U-Net Model In The First Publication Related
COMP 551: Mini-project 4 Report

Zoé Ducroux
School of Computer Science
McGill University
Montreal, QC, CA H3A2A7
`zoe.ducroux@mail.mcgill.ca`

Daisy Jayson
School of Computer Science
McGill University
Montreal, QC, CA H3A2A7
`daisy.jayson@mail.mcgill.ca`

Erjun Zhang
School of Computer Science
McGill University
Montreal, QC, CA H3A2A7
`erjun.zhang@hotmail.com`

Abstract

U-Net is a classic deep learning method and it has been widely applied to image segmentation. The first U-Net publication (*U-Net: Convolutional Networks for Biomedical Image Segmentation*) claimed that U-Net can be used for small dataset medical image segmentation and that data augmentation is the key to improve model performance. To find evidence that supported the two claims, we implemented the model from scratch following authors' original idea in the paper. Then we pre-processed (augmented) the data and applied the model on dataset without augmentation and with different augmentation size. Finally, test accuracy of 81.40% was achieved without data augmentation and accuracy of 86.02% was achieved by augmenting the dataset to from 21 to 147 images. By virtually checking and comparing performance values, we concluded that our results supported the author's two claims. Also, we made huge efforts make our report reproducible by creating GitHub repository, saving results on drive and gave details of our experiments. © 2022 The Author(s)

1 Introduction

U-Net is a classic deep learning method and it has been widely applied to image segmentation. *U-Net: Convolutional Networks for Biomedical Image Segmentation* is the first publication about U-Net[1]. In this paper, U-Net was used to do segmentation on a small dataset of only 30 medical images. The author concludes that data augmentation allows the model be trained with more variation in the images and allows for better training when using a small dataset. This makes data augmentation quite attractive for deep learning researchers, especially for medical researchers, who often suffer from having smaller datasets. This model has been developed since 2012 and has many variants, but we hardly found code (Python) to reproduce the results of this paper.Code found online is either written in MATLAB[1], or does not show the original steps in publication[2], or does not use the original dataset from the original paper[3]. The original publication is meaningful and reproducing the results teaches us more about this popular model. Thus, we set the project goal as reproducing the U-Net architecture for image segmentation following the idea from this landmark publication.

2 Scope of reproducibility

This publication belongs to the research area of medical image segmentation and machine learning (deep learning). It developed a new model, which was called the U-Net model because of its 'U' shape architecture. After the first successful application on 2012 ISBI segmenatation challenge, it began to attract the attention of machine learning researchers and medical imaging researchers. The author was attempting to describe this model in detail and apply it on three small ISBI challenge datasets (rosophila first instar larva ventral nerve cord dataset, Glioblastoma-astrocytoma U373 cells dataset and HeLa cells on a flat glass dataset). In the paper, the author emphasized that they used U-Net model to win ISBI segmenatation challenge with a huge margin in 2012 and 2015. In short, the publication proposed and concluded with two key claims:

- U-Net can be used for medical image segmentation and it can achieve high performance with a small dataset;
- Data augmentation is the key to improve model performance.

We were curious about how this amazing model can deal with small (less than 100) datasets. To our knowledge, there is no python code associated with this paper. Also, considering this model is quite important and widely used in medical imaging, we decided to write the code to implement this model exactly following the author's ideas. Though the publication mentioned the 'secret' technique of dealing with the small data problem is to use data augmentation, there is not enough evidence to prove this directly. Thus, we designed an extra experiment to explore this. The code of the model and experiment, resulting figures, resulting model weights, as well as a report will be yielded at the end (Goal). We will also make sure that our results can be reproduced easily by people interested in our work at the same time.

3 Methodology

To achieve the goal of this mini-project, we planned to implement the model code from scratch, and applied it to one of the three datasets mentioned in this paper and then explored the data augmentation effects on model performance.

3.1 Datasets

As we would just need to find evidence to support the first claim, we chose the simplest dataset (rosophila first instar larva ventral nerve cord dataset) of the three datasets to work on. It is a binary segmentation dataset and contains 30 Transmission Electron Microscopy (TEM) images (512×512 pixels with resolution of $4nm \times 4nm$, see images in Fig. 1. Each image had its corresponding segmented image, where the dark color represents the membranes and the lighter color represents the cells.

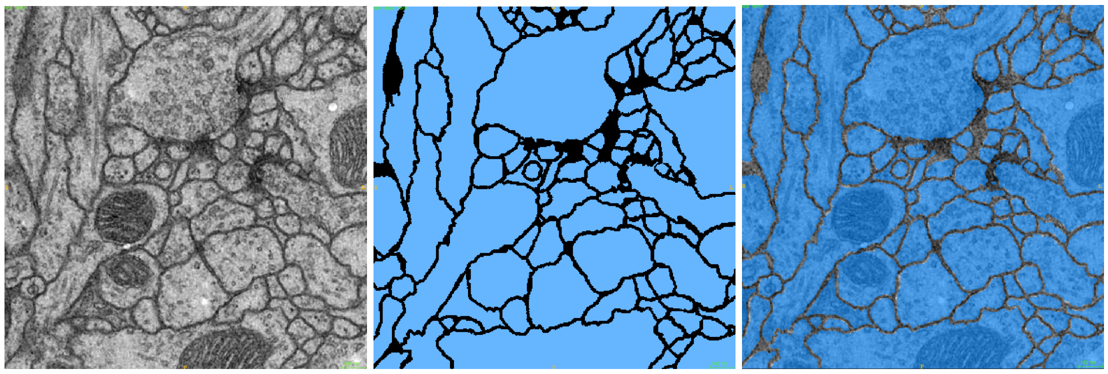

Figure 1: Example image and the corresponding label (segmentation) in the dataset. Left: image, Middle: human-labeled (or human-segmented) image (blue/black pixel value: $1/0$), Right: segmentation overlaid with the example image (with transparency $50\% : 50\%$), Both image and label had pixel of 512×512 and resolution of $4nm \times 4nm$

3.2 Data preprocessing

The data preprocessing includes the parts: downsampling the images, augmenting the images, and padding the images. All the data preprocessing steps were shown in Fig. 2. The first two steps are outside the model and the image padding step is included in the model code.

As we can see, to support the first claim, we just need to down-sample the original dataset from 512×512 to 388×388, which is also the output segmentation shape.

To find evidence of supporting the second claim, augmentation is needed. The augmented images were generated by randomly combining the following operations: rotating the original image $\pm 20°$, horizontally or vertically shifting the image ($\pm 5\%$), shearing ($\pm 5\%$), zooming ($\pm 5\%$), and flipping horizontally with *'nearest'* interpolation. Once the data augmentation was done, we normalized the images, using a Python script we wrote ourselves.

Since this model includes large number of conventional operations and each one will shrink the image, the output image (388×388) is smaller than the beginning image (512×512). As mentioned in the paper, padding also has to been done. Thus, as showing in Fig. 2, images in the training set were first downsampled to the same size as the output image size.

3.3 Model descriptions

We aimed to implement our own U-Net model, following the description in the paper. The paper gives a clear description of the architecture, which allowed us to implement our own model that achieved similar results to the one in the paper.

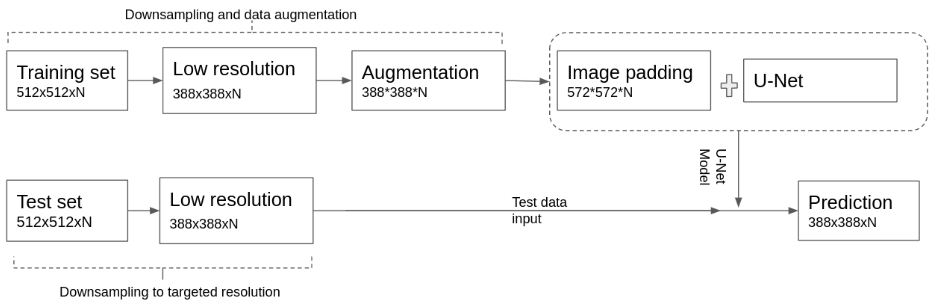

Figure 2: Data preprocessing, training, and prediction. Both training set and test set have downsampling step. Data augmentation was added to the training set in order to solve small data problem. The U-Net algorithm included an image padding step. Input data shape was $388 \times 388 \times N$ for both training and prediction, which was same with the model output. Note: N for training set and test set could be different with each other.

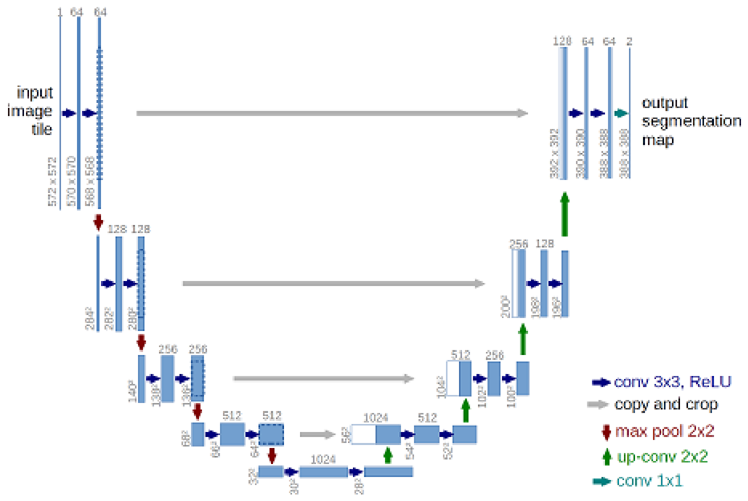

Figure 3: U-Net model architecture (figure from first U-Net paper[4]). It consisted of 5 successive contracting layers and 5 successive expanding layers. In each layer, 2 convolutional operations were included. Down-sampling pool and up-sampling convolutional parts connected the contracting layers and the expanding layers respectively; copy and crop steps were added to connect contracting section and expanding section. Details about this model could be found here.

The U-Net model has 5 successive contracting layers and 5 successive expanding layers. In each layer, 2 convolutional operations were included. Down-sampling pooling and up-sampling convolutional parts connected the contracting layers and the expanding layers respectively; copy and crop steps were added to connect contracting section and expanding section. The overall architecture is represented in Fig. 3, where the left side of the U-shape is considered the contracting path and the right side of the U-shape is considered the expansive path.

3.4 Experimental setup and code

Experiments designed and operations we used were to meet our two general project goal. To see if claims of the author are correct or not:

- For claim 1: we first implemented the model and applied it on the chosen dataset, then tested if it can segment the image well;
- For claim 2: model performance on dataset without data augmentation and with small data augmentation would be compared to see if augmented data can still be used as training dataset;
- For claim 2: model performance on dataset with different augmented dataset would be used to explore the behavior of the model performance while varying augmentation (from no augmentation to 147 augmented samples).

In order to better meet the project goal: reproducibility, we controlled the version of code and created a github repository, which can be found here. This includes the model, the data that we used, the libraries needed, a basic description of this project, as well as our experiments on the model. The model trained, results we got we be saved and upload on drive.

3.5 Computational requirements

This was a simple model and trained on a small-size dataset, thus the computational requirements were normal. If needed, it could be ran on a regular laptop. The experiments we did were on linux machine with *Pop os* 20.10, 64 GB memory and precessor of *Intel core* $i7-9700KF$ *CPU* ©$3.60GHz \times 8$. Tensorflow (Keras) was used for the model and **ITK-SNAP** was used for the viewing images. More details could be found in the *requirement* document and all the requirements can also be found in github.

4 Results

The results of our model support the claims that the U-Net can be used for cell segmentation and that data augmentation improves model performance. With our model, we achieved an accuracy of 90.14%, which is within 2% of the accuracy achieved in the paper using their model. With this high accuracy, we are confident in confirming the claims made in the paper. We can also say that we have implemented a model that is very similar to that of the authors of the paper.

4.1 Results reproducing original paper

4.1.1 Result 1: Experiment 1 (preprocessing steps)

By using method shown above, we obtained the preprocessed images (Fig. 4).

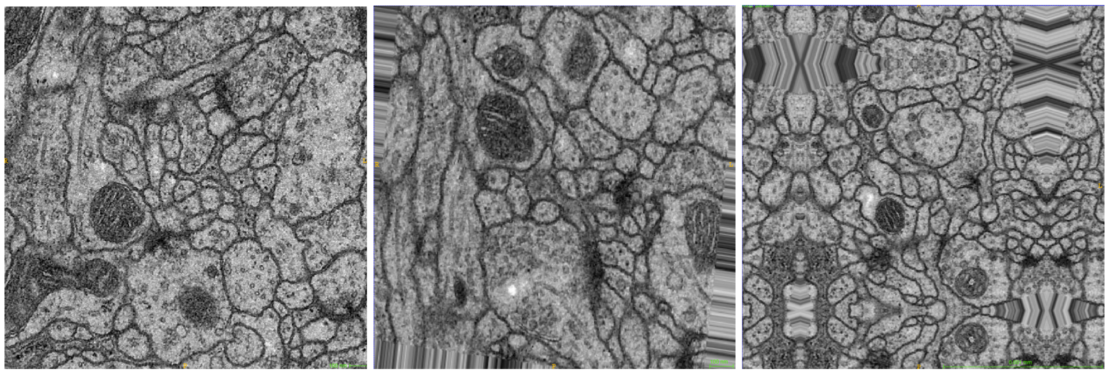

Figure 4: Example steps of image augmentation. Left: resolution downsampled image (pixel: 388×388), Middle: image after elastic deformation (pixel: 388×388), Right: image after mirror-padding (in the final model, pixel: 572×572). Note: its segmentation image were deformed exactly the same and were not shown here, images shown here may be not the same, and the whole samples processed.

4.1.2 Result 2: Experiment 1 (model can work)

After implementing the model and applying it on the dataset, predicted segemntation were output (Fig. 5). After training the model, we see that we achieved an accuracy of 81.40%, 81.56% for test and training, respectively, which we can see is a quite high performance.

4.2 Results beyond original paper

4.2.1 Experiment 2: augmentation can work

By using data augmenation , we trained the model, segmentation results can be seen in Fig. 6.

By varying augmentation size of training set, segmentation results can be seen in Fig. 7.

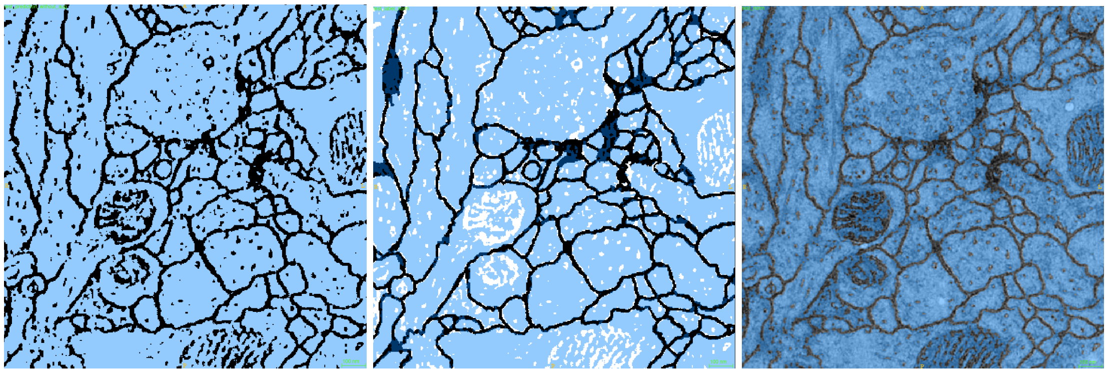
Figure 5: Predicted segmented images by applying coded U-Net model on the dataset with no data augmentation. Left: Predicted segmentation image, Middle: Predicted labels overlaid with the ground-truth (with transparency $50\% : 50\%$), Left: Predicted labels overlaid with the original image. Training set includes 21 images and human-labeled segmentation. Test set includes 9 images from the dataset (30 samples in total). Test accuracy of 81.40% and training accuracy of 81.56% was acheived, which proves that this model can learn the segmentation model well (claim 1).

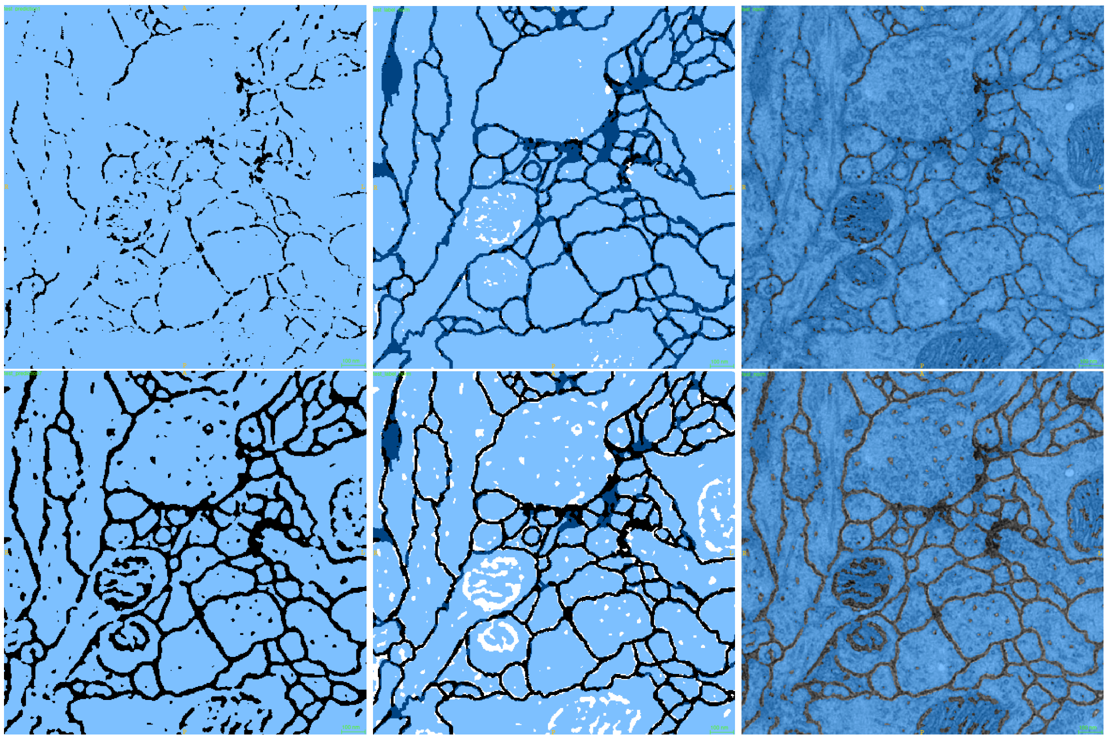
Figure 6: Predicted segmented images by applying coded U-Net model on the dataset with data augmentation. First column: Predicted segmentation, Second column: Predicted segmentation overlaid with the ground-truth (with transparency $50\% : 50\%$), Last column: Predicted labels overlaid with the original image (with transparency $50\% : 50\%$). First row: training set included 42 augmented samples, Second row: training set included 147 augmented samples. Both of them used same 9 samples as the same test samples as experiment with no data augmentation. Test accuracy of 82.99% and 85.66% respectively. Training accuracy of 83.73% and 86.02%. This could support claim 2.

5 Discussion

The model can work with small dataset, even without data augmentation. From Fig 5, we can see most of cell edges have been found out. While inside the cell, it was not clean enough. After increasing dataset size to 147 (secodn row in Fig. 6), it was much cleaner than before. This could be the evidence that this not clean intra cell segmentation could be caused by using two small dataset (20).

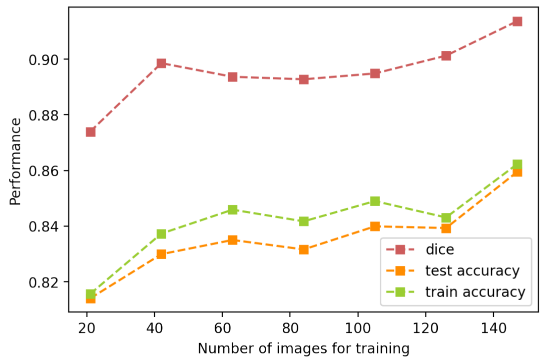

Figure 7: Performance of U-Net model while varying size of augmentation (from no augmentation to 147 augmented samples). Red: dice score, Green: accuracy of training, Orange: accuracy of test. The first point of each curve was the performance without data augmentation. The dataset was used of Drosophila first instar larva ventral nerve cord dataset. This plot could support the claim 2: data augmentation could improve model generalization performance.

While it is not always good to do the data augmentation. Comparing first row (un-consistent segmented cell edges) and second row in Fig. 6 to the no augmentation results, we can see while only use small data augmentation, the segmentation results first became bad and then became better. This could be caused by overfitting. Plot in Fig. 7 could support this. We can see a smaller accuracy gap between training and test performance curves.

5.1 What was easy

The structure of the model was easy to implement. Although this paper did not offer the code in a form that we could use, the authors provided a very clear description of the architecture of the U-Net model that they used. Using this description, we were able to build our own model to reproduce the results given in the model.

The standard to judge the performance is easy. In the article, it uses IOU and smallest error to compare the models to other models. But we can not access these values unless we submitted our results to the ISBI challenge. Despite this, it is not hard to choose a method to judge model performance on the standards that the model is well-suited or that the augmentation improves the performance. Here we choose accuracy and dice score as the performance.

5.2 What was difficult

The difficulties for this project generally are in two parts: making the model clear and making its results producible.

The former involved fixing the difference between the input and output image size. Though the paper mentioned the convolution modules would change image size, how the resolution should be changed was the most confusing part. In order to make this clearer, we made a figure for this, in Fig. 2. The latter difficulty involved the lack of detailed information about image augmentation, which was only provided with Matlab code. The paper also gave a clear description of the data augmentation that they implemented. However, neither the parameters nor the detailed distortion for the augmentation of images were mentioned. For this we had to conduct several trials. The code written through the commercial software Matlab was provided in 2015, but there is still a lack of Python code reproducing the results in this landmark paper. The code we found relating to this paper ignored many details but instead just focused on the general U-shape structure idea of this paper. This all made this a difficult-to-reproduce article.

Throughout this project we were curious about two of the authors' claims: U-Net works well on small medical datasets and data augmentation is key for solving the small dataset problem. We implemented the model from scratch, applied it on the small dataset and did the augmentation experiments. Finally our results gave the evidence to support the authors' two important claims.

Statement of Contributions

All team members worked on literature review and first model developed. Z. Ducroux and D.Jayson wrote the first version of manuscript, did the experiments and final writing quality check. E. Zhang implemented the UNet model, designed parts of the experiments. Group members worked collaboratively on the final data preprocessing, experiments, report and discussion of this project.

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[2] zhixuhao. unet: unet for image segmentation.

[3] Pytorch-UNet: PyTorch implementation of the U-Net for image semantic segmentation with high quality images.

[4] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

A Reproducibility Summary

Scope of Reproducibility

The main goal of this paper was to segment medical images with low contrast. The publication had two main claims:

- U-Net can be used for medical image segmentation and it could achieve high performance
- Data augmentation is the key to improve model performance

Methodology

To reproduce the results of this project, we had to write our own code to recreate the model and data augmentation described within the paper. We first conducted the image pre-processing and data augmentation, following the description in the paper. We then implemented our own U-Net architecture and tested our model.

Results

The results of our model support the two main claims of the paper. We achieved accuracy within 2% of that given in the paper, which shows us that this model works well for medical image segmentation. We also saw that our models performance improved greatly after the data augmentation, again confirming the claim of the paper.

What was easy

Implementing the architecture of the U-Net model was easy to implement, as the authors gave a clear description of the architecture they used, despite not giving the code. It was also easy to judge the performance of our model.

What was difficult

Making our model clear and understandable was a difficulty we faced in this project, as well as being able to reproduce the results given in the paper.

Communication with original authors

We did not have any contact with the original authors of the paper.