COMP9517 Project

Group: WXSZ

SIMIN LI (z5098690)

SHAOYANG ZHANG (z5150006)

Hao AN (z5138496)

Wen XIANG (z5150871)

Jia LIU (z5130484)

Contents

1 Introduction

Image segmentation is a very important and the most challenging part in automatic analysis. Due to the advancements in automated collection of Electron Microscopy (EM) images [1], there is a demand for an efficient membrane detection method so that more study and analysis can be researched. This project is from ISBI 2012 Challenge. A full stack of EM slices will be used to train machine learning algorithms for the purpose of automatic segmentation of neural structures. The aim of the challenge is to compare and analyse the different competing methods based on their pixel and object classification accuracy.

In this project, we implement three different learning-based approaches for neuronal boundary detection. One machine learning based approach is Support Vector Machine (SVM) and two deep learning-based approaches are U-Net and Deep Contextual Residual Network. Firstly, we describe the dataset we used to train our model and what evaluation metrics are used for evaluating our trained model. Then, the details of design choice for each method will be presented in method part. Next, we will show our results of each method including parameters tuning and evaluation. Finally, we will compare three methods in many aspects including performance, advantages, disadvantages in discussion.

2 Method

2.1 Pre-Processing

Image augmentation is a technique that is used to artificially expand the data-set. It is useful in deep networks as this kind of networks need large amount of training data to achieve good performance.
For this assignment, 30 pictures are given as data set. At the beginning, these pictures have been used as training data, but the result is very poor as the boundaries are not clear and the nucleus are not removed (see in Figure1.1).

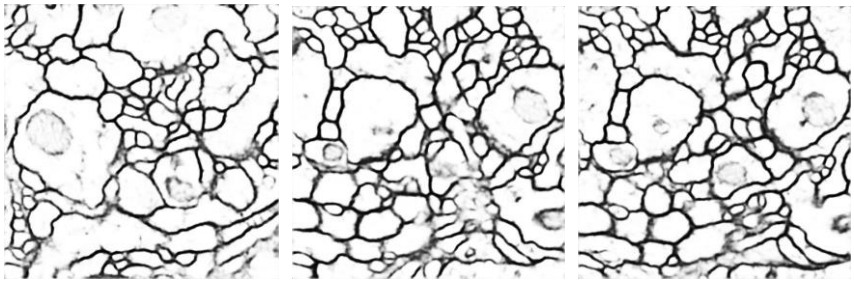


Figure1.1

Affine transformation which composed of rotation, scale, flip and translation is the most commonly used method of data enhancement. Firstly, I tried to use these methods to extend data set, but the final result has not been significantly improved. After that, I tried to use elastic deformation which is mentioned in the paper of U-net. By comparing the results of them, although the affine transformation improved the experimental results, the experiment achieved the best results on the elastically deformed data set.

For elastic deformation, first create a random displacement field to deform the image, ie $\Delta x(x, y) = \text{rand}(-1, +1)$, $\Delta y(x, y) = \text{rand}(-1, +1)$, where rand (-1, + 1) is to generate a random number uniformly distributed

between (-1, 1), and the second step is convolving Δx and Δy with a Gaussian function with σ. For the value of σ, if the value is large, the result value is small because the random value is averaged to zero. If the displacement field has been normalized (up to a norm of 1), then the field is close to a constant and the direction will be randomly; another unexpected situation is σ is small and then the field may look like a completely random field after normalization; for the intermediate σ value, the displacement field looks like elastic deformation, where σ is the elastic coefficient. In this project, 49.06 was chosen as the value of σ. The displacement field is then multiplied by a scale factor α that controls the deformation strength. In the project, the value that produces the best result is α= 1024.The Gaussian convolution displacement field is multiplied by the proportional factor α for the deformation strength to obtain an elastically deformed displacement field. The last step is the displacement field is applied to the image after the transformation to obtain the final elastic deformation enhanced data. The process of action is equivalent to the process of interpolating on an affine image and finally returns the result after interpolation.

2.2 Machine Learning based Model

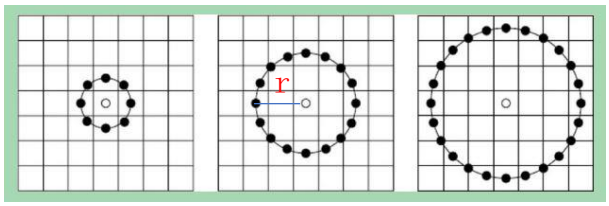
For machine learning based approach, I chose to use Support Vector Machine(SVM) as a classifier to detect membrane structures so that the electron microscopy images can be segmented. SVM is a popular and classic strategies for supervised machine learning and classification, especially binary classification [2]. Although there are many classifiers can be used for image segmentation, such as Random Forest, KNN, Naïve Bayes and so on, Support Vector Machine has its own advantages. These are some reasons why I chose Support Vector Machine as classifier for this project. SVM can works well with unstructured and semi-structured data like text and images. Also, SVM model has very low risk of over-fitting.

For this method, I normalized image pixel value in a range of 0 to1 at first for each image. After data augmentation, 300 images are used for feature extraction. Then, I extract high-dimensional features from patches for each pixel. Patches can be looked at an adjacency matrix for each pixel. After extracted feature vector, normalize each feature so that all features fit in the same range of 0 to 1. The corresponding label can be extracted from the corresponding Ground Truth. They formed the datasets which can be used for training our model. SVM model will learn how to classify a pixel to a label. Since the expecting results are similar like Ground Truth, the label is only 1 or 0, 0 means black and 1 means white. After we get the trained model, we can use the model to predict membrane structures in the given EM slices images.

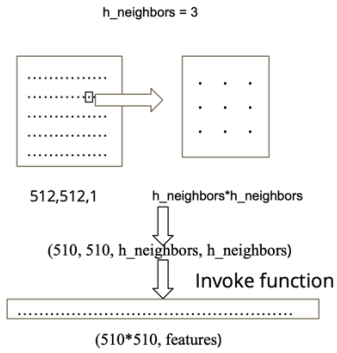
Feature extraction

Feature extraction is the most important part for machine learning based approach compared with deep learning based approach. There are many different ways to get different kinds of features, such as transform features, edge and boundary features, color features, shape features and texture features. Texture feature can be used to partition images into regions of interest and to classify those regions. Texture can present the intensity levels in a neighbourhood. Due to these characteristic, I chose to use pixel value, Local Binary Pattern and Haralick Textures features to form feature vector for each pixel.

Local Binary Pattern depends on the local region around each pixel. Each pixel is looked at individually. Its neighbours are analysed and summarized by a sign numeric code. The theory is that, the reference pixel, which is the canter point, A number of points are defined at a distance r from it. This method will compare the value between center point and those points with distance r (see figure below). Then it can get a binary code and a numeric value represent different texture, for example point, flat area and line. There are two parameters can be tuned for LBP. One is radius. Another one is number of points. Normally, bigger radius can capture bigger texture features. Different images have different details. That is why we need tune parameters to get optimal results. Usually we set a ratio between radius and number points so that these two parameters can change synchronously. The reason is if radius increases, but number of points keep as before, it will lose the ability to capture texture features.



Haralick Textures Features is the other one feature extraction method I used. Haralick Textures Features extracted thirteen textures based on the adjacency matrix. I just use the first nine features to be a part of my feature vector. The first nine features are Angular second moment, Contrast, Correlation, Sum of Square: variance, Inverse difference moment, Sum average, Sum variance, Sum entropy and Entropy respectively. Some of the features are low relevant among the 13 originally texture features, so I chose first nine features to reduce the complexity of the algorithm. Here is an example of how extract haralick textures features. Firstly, build a 3-dimensional matrix to store adjacency matrix for each pixel. Then extract adjacency matrix and store it to the pixel's location. Invoking Haralick Textures features function to compute 13 features (I just slice it to get first nine features). Finally, flatten it and merge with pixel value and local binary pattern.



Feature vector for each pixel contains 11 features, one is pixel value, one is from local binary pattern and rest of them are from haralick textures features. In the experiments, I pick randomly 100 pixels from one original training images then extract the corresponding labels. After data augmentation, we have 300 images as training data, so the total number of training data is 30000. Previously, I tried to select 1000 and 500 points from each image so that we can get a bigger dataset, but the training time is extremely too long.

Building model

I chose to use Support Vector Machine as a classifier. Here are some parameters can be tuned, such as kernel and penalty parameter C of the error term. I just tuned kernel parameters, since choosing a good kernel for a specific problem is very important. rbf kernel and lineal kernel are applied respectively for comparison

2.3 Deep Learning Based Model (Contextual Residual network)

The deep contextual residual network [3] inspires us and based on their method, we make some change and achieve the electron microscopy image segmentation.

Because of the limitation of the hardware, we cannot use too many features in each convolution layer, we decrease them and exchange the last layer to the 'sigmod' layer, using the 'binary-crossentropy' as the loss function.

The input of the model is 300 augmentation images, we normalization them (to make sure the value of each pixel between 0-1 we divide 255 for each of them). And for the labels, we set a threshold (255/2) if the value of the pixel bigger than it, change the value to 1 if not save 0.

network description

before the first maxpooling layer, using the convolution layer to increase the features of each pixel, then deconvolute (upsampling size(2,2)) the output to the same size of the input. In the second block, we use the convolution layer (stride = 2) to decrease the size of the matrix, after that, the output apply a couple of convolution layers for adding their neb's information, after the last layer of this block using the deconvolution(upsampling size(4,4)) to resize the shape of the tensor. We add more convolution layers in third block and at the end of this block also add a deconvolution layer (upsampling size(8,8)). In the merge, we add all the deconvolution layers result, then convolute the output before the sigmod layer. Finally fit the result of this network and label in the model. The overview of this network illustrated by fig 1

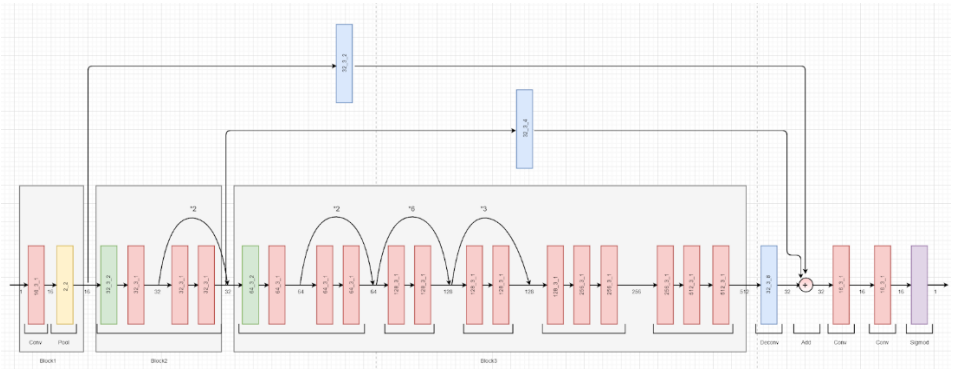


Fig. 1. Red and green blocks imply convolution layers, the strides of red one is 1 which is 2 in the green; blue boxes domate the deconvolution layers; yellow imply max pooling layers with patch size 2*2 strides 2. The output of this network is a 512*512*1 probability matrix

2.4 Deep Learning Based Model (U-Net)

In computer vision area, FCN (Fully convolutional Network) is one of the famous image segmentation networks, and in the medical image processing, U-Net is a more popular network.
According to the U-Net: Convolution Networks for Biomedical Image Segmentation [4], the author proposed the model with 4 times down sampling. Due to the RAM limitation in personal laptop. I would like to try the different depth of the U-Net network, which consider built three different models. One with 3 times down sampling, one is the original model from paper, and one with 5 times down sampling.
The aim to build those three models is to evaluate how deep is the model
Hoping after compared with those three models, I can get the results about how deep is the model, is it as deep as possible?

Network Architecture:
Contracting path (encode) structure:

- 3*3 convolution layer with padding = 'same', and rectified linear unit (ReLU)
- 3*3 convolution layer with padding = 'same', and rectified linear unit (ReLU)
- 2*2 Max Pooling

The contracting path including blocks which the structure shows above. At each pooling layer of each block, the number of feature map get double, starting from 64 at first block and 128 and so on. Contracting path can capture the context of the input image, which makes images are able to be segmented.

Bottle neck structure:

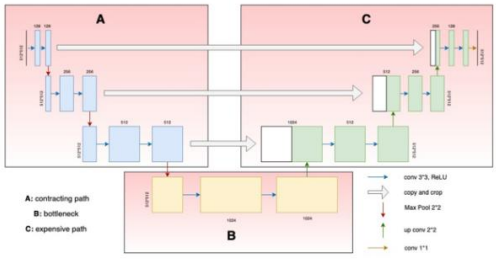
- 3*3 convolution layer
- 3*3 convolution layer
- Dropout (0.8)

The block between the contracting path and expanding path is bottleneck with dropout=0.8 which may avoid overfitting.

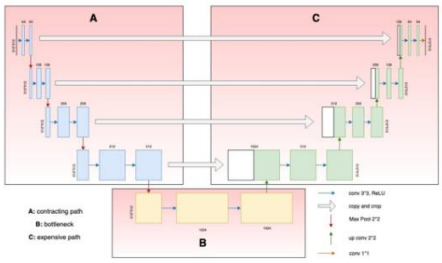
Expensive path (decode):

- 2*2 'up-convolution' layer
- Concatenation with correspondingly cropped feature map from contracting path
- 3*3 convolution layer
- 3*3 convolution layer

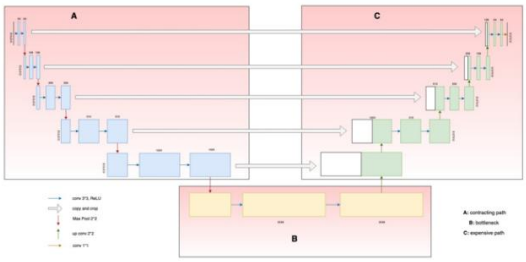
The expensive path including blocks which the structure shows above. Each block combines with contextual information from corresponding contracting path block, which is able to precise localisation.



Model with 3 down sampling



Model with 4 down sampling



Model with 5 down sampling

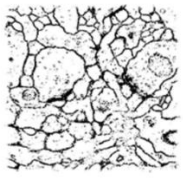
2.5 Post processing

In the two-class classification problem the threshold is usually set to 0.5. However, in our results some pixels whose probability to be membrane is slightly lower than 0.5 will be regarded as non-membrane and this may cause the break of a membrane structure, to reduce the break, first set the classification threshold a bit higher than 0.5, and then label the possible membrane by selecting the pixel whose probability value is lower than the threshold. By doing this, the membrane pixels with higher probability than 0.5 can be included into a possible connected membrane region [5].

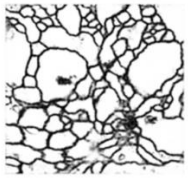
Result with threshold set to 0.5

Results with threshold set to 0.6

Followed by lowering the threshold, using Median Filtering to remove noise in the cytoplasm, and Gaussian Adaptive Thresholding to reinforce the edge of membrane for a better segmentation.

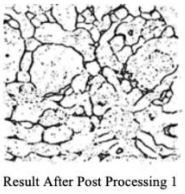


Result Before Post Processing 1

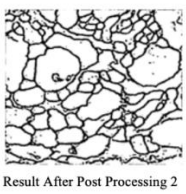


Result Before Post Processing 2

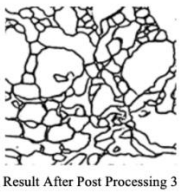
Result Before Post Processing 3



Result After Post Processing 1



Result After Post Processing 2



Result After Post Processing 3

3 Experimental setup:

In this project, we implement three different learning-based approaches for neuronal boundary detection. One machine learning based approach is Support Vector Machine (SVM) and two deep learning-based approaches are U-Net and Deep Contextual Residual Network. Firstly, we describe the dataset we used to train our model and what evaluation metrics are used for evaluating our trained model. Then, the details of design choice for each method will be presented in method part. Next, we will show our results of each method including parameters tuning and evaluation. Finally, we will compare three methods in many aspects including performance, advantages, disadvantages in discussion.

The datasets are 30 ssTEM (serial section Transmission Electron Microscopy) images which are the Drosophila larva ventral nerve cord (VNC) as well as the corresponding segmentation ground truths. All these images represent a set of consecutive slices within one 3D volume. The basic information of the dataset is given in Table 1.

	original image	ground truth
number of images	30	30
size of image	512x512	512x512

Table 1. Basic information of the dataset.

Specially normalized versions of Rand error and Variation of Information were found to be the best for our methods.

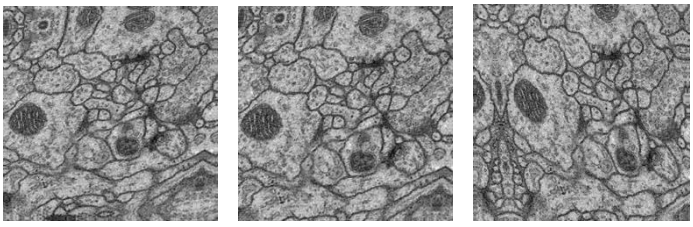
Foreground-restricted Rand Scoring after border thinning: V^{Rand}(thinned) defined as 1 - the maximal F-score of the foreground-restricted Rand index, a measure of similarity between two clusters or segmentations.

Foreground-restricted Information Theoretic Scoring after border thinning: V^{Info}(thinned)

4 Results

4.1 Pre-Processing

After experiment, in Gaussian convolution, sigma = 40.96, alpha = 1024 may have the best result. A part of images after elastic deformation which from original images and flags(see in Figure 1.2).



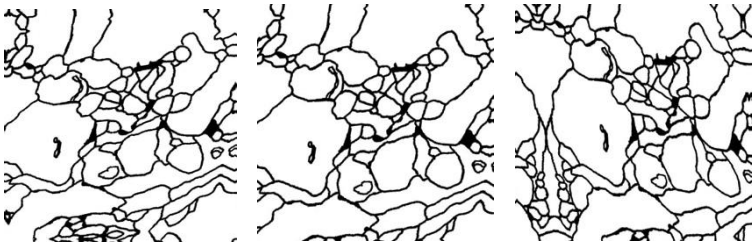


Figure 1.2

After data augmentation, the final result has been improved significantly as the boundary is very clear and most of the nuclei are removed (see in Figure 1.3).

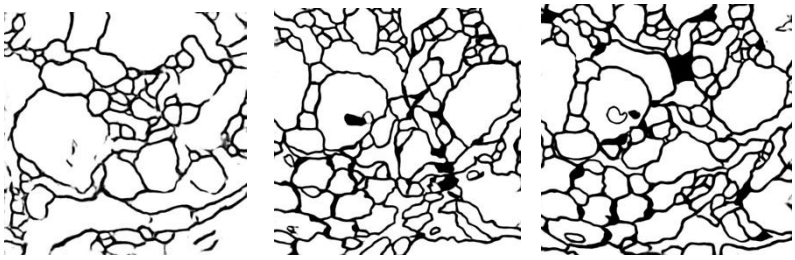
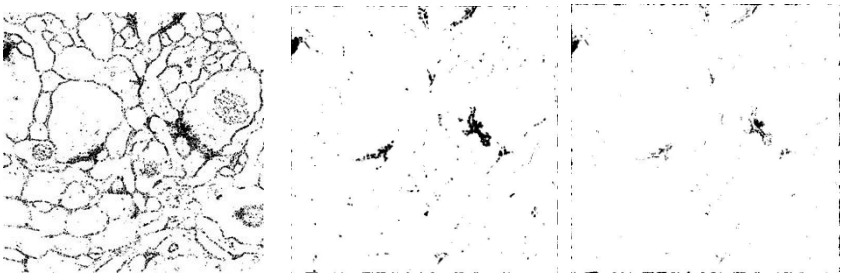


Figure 1.3

4.2 Machine Learning based method

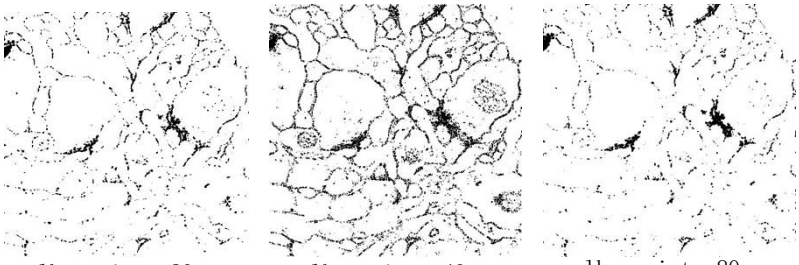
In this part, the process of tuning parameters will be presented. For feature extraction, local binary pattern has two parameters to be tuned, the number of circularly symmetric neighbor points(lbp_points) and radius(lbp_radius) respectively. As for Haralick texture feature, only one parameter can be tuned, called haralick neighbors(h_neighbors). Turning to model parameters, I just tune kernel type which is the most important parameters. rbf kernel and linear kernel are applied in experiments. Here is some comparison of results generated by using different parameters. I just put predict label of first give images and comparison among different parameters setting.

Firstly, I use rbf as SVM kernel.



lbp_points=40 lbp_radius=10 h_neighbors=3	lbp_points=40 lbp_radius=10 h_neighbors=5	lbp_points=40 lbp_radius=10 h_neighbors=9

We can observe that with the increase of h_neighbors, the model lost its ability to capture membranes. Due to this observation, I realized that when I used linear kernel and set h_neighbors to 11(which is the default value I set to SVM model with linear kernel), it always produced a whole white label. From outputs, we can find h_neighbors = 3 performed better. After that I tried to change lbp_points and lbp_radius. As I mentioned before, there is a ratio between lbp_points and lbp_radius. lbp_radius times ratio equals lbp_points. Here the ratio is 4.



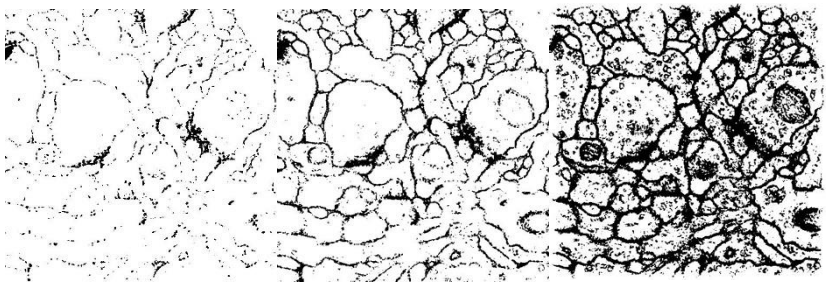
lbp_points=20
lbp_radius=5
h_neighbors=3

lbp_points=40
lbp_radius=10
h_neighbors=3

lbp_points=80
lbp_radius=20
h_neighbors=3

From above comparison, we can see lbp_radius set to 10 can get a best result. Big radius will ignore small details. Membranes are thin in the images, so with the radius increases, membranes become light.

The following is parameters tuning process of SVM with linear kernel.



lbp_points=40
lbp_radius=10
h_neighbors=11

lbp_points=40
lbp_radius=10
h_neighbors=21

lbp_points=40
lbp_radius=10
h_neighbors=41

From above outputs, we can find that with the h_neighbors increase, more details can be captured even noise and cell nucleus.

		kernel and parameters			
		rbf h_neighbors=3	linear h_neighbors=11	linear h_nighbors=21	linear h_nighbors=41
Training	Accuracy	0.8213	0.8394	0.8446	0.8440
Testing		0.8196	0.8340	0.8447	0.8400

Here is the evaluation metrics of results from different model. The predict labels are produced by cross-validation and V^{rand} and V^{info} is calculated by given script. Above images I just used the first image from dataset. It is just for comparison. The rest of predict label may be better than the first one since each images is different.

From the evaluation metrics we can find linear kernel with h_neighbors=21, lbp_points=40 and lbp_radius=10 performs better.

	kernel and parameters		
	rbf	linear	linear
lbp_points=40 lbp_radius=10	h_neighbors=3	h_neighbors=21	h_neighbors=41
V^{rand}	0.4835	0.5026	0.2743
V^{info}	0.7045	0.7062	0.7729

Findings:

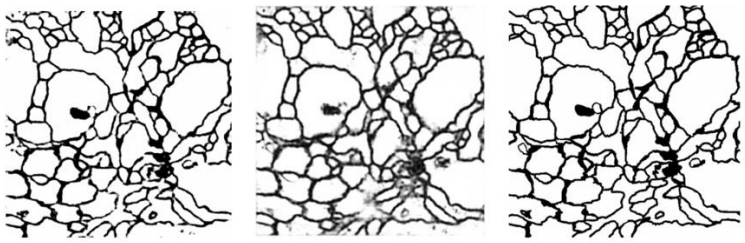
The overall result is not that good. There are many reasons leading to this.

1.My dataset may not big enough to train a high-performance model. After data augmentation (got 300 images),

I randomly selected 100 pixels from each picture. Each image has 515*512 pixels. The picking rate is relatively low. Here is a trade-off. I tried to pick 1000 pixels for each image, but the training time is too long.
2. For machine-learning based approach, it is different from deep learning based approach. From the outputs, we can see that many details such as noises and cell nucleus also showed which should not be segmented. More pre-processing should be done to the dataset since the given images have many noises. Only increasing the size of dataset is not enough.
3.When we tuned the relevant parameters and tried to find the optimal, it is hard to visualize their impact. So, current result may not be the best result of solving this project by using SVM. Also, the choice of features is important. There are many features can be used. Different features will lead to different result. For this specific project, there should be another more effective features can be used. Furthermore, I find normalization is very important for SVM.
4.At the beginning, I did not normalize each feature. The training model time was too long. After I normalized each feature, the training time becomes fast.
5. Most of the 2-class classifiers work best if the number of positive examples is about the same as the number of negative examples. From ground truth, I find the number of two classes is quite different. It may cause bad result.
6. Currently, I applied a pixel-wise classifier, which is quite slow since each image has 262144 pixels. The huge computation load leads to quite long predict time (about 40 mins). Just like Dr. Song suggested, superpixel can dramatically reduce the predicting time. it is faster and more memory efficient, improves segmentation performance.

4.3 Deep Learning Based Model (Contextual Residual network)

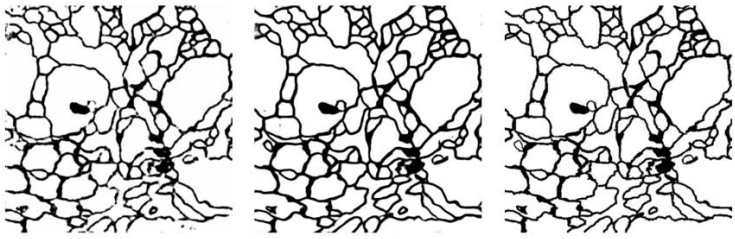
1. Contextual Residual network with 10 times cross-validation (split the augmentation data to 10-fold), and the network without cross-validation the result shown below:



cross-validation no cross-validation ground truth

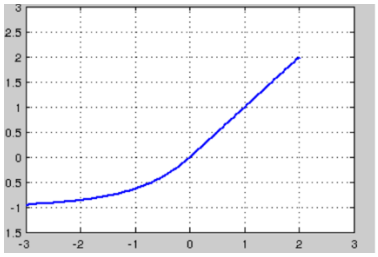
The result of the network with cross-validation is clearer than the one without cross-validation. The larger datasets improve the model. Also, the model is not overfitting, because the test data has no Intersection with train data.

2. To improve the model, we try different activation function (elu and relu). And the result shown below:

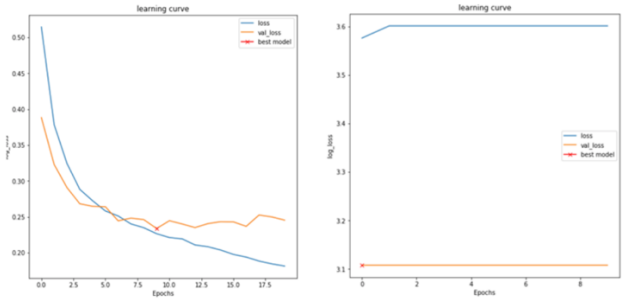


Elu relu ground truth

Their results of relu seems batter， however the relu function will cost more time to train the model. The ELU function speeds up learning in deep neural networks and leads to higher classification accuracies[6]. The output from the relu function is larger than 0, the value which is smaller than 0 will be exchanging to 0. For the ELU function, it has soft saturation in the left part, which can push their average value closer to the 0, convergence faster.

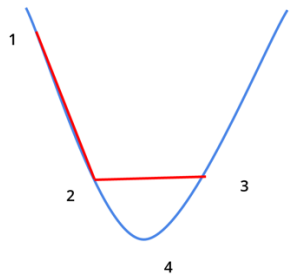


3. Due to the time limitation, Using the network without 10 times cross-validation to compare other parameters. We try to increase the learning rate for conferencing faster, the result shown below:



Learning rate 1e-4 learning rate 0.001

The leaf hand side learning rate is too high to convergence, in the gradient descent process, the point jump over the bowl bottom point, then because of gradient descent, the point jump back. So, this forms a loop



Based on the previous processing (adjustment parameters), the model with relu activation function using 10 times cross-validation seems can get a better result, but value of the average of vrand and vinfo between these

two functions are close. Considering the time complexity, we choose the ELU as the convolution layers activation function.

Because of the limitation of the hardware, we can only use batch_size smaller than 5 in this model, we just using batch_size = 4 and not change this number in this model

So the final model parameter we choose for Contextual Residual network is:

activation function: ELU

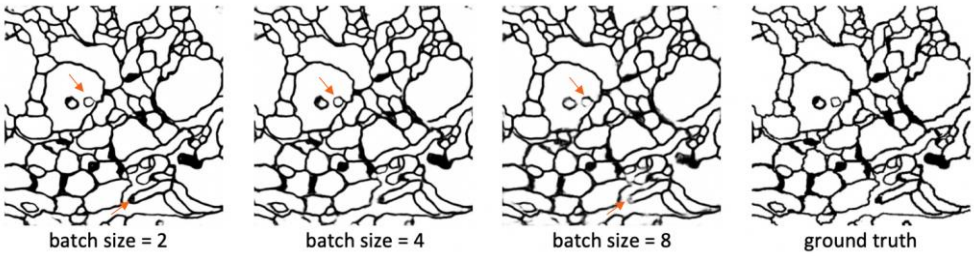
train_data = 270; val_data = 30; 10 times cross-validation

Batch_size = 4; loss function = binary cross entropy

4.4 Deep Learning Based Model (U-Net)

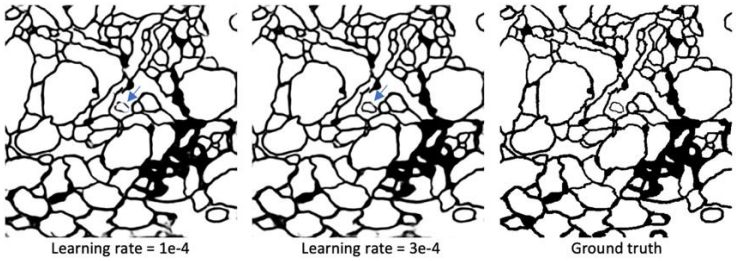
U-Net model with 4 times down sampling

1. I set the learning rate is 1e-4, epoch is 20, dropout is 0.5, the results show below:

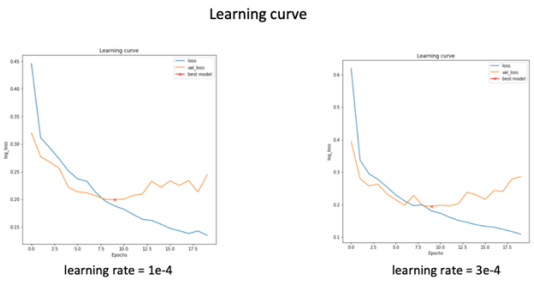


As the orange arrows shows, with the increase of batch size, the test image loose more edge details. Normally batch size as bigger as better. However, due to my experiment, I believe sometimes small batch size can get high random rate, may have better result. Small batch size may have better generalization ability, which also can get better results.

2. Based on the best performance is batch size equals to 2, we try to increase the learning rate. Results shows below:



As the blue arrow shows us, the result from leaning rate equals to 3e-4 is much better in the details than learning rate equals to 1e-4, and the edge is clearer.



According to the learning curve shows above, our ideal loss is starting with a high learning rate then reduce it until converge. Compared with two learning rates, the figure learning rate equals to 3e-4 is more fit our ideal.

3. Because it still has blur area in the result, I increase the drop from 0.5 to 0.8. The results show black areas are fulling-in in drop equals to 0.8 than drop equals to 0.5.
   Therefore, so far, the best result in this model is batch size equals to 2, learning rate equals to 3e-4, and drop equals to 0.8.

Changing U-Net model with 5 times down sampling

Due to the limitation of laptop and huge number of features (up to 2048), the maximum batch size I can set is 2 and the learning rate is 1e-4. Changing the dropout from 0.5 to 0.8 to compare the results.

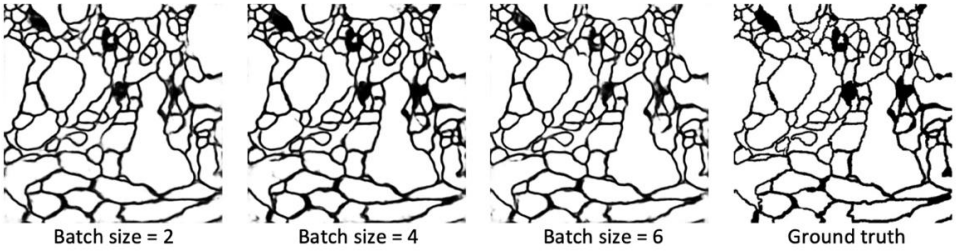
As the picture above shows, the result from small dropout has more edge details.

When I increase the learning rate to 3e-4, loss did not converge at all.

Therefore, the best result in this model is batch size is 2, learning rate is 1e-4, and dropout is 0.8.

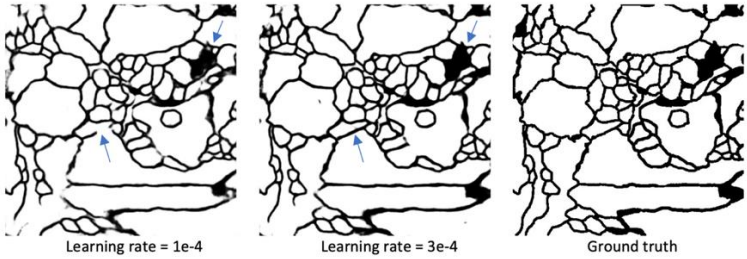
Changing U-Net model with 3 times down sampling

1. Due to huge features captured in U-Net model with 4 times down sampling, I set the features from 128 to 1024 which generally guaranteed the comparison between these two models. First, I adjust two different batch size and the results shows below:



   As we can see, with the increase of batch size, the results loose clear edge and get burl. Except long experiment time, I choose 2 as the batch size followed the rest experiment.

2. Based on batch size equals to 2, the result still not clear in some part of edge. I try to increase the learning rate to 3e-4, the result shows bellow:



As the blue arrows show, increase the learning rate can detective more edge and black area.

3. So far, the best result is batch size equals to 2, learning rate equals to 3e-4. Because this model only contains three down samplings, and I set the drop is 0.8 at first two adjustment steps. The big drop is, the features may lose more. Reduced the drop to 0.2, the result when drop equals 0.2 is much better than drop equals to 0.8, the thickness of the edge was tested.

According to our three different depth models, the model with three times down sampling is impressive me. The result is way much better than other two models. The FIJI results show below:

	vrand	vinfo
0	0.9981	0.9976
1	0.99897	0.9988
2	0.9994	0.9988
3	0.99989	0.9996
4	0.9992	0.9989
5	0.9997	0.9098
6	0.999	0.9988
7	0.9998	0.9995
8	0.9979	0.9978
9	0.9997	0.9989
10	0.9889	0.9964
11	0.9998	0.9996
12	0.9981	0.9979
13	0.9987	0.9973
14	0.9998	0.9991
15	0.9996	0.9983
16	0.9905	0.9956
17	0.9987	0.9978
18	0.9999	0.9997
19	0.9996	0.9992
20	0.9944	0.9962
21	0.9999	0.9992
22	0.9975	0.9967
23	0.9996	0.9989
24	0.995	0.9952
25	0.9994	0.9979
26	0.9998	0.9993
27	0.9992	0.9973
28	0.9993	0.9964
29	0.9848	0.9877

Best model from 3 times down sampling

img	vrand	vinfo
0	0.998	0.9978
1	0.9984	0.99769
2	0.99899	0.99827
3	0.9997	0.99948
4	0.99917	0.9989
5	0.9988	0.99862
6	0.99492	0.99665
7	0.99975	0.99916
8	0.99907	0.99796
9	0.99961	0.99874
10	0.9998	0.99928
11	0.99902	0.99883
12	0.99822	0.99835
13	0.99551	0.9956
14	0.99965	0.99834
15	0.94986	0.98544
16	0.98945	0.99408
17	0.9819	0.99387
18	0.94987	0.98766
19	0.99706	0.99611
20	0.95768	0.98716
21	0.8865	0.97767
22	0.9702	0.9889
23	0.9493	0.9824
24	0.92886	0.95128
25	0.9762	0.9836
26	0.92966	0.9667
27	0.90586	0.8886
28	0.9077	0.9402
29	0.91931	0.95824

Best model from 4 times down sampling

In my opinion, for the feature extraction in the model, the shallow structure can capture some simple features of the image, such as boundary, colour. However, in the deep structure, receptive field increased, and after more convolution operation, it can capture some abstract features of the image. Normally, we believe the more features we captured, the more accuracy we get. In the U-Net model with 5 times down sampling, the feature captured up to 2048, which means the more details we need to consider and also more detail in the image.
In my experiment, I read images as grey scaled image first which only have one channel. In this dataset, it is easier to use shallow model to detective edge and colour.

4.5 Post-Processing

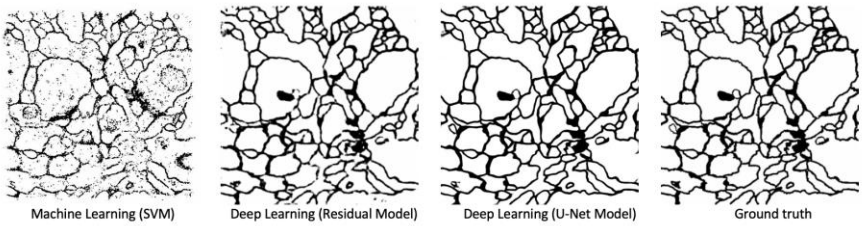
Use K-fold cross validation and set 30 ssTEM with ground truth as dataset, to do the cross validation set K=5, i.e. split the dataset to 5 folds, repeat the experiments for 5 times and each time select one of 5 folds as testing dataset and the rest as training dataset. The average of the evaluation metrics (i.e. VRand and VInfo) for each experiment performed is shown in below table.

	UNet		Deep Residual		SVM	
No.	V_Rand	V_info	V_Rand	V_info	V_Rand	V_info
Fold 1	0.963	0.984	0.991	0.994	0.502	0.708
Fold 2	0.971	0.982	0.975	0.982	0.478	0.682
Fold 3	0.961	0.978	0.981	0.984	0.486	0.712
Fold 4	0.953	0.973	0.971	0.978	0.520	0.710
Fold 5	0.981	0.985	0.976	0.979	0.532	0.690
Fold 6	0.967	0.971	0.982	0.983	0.490	0.692
Fold 7	0.972	0.968	0.976	0.976	0.521	0.716
Fold 8	0.983	0.985	0.973	0.986	0.518	0.718
Fold 9	0.962	0.986	0.983	0.978	0.501	0.687
Fold 10	0.961	0.973	0.977	0.981	0.483	0.692
Average	0.967	0.979	0.979	0.982	0.503	0.701

5 Discussion

Generally, comparing machine learning with deep learning, there are 3 different aspects:

1. The type of training data feed to model. Machine learning need to do the features extraction for the training dataset before fitting it to the model while deep learning fit the whole dataset to the model, followed by automatic convolution of the neighbouring pixels through multi-layer neural network.
2. The value of the label. The machine learning model - SVM is used for the two classification problem so the label is either a 0 or 1 (i.e. black or white which is discrete), while the value of label in deep learning is within range [0, 1] or [0, 255] which means a gray-level value to be continuous.
3. The accuracy of machine learning relies on the feature extraction while the accuracy of deep learning depends on a large scale of training data (therefore data augmentation is done during pre-processing of the training data), if the dataset is not large enough may cause overfitting.



Compared with Deep Contextual Residual Network, the copy and crop which is the highlight of U-Net model, it is necessary to predict a good segmentation map. As the figure above shows us, after 33 times convolutions, the result of Deep Contextual Residual Network is near to the ground truth, but edges are not well connected. On the country, the best results from U-Net is the model with 3 times down sampling which only has 8 times convolutions, and the result is much better than Deep Contextual Residual Network. We believe the copy and crop precise localization and combined with contextual information from the contracting path, which gives us better result. Another advantage of U-Net is that the input is not limited because of no dense layers. The only parameter learned on the convolutional layer is kernel size, which is independent of the size of input image. U-Net model is useful in biomedical segmentation. Combined with data augmentation because the number of samples is usually limited, and post processing also important.

The training time of these two deep learning methods is also quite different. Splitting the 300 augmentation data in two parts (270 training data 30 validation data), batch_size = 4, epoch = 15. The model of u-net network

costs more than one hour for training this data, but for contextual residual network, it only need to spend about half an hour, which is faster than u-net network.

In addition, the results (average of vrand and vinfo) of these two network are similar, and the contextual residual network is better than u-net model in time complexity. So, in this project the best method is contextual residual network in these 3 methods.

6 Conclusion

In this project, 3 learning methods are applied for the electron microscopy (EM) images, which are machine learning (SVM), Contextual Residual network and U-Net network approaches respectively. In order to fit large enough data to deep learning model, data augmentation is used to expand the dataset. And to reduce some membrane structures break, set the threshold a bit higher than 0.5 followed by median filtering and thresholding as post processing. In the end, the metrics of final results are evaluated by the value of V^{Rand} and V^{Info}, which shows that the deep learning is much better than machine learning in the image segmentation.

7 Reference

1. Sporns, Olaf, Giulio Tononi, and Rolf Kötter. "The human connectome: a structural description of the human brain." *PLoS computational biology* 1.4 (2005): e42.
2. Iftikhar, Saadia, and Afzal Godil. "Feature measures for the segmentation of neuronal membrane using a machine learning algorithm." *Sixth International Conference on Machine Vision (ICMV 2013)*. Vol. 9067. International Society for Optics and Photonics, 2013.
3. Xiao, Chi, et al. "Deep contextual residual network for electron microscopy image segmentation in connectomics." *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE, 2018.
4. Ronneberger, Olaf, et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” *Lecture Notes in Computer Science Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015, pp. 234–241., doi:10.1007/978-3-319-24574-4_28.
5. Xiao Tan1⋆, Changming Sun1, and Tuan D, et al,. “Membrane extraction using two-step classification and post-processing” Aizu Research Cluster for Medical Engineering and Informatics, pp. 5-6.
6. Djork-Arné Clevert, Thomas Unterthiner, Sepp Hochreiter(2015). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)

8 Appendix

Name (ZID)	Contribution	Results
z5130484	Pre-processing	Data augmentation (expand dataset)
z5138496	Machine Learning Method	SVM classifier model
z5150871	Contextual Residual Network	Contextual residual model
z5098690	U-Net Network	U-Net model
z5150006	Post-processing	Post Processing (improve accuracy)