Abdominal Adipose Tissues Extraction and Visualization Using Convolutional Neural Network

Binhan Xi, Zhenghan Wang
Shanghai Jiao Tong University

***Abstract*—Abdominal adipose tissues extraction is an important project in both medical image processing and academic research. With much work of other researchers, we analyze comprehensively the contributions and problems left and introduce a new method using U-Net, a Convolutional Neural Network. Our work achieves a better accuracy and efficiency compared with the former.**

In this paper, we first introduce the problem itself and some related works, then propose our method and experience in the implementation process, finally we give some experiment results to evaluate our work.

***Index Terms*—Adipose tissues, U-Net, extraction.**

I. Introduction

OBESITY has become a worldwide healthcare issue. Substantial researches have shown that the accumulation of visceral adipose tissue (VAT) , which covers internal organs within the abdomen, has a stronger association with obesity-related illnesses, compared to subcutaneous adipose tissue (SAT) . Therefore, accurate and rapid measurement of VAT and SAT will be beneficial to assess the severity of the obesity-related diseases.

Currently, extracting the VAT is mainly based on the fundamental digital image processing methods and some multi-scale neural networks. Manual and automatic segmentation are two main approaches for abdominal adipose tissues segmentation. The manual way tends to achieve a relatively higher accuracy while the efficiency is quite low and the process itself is laborious and less objective. Current automatic segmentations, however, have some limitations on the image size, quality and need some pre-processing steps, which is time-consuming. We need to apply a more suitable convolutional neural network to this problem and evaluate the performance of it.

In this paper, a finer abdominal adipose tissues segmentation is proposed. This segmentation mainly consists of two stages. For the first stage, we perform some transformations on the original images, including normalization, converting to tensor and scale processing. For the second stage, we input the images in the last stage to the U-Net network and do the iterations to train it. After the training, we evaluate the trained network with some other images in the angle of accuracy and efficiency.

In sum, our approach has the following contributions:

-**Easier Pre-processing** The characteristics of U-Net network let us get rid of the laborious and time-consuming pre-process on the raw images. Now we only need to do some brief and simple transformations on the images which can be done by some predefined functions.

-**Less Time Needed** In our experiment, our network consumed rather less time than other ones due to a more reasonable and scientific network design and utilization.

-**Higher Segmentation Quality** We evaluate the performance of our work based on some parameters which will be mentioned in this paper and achieve a satisfactory result in the abdominal adipose tissues segmentation problem.

II. Related Work

In this section, we introduce some related work in the field of abdominal adipose segmentation and extraction to give a overall understanding of this problem.

Related Work One In the paper[1], Fei Jiang et al proposed a two-stage coarse-to-fine algorithm for AAT (abdominal adipose tissues) segmentation using a novel deep neural network model, MSDNN, which employs multiple parallel networks to extract high-level features from multi-scale inputs.

Related Work Two In the paper[3], Mitsutaka Nemoto et al use threshold processing and connection component analysis in each axial slice to do the body trunk extraction based on 24 sets of whole-body CT volume data with anthropometric measurement data.

Related Work Three In the paper[2], Hyunkwang Lee et al use the improved fully convolutional network (FCN) to do muscle segmentation, which shares some similarities with our abdominal adipose segmentation.

III. Approach Overview

In this section we will present a brief glimpse on our approach to solving this problem. Fig 1 shows the flow diagram of our experiment and codes.

First, we have a dataset of 140 labeled 512×512 png images. In these images, the SAT is marked in blue and the VAT is marked in green. These data have relatively high quality so we can make full use of them in our training process.

Second, we utilize the transform functions imported from the torchvision library to do some transforms on the original

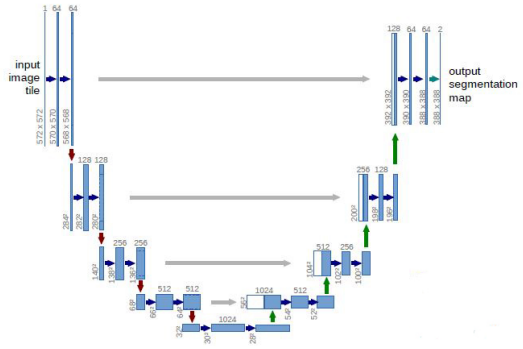

Fig. 2: The brief architecture of U-Net. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

images so that they ca meet the demands of the network training process.

Third, we feed the data into the U-Net network to do the training work. We iterate the training for 150 epochs and use some parameters to evaluate the network performance on the training set.

Finally, we apply the trained network on the validating set and use the same parameters to do the evaluation. The evaluation result will be shown in the following sections.

IV. Method

In this section, we will explain the method, mainly the U-Net network comprehensively in terms of its history, main ideas, implementation and performance.

U-Net, first presented by Olaf Ronneberger at al in 2015 in the paper [4] is a typical fully convolutional network (FCN), which differs from the well-known convolutional neural network (CNN). They modify and extend this architecture such that it works with very training images and yields more precise segmentations. The biggest contribution of this architecture is that the pooling operators are replaced by upsampling operators so that these layers can increase the resolution of the output. The U-Net architecture is shown in Fig. 2.

The U-Net network is open source so that we get it from the GitHub. In the implementation of this network, we introduce the energy function it uses, which is computed by a pixel-wise soft-max over the final feature map combined with the cross entropy loss function. The soft-max is defined as $p_k(\mathrm{x}) = \exp(a_k(\mathrm{x}))/(\sum_{k'=1}^{K}\exp(a_{k'}(\mathrm{x})))$ where $a_k(x)$ denotes the activation in feature channel k at the pixel position $\mathrm{x} \in \Omega$ with $\Omega \subset \mathbb{Z}^2$. K is the number of classes and $p_k(\mathrm{x})$ is the approximated maximum-function, i.e. $p_k(\mathrm{x}) \approx 1$ for the k that has the maximum activation $a_k(\mathrm{x})$ and $p_k(\mathrm{x}) \approx 0$ for all other k. The cross entropy then penalizes at each position the deviation of $p_\ell(\mathrm{x})$ from 1 using

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

where $\ell : \Omega \to \{1, \ldots, K\}$ is the true label of each pixel and $w : \Omega \to \mathbb{R}$ is a weight map that we introduced to give some pixels more importance in the training.

In order to apply the U-Net tool to our problem, we need to write a data loader first. In our data loader, the pillow and torch library is used and our coding language is Python 3. Our program has two modes, one is train mode and the other is validate mode. The loader does the job to fetch a certain number of images from a given directory, to perform some transforms, including compose, converting to tensor and scale transform, on the image. Finally, it outputs the modified images to the network.

V. Experiment

In this section, we introduce our experiment experience and some important points together with some typical problems and errors we met in writing and running our programs.

A. Step 1: Environment

Our network runs on the server provided by our TA so we first learned some remote access methods, some linux terminal operations and the usage of vim editor.

After that, we begun to configure our running environment. The network needs the support of CUDA so we first install CUDA on the server. Noticing that the Python program lacks some necessary libraries, we followed the commander lines' instruction to install these libraries step by step until our program runs without bugs.

B. Step 2: Parameter Adjustment

In our data loader, the most important variables are the number of epochs and the batch size. If the epoch number is set too big, the program will have to run for a long time although it changes little within an iteration in the end and if it is set too small, the performance of the training may be literally bad. As to the batch size, which is the number of images the loader read in each time, should also be set to a suitable number because if it is set too big, the CUDA memory will fail to stand it and if it is set too small, the program will also run slowly.

C. Step 3: Validation

When the network is finally trained, we should apply it to the validate data set which is new for it to test if it can accurately segment the image. In this step, we have several parameters to evaluate the network, including accuracy, mean intersection over union (mean iu) and frequency weighted intersection over union. We should notice that among them, the mean iu is the one that we should use in the evaluation instead of accuracy. This is because mean iu is the standard measurement of semantic segmentation. It calculates the fraction of the intersection and union of two sets and in semantic segmentation problems, these two sets are ground truth and predicted segmentation. The calculating method is

$$MIoU = \frac{1}{k+1}\sum_{i=0}^{k}\frac{p_{ii}}{\sum_{j=0}^{k}p_{ij} + \sum_{j=0}^{k}p_{ji} - p_{ii}} \tag{2}$$

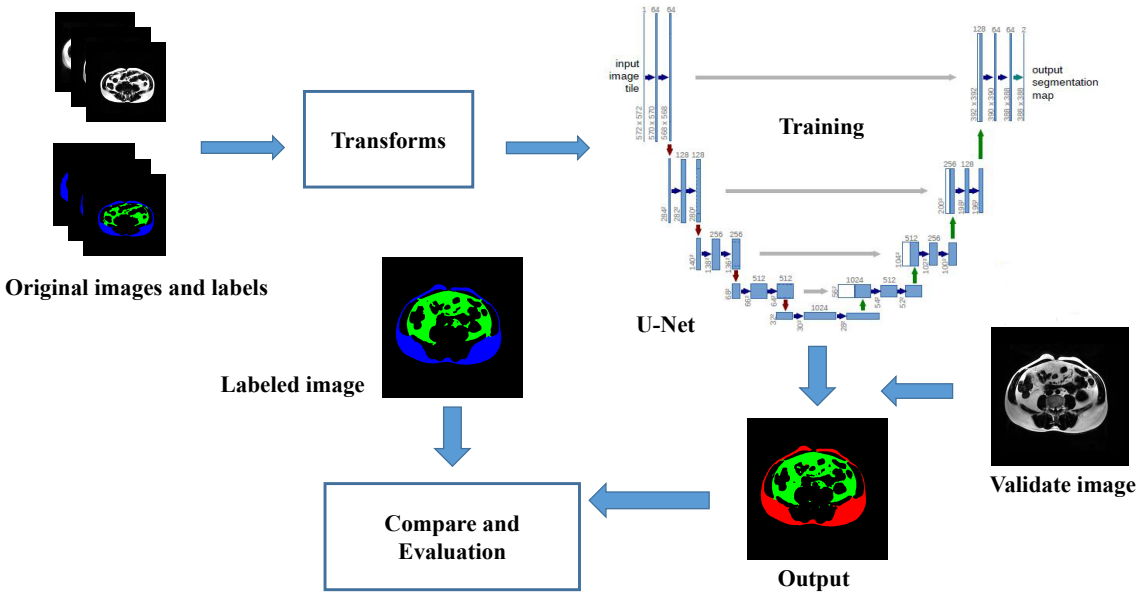

Fig. 1: Flow chart for abdominal segmentation. Stages include image transforms and FCN training. When the training is done, the network is applied to prepared labeled images to evaluate its performance.

.

On the contrary, the accuracy seems useless in semantic segmentation problems because we are not pursuing the fit accuracy. As a result, we did not use it as an evaluation parameter.

In addition, instead of validating the network after the whole training process is done, we do this work every time an epoch finishes so that we can trace the performance of the network in real time. Although this will cost more time to run the code, we believe it is worthwhile because we have to know the evolution of the network so that we can analyze this problem more deeply.

Fig.3 shows some runtime information provided by our program including the epoch number, the loss function and some evaluation parameters.

VI. Experimental Results and Analysis

In this section, we will introduce the result of our experiment and give some analysis both on the fruit of it and some problems await for further progress.

First we present our segmentation images after epochs' training. We can see from the image 4 that at first, the network failed to do a good segmentation because in its output image, the edge of the SAT is quite ambiguous and the shapes of the two types of abdominal adipose tissues do not fit the labeled image well. As the training goes on, we can clearly see the improving quality of our network. After epoch 125, there are almost no differences between our output and the ground truth.

However, we still find that our output images differ from the ground truth in some subtle details. Our result sometimes extract some small regions and dots in the image, which are

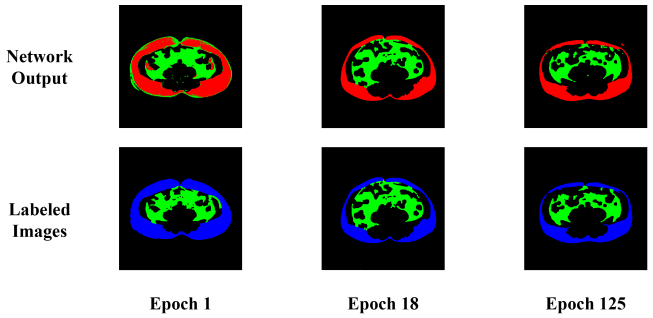

Fig. 4: The changes on the output of the network as the training goes on corresponding to the epoch number.

not shown in the ground truth. The reason may be the lack of training samples. We only have less than 150 images to train the network, which seems too few for it to adjust itself to a best condition. Additionally, since we only ran the program for 150 epochs, which costs 4 to five hours to finish the whole training, the network training may be better with longer training time.

Second we give a table that shows how the parameter (mean iu as an example) evolves during the whole training process. In this table, every time the current best match epoch changes, the program save the information of this iteration. The four parameters we select are Loss, Mean iu, Accuracy and fwavacc.

We also draw the data in a line chart (see Fig. 5) which shows the tendency of two main parameters' change.

In this chart we can easily see that the two parameters change rapidly within the first 20 epochs and tend to be steady after that. Thus, we have reason to believe 150 epochs' training

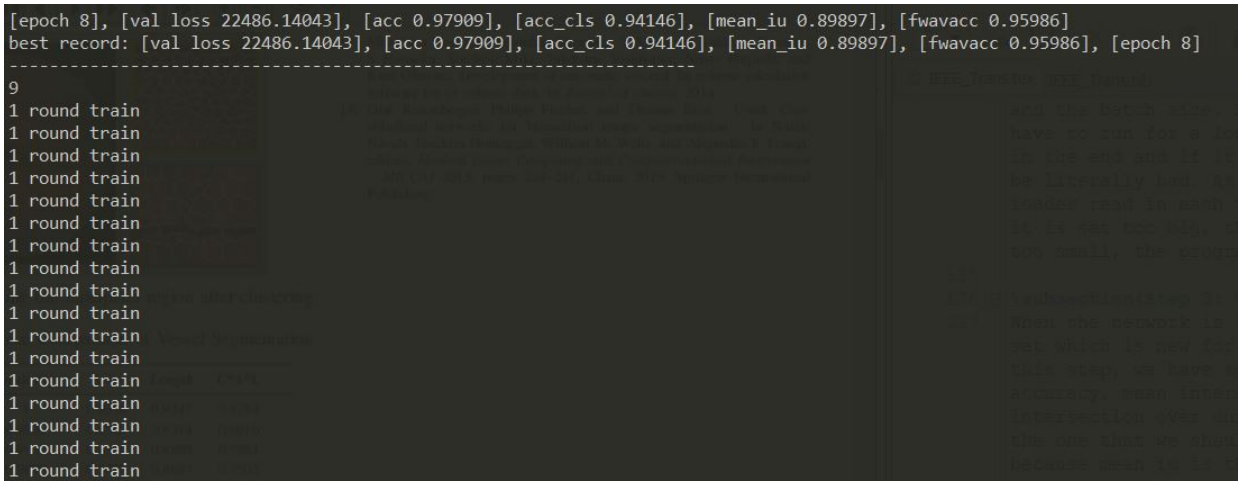

Fig. 3: The runtime screenshot of our program. Notice that this is the state when the 8th epoch finishes and some information such as best record and its loss function value, accuracy and mean iu is shown in the command line.

TABLE I: Performance of the Network Over Epochs

Epochs	*Loss*	*Mean iu*	*Accuracy*	*fwavacc*
1	79230.13359	0.73469	0.94349	0.91928
8	22486.14043	0.89897	0.97909	0.95986
35	14344.99180	0.90920	0.98121	0.96398
71	17024.46548	0.91089	0.98102	0.96366
125	21725.42720	0.91268	0.98160	0.96464
Best	**14344.99180**	**0.91268**	**0.98160**	**0.96464**

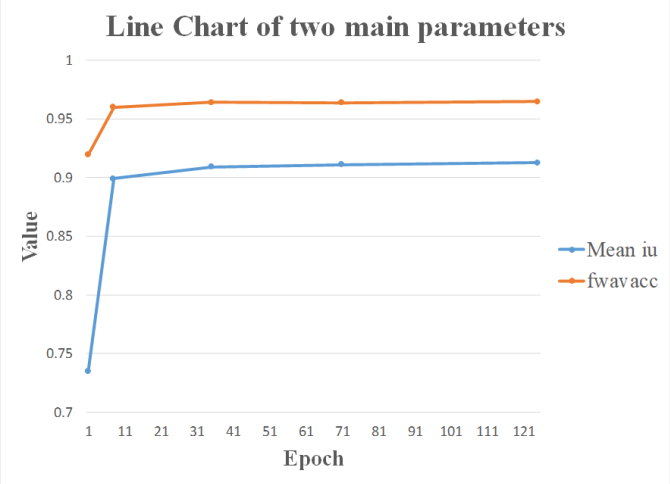

Fig. 5: This chart shows the change tendency of two main evaluation parameters, mean iu and fwavacc as the epochs go bigger.

is enough for the network to adjust itself to a best condition. We can also see that the mean iu over the validating data set reaches over 0.9, which is a very good value showing that our network performs well on the segmentation and the results match the ground truth well.

VII. Conclusion and Future Work

In conclusion, our application of the U-Net to the abdominal adipose tissues segmentation achieves a satisfactory result with the mean iu parameter being very high. We would like to say this is a successful attempt and the method itself is suitable for this kind of problems.

In the future, we believe it is necessary to more finely and carefully tune the parameters, especially the epoch number and loss function, in order to make the result get rid of some unexpected mis-extraction.

In addition, we think it is possible to apply this method to some other medical image extraction and segmentation works.

VIII. Group Members' Work Division and Contributions

In this section, we introduce the group members of this project. We also explain the division of work while doing the experiment work.

Binhan Xi, undergraduate student of Department of Computer Science and Engineering, Shanghai Jiao Tong University. In this project, he completes the environment setting and most of the code work is done by him. In addition, he makes the demonstration slides and the final project report. During the project process, he solves some of the problems and contacts with the professor and TAs.

Zhenghan Wang, undergraduate student of Department of Computer Science and Engineering, Shanghai Jiao Tong University. In this project, he helps Binhan Xi with the code work and do the mid-term demonstration. In addition, he also attends the group meeting regularly.

IX. Acknowledgement

We would like to present our faith and sincere thanks to our professor Bin Sheng and TA Siyuan Pan for their patience and instruction on our work.

REFERENCES

[1] Fei Jiang, Huating Li, Xuhong Hou, Bin Sheng, Ruimin Shen, Xiao-Yang Liu, Weiping Jia, Ping Li, and Ruogu Fang. Abdominal adipose tissues extraction using multi-scale deep neural network. *Neurocomputing*, 229:23 – 33, 2017. Advances in computing techniques for big medical image data.

[2] Hyunkwang Lee, Fabian M. Troschel, Shahein Tajmir, Georg Fuchs, Julia Mario, Florian J. Fintelmann, and Synho Do. Pixel-level deep segmentation: Artificial intelligence quantifies muscle on computed tomography for body morphometric analysis. *Journal of Digital Imaging*, 30(4):487–498, Aug 2017.

[3] Mitsutaka Nemoto, Tusufuhan Yeernuer, Y Masutani, Yukihiro Nomura, S Hanaoka, Soichiro Miki, Takeharu Yoshikawa, Naoto Hayashi, and Kuni Ohtomo. Development of automatic visceral fat volume calculation software for ct volume data. In *Journal of obesity*, 2014.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.