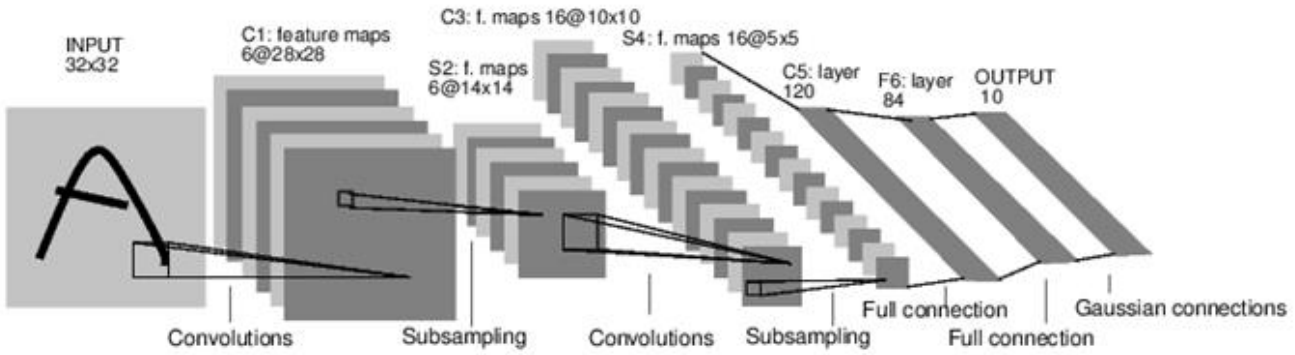
2020.03.02

U-Net

Youngtaek Hong, PhD

Fully-Convolution Net

- U-Net이란, 기존 CNN(Convolution Neural Network)의 한계점을 극복하기 위해 만들어진, 이미지 영역화의 끝!
- U-Net 모델의 등장 전까지 모든 CNN은 단순히 하나의 명확하게 나뉘어지지 않은 Object를 학습할 수 밖에 없었습니다.
- 이러한 이유는 기존 CNN의 단점에 의한 것으로 볼 수 있습니다.
- 초기 CNN의 모델은 LeNet을 활용하였으며, CNN 적용의 가장 기초적인 Layer 구성 모델입니다.



A Full Convolutional Neural Network (LeNet)

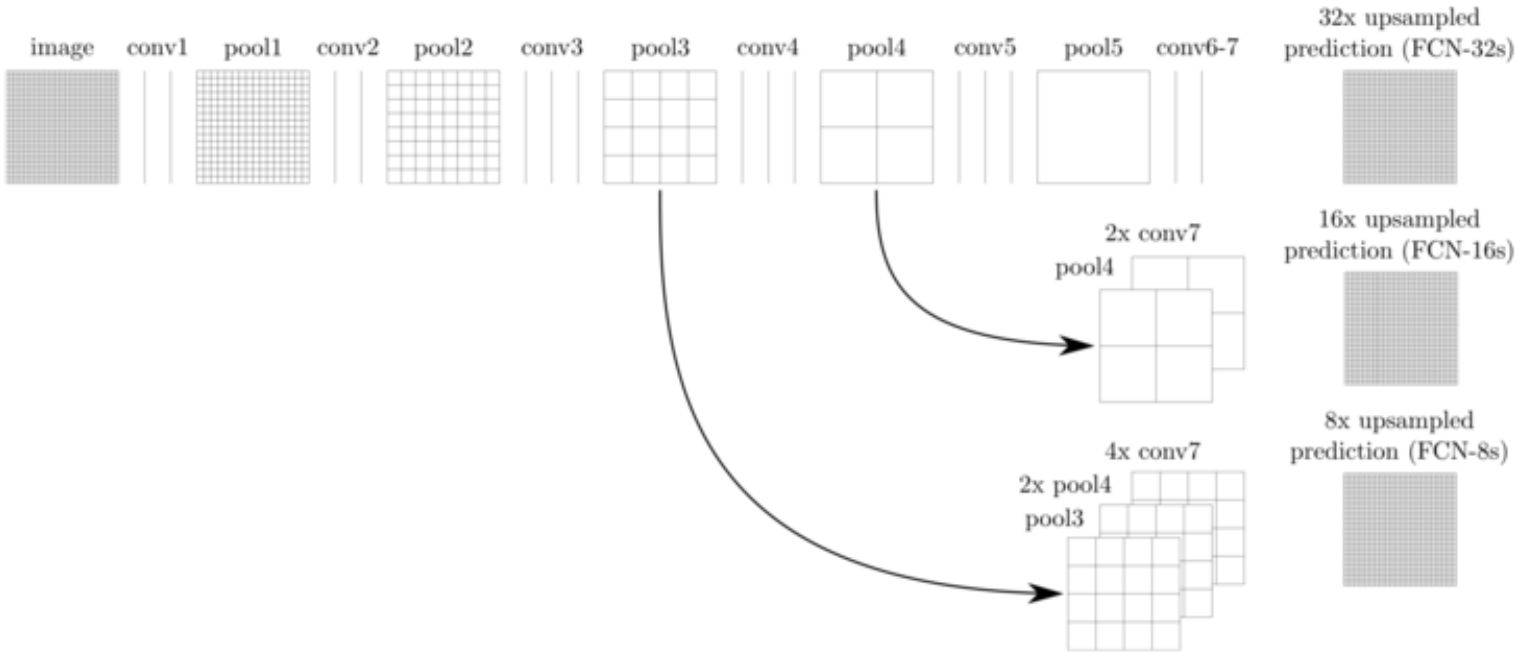
Fully-Convolution Net

- 기존 CNN은 Convolution Filter를 활용하여 이미지의 특징을 간추려내고, 마지막에 Fully-Connected Layer 형태를 구성하기 위해 추출된 Feature들을 1-d Vector로 배치합니다.
- 그리고 최종적으로 특징들을 연산한 결과가 어떤 값을 가르키는 지 Output Neuron을 통해 나타내게 됩니다.
- 하지만, 이러한 CNN의 기본적인 틀은 1차원 Matrix의 대해서만 연산이 가능하며, 따라서 보다 상세하게 Feature를 추출하기가 어려웠습니다.
- 이러한 단점은 이미지가 특징만 포함하지 않고 특징을 포함한 어떤 영역의 데이터를 학습해야만 하는 결과를 가져왔으며, Detection 분야에서 한계를 맞이하게 됩니다.

Fully-Convolution Net

- Fully-convolution Net는 기존 CNN의 가장 대표적인 초기 모델인 LeNet의 한계를 극복하고자 만들어 졌습니다.
- Matan Sela[]의 말에 의하면, 초기 CNN의 모델은 One-Dimensional Input의 대해서만 처리가 가능한 좋지 못한 모델의 대해 Multi-dimensional input을 위한 Decoding 모델을 발표했습니다.
- 이 모델이 향후 Wolf&Platt[] 두 저자에 의해 발전되어지고, Fully-Convolution Net 모델이 만들어지게 되었습니다.

Fully Convolutional Network



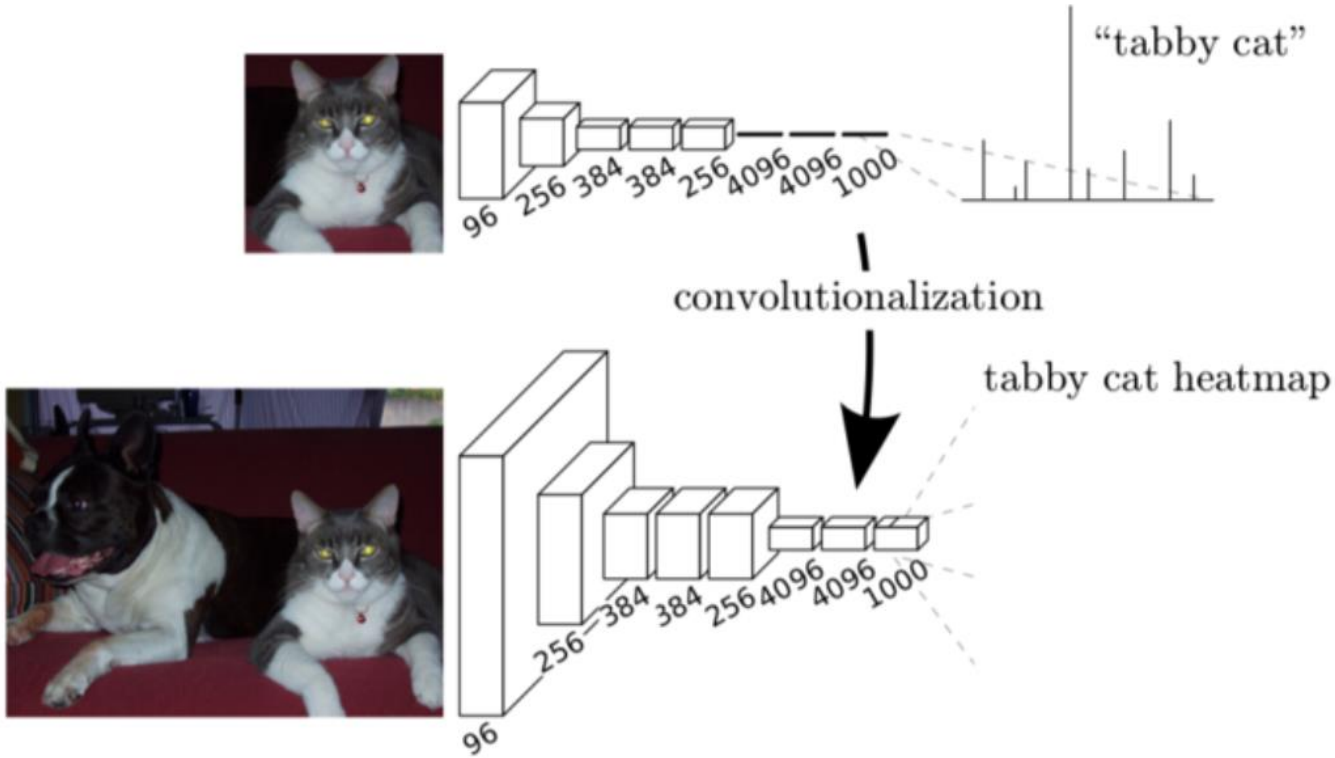
- 우선 Fully-Convolution Net은 3차원 데이터를 곧바로 분류할 수 있도록 만들어진 모델입니다.
- 따라서, 너비(w)와 높이(h), 그리고 채널(c)라는 3가지 차원을 가진 기본적인 이미지 데이터를 입력으로 받습니다. 이때 입력되는 채널은 일반적으로 3의 크기를 가지며, 이를 "RGB"영역에 의한 3의 vector를 가진다고 표현합니다.

Fully Convolutional Network

- Fully-Convolution Net은 Output을 연산하기 위해 기존의 Fully-Connected의 선형 패턴을 통해 출력되는 결과들을 비선형 패턴에 적용되는 함수로 탈바꿈하게 됩니다.
- 쉽게 말해서, 특징을 특정하기 위해 transpose convolution filter를 활용하여 이미지 자체를 복원하는 시간 Decoding 과정을 가지게 됩니다.
- 이때 Filter, 즉 이 Filter를 구현하기 위해 사용된 모델을 지칭하여 Deep Filter, 또는 Fully-Convolution Network라고 부르게 됩니다.
- 기존의 Fully-Connected Layer는 이미지의 특정 부분을 단순이 Scoring하는 방법이였으며, 이는 현실적으로 대량의 학습 이미지와 더불어 이미지 자체를 상세하게 분석할 수는 없었습니다.

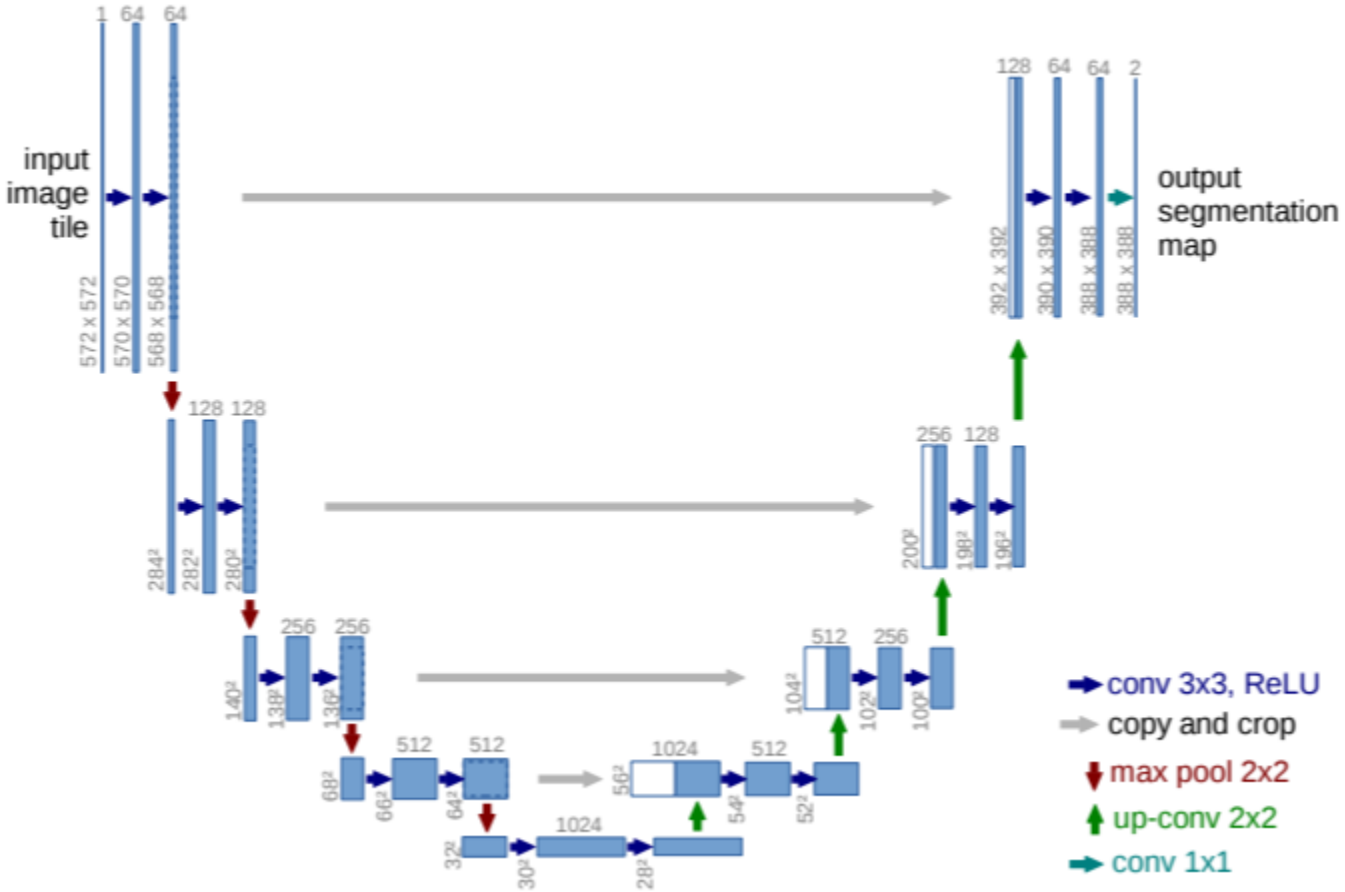
Fully Convolutional Network

- 기본적으로 Fully-Connected Layer는 MLP 이론에 따라 특정하고자 하는 고정된 Dimension 내 모든 Pixel값들을 입력으로 진행합니다.
- 이러한 방법은 바꿔말하면, 특정하고자하는 일부 영역(Kernel)을 하나의 입력 매개체로 사용할 수도 있다고 해석할 수가 있습니다.
- 따라서 이 Idea를 활용하여 Convolution Filter가 등장하였고, 실제로 그 동안 인공지능 분야에서 이미지를 처리하는 알고리즘 모델 LeNet의 한계점을 돌파하고, 그 모델이 군림하던 시대를 끌어내리게 되었습니다.
- 또한 FCN(Fully-Convolution Net)모델을 이용하여 점차 더 많은 이미지 처리, 영상 비전 등의 다양한 영역에 적합한 새로운 알고리즘 모델들이 탄생하게 되었고, 지금도 가장 많이 활용되고 있는 핵심 모델이 됩니다.



(Figure 3) quote to figure-2 in Fully Convolutional Networks for Semantic Segmentation

U-Net


input image tile
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

Abstract. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. We show that such a network can be trained end-to-end from very few images and outperforms the prior best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. Using the same network trained on transmitted light microscopy images (phase contrast and DIC) we won the ISBI cell tracking challenge 2015 in these categories by a large margin. Moreover, the network is fast. Segmentation of a 512x512 image takes less than a second on a recent GPU. The full implementation (based on Caffe) and the trained networks are available at http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net.

논문에서 처음에 소개하는 내용은 지난 2년동안 (U-Net은 2015년 5월에 발표되었습니다.) deep convolution network는 많은 visual recognition 작업에서 매우 좋은 성능을 보였지만, training set의 크기와 고려할 네트워크의 크기 때문에 그 성공은 제한적이었다고 말하고 있습니다.

Convolution네트워크의 일반적인 용도는 이미지에 대한 출력이 단일 클래스 레이블인 분류 작업에 있지만, 많은 시각 작업, 특히 biomedical processing에서 원하는 출력은 localization을 포함해야 하며, 즉 클래스 라벨은 각 pixel에 할당 되어야 한다고 합니다.

U-Net에서 핵심으로 말하고 있는 내용은 세가지로 생각됩니다.

1. Convolution Encoder에 해당하는 Contracting Path + Convolution Decoder에 해당하는 Expanding Path의 구조로 구성. (해당 구조는 Fully Convolution + Deconvolution 구조의 조합)
2. Expanding Path에서 Upsampling 할 때, 좀 더 정확한 Localization을 하기 위해서 Contracting Path의 Feature를 Copy and Crop하여 Concat 하는 구조.
3. Data Augmentation

두가지 단점은,

1. 네트워크가 각 패치에 대해 개별적으로 실행되어야 하고 패치가 겹쳐 중복성이 많기 때문에 상당히 느리다.
2. localization과 context사이에는 trade-off가 있는데, 이는 큰 사이즈의 patches는 많은 max-pooling을 해야해서 localization의 정확도가 떨어질 수 있고, 반면 작은 사이즈의 patches는 협소한 context만을 볼 수 있기 때문입니다.

Contracting Path에서 Pooling되기 전의 Feture들은 Upsampling 시에 Layer와 결합되어 고 해상도 output을 만들어 낼 수 있습니다.

하나 더 중요한 점은! 많은 수의 Feature Channels를 사용하는데요. 아래 네트워크 아키텍쳐를 보시면 DownSampling시에는 64 채널 -> 1024채널까지 증가 되고, UpSampling시에는 1024 채널 -> 64채널을 사용하고 있습니다.

네트워크는 fully connected layers를 전혀 사용하지 않고, 각 layer에서 convolution만 사용합니다.

다음으로, U-Net에서는 Segmentation시 overlab-tile 전략을 사용합니다. (그림.2)

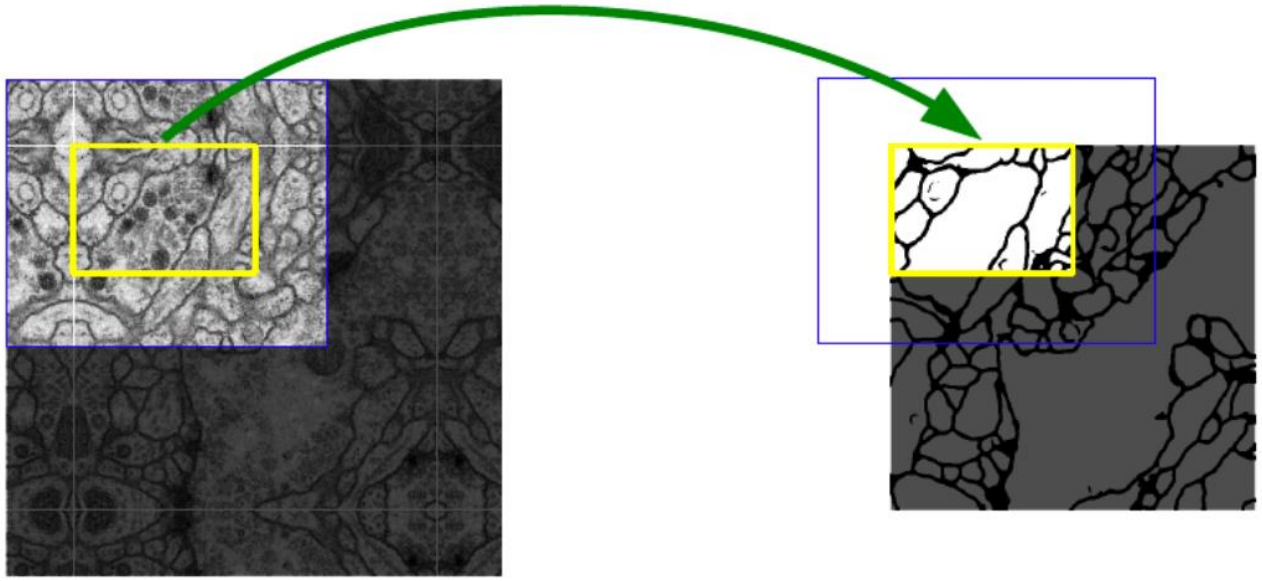
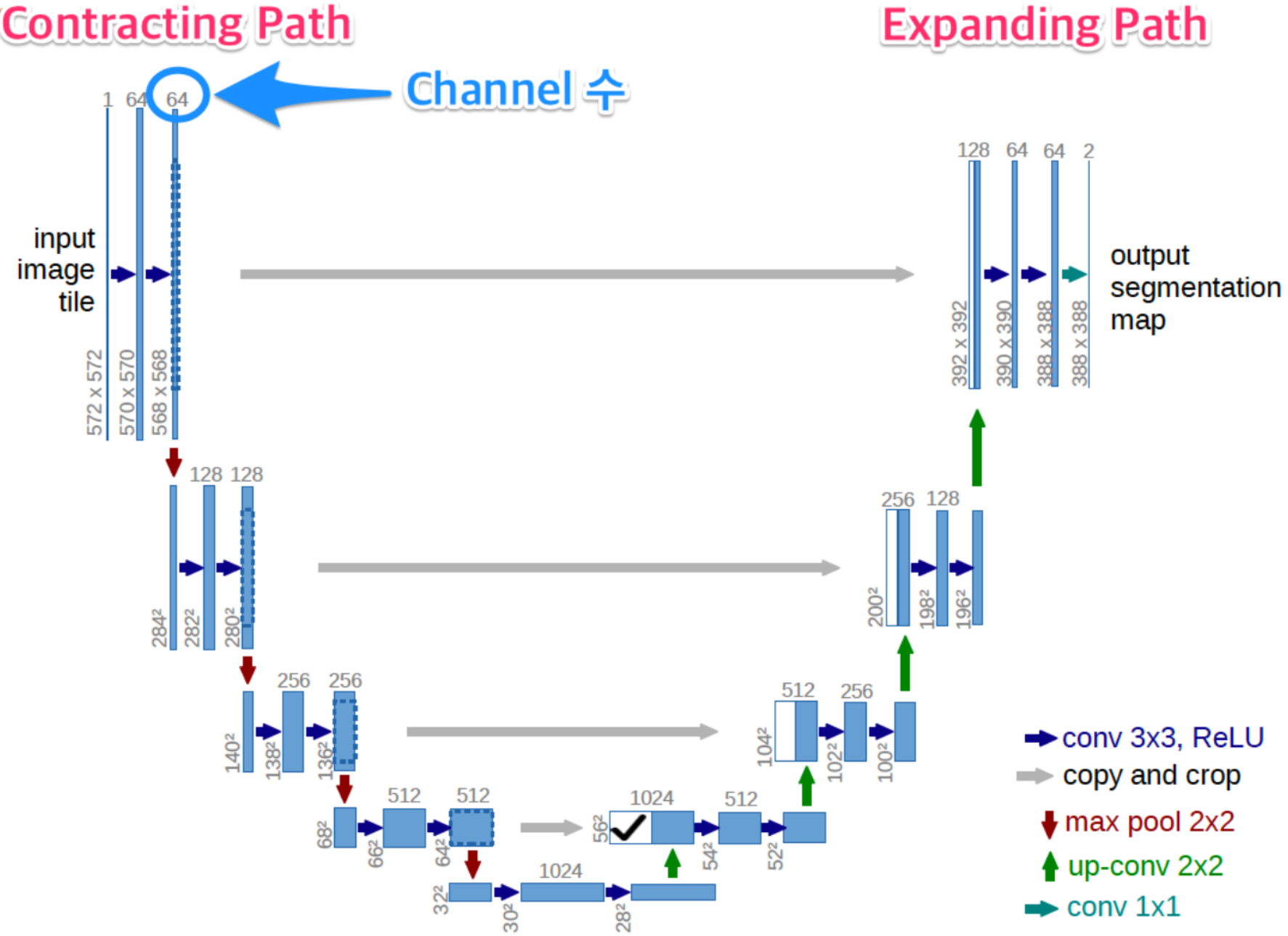


Fig. 2. Overlap-tile strategy for seamless segmentation of arbitrary large images (here segmentation of neuronal structures in EM stacks). Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input data is extrapolated by mirroring


Contracting Path
Expanding Path
Channel 수
input image tile
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

Contracting Path는

1. 전형적인 Convolution network 이고,
2. 두번의 3X3 convolution을 반복 수행하며 (unpadded convolution를 사용),
3. ReLU를 사용합니다
4. 2X2 max pooling 과 stride 2를 사용함
5. downsampling시에는 2배의 feture channel을 사용하고

Expanding Path는

1. 2X2 convolution (up-convolution)을 사용하고,
2. feature channel은 반으로 줄여 사용합니다.
3. Contracting Path에서 Max-Pooling 되기 전의 feature map을 Crop 하여 Up-Convolution 할 때 concatenation을 합니다.
4. 두번의 3X3 convolution 반복하며
5. ReLU를 사용합니다

마지막 Final Layer에서는 1X1 convolution을 사용하여 2개의 클래스로 분류합니다.

3. Training

학습은 Stochastic gradient descent 로 구현되었습니다.

이 논문에서는 학습시에 GPU memory의 사용량을 최대화 시키기 위해서 batch size를 크게 해서 학습시키는 것 보다 input tile 의 size를 크게 주는 방법을 사용하는데요, 이 방법으로 Batch Size가 작기 때문에, 이를 보완하고자 momentum의 값을 0.99값을 줘서 과거의 값들을 더 많이 반영하게 하여 학습이 더 잘 되도록 하였습니다.

a: HeLa cells 현미경 raw image
b: color를 다르게한 ground truth
c: segmentation mask white - foreground / black-background
d: pixel-wise loss를 사용해서 세포의 경계선을 학습시킴

5

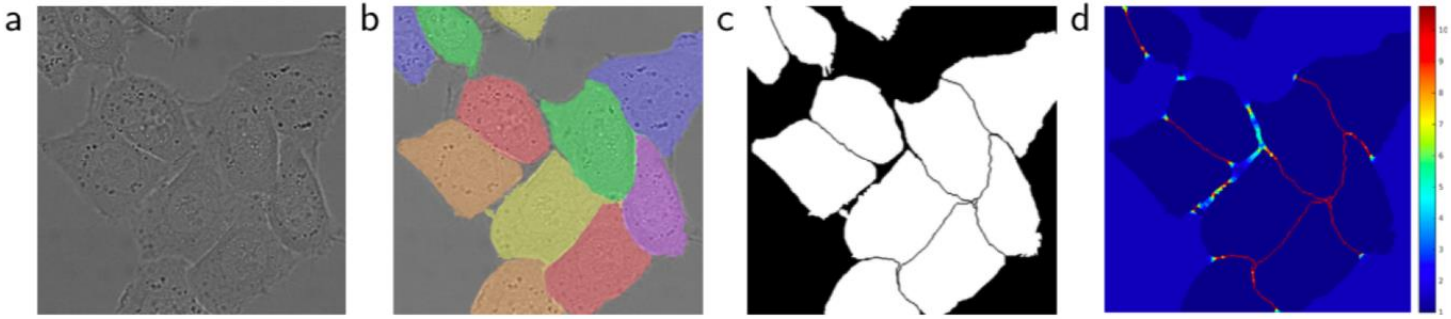


Fig. 3. HeLa cells on glass recorded with DIC (differential interference contrast) microscopy. (**a**) raw image. (**b**) overlay with ground truth segmentation. Different colors indicate different instances of the HeLa cells. (**c**) generated segmentation mask (white: foreground, black: background). (**d**) map with a pixel-wise loss weight to force the network to learn the border pixels.

softmax

$$p_k(\mathbf{x}) = \exp(a_k(\mathbf{x})) / \left(\sum_{k'=1}^{K} \exp(a_{k'}(\mathbf{x}))\right)$$

Cross Entropy Loss with w(x)

> *각각 정답 픽셀에대한 cross entropy loss에는* $w(\mathbf{x})$ *라는 가중치 값이 추가됩니다. 여기서* $p_{l(x)}(x)$ *의* $l(x)$ *는 정답 클래스 즉 위* ***softmax 수식****에서 정답의 레이블에 해당하는* k *값을 반환하는 함수 입니다.*
>
> *cross entropy 함수는 정답의 추정값을 log에 사용하기 때문에 이에 해당하는 정답의 확률을 가져오는 것이죠. 수식 (1)은 loss 값에 가중치* $w(\mathbf{x})$ *를 곱한 형태이며 이제 우리가 살펴볼 것은* $w(\mathbf{x})$ *입니다.*

$$E = \sum_{\mathbf{x} \in \Omega} w(\mathbf{x}) \log(p_{\ell(\mathbf{x})}(\mathbf{x})) \qquad (1)$$

$w(\mathbf{x})$ **구하는 법 :**

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right) \qquad (2)$$

$w(\mathbf{x})$는 두개의 텀의 합으로 구성됩니다. $w(\mathbf{x})$는 x 위치의 픽셀에 가중치를 부여하는 함수입니다.

$w_c(\mathbf{x})$는 $\mathbf{x}$ 의 위치에 해당하는 클래스의 빈도수에 따라 값이 결정됩니다.

즉 학습데이터에서 $\mathbf{x}$ 픽셀이 background일 경우가 많은지 foreground일 경우가 많은지의 빈도수에 따라 결정된다고 보시면 됩니다.

그 뒤의 exp 텀은 d_1, d_2 함수를 포함하는데 d_1 은 $\mathbf{x}$ 에서 가장 가까운 세포까지의 거리이고 d_2는 두번째로 가까운 세포까지의 거리를 계산하는 함수입니다.

즉 $\mathbf{x}$는 세포사이에 존재하는 픽셀이며 두 세포사이의 간격이 좁을 수록 weight를 큰 값으로 두 세포 사이가 넓을 수록 weight를 작은 값으로 갖게 됩니다. 이는 그림 3(d) 를 보시면 명확하게 확인하실 수 있습니다.

네트워크 파라미터의 초기화는 He 초기화 방법을 적용하였습니다.

Table 2. Segmentation results (IOU) on the ISBI cell tracking challenge 2015.

Name	PhC-U373	DIC-HeLa
IMCB-SG (2014)	0.2669	0.2935
KTH-SE (2014)	0.7953	0.4607
HOUS-US (2014)	0.5323	-
second-best 2015	0.83	0.46
u-net (2015)	**0.9203**	**0.7756**

5. Conclusion

마지막으로 결론입니다.

U-Net 구조는 매우 다른 biomedical segmentation applications에서 좋은 성능을 보였고, 이 성능을 보일 수 있었던 것은 Elastic 변환을 적용한 data augmentation 덕분이고, 이것은 annotated image가 별로 없는 상황에서 매우 합리적이었다고 합니다.

학습하는데 걸렸던 시간은 NVidia Titan GPU(6GB)를 사용했을때, 10시간이었습니다.

이 U-Net 구현은 Caffe 기반으로 제공되고 있으며, U-Net의 아키텍쳐는 다양한 task에서 쉽게 적용되서 사용될 것을 확신한다고 하면서 논문은 마무리 됩니다.

Image Segmentation Task에서 가장 많이 쓰이는 U-Net은 U자형 아케텍쳐와 Fully Convolution & Deconvolution 구조를 가지고 있는 것으로 좀 더 정확한 Localization을 위해서 Contracting Path의 Feature를 Copy and Crop해서 Expanding Path와 Concat하여 Upsampling을 한다는 것을 확실하게 알아두면 좋을 것 같습니다.