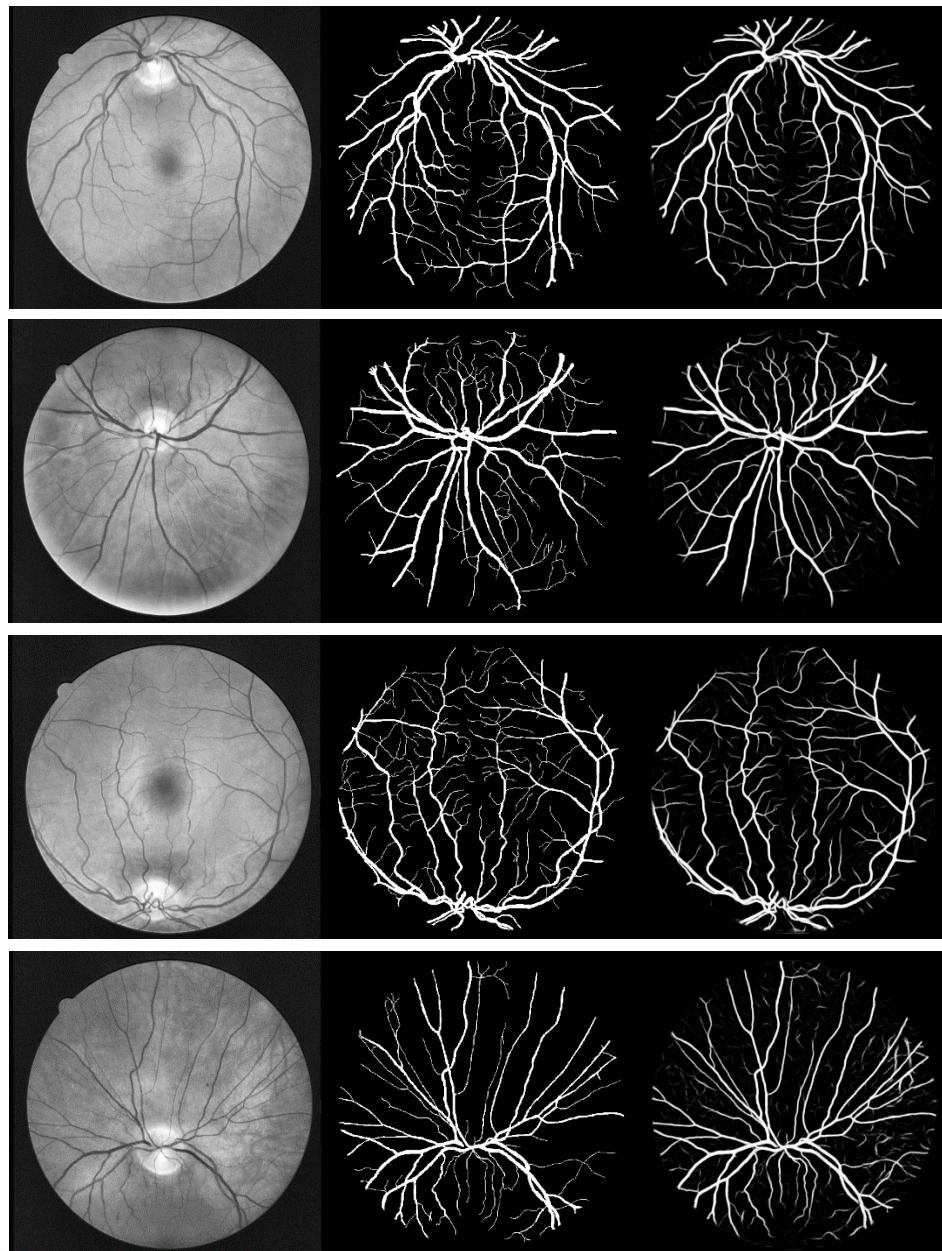


Figure 40 Quelques résultats de segmentation pour la base DRIVE

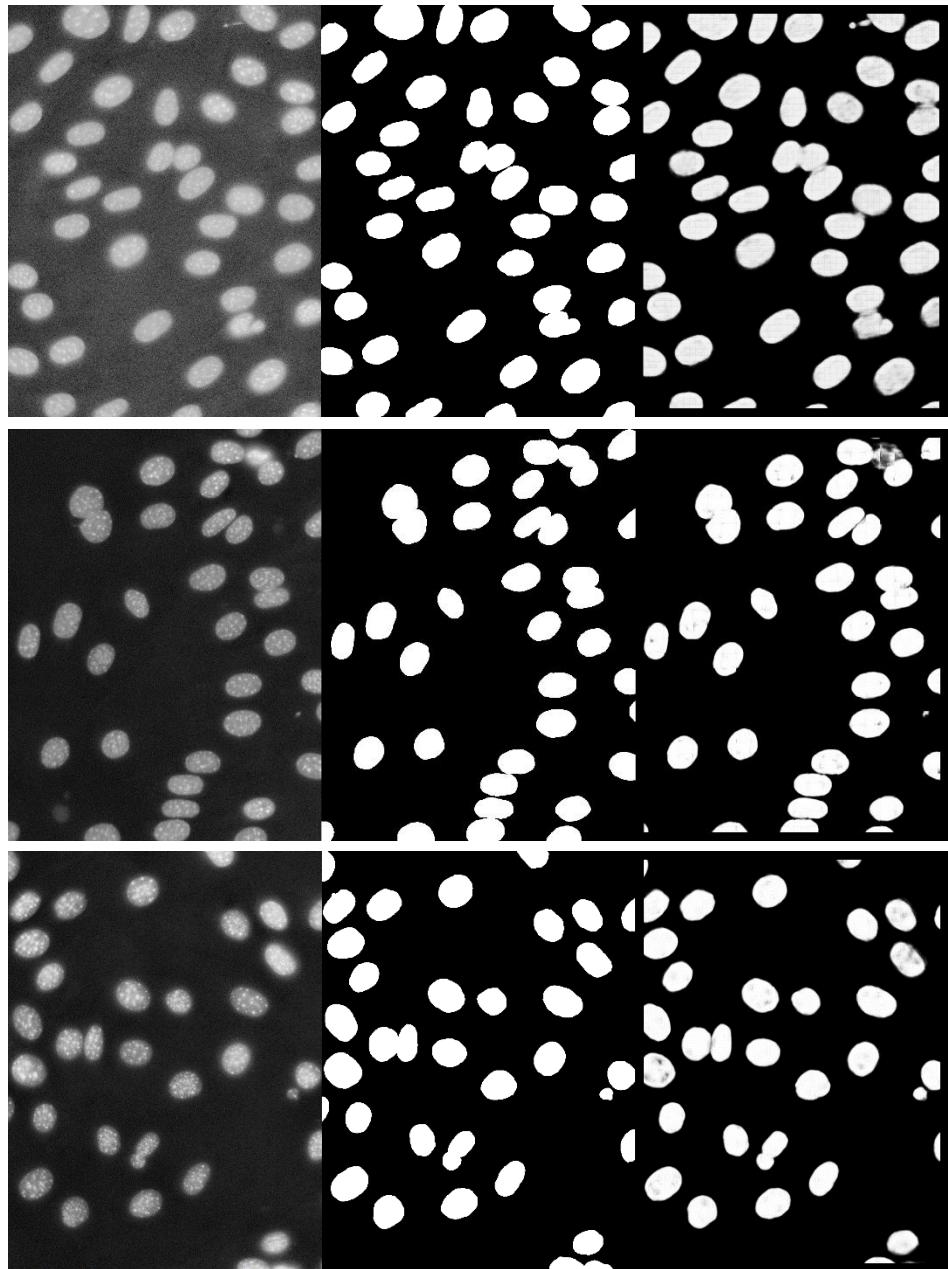


Figure 41 Quelques résultats de segmentation pour la base Cellules