Exploring hippocampus segmentation on unbalanced data set using U-Net-based models

Kaiyuan Wu

Department of Bioengineering, Rice University, 6100 Main Street, Houston, TX 77005, USA

kvw1@rice.edu

Abstract. This paper studies the performance of training the U-Net-based models on an unbalanced data set for hippocampus segmentation. It investigates through a series of ablation studies the effect of the weights in loss function, sampling method, model architecture, and learning rate type, and compare across different trials regarding their dice score and accuracy to identify the best strategy under class imbalance. Lastly, it displays numerical and graphical results before discussing potential implications and future directions.

Keywords: Class Imbalance, Image Segmentation, U-Net, U-Net++, Dice score.

1. Introduction

The hippocampus (HC) is a region of the brain that is embedded in the temporal lobe. It plays a crucial, versatile role in cerebral functions such as memory, learning and spatial reasoning; damage to hippocampus often leads to neurodegenerative diseases such as head trauma, stroke, and Alzheimer's disease (AD). Accurate early-stage diagnosis is important for preventing and intervening disease progression; therefore, the shape and volume of the HC are often measured using structural magnetic resonance imaging (MRI), and HC is segmented to render a more clear and detailed view of it.

Image segmentation is widely used in fields such as computer vision, biomedical engineering, and bioimaging. It is defined as the process of dividing the image into distinct groups of pixels according to certain rules, outputting a mask where each segment is assigned a unique grayscale value or color to identify it. Ideally, each class should have equal proportion; however, in actual medical data sets, this rarely holds true, leading to the issue of class imbalance that drastically affects model performance. It is common that HC images have a much larger background class than the others. Under such circumstance, even blindly predicting a fully black image would be rewarded with an accuracy of over 90%, which is not a fair, accurate representation of the model performance. Thus, it is worth the effort to investigate the specific combination of model parameters (and hyperparameters) that performs the best under class imbalance.

In this paper, the experiment is conducted on the Kulaga-Yoskovitz (K-Y) data of HC [1], a representative example of unbalanced data sets. It focuses on testing different combinations of sampling method, loss function weights, model architecture, and learning rate type to examine how the performances vary. For model assessment, dice score and accuracy are used.

2. Method

2.1. *Preliminary steps*

2.1.1. Data set used. The Kulaga-Yoskovitz (K-Y) data set used belongs to the MNI-HISUB25 resource, which, based on 25 healthy subjects, contains their manual hippocampal subfield labels in forms of T1- and T2-weighted images [1]. This study mainly focused on the T1-weighted, standard defaced MNI images, which corresponded to 50 labels in total (left and right). Adopting the manual labelling proposed by Kulaga-Yoskovitz *et al*, each label was delineated by four classes of pixels, denoted by integers zero to four in the model, with zero being the background and one to three the three subregions, including subicular complex, Cornu Ammonis 1, 2 and 3, and CA4-dentate gyrus, respectively [1]. As displayed in figure 1(b) on the right, HC was a small, colored region within the brain.

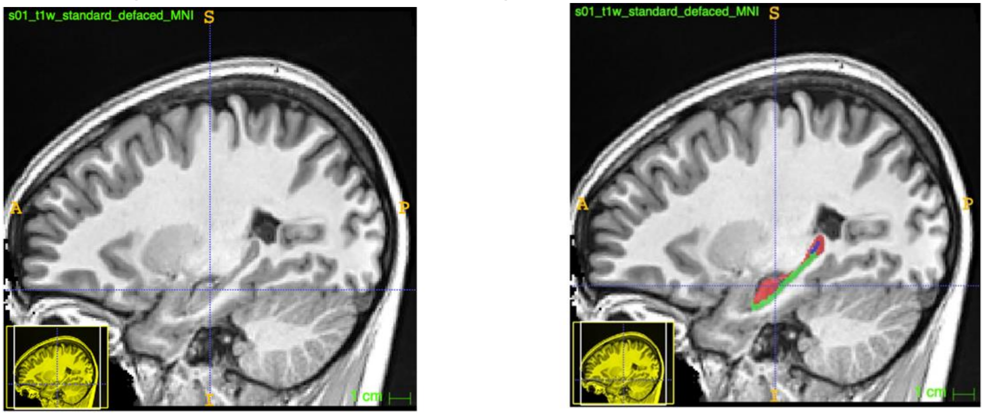

Figure 1. (a) The exemplary T1w image of subject 1 from K-Y data without labels, (b) with labels. All images rendered by ITK-SNAP v. 3.6.0, 2017.

2.1.2. Data pre-processing. First, to reduce the dimension of data, cropping was carried out on both images and associated left and right labels. Specifically, the starting and ending positions for cropping along each spatial dimension were determined by widening the actual length of HC by ten on both sides to allow some additional space while ensuring that the widened interval would not exceed the original data size. Since there were left and right labels, there would be two different sets of 3D coordinates for cropping, each of which would be used to crop the same image. As a result, each image would be cropped twice. To facilitate data storage, each cropped image with its associated cropped left and right HC label were combined and saved together, respectively. In the end, 50 image-label data sets were saved in the format of nii and numpy files, with two for each of the 25 subjects. Subsequently, these data were randomly split into 40 training samples (20 subjects) and ten testing samples (five subjects), with subject’s index in the testing set being determined via random sampling without replacement.

2.1.3. Data augmentation. Firstly, images in both training and testing sets underwent normalization - with mean zero and standard deviation one. Subsequently, for images in the training set, random crop was applied (not the same as previous cropping), with the multiplier in each dimension (between zero and one) being determined using either uniform or normal sampling. This multiplier then multiplied the difference of actual and crop size in each dimension to get the starting position. The shape for random crop was specified to be 32 by 32 by 32, and for those less than 32, padding was introduced to ensure dimensional consistency. Besides that, random flip was also introduced along the width dimension (i.e., left-right axis) if the random number generated was less than 0.5. This intended to add more variations into the training set. On the other hand, for images in the testing set, a 3D sliding-window approach was adopted, with the window size being 32 and overlap being ten.

2.2. Model

The model architecture was largely based on the classic U-Net. To generalize it to 3D, the approach similar to Manjón et al [3] was followed. To move beyond Monjón et al, the U-Net++ was also incorporated and adapted from Zhou et al [4] to the task of 3D HC segmentation.

2.3. Variables of interest

2.3.1. Model group number. In case of applying a 3D convolutional layer, the model group number controlled the ways the inputs were connected to the outputs. In this study, model group was selected to be one or four for analysis, with one being all inputs convolved to outputs, while four being four subgroups of inputs leading to one-fourth of all output channels before concatenation.

2.3.2. Sampling method. Two different statistical sampling methods were considered for deciding the multiplier in random crop, including uniform and Gaussian sampling. For uniform sampling along all three dimensions, the interval was chosen to be [0, 1]. For Gaussian sampling, while the mean was held constant at zero, the standard deviations were given a range of random values such as five, two, one, 0.3, and 0.1. Furthermore, for Gaussian sampling, clipping was used to guarantee that the multiplier sampled would always fall between zero and one.

2.3.3. Loss function. Cross entropy was used since it is one of the most popular loss functions for classification tasks in image segmentation. In case of class-imbalanced data set, equal-weight loss function, the default choice, treated the over-represented and under-represented classes equally. Besides equal-weight loss function, this experiment also introduced random-weight and normalized class-weight loss functions, with the former being specified to be [0.01, 1, 1, 1], or [0.003322, 0.3322, 0.3322, 0.3322] after normalization. Class-weight loss function addressed the issue of class imbalance by forcing the model to possibly sample more from the under-represented group and less from the over-represented. To obtain the class weights, each class's proportion was calculated, taken the respective reciprocal, and normalized to sum up to one. In this case, the normalized weights for class zero (background) to three (subfields) were calculated to be [0.00216, 0.110, 0.241, 0.647], respectively. For both, the background weight was designed to be roughly the same for comparison.

2.3.4. Learning rate. Two types of learning rate were probed in this study, including the fixed learning rate and the learning rate scheduler called "StepLR". The fixed learning rate was picked to be 0.001, while the scheduler was specified to have a step size of one and gamma 0.5, decreasing the learning rate by half for every epoch and enabling the model to learn more and more slowly.

2.4. Evaluation criteria

Accuracy alone may not be sufficient, since it tends to give misleading result for models that mis-classify minority class on a data set predominated by the majority class. Thus, dice score was also used, with its value ranging from zero to one. Unlike accuracy, dice score also accounted for the overlap between the ground-truth area and segmented area. In this study, the mean dice score of all four classes was calculated per epoch to monitor the training progress, and each final dice score displayed in the experiment section was averaged over three repeated trials.

3. Experiment

3.1. Implementation details

In each trial, with an initial learning rate of 0.001, the model was trained for 100 epochs with early stopping. An Adam optimizer was used, and a batch size of four was specified. For the sake of computational efficiency, the depth of both the U-Net and U-Net++ was limited to be three only rather than four. Aside from the 3D convolutional layer, 3D batch normalization layer, ReLU, and other necessary components generalizable from 2D to 3D, the 3D max-pooling layer was also incorporated into the architecture, with the kernel size of two and stride two. The dropout rate was chosen to be 0.3 to minimize the overfitting issue. The model output had the size of four, corresponding to the background class and the three HC subregions in the K-Y data. Experiments were conducted by changing only one variable at a time, while other variables were held constant as default. The default choices of all variables were displayed below in table 1, where standard deviation was abbreviated to "std".

Table 1. Default choice of variables (achieved dice score of 0.8543 and accuracy of 99.1574%).

	Model architecture	Loss function	Sampling	Model group number	Learning rate
Default	U-Net	Equal weight	Gaussian (std=1)	1	0.001 (fixed)

3.2. Comparison of results

3.2.1. Changing model architecture. Based on the results below in table 2, despite the similarity in accuracy, the U-Net++ model still outperformed the traditional U-Net in HC segmentation task, achieving a dice score of 0.8555. This is not surprising, given that the U-Net++ has nested convolutional layers and dense skip connections, and by bridging the sematic gap between the encoder's and decoder's feature maps, it is able to achieve superior segmentation result.

Table 2. Comparison of experimental results with different models.

	3D U-Net	3D U-Net++
Dice score	0.8543	0.8555
Accuracy (%)	99.1574	99.1576

3.2.2. Changing loss function weights. From table 3 below, the model with the class-weight loss function achieved the highest dice and accuracy, while the one with random-weight loss function performed the worst among the three. This was expected since the former considered the class distribution within the data and adjusted the sampling strategy accordingly. On the other hand, the fact that equal-weight loss function outperformed the random-weight indicated that reducing background weight alone while treating other classes equally may not be sufficient. It did not lead to model improvement, and the output was worse than treating all four classes equally. Only when both the background class was under-sampled, and the sampling of HC subfields was adjusted based on their respective proportion, would the segmentation result be optimal.

Table 3. Comparison of experimental results with loss function weights changing.

	Equal weight	Class weight	Random weight
Dice score	0.854	0.857	0.849
Accuracy (%)	99.157	99.159	99.072

3.2.3. Changing sampling method. The results in table 4 below showed that normal sampling multiplier with mean of 0 and standard deviation of 1 appeared to have the highest dice score - outperforming the uniform sampling with interval [0, 1] - while normal sampling with the same mean and standard deviation of 0.3 obtained the highest accuracy. This suggested that normal sampling may be more suitable for determining the multiplier on HC data than uniform sampling. Moreover, regarding normal sampling, as standard deviation alone decreased from 5 to 0.1, the dice score first increased then decreased. This seemed interesting, since it's worth wondering if there existed a quadratic relationship between standard deviation and dice score in case of normal sampling.

Table 4. Comparison of experiments with sampling method changing.

	Unif(0,1)[a]	N(5)[b]	N(2)	N(1)	N(0.3)	N(0.1)
Dice score	0.846	0.84	0.847	0.854	0.849	0.822
Accuracy (%)	99.136	99.093	99.172	99.157	99.208	98.984

[a] Unif stands for uniform sampling. The first and second number within the parenthesis specify the interval associated with it.

[b] N stands for normal sampling. The number within the parenthesis is the standard deviation.

3.2.4. Changing model group. The results in table 5 below suggested that the performances of model with group of one and four were similar with negligible difference. In other words, the way the inputs were connected to the outputs barely had any effect in HC segmentation result. Nevertheless, the accuracy somehow improved as the model group number increased.

Table 5. Comparison of trials with model group changing.

	Model group of 1	Model group of 4
Dice score	0.854	0.854
Accuracy (%)	99.157	99.161

3.3.5. Changing learning rate type. From table 6 below, it could be seen that compared to the fixed learning rate, the step learning rate scheduler alone was not sufficient in boosting the model performance – even led to a decrease in dice score. This dip may be explained by the scheme of continuously decreasing half the learning rate that eventually it became so small that the model learned really slow to the point of stagnation. In this light, an exponentially decaying learning rate scheduler may be a more proper choice since it would enable the learning rate to decrease drastically in the beginning and asymptotically slow in the end.

Table 6. Comparison of results for fixed learning rate and learning-rate scheduler.

	Fix learning rate (0.001)	StepLR (step=1, gamma=0.5)
Dice score	0.854	0.851
Accuracy (%)	99.157	99.158

3.3. Visualization of segmentation result

For purpose of illustration, only the segmentation result of the default parameter (and hyperparameter) choices for one exemplary subject, the subject one with left label, was displayed. Juxtaposed with it were the input image and ground-truth label associated with it, as shown below in figure 2. It could be seen that by and large, the segmentation result by 3D U-Net was quite similar to the ground truth, except that the top right blue area within the label differed a little. This may be that the HC subregion in blue was the most scarce class of all; consequently, using it to train the model and generate predictive results accordingly tends to be more difficult than other classes. A closer examination on its individual class dice score revealed that the class 3 (the blue subfield) had the lowest dice score of 0.7420 compared to others, suggesting that the model segmented the blue subregion least well.

4. Conclusion

In a nutshell, a series of ablation studies was conducted on the Kulaga-Yoskovitz (K-Y) data set using the U-Net-based models. According to the results obtained, the U-Net++ model outperformed the U-Net, and the model with normalized class-weight loss function obtained the best result in dice score and accuracy. Concerning sampling method, the model using normal sampling with mean of zero and standard deviation of 1 has the highest dice score, while the model using normal sampling with standard deviation of 0.3 has the highest accuracy. Interestingly, there seems to be a quadratic relationship between dice score and standard deviation in normal sampling. If further studied, it would help pinpoint the exact standard deviation value that leads to the highest dice score. Furthermore, model group number and learning rate type appear to have only negligible impact on model performance. Compared to some other literature, the performance metrics obtained herein may not be the highest given the use of simplified model architecture; nevertheless, it is adequate for the purpose of this study since it is mostly the comparative difference between experimental results that matters.

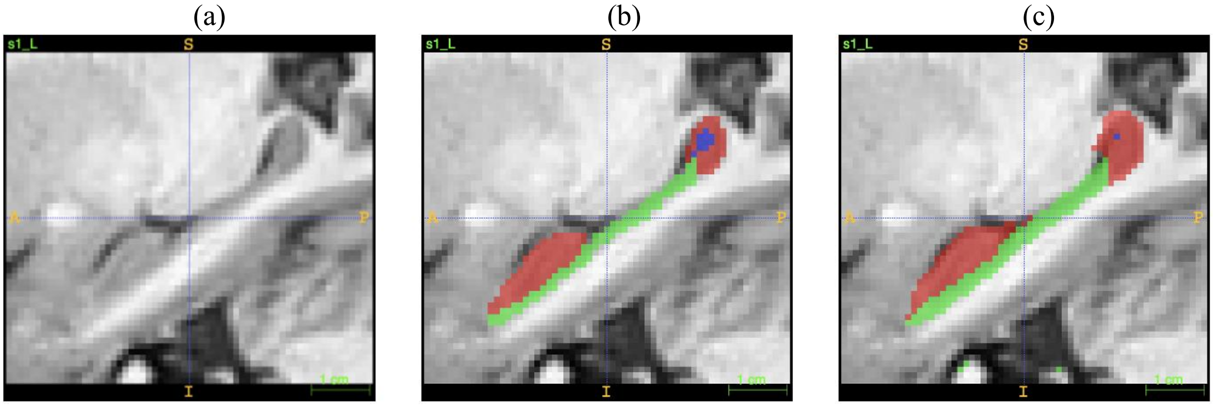

Figure 2. (a) Visualization of the input image of K-Y data, subject one left, (b) with ground-truth labels, and (c) with the automatic segmentation result using the 3D U-Net. All images rendered by ITK-SNAP v. 3.6.0, 2017.

While this paper focuses primarily on the effect of adjusting different model parameters (and hyperparameters) of 3D U-Net-based models in the class-imbalanced setting, several things were not considered, one of which being the cost-sensitive analysis, for instance, as implied by Lopez et al [2]. For future work, it would be worthwhile examining how incorporating this argument into the study would affect the segmentation result. Moreover, some other variable choices can also be tested on the K-Y data for further improvement on segmentation result, including but not limited to, other loss functions such as Jaccard, focal, Tversky loss, other U-Net variants with the ResNet backbone and attention mechanism, or adapting other approaches for addressing class imbalance such as Synthetic Minority Over-Sampling Technique (SMOTE) [5], balanced group softmax (BAGS) [6], and so on.

References

[1] Kulaga-Yoskovitz J et al 2015 Multi-Contrast Submillimetric 3 Tesla Hippocampal Subfield Segmentation Protocol and Dataset *Sci. Data* **2** 150059

[2] López V, Fernández A, García S, Palade V and Herrera F 2013 An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics *Information Sciences* **250** 113-141

[3] Manjón J, Romero J and Coupe P 2022 A Novel Deep Learning Based Hippocampus Subfield Segmentation Method *Scientific Reports* **12** 1333

[4] Zhou Z, Siddiquee M, Tajbakhsh N and Liang J 2018 UNet++: A Nested U-Net Architecture for Medical Image Segmentation *Deep learning in medical image analysis and multimodal learning for clinical decision support* (Springer) pp 3-11

[5] Chawla N, Bowyer K, Hall L and Kegelmeyer W 2002 SMOTE: synthetic minority over-sampling technique *Journal of artificial intelligence research* **16** pp 321-357

[6] Li Y et al 2020 Overcoming Classifier Imbalance for Long-tail Object Detection with Balanced Group Softmax *Conference on Computer Vision and Pattern Recognition (CVPR)* pp 10988-10997