# U-Net's Excellence and Optimal Loss Function in Medical Segmentation

Sanghyun Kim

Submitted for the Degree of Master of Science in

Your MSc Programme


Department of Computer Science
Royal Holloway University of London
Egham, Surrey TW20 0EX, UK

September 7, 2020

# Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

**Word Count: 9919**

**Student Name: Sanghyun Kim**

**Date of Submission: September 7, 2020**

**Signature:**

# Abstract

With recent advances in segmentation methods and loss functions it is now possible to generate segmentation medical images which quite resemble manual depictions of the doctors. U-Net, the Convolution neural network-based semantic segmentation technique, and its various loss function have allowed this marvelous performance. Our main goal is to show the excellence of U-Net in medical segmentation and finding its optimal loss function. This project provides concepts, mechanisms, and taxonomies of segmentation, Deep neural network, and Convolution neural network, which are base theories of U-Net. Consequently U-Net and loss functions for medical segmentation are intuitively interpreted with mathematical and infographic approaches. After theory research, experiment of U-Net and loss function variation is described step by step, accompanied by specific information. At the last, there are conclusions, U-Net generates segmentation images from input data, which operate quickly with very competitive performance. The best loss function is Focal loss in case of loss rate. When it comes to running time, IoU loss is the optimal.

# Contents

# 1 Introduction

## 1.1 Advent of computer-aided biomedical Imaging

In Biomedical industry, Imaging analysis conventionally has depended on expert in both clinical and research areas. However, recent deep learning developments have shown that artificial intelligence could perform better than humans in recognition tasks in medicine and healthcare, especially in medical imaging. In clinical pathology, usage of deep learning is surging for diagnosing medical imaging such as Positron Emission Tomography (PET), magnetic resonance imaging (MRI), computed tomography (CT) and ultrasound. On the research side, it is being produced visible images of inner structures of the body and interior tissues which seeks the disease identification and management.

## 1.2 Image segmentation in biomedical Imaging

Image segmentation is a main domain of medical image process which extracts the region of interest (ROI). The purpose is easy analysing, interpreting medical images with preserving the quality and Tracing objects' borders in the images. With defining the borders of the ROI, medical analysts can simplify the decisions. It is also a vital step that determines the result of the whole process because the rest of the analysis fully depends on the output from image segmentation phase. When there is not much medical image information, experts can directly perform image segmentation in hands. However explosive increase in the amount of medical image information, they could not solve in handwork method anymore. This causes the deep learning based automatic segregation method in Image segmentation. For example, Cancer has long been a deadly illness. Despite of contemporary technological developments, cancer can be lethal unless identifying it at an early stage. Quick detecting cancerous cells can save patients from cancers. The cancerous cells' shape is critical when medical analysts determining the severity of the cancer which means that false identifying (putting the healthy cells and cancerous cells together) can cause death of patients. Deep learning-based image Segmentation method can be powerful solution of this kind of problem.

## 1.3 Requirements in biomedical image segmentation

### 1.3.1 Several deep learning-based techniques and their limitation

Many deep learning-based techniques have been devised for image segmentation

such as region-based [1], edge detection [2] and pixel-based clustering segmentation [3]. Region-based segmentation is method of separating the objects into different regions with some threshold value. The drawback is that insignificant grayscale difference or an overlap of the grayscale pixel values in data set can occur difficulties in accurate segments. Making use of an image's discontinuous local features to detect edges and defining a boundary of the object is edge detection segmentation, inconsistent too many edges and less contrast between objects in the image. Separating the pixels of the image into homogeneous clusters is pixel-based clustering segmentation. Its limitation is computational expensive and inconsistent for clustering non-convex clusters. For these reasons, they are not suitable for medical image segmentation which need the amount of relevant labelling rand has monotonous colour data with a few edge objects.

### 1.3.2 Necessity of Semantic segmentation

In biomedical imaging, annotating on pixel's label, for example the different types of diseases like cancer, tumour, play crucial role. The types of deadly maladies could be hundreds or more, so analysts are studying pixel's class whether it is malady or what kind of maladies. In this manner, Semantic segmentation is major method in biomedical image segmentation because it identifies the object class of each pixel for whole objects within an image.

## 1.4 Motivation for U-Net and its optimal loss functions

U-Net is a CNN-based semantic segmentation model which is significant breakthroughs in the medical image segmentation. Recently many researches have been improved based on U-Net to develop the performance of semantic segmentation. From V-Net [4] in 2016 to UNET 3+ [5] in 2020, 54 models have been devised and advanced from U-Net. I aim at anatomizing architecture and advancement of U-net with variation of loss function and experiment because I want to know what features have made mainstream of medical image segmentation in both architecture and loss function. Furthermore, I comprehend Convolution neural network, concept of segmentation etc. Using mathematical methods which are basement of understanding in U-net model mechanism. During researching, I could grasp deeply in basic deep learning technology and how medical image analyst's work in industry which allows me to think about that aspect of our work.

# 2 Background research

## 2.1

## 2.1 Type of segmentation

Object detection, instance segmentation and semantic segmentation are fundamental of computational visual tasks in medical imaging. Figure 1 is general concept examples of three tasks to understand simply.

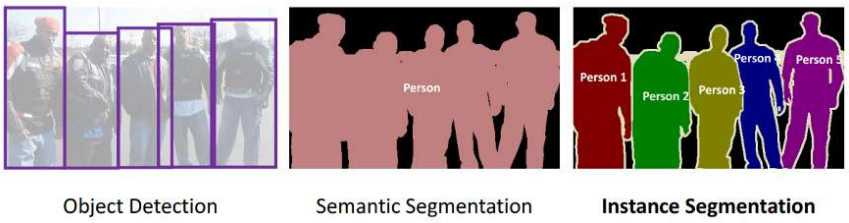



Figure 1

### 2.1.1 Object Detection

Object detection is a kind of computer vision task which is basic step of segmentaion. It carries out instance detection of visual objects in digital images. For example, humans, animals, or cars are classes of visual objects. [6]; This process consists of two steps. At first, object localization is the task that determining a bounding box as tightest as possible to locate the exact position of the object in the image. Secondly, image classification is classifying localized object to label. Common object detection tasks assume multi classes objects in an image at once. In other words, multi-labelled classification, and bounding box regression (predicting box location) are combined. Figure 2 shows object detection that an object in an image. But Figure 3 shows object detection that multi-labelled multiple objects in an image.

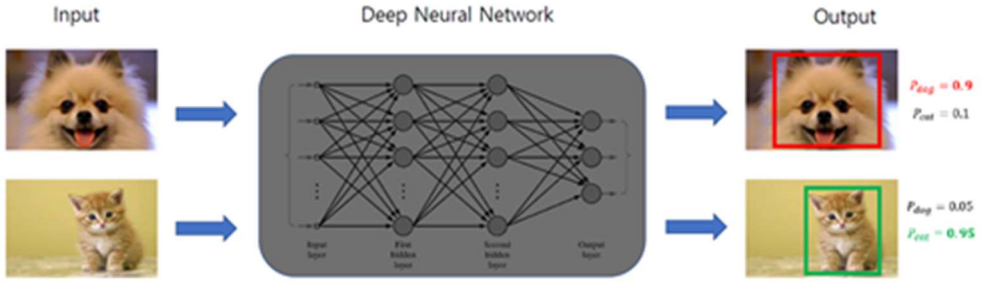



Figure 2: one object in one image

Figure 3: multiple objects in one image

The object detection models can be divided into two different categories. two stage approach and one stage approach. [7] In two stage approach, reginal proposal and classification are performed sequentially. Regional proposal is object localization step, finding regions which have high likelihood of object; it creates sparse proposal set. Classification encodes feature vector of created proposal set with deep convolutional neural networks and predict class of objects. Two stage approaches have high accuracy but slow. One stage approach conducts reginal proposal and classification simultaneously unlike two stage approaches. Considering whole location in an image as potential objects, each regions of interests are classified as background or target objects. These methods are quite faster than two stage approaches, so they are suitable to real time system. However relatively lower output.

### 2.1.2 Instance segmentation

Instance segmentation models can be defined as the method of detecting and describing each distinct object of interest in an image [8]; It was combination between classification method and object detection method to aim at predicting the object class-label and the pixel-level object instance-mask. Instance segmentation model typically is used in splitting objects of the same class into different instances. However, it is not easy to automize this process which means that the number of instances is initially unknown and pixelwise evaluation cannot be applied in prediction. Despite this problem, Instance labelling is advantageous; extra reason information about occlusion situations, counting elements the same class and detecting an object. It is applied to robotics and automatic drive tasks.

Instance segmentation methods are typically divided two categories: detection-based methods and segmentation-based methods, as below Figure 4 and 5. The detection-based methods concentrate on performing proposal that objects are tightly bounded by rectangular boxes and objecting detection which masks the objects in the predicted bounding boxes. [9,10] On the contrary, the segmentation-based methods [11,12] are different from detection-based methods. At the first step, a pixel-level segmentation map is synthesized from the entire image. After, the map recognizes target instances.

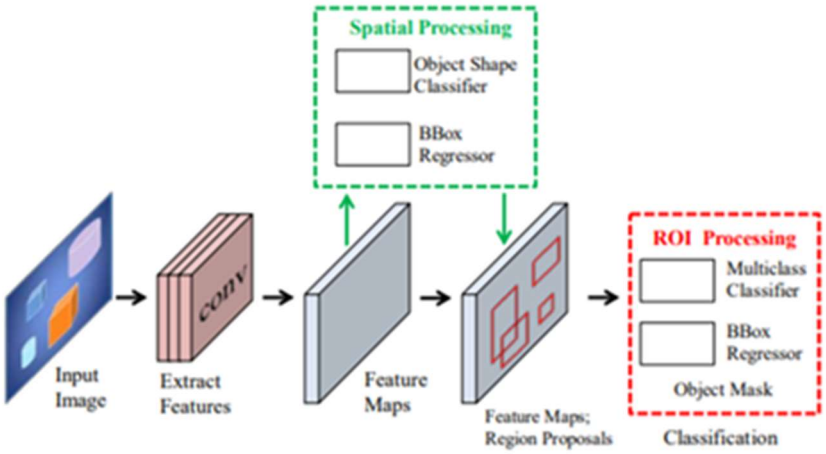



Figure 4 General framework for detection-based methods


Figure 5: General framework for segmentation-based methods

Instance segmentation models have two advantages: segmentation accuracy and overlapping object differentiation. Since considered not whole image but individual ROIs, the segmentation accuracy improves. Semantic segmentation methods always use pixel-level classification which make separation hard. But, in instance segmentation techniques, overlapping objects of the same class are easily separated. it is vital to diatom identification such as count the number of specimens of each class. Nevertheless, instance segmentation has an insignificant harm. Detected individual instances will be segmented exclusively when object detection methods are used in finding the individual instances. If objection detection gets mistakes, the output will be bad. When it comes to category of instance segmentation, detection-based methods are relatively simple to implement and have modest segmentation accuracy. But they have difficulty to optimize training and are slow in both training and test. Also, there are storage, time, and detection-scale issues during training. Lastly, they are not suited for real time applications. Segmentation-based methods' benefits are simple in training, better generalization, relatively faster and good segmentation accuracy. Drawbacks are that they depend on a complicated training pipeline which is difficult to train and to optimize. [13]

### 2.1.3 Semantic segmentation

Semantic segmentation is the task of assigning a class to every pixel in each image. The prediction of pixel deals with both the class and the boundaries of each kind object; it reflects the spatial relationship among all objects in one image. [14] In short, the purpose of semantic segmentation is partitioning an image into subsets in mutually exclusive manner; each subset embodies a meaningful region of the original image.

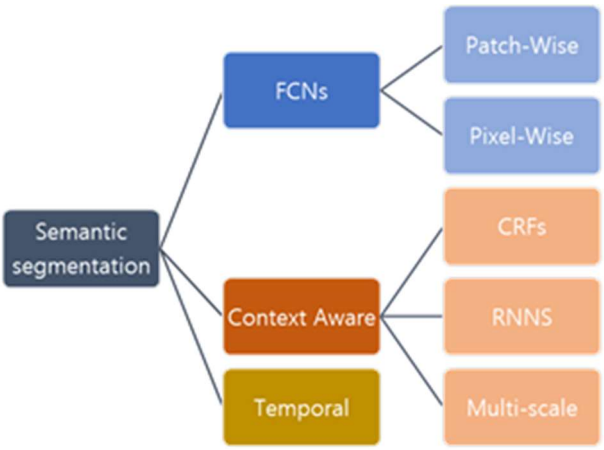



Figure 6: Taxonomy of semantic segmentation approaches

There are three main categories in semantic segmentation methods as you see above (Figure 6): Fully Convolutional Networks, Context Aware Models and Temporal Models. [15] FCNs family are divided patch-wise and pixel wise. Context Aware have three subcategories such as CRFs, RNNS, Fully Convolutional Networks

Fully Convolutional Networks (FCNs) is named from their architecture, which is devised by Long et al in 2015. [16] FCNs and its family's main task is to propose a novel solving techniques with convolutional neural network structure which is designed to get efficient inference result; it can allow the output prediction consistently sized with a random input size and get a more competent result. General architecture of FCNs is consists of down sampling path and up sampling path. (Figure 7) Down sampling path capture semantic and contextual information, using pooling and stride convolution. Specifically describing, an input image is downsized and undergoes fully connected (FC) convolution layers, consequently output one predicted label for the input image. fully connected layer of down sampling is regarded as $1 \times 1$ convolution which can remain information of location and. Up sampling path is used to allow precise localization, using deconvolution. Deconvolution is process get the output size larger. There are several methods of deconvolution such as unspooling, max unspooling and transpose convolution. FCNs use transposed convolution, the network can learn heatmaps to

get dense predictions.

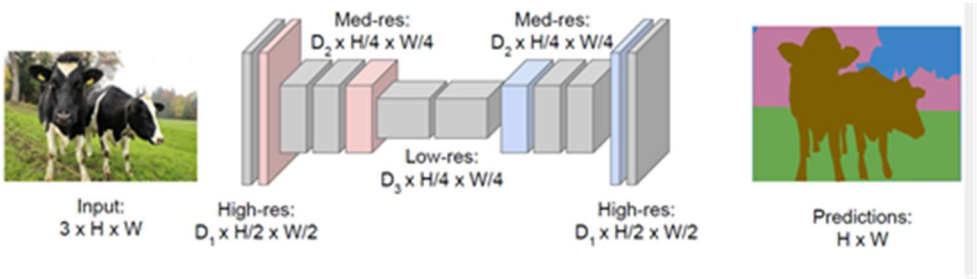



Figure 7: General framework for FCNs

Patch-wise strategy do not use training set of full images to avoid the redundancies of full image training. The method called "patch wise sampling" feeds the network accompanied by batches of random patches from the training set; patch means rectangular image regions around the objects of interest. patch wise sampling makes sure the enough variance input, representing the training dataset's validation, faster converging, and the classes' balance. There are several breaks thorough of patch-wise techniques such as MINC [17] and MultiPatch Decon. [18] Pixel-wise techniques use classification which predicts a class label for every pixel and take fundamental difference with entire image classification matters. Because every pixel has a label, Pixel-wise classification allow easy collection training image patches from an image. Furthermore, the network learned heatmaps that were up sampled with subsampled pooling and transposed convolution to get dense predictions. The full image is utilized for inferring dense predictions, which is different from patch-wise techniques. U-Net and SegNet [19] are representative of these methods.

Context aware models give explanation of similarity of pixels that belong to an individual object when image segmentation; they are categorized into multi-scale support, utilizing conditional random fields (CRFs), or recurrent neural networks (RNNS). Generally, multi-scale methods use a series of feature vectors that are calculated from regions of multiple sizes near every pixel in the image; the convolution neural network which is multi-scale consists of multiple copies of a single network. The input image go through makes the single network with different scales of a Laplacian pyramid. These mechanisms cover a large contextual information. Dilated Conv [20] is exemplified of this technique. During semantic segmentation, feature map' size decrease so detail information can be lost. Conditional random field (CRF) methods can solve this problems, utilizing the fully connected conditional random fields as a post processing ; the first CRF is devised by Chen et al. [21] The unary potentials of the CRF are set to the probabilities based on their convolutional network, on the other hand paired potentials are gaussian kernels from the spatial and colour features. Another subcategory to be contextual is using recurrent neural networks (RNN). Vanilla RNNS' family method makes a set of masks according to their labels with recurrent neural networks to capture the long-range dependencies of various regions, which ensures a better Context-aware segmentation. Drawbacks of vanilla RNN is the vanishing gradients, but gated

recurrent architectures can solve this problem. Reseg[22] can be representative of RNNS family.

Temporal models are segmentation methods that extracting the region of interest of each frame, using temporal information of sequences extracted from different frames in a given time interval; there are temporal information categories such as Background Subtraction, Temporal Differencing, Optical Flow, and the combinations of them have been analyzed.[23] Background Subtraction is detecting the moving objects from the difference of the current frame from a reference frame. Temporal differencing utilizes value difference between pixels in the same position in consecutive frames to extract moving regions. Optical flow is a method of moving extraction from relative local moving between two observations of an object. Temporal models are mainly used in video semantic segmentation. ClockWorks[24] is exemplified in these methods.

## 2.2 Deep Neural Network

Deep neural network (DNN) is a neural network accompanied by a degree of complexity, a neural network with more than two hidden layers. [25] It can construct complex non-linear relationship model like common neural network. [26] For instance, each object constitutes hierarchy layers of basic image elements In DNN for object recognition; additional layers can take features of lower layers. Due to these features, DNN with fewer units performs better than that of shallow network. Furthermore, NN can get one solution such a word, an action, and a number, while DNN can resolve the problem more globally and predict rely on the information supplied. DNN do not require a significant amount of marked data when solving problem, unlike NN. Figure 8 shows difference between NN and DNN.

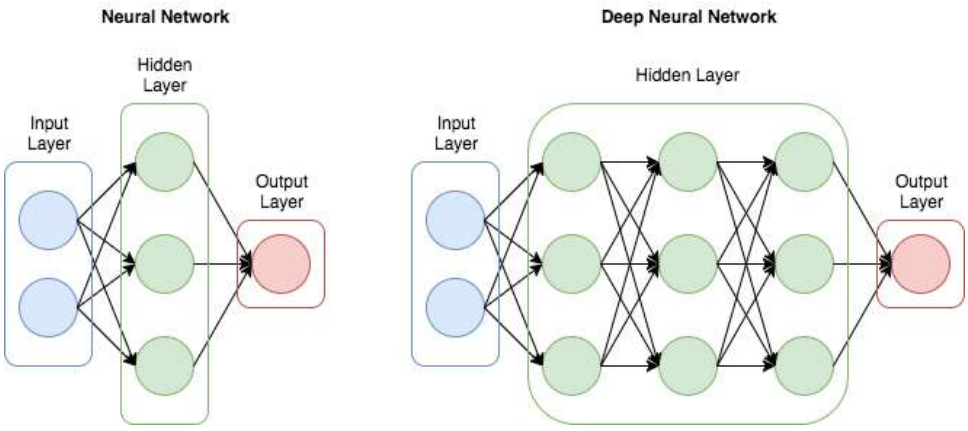



Figure 8: Difference between NN and DNN

### 2.2.1 Mechanism of Deep neural network

Specific DNN mechanism is like that of NN. [27] As shown in Figure 9, a series of vectors ($X_1$, $X_2$, … $X_n$) is entered into input layers. Sequentially, multiplying each data by each weight, their weighted sum is added to bias. The total sum goes through to next neuron layer after passing activation function which determine a neuron feature. Activation function have various method as shown in table 1. NN use sigmoid and hyperbolic tangent mostly, while they are not suitable for DNN. Several hidden layers cannot allow loss correction of weighted sum among layers, which can bring out gradient vanishing; The more hidden layers, the more output error cannot feedback input layers. After driving forward pass once, there are several final processes such as selecting batch of observations, allocation weights among whole node connections randomly, and prediction of the output. And they need to be incorporated, which is called "Backpropagation". The backpropagation is the process that evaluate difference between output and real target and automatically updates all the node connections' weights to try improving loss rate.

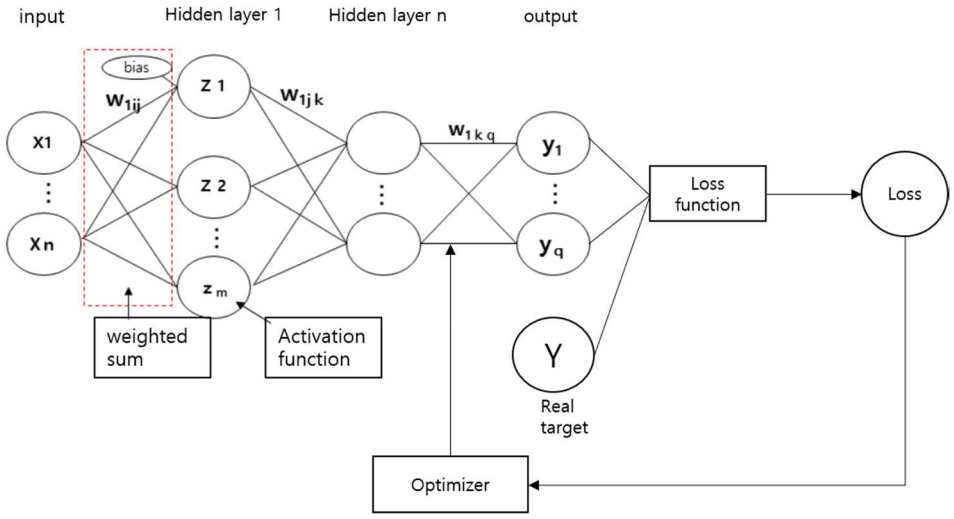



Figure 9: Architecture of DNN

| Activation Function | Formula |
|---|---|
| Step function | $h(u) = f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$ , $u = \sum_{i=1}^{n} \omega_i x_i$ |
| sigmoid | $S(u) = \frac{1}{1 + e^{-u}}$ |
| Hyperbolic tangent, tanh | $\tanh(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}}$ |
| Rectified linear unit, ReLU | $R(u) = \max(u, 0)$ |
| softmax | $p(y^k) = \frac{e^{u_{2k}}}{\sum_{q=1}^{Q}(e^{u_{2k}})}$ |

Table 1: Activation Function

### 2.2.2 Loss function and Optimizer

Loss function and optimizer are crucial techniques of the backpropagation. Loss function is the metric which make DNN learning. In short, error between actual values and predicted values; Learning is finding weight and bias for minimal loss values. The mean squared error (MSE) and categorical cross entropy are selected as loss function, respectively regression problems and classification problems. DNNs can have multiple loss functions nevertheless binary cross entropy is mainly used in medical imaging. [28] Binary cross-entropy is a metric of the dissimilarity between two distributions of probability over a given set of events. Instinctively speaking, it is probabilistic prediction with only two choices (yes or no, A or B, 0 or 1, left or right). Binary cross entropy' formula is described on (1).

$$-\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{c} L_{ic}\log(P_{ic}) \qquad (1)$$

Where, n= number of data, c= number of categories, L=ground truth values (mostly 0 or 1), P=probabilities of ground truth (range from 0 to 1)

Suppose a data set named $D_1$ that n=3, $L_1$= [0, 0, 1], $P_1$= [0.1, 0.2, 0.7]. binary cross entropy of $D_1$ is 0*log(0.1) + 0*log(0.2) + 1*log(0.7) = -log(0.7) = 0.35. If probabilities are changed to [0.5, 0.3, 0.2], binary cross entropy is 0*log(0.5) + 0*log(0.3) + 1*log(0.2)= -log(0.2) = 1.6. Then model with changed probabilities did not predict well because of higher loss. For this example of calculation, we know the method that Model of DNN aim to minimize the loss. Optimizer is aimed to determine update the parameters based on loss function. Specifically describing, optimizer solve plateau problem in cost(loss) (Figure 10); solving local minimum problems is not essential because it is rare case, loss in whole variables should increase, in high dimensional model. There are several optimization algorithms, but Adam has conventionally dominated as optimizer in medical imaging. [29] Computational economy and generalization are valuation basis for selecting optimizer. So, A lot of researchers choose Adam for its advantages in both.

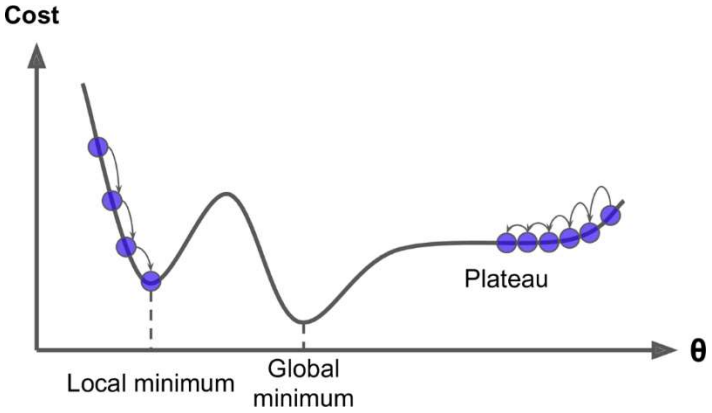


Figure 10

### 2.2.3 Limitations of Deep neural network

DNN have alleviated many problems which cannot be solved by previous methods. While there are some drawbacks in vision tasks. Almost all vision tasks have large number of parameters which can bring out curse of dimensionality. It makes easy to be over-fitted and needs large memory consumption for calculation. Secondly, DNN does not consider information of location which means that Data input go through with one dimension. If images are little shift or distorted, DNN model regards same image as different image. As shown in Figure 4, humans identify six characters "A", however DNN identify six characters whole different classed objects because of absence in location information. So, it should learn more. At last, DNN has fully connected layer structure which means that whole former neurons are connected to whole next neurons. Training time will be longer due to this structure's feature.

Figure 11

## 2.3 Convolution neural network

Convolutional neural network (CNN) is a subcategory of deep neural network; CNN replace matrix multiplication in neural nets with convolution, unlike conventional DNN. Yann Le Cun at el introduced the concept of Convolutional Neural Networks In 1995. [30]; It is influenced by the organ of animal visual cortex [31,32] The advent of large datasets and compute resources allow CNN the mainstream for many computer vision applications such as image segmentation. CNN's is a mathematical construct that several building blocks: convolution layers, pooling layers, and fully connected layers. (Figure 12)

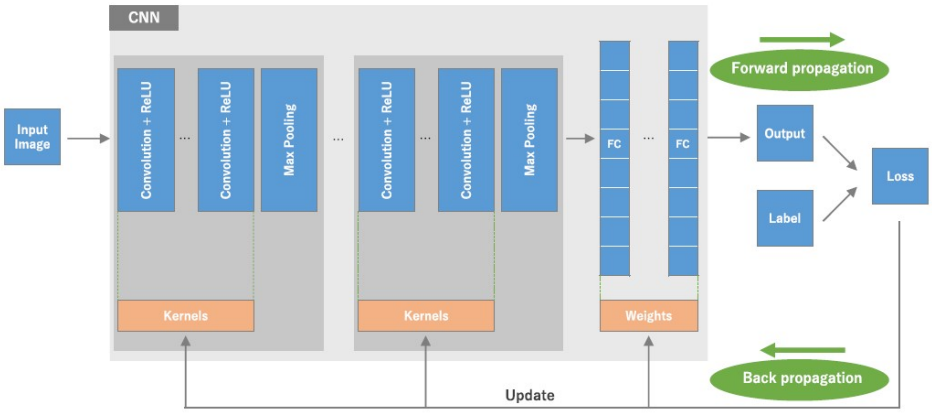
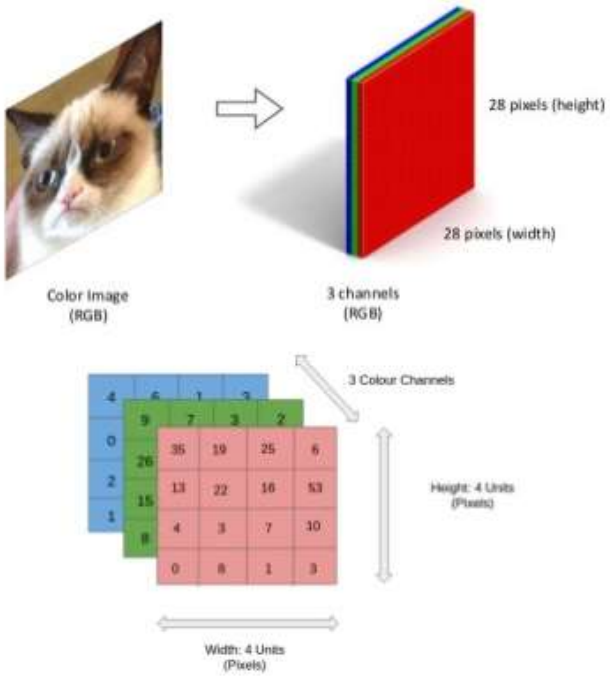



Figure 12: Structure of CNN

### 2.3.1 Mechanism of Convolution neural network

Convolution layer is an essential constituent of the CNN architecture that conducts feature extraction, which commonly compose of a combination of convolution operation and activation function. Convolution is a specialized operation applied on a matrix called a tensor (from image input) using another matrix called a kernel for feature extraction. As shown in Figure 13, image data can be transformed to 3-dimensional tensor of height * width * channel. Kernel or filter is an image that depict a feature. A set of curve picture and its kernel is exemplified. (Figure 14) This can be a sample feature that CNN will recognize.




Figurer 13: Tensor of color image.

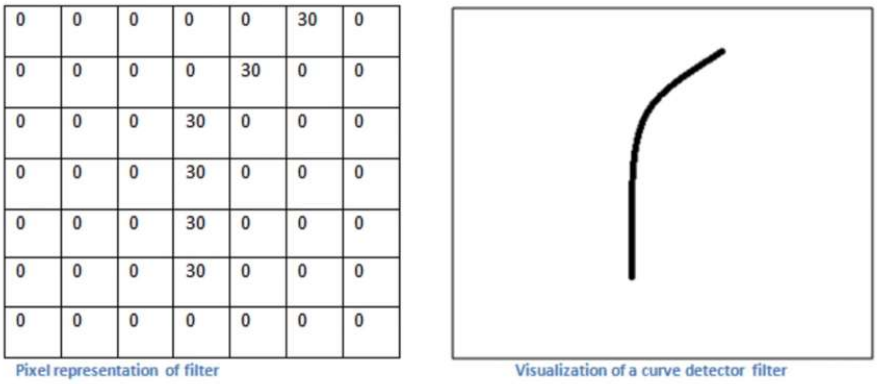



Figure 14: A simple filter depicting a curved line.

The operation of product between each element of the kernel and the input tensor in element-wise manner is multiplying the values of a cell in accordance with a row and column, of the tensor matrix, with the value of the corresponding cell in the kernel matrix. Consequentially, obtained output value is called a feature map. (Figure 15) This procedure is repeated applying multiple kernels, which stands for different property of the input tensors. Stride is the number of columns shifts over the input matrix. Figure 16 shows convolution would work with a stride of one and two, respectively. When convolution run, output image size goes through smaller than size of input in as you see in Figure 16; These can cause loss of information on edge pixels. To alleviate this problem, most CNN model use padding which means that surrounding input image tensor with a certain value. Padding make output size same or little smaller as input size. Surrounding input tensor with zero is called zero-padding. (Figure 17)

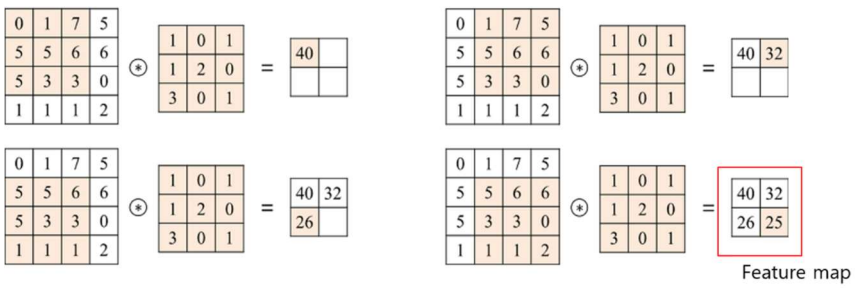



Figure 15: Convolution operation

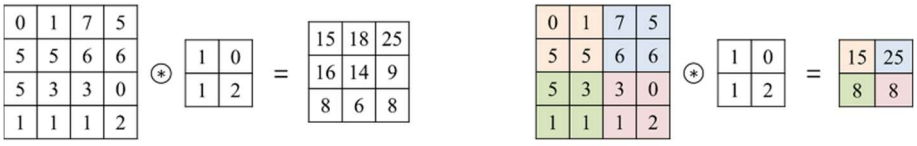


Figure 16: Stride 1 convolution and Stride 2 convolution.

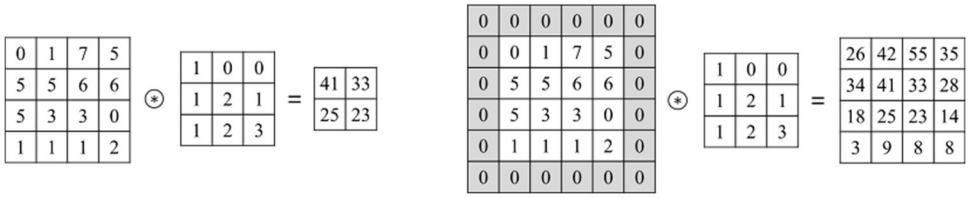



Figure 17: No padding convolution and Zero padding convolution.

Activation function layer is to introduce non-linearity in CNN layer. The purpose of applying activation after CNN is that the real-world data would want CNN to learn would be non-negative linear values. Sigmoid and hyperbolic tangent (tanh) function used to be selected as smooth nonlinear functions because they are mathematical representations of biomimetics in neuron. However, most of the data scientists use ReLU. performance wise ReLU is superior to the other two. Figure 18 shows how apply ReLU to CNN.

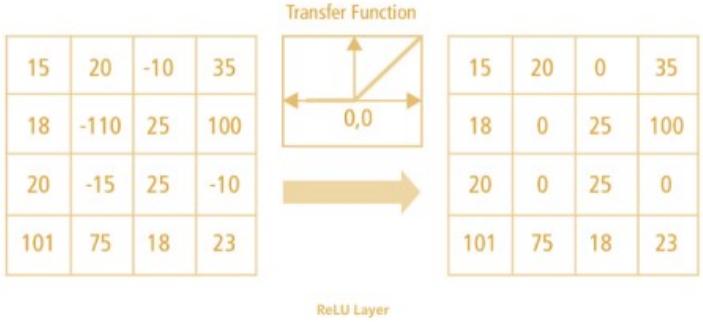



Figure 18: Applying Relu to CNN

The Pooling layer compose of providing the process of extracting a value from a set of values. This reduces the dimensionality of the feature maps to induce a translation invariance to small shifts and deformation and lessen the number of parameters and hence to also control overfitting. Pooling operates by sliding a kernel across the feature map and applying the content of the kernel to a pooling function. There are several pooling methods, but Max pooling is mainstream; it takes the largest element from the sub section of feature map. Figure 19 shows how to operate max pooling. With pooling operation, image goes through smaller, called down sampling. (Figure 20); kernel of convolution can be learnable abstract feature in image.

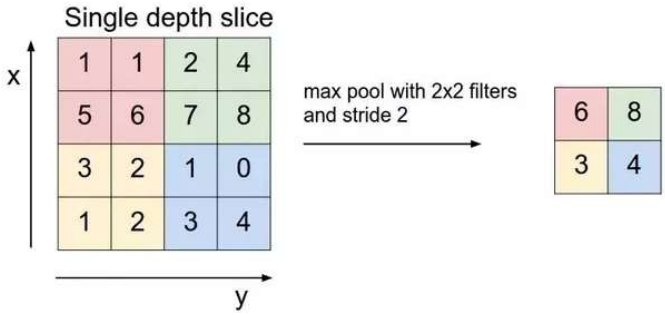



Figure 19: Example of Max pooling.

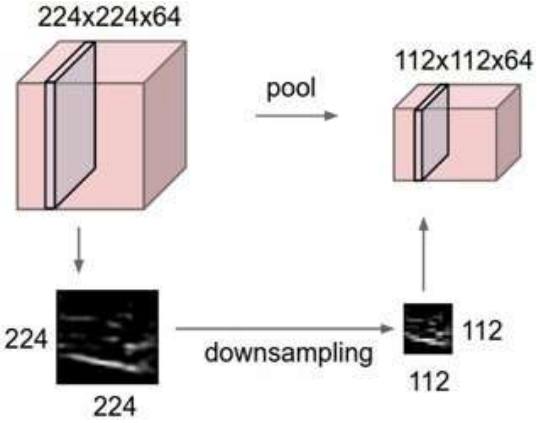



Figure 20: Down sampling image with pooling.

Fully Connected Layer is last step of the CNN structure. This process flattens output of last pooling layer into a one-dimensional array of numbers and linked to one or more fully connected layers, that conducts the work of classification task. Each fully connected layer is followed by a nonlinear function, such as ReLU. The input tensors pass through convolution and pooling layers, keeping the visual information. And feature maps transform Neural network with one hidden layer, classified finally. (Figure 21)

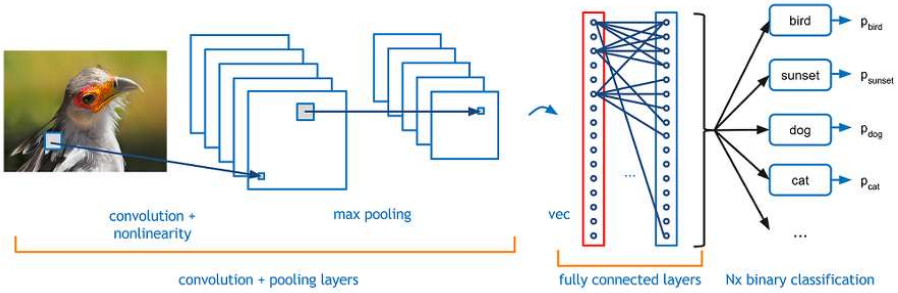



Figure 21

### 2.3.2 Competence in Vision task

CNN is competitive in vision task due to stationarity of statistics and locality of pixel dependencies [33] Stationarity of statistics is the concept that values of the pixels is time invariant. Images have feature that the same pattern repeats regardless of locations. In other words, learnable parameters of an image in certain region can be used extraction of same feature in other regions. In Figure 22, Two people's mouths are located on different regions which can be same pattern with stationarity assumption. When extracting feature of mouth, there are two ways. Considering two mouths as a feature regardless location and extracting two mouths. Taking account of location, extracting two different features (mouth located on left and right). Former is more efficient because of sharing parameters which is same manner as CNN method. (Figure 21)

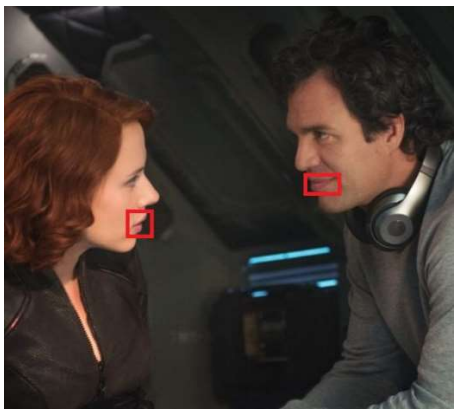


Figure 22

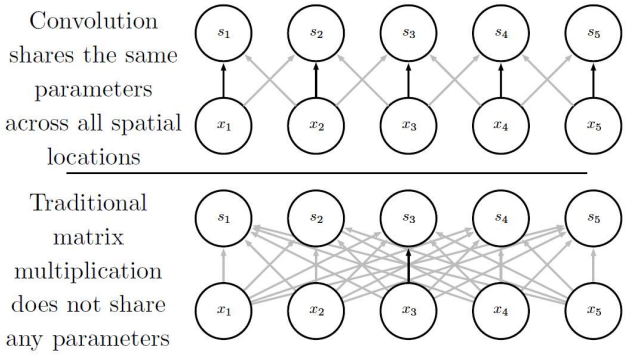



Figure 23

locality of pixel dependencies means that close pixels tend to be correlated to each other, so we should leverage this dependency to operate them together. The size and the shape of the neighborhood could vary, rely on the region of the image. Feature of nose is represented and are correlated to each other in only blue rectangular region's pixels (Figure 24); The pixels in red rectangular have no dependencies to pixels in blue rectangular. This assumption matches that CNN

operate convolution with kernel which has sparse interactions feature. Sparse interaction is concept that each output is connected to a small number of inputs; locality of pixel dependencies exactly matches that.

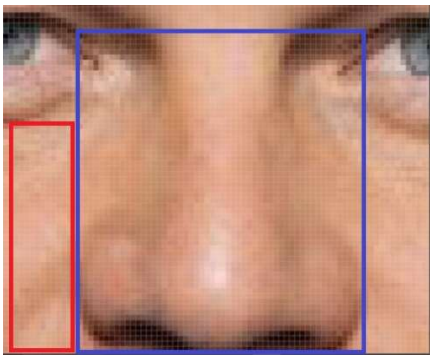


Figure 24

# 3 U-Net

U-Net is CNN-based semantic segmentation approach evolved from FCN [34], which was first designed by Olaf Ronneberger et al in 2015 [35] for better biomedical images segmentation. Ciresan et al devised the network for biomedical image segmentation and won the ISBI 2012, EM segmentation challenge with their own method. However, there were two drawbacks which were found by Olaf Ronneberger et al. At First, each patch runs individually, which cause slowness and waste when it comes to overlapping. Secondly, trade-off between localization accuracy and use of context. U-Net is dedicated to solving these problems, using its architecture and peculiarities. In other words, U-Net classify on every pixel which leads to localisation and distinguishing borders; sharing same size of input and output.

## 3.1 Architecture of U-Net

U-Net's name is derived from its architecture, which is like the letter U, as shown in Figure 25. The Architecture composed of contracting path and expansive path. Contracting path is aimed at catching the context of image which is on left side. Specifically describing, each contracting step repeat 3 x 3 convolution twice including ReLU operation but there is no padding which bring out loss of feature map (blue arrows). Also, each contracting step, 2x2 max pooling which stride is two is operated (red arrows). This process makes feature map half. Whenever down sampling, feature map channel increases twice. Expansive path expands the feature map which I on right side. Each expanding step operate 2x2 up-convolution (green arrows). This process makes feature map double. Also, each expanding step, repeat 3 x 3 convolution twice including ReLU operation but there is no padding which bring out loss of feature map (blue arrows); it is same as contacting path. Whenever up sampling, feature map channel increases half. On last layer, 1 x 1 convolution is operated for the prediction of non-linear. When down sampling and up sampling, sophisticated pixel's information lose. This is serious for segmentation which needs dense prediction. Skip connection (gray arrows) sends significant information from contracting path to expansive path, which leads to more accurate prediction. Segmentation map's size, the final output, is smaller than input image size for no padding in convolution.

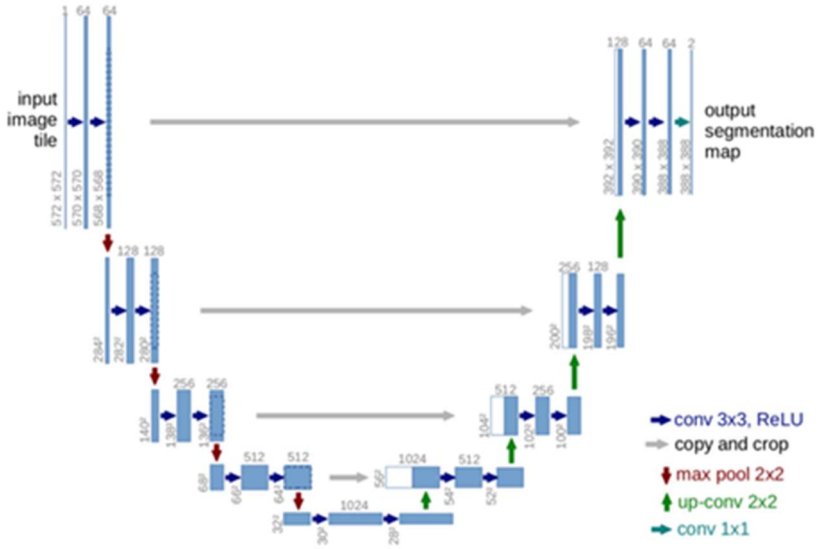



Figure 25: Architecture of U-Net

## 3.2 Strategies of U-Net

Overlap-tile strategy means that the pixels in the border region are mirrored around the image so that images can be segmented continuously. Prediction of the segmentation in the yellow region, needs image data within the blue region as input. (Figure 26) When it comes to large image data sets, the padding technique is essential for applying the U-Net model; Unless, the lack of the GPU memory make the resolution. Conventional CNN typically used sliding window approach, while intensely redundant computations are carried out, using sliding windows. [36] Sliding window process a patch around every pixel which leads to overlap where many computations are performed redundantly. In contrast, U-net's method omit the region, which is already operated, and start operates new patch. This decrease redundant computations. (Figure 27)

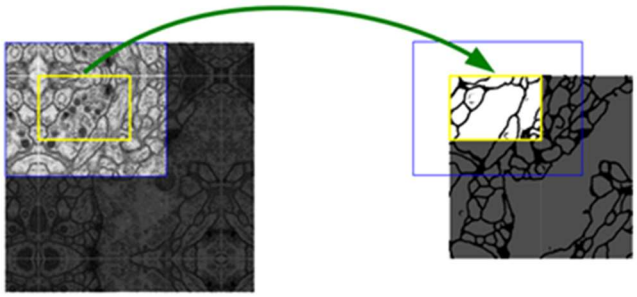


Figure 26: Overlap-tile strategy

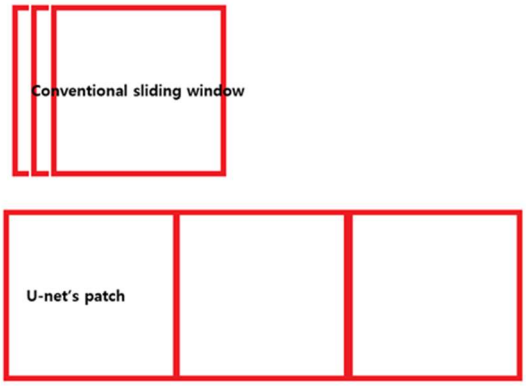



Figure 27

### 3.3 Advantages of U-Net

U-Net is faster than conventional segmentation models. Its patch overlapping percentage is low which leads to low redundancies of computation. Also, U-Net does not have trade-off between context and localization. Segmentation Network should conduct context and localization for classification at once. Each performance depends on size of patch, which is related to trade-off. When patch bigger, it can recognize larger regions at once. This is beneficial for context, while detrimental to localization performance during a lot of max pooling. In contrast, when patch smaller, localization performance gets better; recognition areas gets too smaller make bad context performance. The U-Net' skip connection sends location information from contracting path to expanding path. This process can make combining localization and context which can avoid trade off context and localization. When it comes to training data insufficiency, data augmentation method be useful. It is rotating and reversing data for increasing data. This method is not mandatory and not always advantageous. However, data augmentation effects positively in cell data, Olaf Ronneberger et al applied the method; it resembles elastic deformation of tissue organ.

## 4 Loss Functions for Medical Image Segmentation

Selecting loss function method is crucial for medical image segmentation tasks. Although a architecture is fair, selecting bad loss function can bring out bad results. It is commonly adopted that cross entropy is loss function in CNN based segmentation methods. [37] However, selecting cross entropy can bring out a bottleneck to obtain high precision for sophisticated model. For example, there are prevalent disconnection among long, thin, and weak vessels or missing under the supervision of a cross entropy in the retinal vessel segmentation task. Therefore, many researchers have devised for this problem in recent several years. Here are taxonomy and introduction of loss function for medical image segmentation.

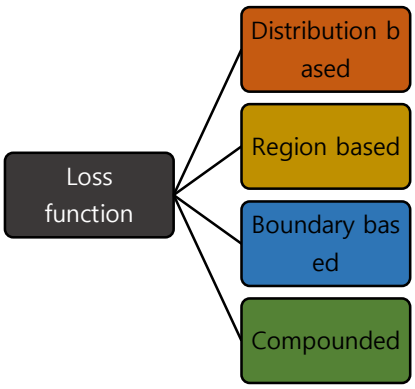



Figure 28

Figure 28 shows that main taxonomy of loss function in medical image segmentation.

## 4.1 Distribution based loss function

Distribution based loss are dissimilarity metric among distribution. All methods are derived from cross entropy. Cross entropy (CE) is common and works well in equal data distribution. It is explained well in 2.2.1. Weighted cross entropy (WCE) is a development of CE, which assign different weight to each class. Skewed dataset matches this method. Balanced Cross-Entropy resembles WCE, but it allows negative weight. Distance map derived loss penalty term is WCE with distance map from ground truth mask. It is used for guiding the network' focus on hard-to-segment boundaries. Lin et al have devised the focal loss for Facebook AI Research in 2017. [38] The Focal Loss is variation of CE, which weighs down the contribution of easy samples, but make model concentrate on hard samples. Sample brain lesion segmentation CT scan image can be exemplified. (Figure 29) (a) is training image, (b) is the segmentation mask; a small number of pixels are in white area (targeted lesion) but amount number of pixels are in black pixels.

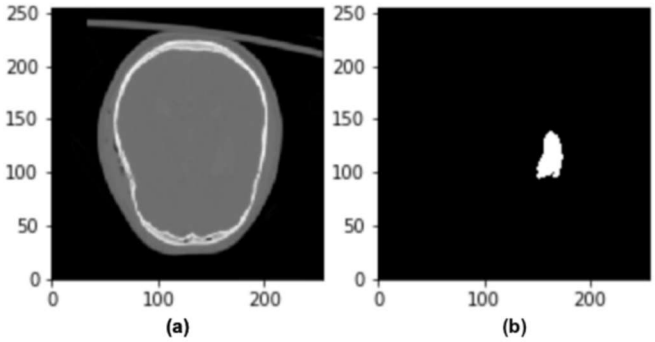



Figure 29

CE is cross entropy

$$CE = \begin{cases} -\log(p), & if\ y = 1 \\ -\log(1-p), & otherwise \end{cases} \quad (2)$$

For notational convenience, focal loss defines the estimated probability of class as:

$$p_t = \begin{cases} p, & if\ y = 1 \\ 1-p, & otherwise \end{cases} \quad (3)$$

Now Cross-Entropy can be rewrite, CE (p, y) = CE ($p_t$) = -log ($p_t$)

So, formula of focal loss is

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \quad (4)$$

$\alpha$ generally range from [0,1] and $\gamma$>0.

## 4.2 Region based loss function

Minimalizing the mismatch or maximizing the overlap regions between labels and prediction is approach of region-based loss functions. Sensitivity-specificity Loss is the weighted MSE of sensitivity and specificity. It aims to focus on true positives. Dice Loss (DL) directly optimize the dice coefficient, which is the non-convex in nature, modified to manage easily.[39] it is commonly used metric in computer segmentation tasks. Formula is below.

$$DL(y,\hat{p}) = 1 - \frac{2y\hat{p}+1}{y+\hat{p}+1} \quad (5)$$

the function can be undefined in edge case scenarios such as when

$$y = \hat{p} = 0 \quad (6)$$

So, 1 is added in numerator and denominator to avoid above problem.

Tversky Loss is variant of DL which applies different weights to false negative and false positive. [40]

$$L_T = T(\alpha,\beta) = \frac{\sum_{i=1}^{N}\sum_{c=1}^{C} g_i^c s_i^c}{\sum_{i=1}^{N}\sum_{c=1}^{C} g_i^c s_i^c + \alpha\sum_{i=1}^{N}\sum_{c=1}^{C}(1-g_i^c)s_i^c + \beta\sum_{i=1}^{N}\sum_{c=1}^{C} g_i^c(1-s_i^c)} \quad (7)$$

When $\alpha$=$\beta$=0.5, this loss function is like equivalent to dice loss.

The Intersection over Union (IoU) loss is derived from dice loss. [41] It operates intersection of two different set by union them. Formula is below.

$$L_{IoU} = 1 - \frac{\sum_{i=1}^{N}\sum_{c=1}^{C} g_i^c s_i^c}{\sum_{i=1}^{N}\sum_{c=1}^{C}(g_i^c + s_i^c - g_i^c s_i^c)} \quad (8)$$

## 4.3 Boundary based loss function

Boundary-based loss is approach to minimize the distance between ground truth and predicted segmentation. This method trains in more robust manner. Hausdorff distance loss is estimating Hausdorff distance from the CNN output probability for minimizing it. [42] Hausdorff distance is measure the largest distances from a point in one set to the nearest point in the other set. Boundary loss aims to compute the distance between two boundaries: not regions but contours. This approach is beneficial to mitigate regional loss in context of highly unbalanced segmentation.

## 4.4 Compounded loss function

Compounded loss is combination of different loss functions. Exponential logarithmic loss (ELL) or DiceCE [43] is sum of DL and CE, which match cases of less accurate prediction.

$$L_{ell} = \omega_{DL} E\left[-(\log Dice_c)^{\gamma^{DL}}\right] + \omega_{CE} E\left[\omega_c (\log(s_i^c)^{\gamma^{CE}}\right] \quad (9)$$

$$where\ Dice_c = \frac{2\sum_{i=1}^{N} g_i^c + s_i^c}{\sum_{i=1}^{N}(g_i^c + s_i^c) + \epsilon} \quad (10)$$

# 5 Experiment

## 5.1 Aim

I conducted this experimentation that aim to highlight U-Net's excellence in medical image segmentation and find optimal loss function for U-Net.

Criteria is loss rate and calculating time. Graphical check is added.

## 5.2 Knowledge acquisition

- Choice of Programming Language: Python was the language of choice. Tool is jupyter notebook because it is intuitive and used in colab.
- Choice of Packages: Architecture package is Pytorch. Other utility packages are matplotlib, pillow, os, time

## 5.3 Dataset

ISBI 2012

### 5.3.1 The background of dataset

"An Integrated Micro and Macro architectural Analysis of the Drosophila Brain by Computer-Assisted Serial Section Electron Microscopy" [44]

Synaptic contacts can be observed using only high-resolution electron microscopy (EM) because of their small size which means that complete series of ultrathin sections are mandatory to reconstruct neuronal micro circuitry (the connectivity at the level of individual neuronal processes and synapses) . Due to the amount of data size (15,000 sections per millimeter of tissue), computer-assisted method is required to acquire and analyze of EM sections. Cardona, Albert, et al demonstrate the utility of the software package TrakEM2. The purpose is modeling nerve fibers interconnections from consecutive EM sections and efficient reconstructing the neural networks which is encountered in different parts of the fruit fly Drosophila melanogaster's early larval brain. With TrakEM2, neuronal networks are made up patterns of axons and dendrites (extended neurons that transmit and receive signals, respectively), describing the most common motifs. Cardona, Albert, et al showed a comprehensive anatomical reconstruction and delivered micro circuitry comparisons between vertebrate and insect brains in neuronal microcircuits.

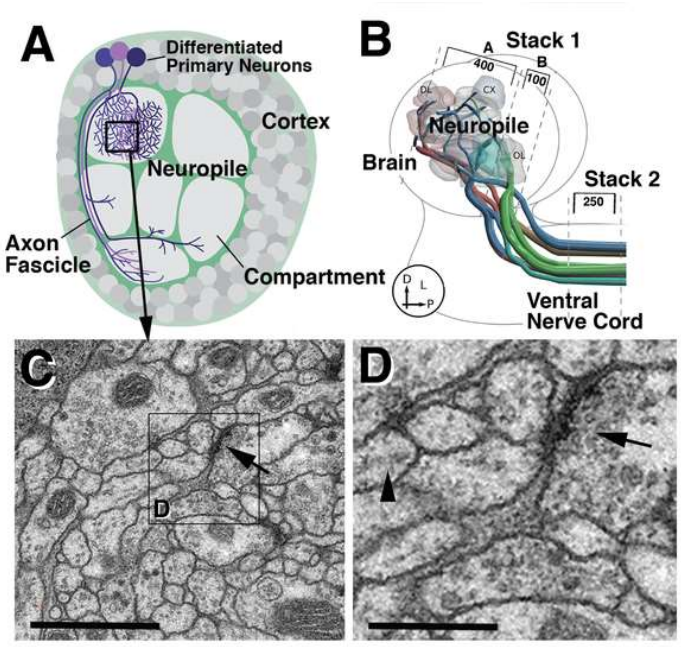



Figure 30

(A) Diagrammatic description of cross-section of one brain half sphere, showing outer cortex of neuronal cell bodies in central neuropile, built by neuronal processes (neurites). One lineage of primary neurons is highlighted in purple color. Processes of glial cells (green) enclose the cortex and neuropile and build boundaries around

sections within the neuropile.

(B) Diagrammatic of larval brain and ventral nerve cord, indicating positioning and orientation of serial sections. The first stack contains the neuropile of one brain hemisphere in two closely adjacent sets of 400 and 100 sections, respectively. The second stack includes 250 sections of the ventral nerve cord (corresponding to approximately two consecutive neuromeres). Colored lines represent axon tracts connecting brain and ventral nerve cord (after 69).

(C, D) Electron micrographs of sections of neuropile illustrating resolution that can be achieved at 5,000× primary magnification. Arrow in (C) points at synapse; arrow in (D) indicates presynaptic vesicles; arrowhead shows obliquely sectioned microtubule. At a resolution of 4 nm per pixel, these structures can be clearly resolved. Scale bars: 1 μm (C); 350 nm (D).

### 5.3.2 Dataset Image Acquisition

There are two datasets, training, and test. Both were a set of 30 grayscale successive images (512 × 512 pixels) from a serial section Transmission Electron Microscopy (ssTEM) which involves the Drosophila first instar larva ventral nerve cord [44].

Training data: The imaged volume measures 2 × 2 × 1.5 μ, accompanied by a resolution of 4 × 4 × 50 nm/pixel. With the object driving a FEI electron microscope, the images were captured using Leginon [45]; it was equipped with a Tietz camera and a goniometer-powered mobile grid stage, accompanied by an enlargement of 5600× binned at 2, which generates the 4 × 4 nm per pixel resolution. The imaging is too anisotropic; the x- and y-directions deliver a high resolution, but the z-direction has a low resolution because it is limited by physical sectioning of the tissue block. Electron microscopy generates the images as a projection of the whole part, so some of the membranes that are diagonal to the cutting plane can occur blurred.

AC created the ground truth segmentation maps. This aimed to the training images which are segmented each neurite of the training volume. This process conducted marking its borders on each 2D plane in handmade. The training dataset was made publicly available, so that participants in the challenge could use it for developing algorithms.

Test data: IA and DB independently segmented from the whole test volume to the ground truth segmentation maps for the test images. AC and IA manually depicted the neurite boundaries using the open-source software TrakEM2, while DB used the freely available software VAST3. After the two test boundary maps had agreement, the final test labels were completed. With that purpose, the labels from IA got visually check and compared with the labels of DB. A disagreement (it happened object splitting or merging) was found at any moment, a manual rectification was conducted to guarantee the 3D object continuity. Only the grayscale images were generated publicly available from the test dataset. The ground truth segmentation maps of the test images were kept confidential and only a secret part of them were used to calculate the public test score (Figure 30D). The participants

submitted predicted segmentation maps for the test images. The organizers evaluated the predicted segmentation maps by contrasting them to the withheld ground truth.

Results format

The results are expected to be submitted as a 32-bit TIFF 3D image, which values between 0 (it is absolutely membrane) and 1 (it is not absolutely membrane).

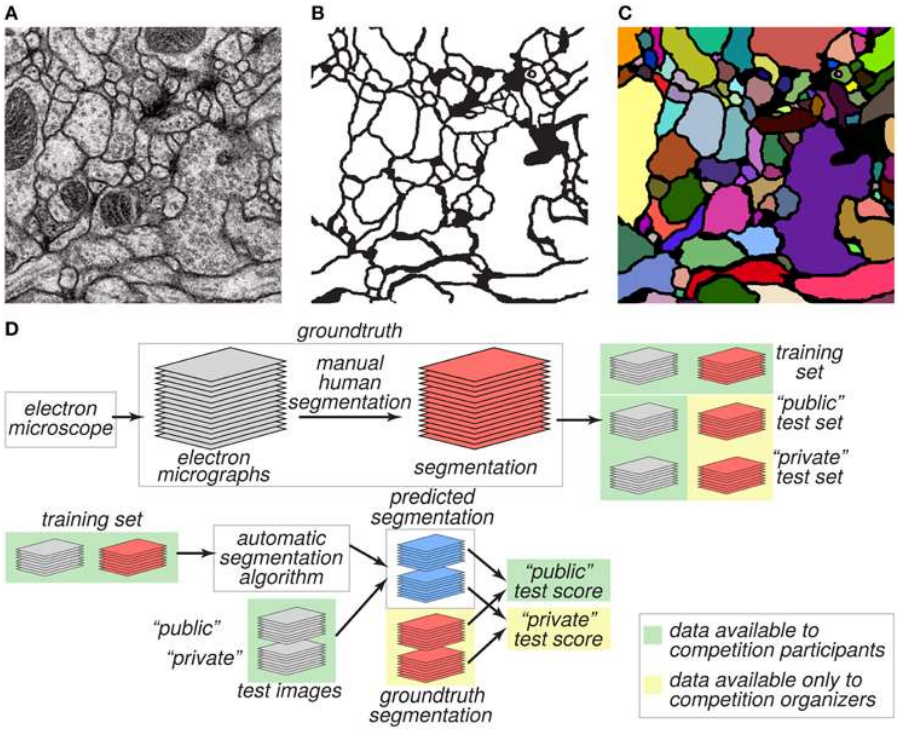



Figure 31

(A) EM image of a larval Drosophila's ventral nerve cord.

(B) Boundary map annotated by human experts.

(C) Segmentation into neurite cross-sections.

(D) The annotated dataset was split into training and test sets and distributed publicly. Ground truth labels for the test set were withheld and used to evaluate the predictive performance of candidate algorithms.

AC: Albert Cardona. Howard Hughes Medical Institute, Janelia Research Campus, Ashburn, VA, USA

IA: Ignacio Arganda-Carreras: UMR1318 French National Institute for Agricultural Research-AgroParisTech, French National Institute for Agricultural Research Centre de Versailles-Grignon, Institut Jean-Pierre Bourgin, Versailles, France

DB: Daniel R. Berger: 2eCenter for Brain Science, Harvard University, Cambridge, MA, USA

## 5.4 Method

There are two classes membrane and non-membrane. U-Net calculates the probability whether membrane or not in every pixel. At first, the classifier is trained using the provided training images and validation set. Consequently, the classifier is applied to test set, generating a map of membrane probabilities. Finally, it segments test images. Data is loaded with custom data loader, using torch. Also, networks are computed with torch, GPU, and tensor. Custom normalization is used in transforming data to tensor. Training epoch is 100. Optimization function is Adam. Saved checkpoint

### 5.4.1 Data augmentation

Training data is very little, so I applied data augmentation. Data augmentation methods are various such as resizing, scaling, translating, rotating, and noising. But I used image cropping and flipped the cropped images randomly. Tissues in biomedical images commonly are deformed, and realistic deformations can be efficiently simulated. By the way, it is found that data augmentation is very similar to elastic deformation. [46] In this manner, the training is more powerful with increasing size of dataset, which can be elastic deformation.

### 5.4.2 Architecture

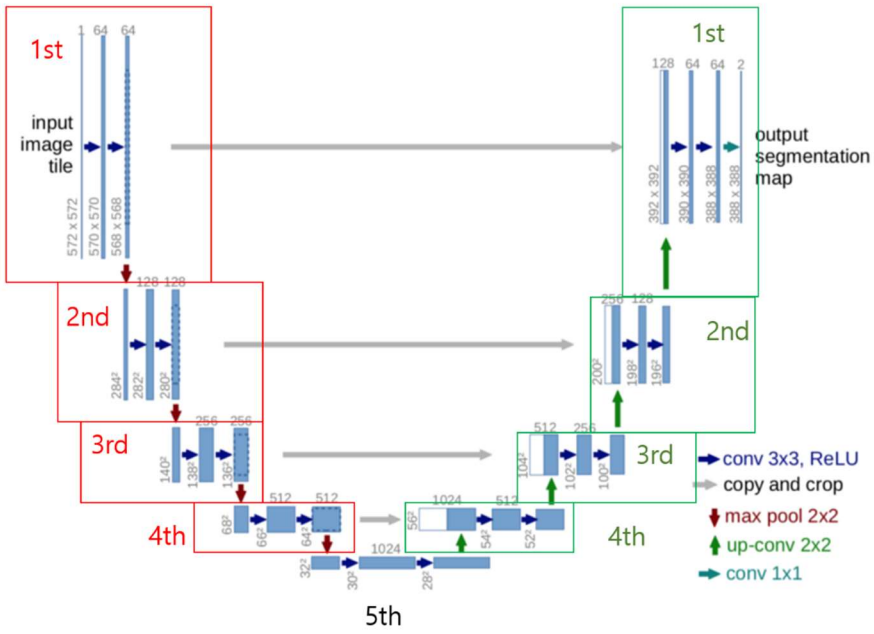



Figure 32: explanation of architecture.

Firstly, I constructed a succession of convolution, batch normalization and ReLu for blue arrows process on figure 1. I call this CBR2d. Contracting path consists of red blocks. Each red block composed of CBR2d- CBR2d- MaxPool2d(kernel_size=2). 5Th stage is just CBR2d- CBR2d. Expanding path consists of green blocks. Each green block is named corresponding to red blocks name because of concatenation process, which is ConvTranspose2d(in_channels, out_channels, kernel_size=2, stride=2, padding=0, bias=True)- CBR2d- CBR2d. 1st green block is exceptional, adding Conv2d(in_channels=64, out_channels=1, kernel_size=1, stride=1, padding=0, bias=True) at last. Gray arrows are torch.cat with dimension one, which are connecting from red block to green block, respectively.

### 5.4.3 Loss function

I used six loss functions for compare each other. First, I used BCEWITHLOGITSLOSS (binary cross entropy plus a sigmoid layer), which is built-in loss function, with no parameters. Other five loss functions are custom loss functions. Dice loss, Dice BCE loss (combination of Dice loss and BCE), IoU loss, Focal loss and Tversky loss. Focal loss is required to input parameters. Gamma is 2 which is default, Alpha is 0.8 is approximate of optimal value in other experiment. [47] Tversky loss is also require inputting parameters. Alpha=0.3 and beta=0.7 shows best result in other experiment. [48] Dice BCE loss(Exponential logarithmic loss) could tune parameter, but I followed the way that gamma of dice is same as gamma for simplicity.

## 5.5 Results

There are results: time to run train and validation, loss rates graph of train and validation, loss rates of each sample test segmentation, visualizations of each sample test segmentation (input, output, label). They are conducted in six models.

- U-Net with BCE loss
- U-Net with Dice loss
- U-Net with Dice BCE loss
- U-Net with Focal loss
- U-Net with IoU loss
- U-Net with Tversky loss

| Model | Time (Second) |
|---|---|
| U-Net with BCE loss | 919.3693518638611 |
| U-Net with Dice loss | 891.8287534713745 |
| U-Net with Dice BCE loss | 964.0130093097687 |
| U-Net with Focal loss | 961.0538156032562 |
| U-Net with IoU loss | 886.8247578144073 |

| | |
|---|---|
| U-Net with Tversky loss | 962.9740669727325 |

Table 2: Running time of each model

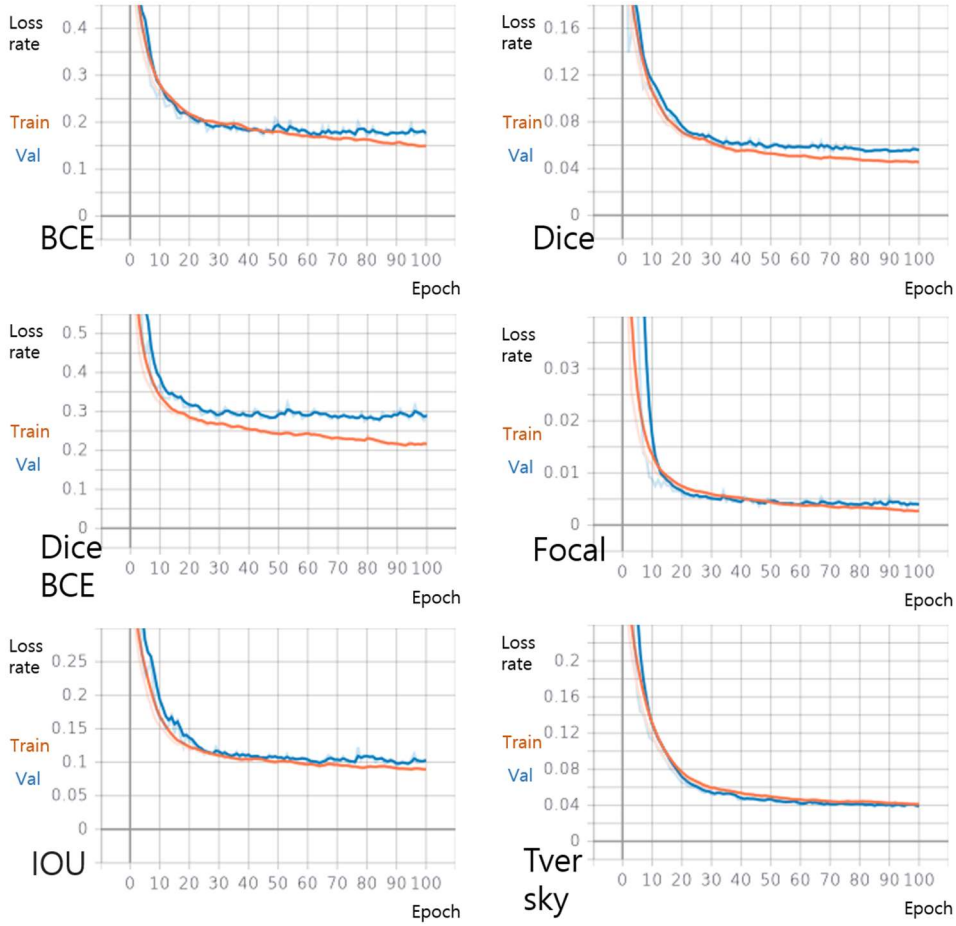



Figure 33: Train and validation loss rate for each model

| Model | Loss rate |
|---|---|
| U-Net with BCE loss | 0.2156 |
| U-Net with Dice loss | 0.0539 |
| U-Net with Dice BCE loss | 0.2476 |
| U-Net with Focal loss | 0.0045 |
| U-Net with IoU loss | 0.1007 |
| U-Net with Tversky loss | 0.0612 |

Table 3: loss rates of a test sample in each model

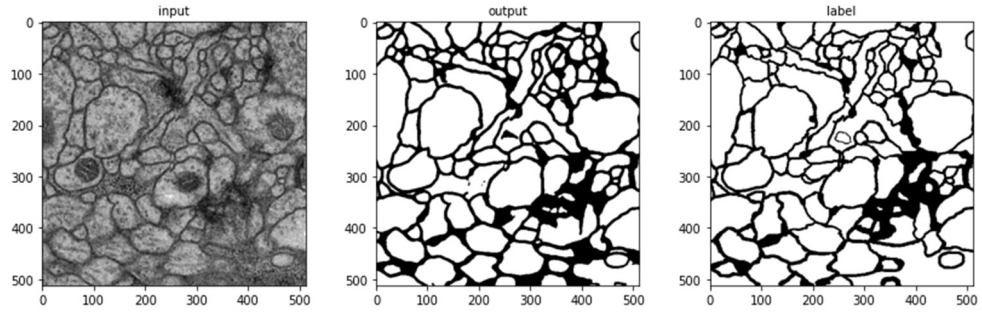



Figure 34: visualization of sample test in BCE loss

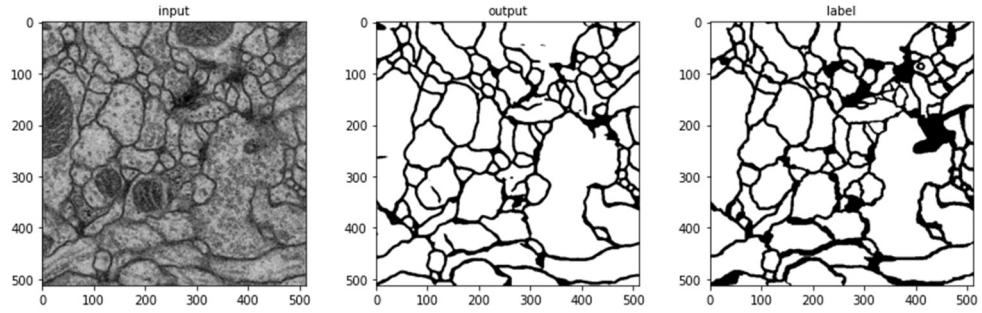



Figure 35: visualization of sample test in Dice loss

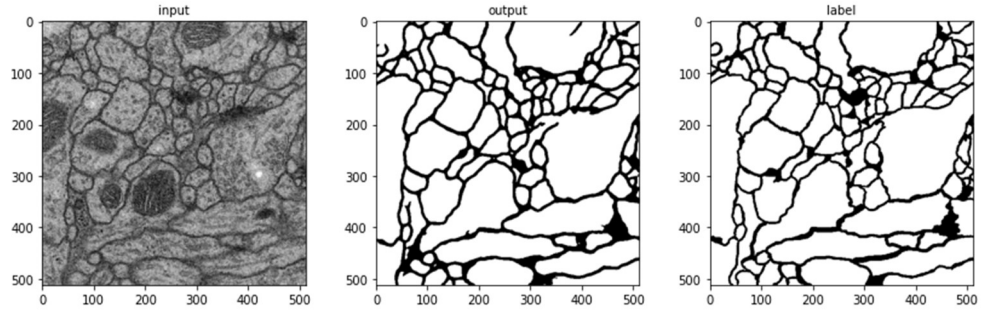



Figure 36: visualization of sample test in Dice BCE loss

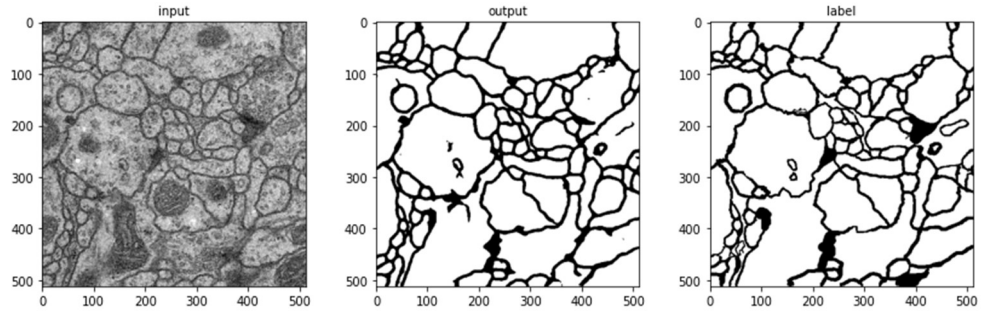



Figure 37: visualization of sample test in Focal loss

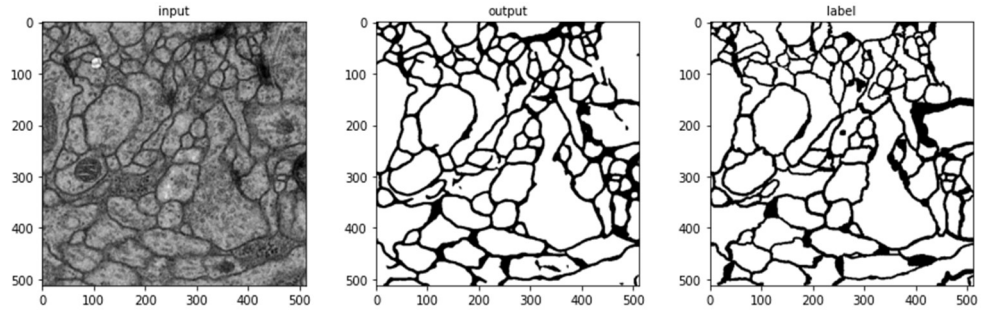



Figure 38: visualization of sample test in IoU loss

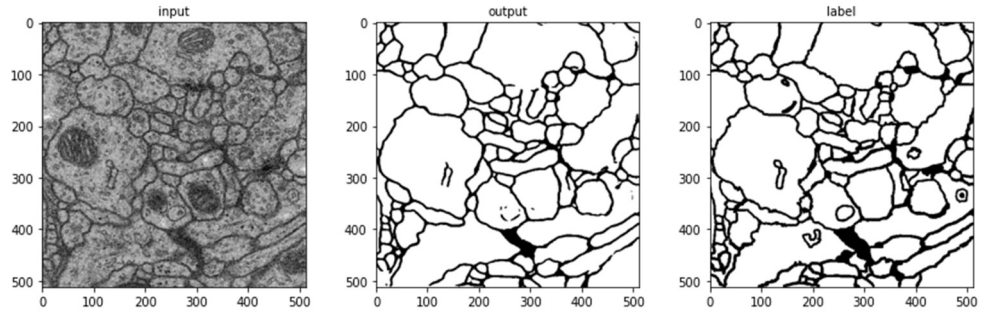



Figure 39: visualization of sample test in Tversky loss

## 5.6 Conclusions

When it comes to time, it did not take to run over 16 minutes in all models; the fastest model is eight percent faster than the slowest model. However, U-Net with IoU loss is the fastest model among them. If dataset is much larger than ISBI 2012, using IoU will show significant difference in running time.

U-Net with BCE loss and U-Net with Dice BCE are not compatible on the face of the loss rate. Their loss rate in train and validation converged to around 20 percent, also a test sample did so. Although U-Net with IoU loss is better than them, its loss rate went near 10 percent during hundred epochs, which is not good. U-Net with Dice loss and U-Net with Tversky are fair among six models. U-Net with Focal loss is the best, which converged under 0.5 percent.

As graphical qualitative analysis, U-Net generate good segmentations image from inputs. Output images resemble label images; locations and sizes of membrane are almost same except several ones. As you see Figure 40, a segmentation image, using Ciresan's method has some blurred regions. This is an evidence that U-Net is better than previous model; Ciresan's method used ISBI 2012 like U-Net.

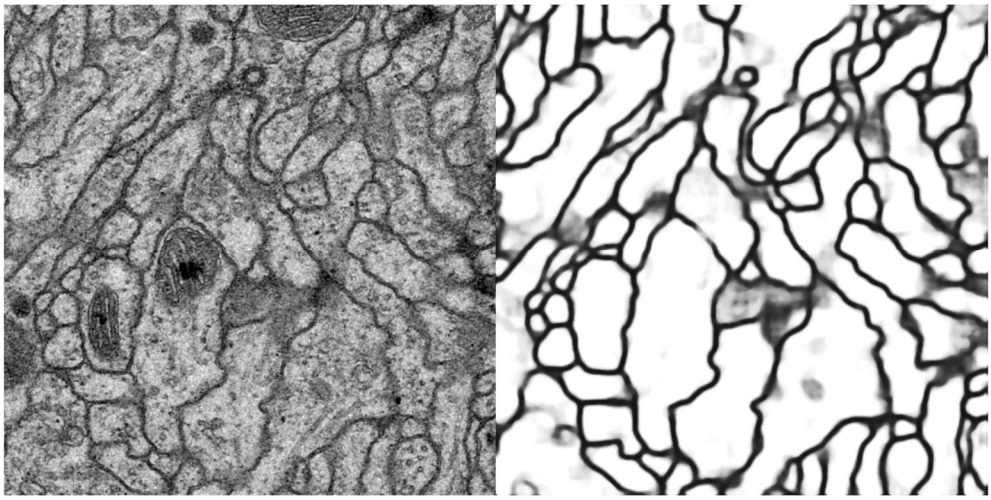


Figure 40: Ciresan's segmentation method. [49] left is input image, and right is output image.

I found the difference between outputs and labels with visualization check. Difference patterns are not same among whole models. BCE loss model tends to generate membrane thicker, which makes non membrane regions small. Dice loss model make redundant short slim membranes which cannot close curve. Also, small closed curves are not completed in dense small closed curves area. Dice BCE loss model prone to uncomplete lean middle size closed curves. Focal loss model is little different form former models, there are uncompleted lines in relatively larger closed membranes. U-Net with IoU loss and U-Net with Tversky loss resemble Dice loss model. Medical image segmentations are used in the number of different parts. Different usage can vary importance of feature in segmentation images, so it is hard to select optimal loss function with graphical qualitative analysing.

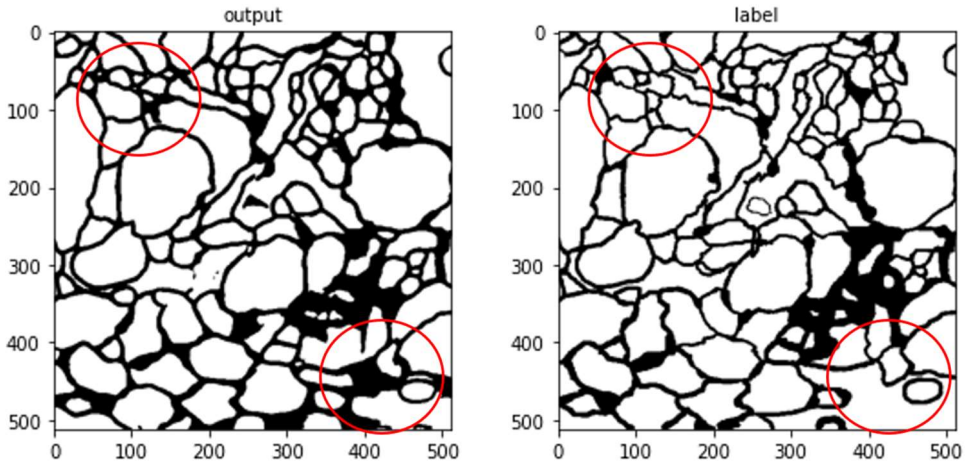



Figure 41: difference check in BCE loss model

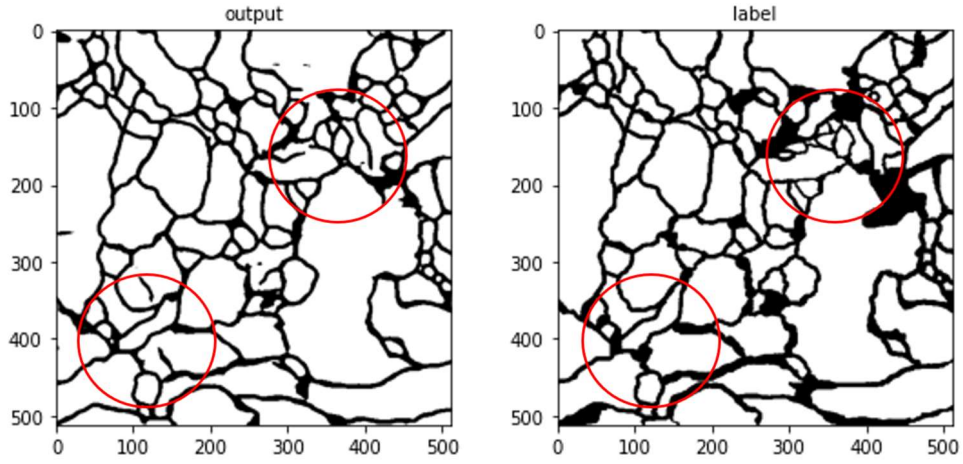



Figure 42: difference check in Dice loss model

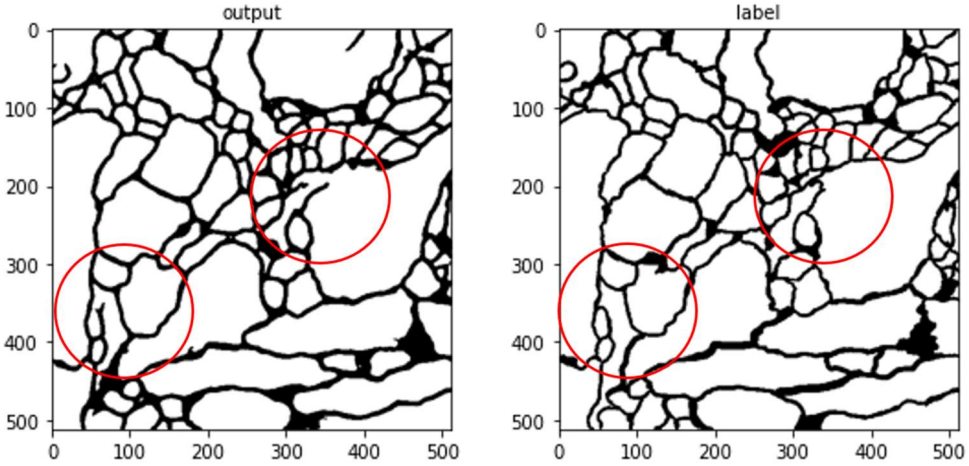



Figure 43: difference check in Dice BCE loss model

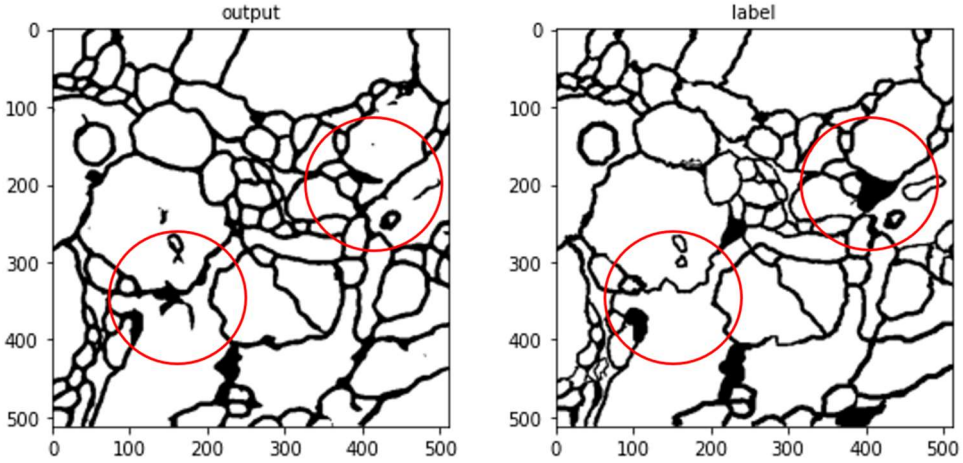


Figure 44: difference check in Focal loss model

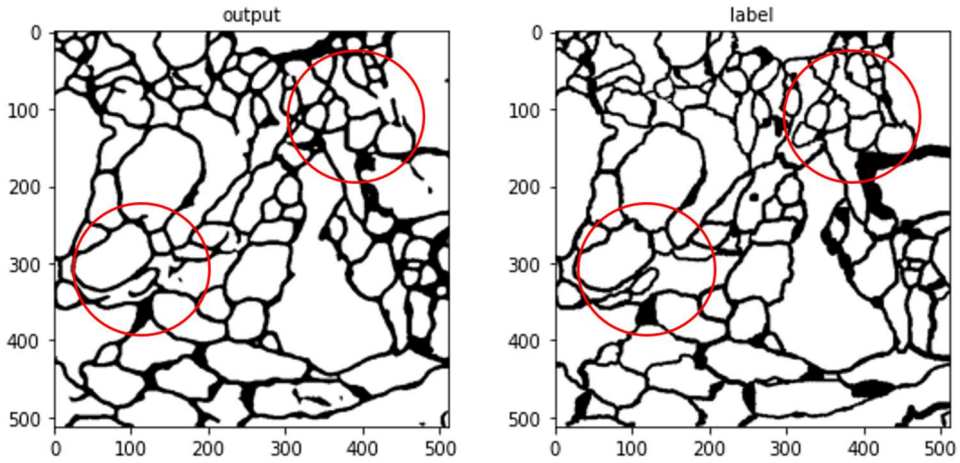



Figure 45: difference check in IoU loss model

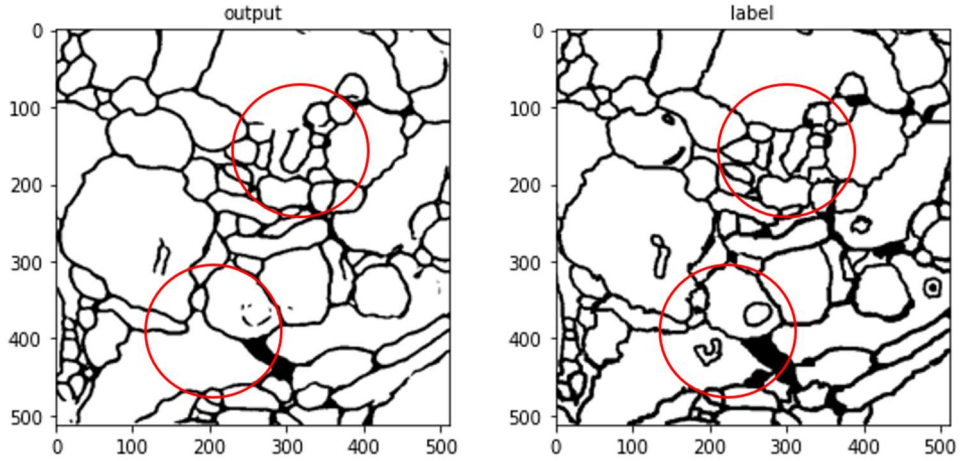



Figure 46: difference check in Tversky loss model

In brief, IoU loss should be selected if time is important. Focal loss should be selected if loss rate is important. And U-Net is superior to previous model such as Ciresan's method.

# 6 Possible Extension: Ciresan's deep neural networks segmentation

Olaf Ronneberger et al have devised their model to compare Ciresan's model. [49] I thought it could be more explainable if I study model of Ciresan. So, I researched the model.

## 6.1 Multi-column Deep Neural Networks

Multi-column deep neural networks (MCDNN) is deep neural network which has independent working columns [34]; Since they tend to have large vector adders that can do many operations at once, MCDNN can alleviate problems that vanilla DNN have in error rates and computational efficiency.

A single DNN is built up of a series of convolutional, max-pooling and 2-dimensional fully connected layers with shared weights as it can be seen Figure 47. Each of DNN is randomly initialized, and the input data may be pre-processed in different ways for each column. But each column shares the same network configuration and training data. Ciresan et al applied winner-take-all method on each layer to train only the winner neurons, but it allows the other neurons not to forget what their have learned. If it is given some input pattern, winning neurons are determined by segregating layers into quadratic sections of local inhibition, choosing the most energetic neuron of each section with simple max pooling technique. Some layer's winners go through a smaller, down-sampled layer with lower resolution, and it feeds the next hierarchy.

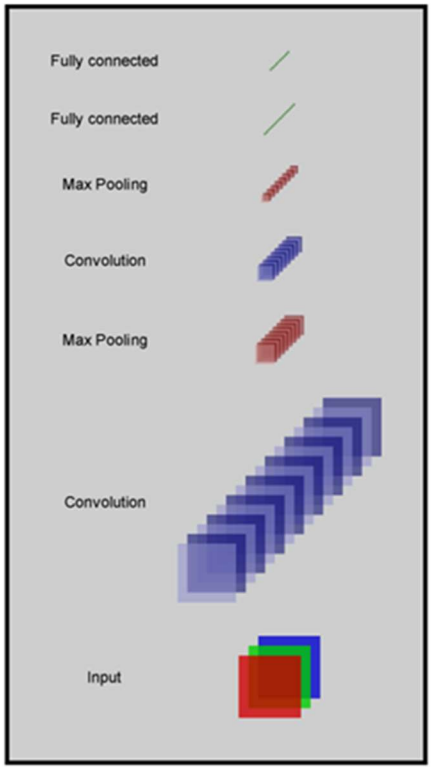



Figure 47: the structure of each DNN

Down sampling process goes through the first 1-dimensional layer and repeat same process. Sequentially, only winning region of 1-dimensional layer leads to the top part of the hierarchy, as called a standard multi-layer perceptron (MLP). The size of receptive fields and winning regions of DNN often are minimal, which

bring out maximal depth of layers with 2-dimensional winning regions. In fact, insisting on minimal 2x2 fields automatically interprets the entire deep architecture, aside from various convolutional kernels per layer and the depth of the plain MLP on top. Despite of more peripheral layers' weight changes affection, as mentioned above, only winner neurons are trained which means that other neurons cannot lose what they have learned. The resulting shrink of synaptic periodical changes exact match biologically plausible reduction of energy consumption. Training algorithm is based on online, i.e. after every gradient calculation phase, weight is updated.

At the last, several DNN columns are combined into MCDNN. In this Multi-column predictions from each column are averaged. Before training, weights of each column are randomly initialized; the columns can be trained on the same inputs, which can be pre-processed indifferent ways. Formula 11 show the method.

$$y^i_{MCDNN} = \sum_{j}^{Columns} y^i_{DNN_j} \quad (11)$$

where i corresponds to the ith class and j runs over all DNN

The MCDNN architecture is illustrated in Figure 48.

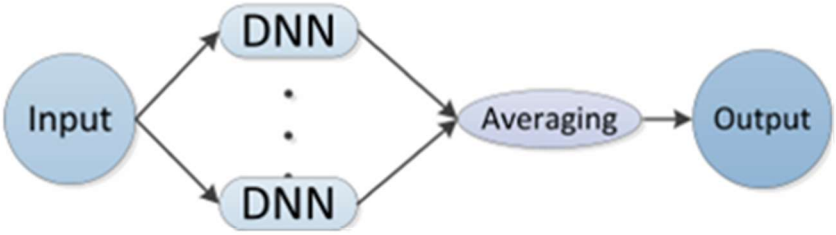



Figure 48: General architecture of MCDNN

## 6.2 General idea of Multi-column Deep Neural Networks

Deep Neural Networks Segmentation is a DNN-based "Semantic" image segmentation method by Ciresan et al. [49] It is based on a MCDNN used as a pixel classifier. Semantic segmentation explains the process of relating each pixel of an image as a membrane with a class label; this acquire characteristics and represent the whole original image. With using as input the intensities of image in a square window cantered on the pixel itself, MCDNN calculates the probability of a pixel and segments an image into classified pixels. As shown in Figure 49, MCDNN is trained about similar featured different stack with manually annotated membranes.

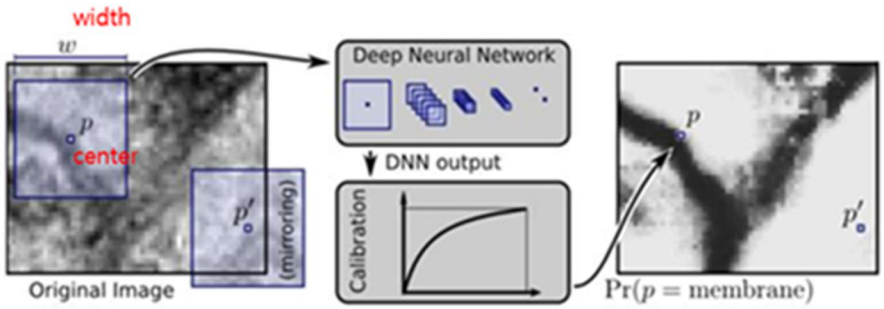



Figure 49

The MCDNN structure consists of a 4-layer convolutional layer plus two fully connected layers. and pooling method is max pooling. The last fully connected layer has only two neurons and uses the SoftMax activation function to obtain the probability of whether the input picture has a membrane. The network structure seems relatively simple. (Table 2)

| Layer | Type | Maps and neurons | Kernel size |
|---|---|---|---|
| 0 | input | 1 map of 95x95 neurons | |
| 1 | convolutional | 48 maps of 92x92 neurons | 4x4 |
| 2 | max pooling | 48 maps of 46x46 neurons | 2x2 |
| 3 | convolutional | 48 maps of 42x42 neurons | 5x5 |
| 4 | max pooling | 48 maps of 21x21 neurons | 2x2 |
| 5 | convolutional | 48 maps of 18x18 neurons | 4x4 |
| 6 | max pooling | 48 maps of 9x9 neurons | 2x2 |
| 7 | convolutional | 48 maps of 6x6 neurons | 4x4 |
| 8 | max pooling | 48 maps of 3x3 neurons | 2x2 |
| 9 | fully connected | 200 neurons | 1x1 |
| 10 | fully connected | 2 neurons | 1x1 |

Table 2

Like U-net, Ciresan et al used data augmentation, mirroring and rotating the training image by plus or minus 90 degrees at the beginning of each epoch.

Since each class is equal in the training set, but not in the test data, the network output cannot be directly considered as a probability value. On the contrary, they tend to seriously overestimate the probability of membranes. To solve this problem, the author performs polynomial function post-processing on the output of the network. The trained function is well approximated by monotone cubic polynomials, where the coefficients can be calculated by least squares fitting. Then use the same function to calibrate the output of all trained networks.

They conducted with foveation and nonuniform sampling to improve the network performance by manipulating its input data. (Figure 50) It imposes a spatially variable blur on the pixels of the input window, so that all details remain in the concave central part and the peripheral part is defocused by convolution with the disk kernel to remove the details. The advantage of this is that the task of the

network is to classify the central pixel of the window, and then ignore the most likely unrelated peripheral details in this way, but still maintaining the general structure of the window (context).

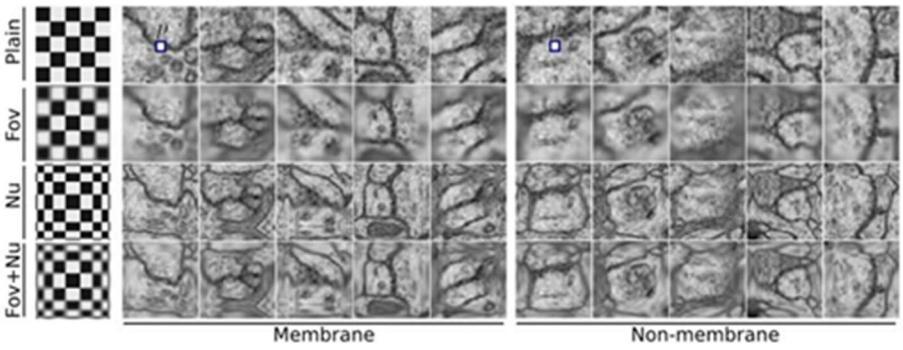



Figure 50

w = 65 input windows, from the training set. First row shows original window (Plain) but foveation (Fov) influenced other rows, nonuniform sampling (Nu), and both (Fov+Nu). Samples on the left and right accord with instances of class Membrane and Non-membrane, individually. The leftist image depicts how a pattern of checkerboard is affected by such transformations.

The larger the width w of the window can improve the performance, but the change of the value of w will lead to a larger network, and it will take more time to train and more training data. Image pixels are mapped to neurons only in the central region of the window in uneven sampling manner; elsewhere, as the distance from the centre of the window increases, their source pixels are sampled at a reduced resolution. As shown in Figure 50, the image in the window is distorted in a fisheye-like manner and covers a larger region of the input image which has fewer neurons.

Average output of different networks could solve the problem of overfitting. In this paper, the calibration output of multiple networks with different architectures is averaged to try to reduce this large variance problem.

## 6.3 Drawbacks

At first, it is slow in running because the network must be operated individually for each patch. Also Overlapping patches strategy occurs redundant process. A window move top to bottom, left to right on image even though previous process know probability of membrane. (Figure 51) Secondly, there is a trade-off between localization accuracy and the use of context. Larger patches need more max-pooling layers for reducing the localization accuracy, while small patches allow the network to see only little context.

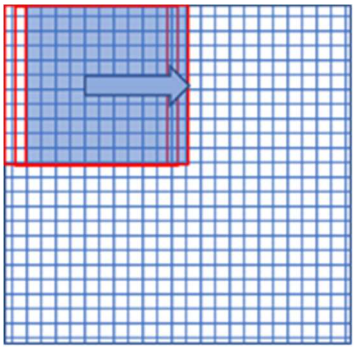


Figure 51: Sliding window approach

# 7 How to use my project

My project files are based on Colab pro and google cloud. Google cloud storage should be over 100 giga byte for this project. This is because checkpoint saves about 17 giga byte in each model. So, I suggest that you upgrade google cloud. Colab pro gives more GPU capacity which allows operating code faster. It is better to upgrade Colab also. Also, all packages are already installed in Colab. No package installation is required. Follow the instructions step by step

- unzip extracted file and upload on google cloud if checkpoint or log folders are not uploaded, create them on cloud.

- Open each jupyter notebook file and check the paths for your clouds. Some paths are '…/result' or '…/numpy'; these are not mistakes. When codes run, they will be created.

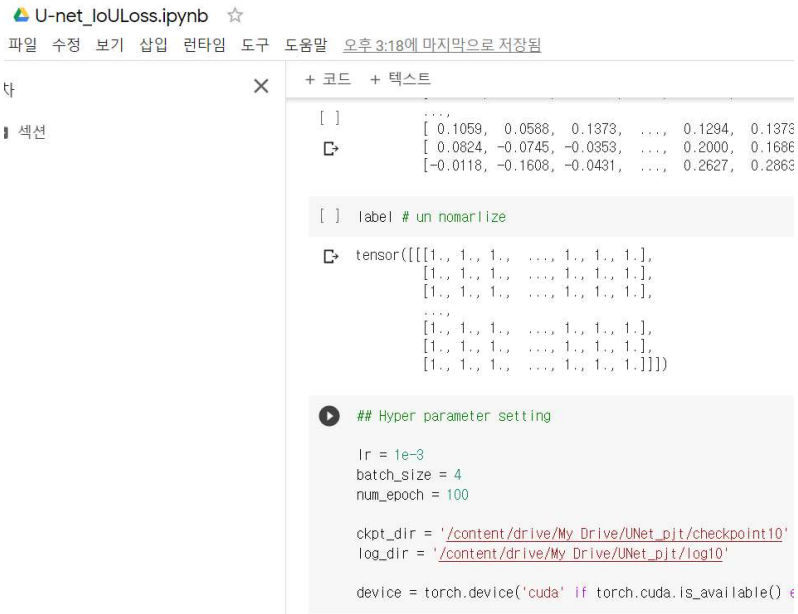


Figure 52: path examples.

- Mount google drive

```
[ ]  from google.colab import drive
     drive.mount('/content/drive')

     Drive already mounted at /content/drive; to attem
```

Figure 53: drive mount for load and save the files.

- Runtime type checking: select GPU/maximum RAM

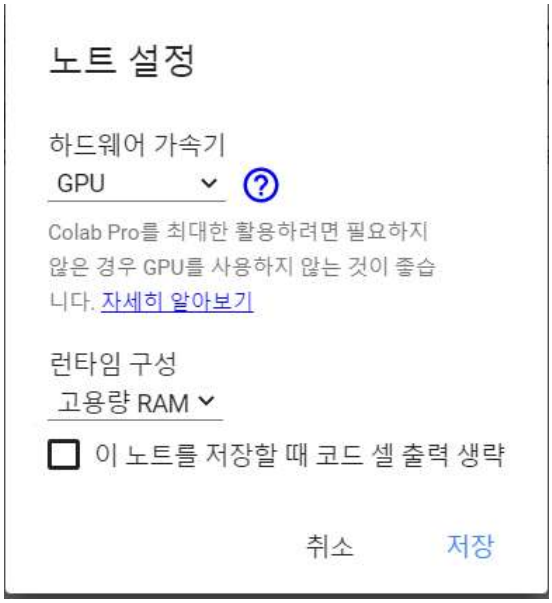



Figure 54: default is None and common RAM but you should change

- Run cells top to down order.

# 8 Self-Assessment

During my project, there have been several obstacles. At the fist, knowledge of bio medical dataset. Deep learning researchers and engineers who work in bio medical industry need to study about the specificity of medical dataset; For example, 224x224 pixel image is enough size to classify cat or dog. In contrast, in the case of X-ray image, only a few doctors can classify it whether cancer or not. It means that methods using in other are can fail in medical imaging. I got medical imaging study club last year, which was 10-week duration. Therefore, I thought I have knowledge in medical imaging. But It was my mistake. For instance, I applied model with SGD as optimizer and big input tile size firstly. I thought cropping patch in too small make bad context in image. When I run my code, I found it was wrong selection. It took too long time (6 hours) and loss rate was too high. Decisively, segmentation

imaging was too bad; it was far from ground truth. I recognized I was not enough competent in knowledge of medical imaging.

Moreover, insufficient experience in setting environment. My friends used their laptops for project coding. I decided to use my laptop like them; my laptop was cutting edge model (RAM is 16GB, GPU is RTX2060). Thus, I reckoned that my laptop has no limitation during project. I tried to use ISBI 2012 original dataset. However, when I use the original dataset, there is an error "Cuda out of memory" and "GPU already allocated". I did not know why these errors occurred. So, using other version of ISBI 2012 dataset was inevitable which leads to bad segmentation results. I changed my mind of laptop performance and I started to use colab pro and google one. They allowed to use two tera byte space and P100 GPU and 25 giga byte RAM, which can make a model with using original dataset. Furthermore, I do not have to worry about setting framework environment. When I was my laptop, I reinstalled ubuntu, nvidia driver and Cuda several times. It was waste of time, which bring out rack of the time. But after using colab, I could all framework for running networks. If I knew the these earlier, it is less troubles to conduct my project.

If I construct Ciresan's method, it could be more powerful explain of U-Net's excellence. Comparison between several different networks is common to show a certain network's competence; loss rate and running time comparison are exemplified. However, I could not model of Ciresan which leads to concentration on graphical analysis for showing U-Net's advantages.

Despite of these weakness, I have learned some knowledges in biomedical imaging and deep learning. During researching background researches, I have acquired general concepts of segmentation and loss function with their taxonomies and architectures, which can be beneficial for planning model of segmentation. Also, I have experienced distinctive feature of medical data. I got knowledges that I should avoid way in data loading and model development from trials and errors. If I participate biomedical project, it will be beneficial for dataset preprocessing. Most of all, modeling U-Net is satisfying achievement of my project. I did not expect good result of segmentation image. With implementing various loss functions which are used in medical imaging, I grasped the way how to improve segmentation result. Not only selecting network, but also selecting loss function is important. I found my strength from this experience.

In general evaluation, my project results, its schedule, and coding part are not quite fair, I have good potentials to be fair in medical imaging project.

# 9 Professional issues: Cloud computing flatform using management

A lot of artificial intelligence researchers in companies used to develop their projects, using computer in their company's server room. This was a kind of burden in computing resources. Researchers should predict usage of computing resources and

prepare of that. If prediction is larger than usage, Idle resources occur. Vice versa, project could go through difficulties. It is hard to get new servers or reuse old servers, which means that infra structures of software should be set for each project. Hardware performance also should be adaptable. When it comes to server schedule clash, server room manager should orchestrate resources depending on servers and project. Individual developer and students were in worse environment. Their computers performance was not enough to handle amount of data at once. They should change configuration of drivers and frameworks for project in their computer. In some case, it can take a long time or almost impossible to install. (Figure 55) I have also experienced these harsh troubles when I started first modelling.


Figure 55: An example of installing infra structure of software in deep learning.

Conventional deep learning infra structure software installation needs not only supports of many people but also long time. Hardware procurement and environment setting can take months. However, advent of cloud computing in deep learning has alleviated these problems and additional benefits.

- Data transfer and connection

Cloud computing flatforms give services such as GUI and coding type for various data in real-time and batch. There are service about atypical or big data scale data. In case of data in company's own computer network, it is possible to train model without data transfer; cloud can control resources and data.

- Data processing

Scalability is a feature of cloud. Regardless of data size, cloud serves techniques without any big change. For instance, big data clusters based on Spark can be operated in automatic adjustment manner corresponding to usage. It makes optimization of computational costs.

- Model training

Cloud can operate model training with CPU and GPU. It also supports optimized selecting data processing, algorithm, hyper parameter. In auto machine learning, which is for efficient using computer resources, GPU cluster

nodes automatic generate and operate. After operation finished, nodes are removed automatically. This is cost saving manner.

- Model interpretability

There are service about general validation and interpretability of model. Explaining models to executives and stakeholders is important. If data scientists can interpret the values, accuracy and debugging model, they can ensure the results and effect their decisions.

- Model packing and distribution

Software environments can be different rely on projects. Most cloud flatforms control environments and package distribution.

- Service monitoring.

Cloud platforms can collect and managing input data, prediction. If pattern of input data is different from pattern of used data, which is called data drift, prediction would be bad. Cloud platforms schedule regular monitoring, so it can take information for retraining time.

- Managing process and results

Above all advantages are organically connected. For example, what datasets and codes are utilized, how good performance is, distributed by what service in a certain deep learning experiment. Lineage of that should be checked.

The researchers have only to concentrate on develop model and data acquisition if they use cloud flatforms.

There are many cloud computing flatform for deep learning development but Amazon Web Services (AWS), Microsoft Azure, Google cloud platform (GCP) are mainstream. AWS is cloud flatform service of amazon, which take number one market share. (https://aws.amazon.com/) Azure is service of Microsoft. (https://azure.microsoft.com/ ) GCP is increasing fastest in the market share. (https://cloud.google.com/) Table 4 shows that pros and cons of three cloud flatform.

| | AWS | Azure | GCP |
|---|---|---|---|
| User interface | Easy | Not good | Easy |
| Instance speed | Fast | Slow | Moderate |
| Start speed | Moderate | Slow | Fast |
| Pricing Models | Expensive | Expensive | Cheap |
| Number of services | Many | Moderate | Few |
| Integration with open-source | Good | Best | Good |

Table 4

# References


1. Gould, S., Gao, T., & Koller, D. (2009). Region-based segmentation and object detection. In *Advances in neural information processing systems* (pp. 655-663).

2. Al-Amri, Salem Saleh, N. V. Kalyankar, and S. D. Khamitkar. "Image segmentation by using edge detection." *International journal on computer science and engineering* 2.3 (2010): 804-807.

3. Ng, H. P., et al. "Medical image segmentation using k-means clustering and improved watershed algorithm." *2006 IEEE southwest symposium on image analysis and interpretation*. IEEE, 2006.

4. Milletari, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. "V-net: Fully convolutional neural networks for volumetric medical image segmentation." *2016 fourth international conference on 3D vision (3DV)*. IEEE, 2016.

5. Huang, Huimin, et al. "UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation." ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020.

6. Zou, Zhengxia, et al. "Object detection in 20 years: A survey." arXiv preprint arXiv:1905.05055 (2019).

7. Sultana, Farhana, Abu Sufian, and Paramartha Dutta. "A review of object detection models based on convolutional neural network." Intelligent Computing: Image Processing Based Applications. Springer, Singapore, 2020. 1-16.

8. Romera-Paredes, Bernardino, and Philip Hilaire Sean Torr. "Recurrent instance segmentation." *European conference on computer vision*. Springer, Cham, 2016.

9. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid Task Cascade for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.

10. J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In ECCV, 2016. 1, 3, 7

11. H. Riemenschneider, S. Sternig, M. Donoser, P. M. Roth, and H. Bischof. Hough regions for joining instance localization and segmentation. In ECCV. 2012. 1, 2, 3

12. Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In CVPR, 2016. 1,

2, 3

13. Mueed Hafiz, Abdul, and Ghulam Mohiuddin Bhat. "A Survey on Instance Segmentation: State of the art." arXiv (2020): arXiv-2007.
14. Li, Biao, et al. "A Survey on Semantic Segmentation." 2018 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE, 2018.
15. Siam, Mennatullah, et al. "A comparative study of real-time semantic segmentation for autonomous driving." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2018.
16. Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
17. S. Bell, P. Upchurch, N. Snavely, and K. Bala. Material recognition in the wild with the materials in context database. In Computer Vision and Pattern Recognition (CVPR). IEEE, 2015.
18. D. Grangier, L. Bottou, and R. Collobert. Deep convolutional networks for scene parsing. In ICML 2009 Deep Learning Workshop, volume 3. Citeseer, 2009.
19. V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv preprint arXiv:1511.00561, 2015.
20. F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122, 2015.
21. L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. arXiv preprint arXiv:1606.00915, 2016.
22. F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho,Y. Bengio, M. Matteucci, and A. Courville. Reseg: A recurrent neural network-based model for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 41–48, 2016.
23. Saval-Calvo, Marcelo, Jorge Azorín-López, and Andrés Fuster-Guilló. "Comparative analysis of temporal segmentation methods of video sequences." Robotic Vision: Technologies for Machine Learning and Vision Applications. IGI Global, 2013. 43-58.
24. E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. CoRR, abs/1608.03609, 2016
25. Bengio, Yoshua (2009). "Learning Deep Architectures for AI" (PDF).

Foundations and Trends in Machine Learning. 2 (1): 1–127.

26. Szegedy, Christian, Alexander Toshev, and Dumitru Erhan. "Deep neural networks for object detection." Advances in Neural Information Processing Systems. 2013.

27. Bang DaeHwa, Estimation of Travel Mode Choice Models using Deep Neural Networks, 2019 [In Korean].

28. Pasupa, Kitsuchart, Supawit Vatathanavaro, and Suchat Tungjitnob. "Convolutional neural networks based focal loss for class imbalance problem: A case study of canine red blood cells morphology classification." Journal of Ambient Intelligence and Humanized Computing (2020): 1-17.

29. Mortazi, Aliasghar. "Optimization Algorithms for Deep Learning Based Medical Image Segmentations." (2019).

30. LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

31. Hubel DH, Wiesel TN (1968) Receptive fields and functional architecture of monkey striate cortex. J Physiol 195:215–243

32. Fukushima K (1980) Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern 36:193–202

33. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

34. Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

35. A sliding window-like approach: a classification network is used over different patches of original image to reconstruct a pixel-by-pixel estimates of the probability maps. A full-image approach: like the FCNN and UNET approach, rely on fully convolutional architectures and the upscaling phase is incorporated in the network itself using transposed convolutions.

36. A sliding window-like approach: a classification network is used over different patches of original image to reconstruct a pixel-by-pixel estimates of the probability maps. A full-image approach: like the FCNN and UNET approach, rely on fully convolutional architectures and the upscaling phase is incorporated in the network itself using transposed convolutions.

37. Lan, Yuan, Yang Xiang, and Luchan Zhang. "An Elastic Interaction-Based Loss Function for Medical Image Segmentation." arXiv preprint arXiv:2007.02663 (2020).

38. Lin, Tsung-Yi, et al. "Focal loss for dense object detection." Proceedings of the IEEE international conference on computer vision. 2017.

39. Milletari, F., Navab, N., Ahmadi, S.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 2016 Fourth International Conference on 3D Vision (3DV). pp. 565{571 (2016)

40. Salehi, S.S.M., Erdogmus, D., Gholipour, A.: Tversky loss function for image segmentation using 3d fully convolutional deep networks. In: International Workshop on Machine Learning in Medical Imaging. pp. 379{387 (2019)

41. Rahman, M.A., Wang, Y.: Optimizing intersection-over-union in deep neural networks for image segmentation. In: International symposium on visual computing. pp. 234{244 (2016)

42. Javier Ribera, David G¨uera, Yuhao Chen, and Edward J. Delp. Weighted hausdorff distance: A loss function for object localization. ArXiv, abs/1806.07564, 2018.

43. Ken CL Wong, Mehdi Moradi, Hui Tang, and Tanveer Syeda-Mahmood. 3d segmentation with exponential logarithmic loss for highly unbalanced object sizes. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 612–619. Springer, 2018.

44. Cardona, Albert, et al. "An integrated micro-and macroarchitectural analysis of the Drosophila brain by computer-assisted serial section electron microscopy." PLoS Biol 8.10 (2010): e1000502.

45. Suloway, C., Pulokas, J., Fellmann, D., Cheng, A., Guerra, F., Quispe, J., et al. (2005). Automated molecular microscopy: the new Leginon system. J. Struct. Biol. 151, 41–60. doi: 10.1016/j.jsb.2005.03.010

46. Dosovitskiy, A., Springenberg, J.T., Riedmiller, M., Brox, T.: Discriminative unsupervised feature learning with convolutional neural networks. In: NIPS (2014)

47. Lin, Tsung-Yi, et al. "Focal loss for dense object detection." Proceedings of the IEEE international conference on computer vision. 2017.

48. Salehi, Seyed Sadegh Mohseni, Deniz Erdogmus, and Ali Gholipour. "Tversky loss function for image segmentation using 3D fully convolutional deep networks." International Workshop on Machine Learning in Medical Imaging. Springer, Cham, 2017.

49. Ciregan, Dan, Ueli Meier, and Jürgen Schmidhuber. "Multi-column deep neural networks for image classification." 2012 IEEE conference on computer vision and pattern recognition. IEEE, 2012.