ACU-NET: A FAST AND EFFICIENT NEURAL NETWORK FOR WHITE MATTER HYPERINTENSITY SEGMENTATION

A PREPRINT

Tianyu Ding
Department of Biostatistics
University of Pittsburgh
Pittsburgh, PA 15260
tid16@pitt.edu

Robert Krafty
Department of Biostatistics and Bioinformatics
Emory University
Atlanta, GA 30322
rkrafty@emory.edu

Dana L. Tudorascu
Department of Psychiatry and Biostatistics
University of Pittsburgh
Pittsburgh, PA 15213
dlt30@pitt.edu

May 6, 2021

ABSTRACT

In this paper we developed ACU-Net, an efficient convolutional neural network for medical image segmentation. The proposed deep learning network overcomes the small sample size problem of training a deep neural network when used for medical image segmentation. It also decreases computation cost by increasing the effective degree of freedom through data augmentation and the novel use of convolutional layers blocks to compress the model. We show that ACU-Net can achieve competitive performance while dramatically decreases the computation cost compared with other modern popular CNNs used for biomedical image segmentation.

Our approach is based on U-Net, a famous biomedical image segmentation neural network. We built convolutional blocks by adopting depth-wise convolution, inverted residual blocks and SE(Squeeze-and-Excitation) mechanism to U-Net. To compress the heavy structure of U-Net, we proposed an 'L' shaped neural network that focuses mainly on encoder part. We evaluate the trade-offs between model performance and computation cost on two white matter hyperintensity datasets through (1) regular segmentation accuracy metrics; (2) number of floating point operations per second (FLOPs).

1 Introduction

In 1998, "Lenet-5", one of the earliest convolutional neural networks, was proposed in [1] and achieved great success for the classification of handwritten numbers on MNIST set. However, due to hardware limitations, deep neural networks (DNN) did not attract widespread attention until 2012 when AlexNet [2] won the ImageNet competition by an 11% margin. After that, both academic and IT industrial researchers developed multiple well-validated deep networks of computer vision such as Visual Geometry Group (VGG) in [3], GoogLeNet in [4], and ResNet in [5]. Building upon these established deep network architectures, researchers developed more and more networks for specific areas or tasks. U-Net was developed in [6] and provides a practical deep network on training data with relative small sample size (i.e. 30 medical images and 512×512 pixels).

Recently, researchers have focused on building compact deep neural networks not limited to CNN. [7] proposed a Structured Sparsity Learning (SSL) method to regularize the structures of DNNs by introducing sparse group lasso regularization, both filter-wise and shape-wise. [8] assumed weight filters to be both low-rank and sparse, and split the weight matrix into the sum of a low-rank matrix and a sparse matrix, then applied several of thefamous networks

listed above. [9] proposed DeepTwist, a technique to compress CNNs by low-rank approximation to injected noise into weights. [10] proposed T-Net, a parametrizing fully convolutional network with a single high-order tensor that is different from previous layer-by-layer tensorization. Compared with the popular dropout technique in [11], which shrunk DNNs by randomly dropping units (along with their connections) from the neural network during training, low-rank approximation and sparsity regularization provide a more interpretable approach for dimension reduction and feature selection.

In state-of-the-art biomedical image segmentation deep neural models, U-Net [6] is the most famous and well-validated structure. Recent deep neural networks for biomedical image segmentation frequently use U-Net as basic structure or for comparison. Our work is also inspired by U-Net. In addition, among most recent researches on compression of deep neural networks, depth-wise separable convolutions [12], inverted residual block [13] and squeeze-and-excitation networks [14] are proved to be very useful and popular. Thus, we proposed ACU-NET, an asymmetric compact U-Net by applying the depth-wise separable convolutions in an inverted residual block with squeeze-and-excitation to convolutional layers.

This work describes the ACU-Net model in order to deliver the next generation of high accuracy efficient networks to improve biomedical imaging segmentation tasks by reducing computation cost while maintaining predictive performance. This could enable the segmentation tasks to even be performed on mobile devices in the future.

The goal of this Chapter is to optimize the trade off between accuracy and model size. To realize this we have introduced: (1) an efficient convolutional layer block design and (2) a new network architecture. We presented experiments on two white matter hyperintensity datasets: (1) a private normal aging cohort; (2) a public white matter hyperintensity segmentation challenge [15] to demonstrate the breakthrough efficacy of ACU-Net.

2 Method

First, we introduce U-Net architecture, which is illustrated in Fig 1. It is mainly established with:

- **Convolutional layers with ReLU (Rectified Linear Unit)** i.e. $f(x) = X^+ = max(0, x)$. Convolutional operation is sliding a convolutional filter over an input feature map. The output feature map is built by the dot products between the filters and input feature map.
- **Max-pooling layers** are operating independently on every depth slice of the input feature map and resizes it spatially, using the max function. These layers are often used to decrease the size of the input feature map.
- **Up-convolutional layers** are doubling the input feature map size.
- **Sigmoid activation function** i.e. $f(x) = 1/(1 + e^{-x})$ is used in the last fully connected layer to create an output probability map.

Second, U-Net has symmetric architecture, in each convolutional layers block, the last convolutional layer is cropped and copied to corresponding up-convolutional layer block in decoder. U-Net has following properties: (1) In each convolutional layer block, it includes two convolutional layers followed by a max-pooling layer to halve the input feature maps dimension and then, double the number of channels. For example, the last feature map of first convolutional layer block is $568 \times 568 \times 64$ which is corresponding to $H(height) \times W(width) \times C(channel)$, then after a max-pooling layer, the feature map becomes $284 \times 284 \times 64$. Next, a convolutional step makes this feature map become $284 \times 284 \times 128$. (2) In the decoder part of U-Net, which is the right part of the U-Net architecture, it is symmetric to its corresponding encoder part. Thus it costs similar even higher computation cost compared with its corresponding encoder part as it concatenates the encoder part at the beginning of each decoder block.

2.1 ACU-Net convolutional layer block

Although U-Net is effective in biomedical imaging segmentation, it is "overweight" compared with modern compact models. To compress U-Net while maintaining its capacity, we have developed ACU-Net. Before we demonstrate ACU-Net architecture, we first introduce several techniques we have used to build ACU-Net convolutional layers block.

Depthwise separable convolution Depthwise separable convolution was proposed in [12] and described in Fig 2. The classic convolutional filters, for example, with filter size $D_K \times D_K$, input channel number as M, output channel number as N in Fig 2.(a) has been decomposed into two parts: depthwise convolutional filters in (b) and pointwise convolutional filters in (c). It is called "depthwise" because this technique first looks at each channel as shown in

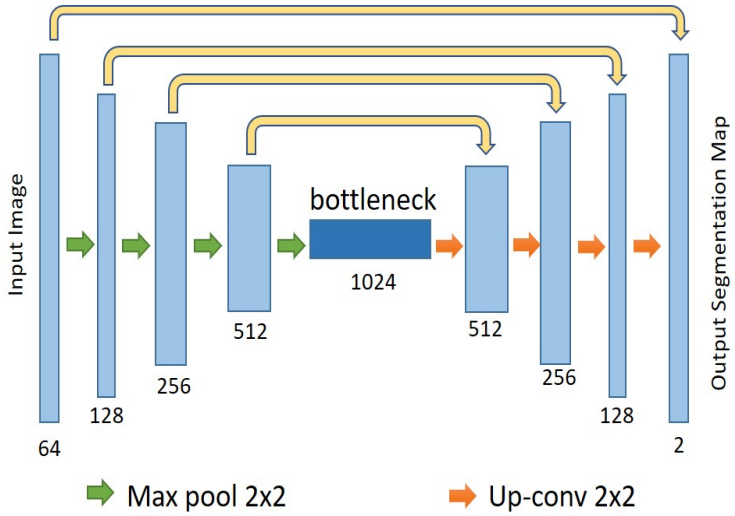

Figure 1: U-Net architecture. Adapted from 'U-Net: Convolutional Networks for Biomedical Image Segmentation,' by O.Ronneberger, P.Fischer and T.Brox, 2015, International Conference on Medical image computing and computer-assisted intervention, p.234–241.

(b), which is similar as decomposing a length M channel tensor into M length 1 tensor. Thus, this step generates a temporary output feature map with dimension $D_G \times D_G \times M$ where D_G is the spatial width and height of a square output feature map (for simplicity of illustration, we use square feature map here). Next, in order to transform the channel number from M to desired N, pointwise convolutional filters in (c) are used as 1×1 convolutional filters to transform channel numbers to desirable ones. As described in [12], depthwise separable convolution can get a reduction in computation of $\frac{1}{N} + \frac{1}{D_k^2}$. Using 3×3 convolutional layer in U-Net as example, this technique leads to around $\frac{1}{8}$ computation cost compared with the classic convolutional filters.

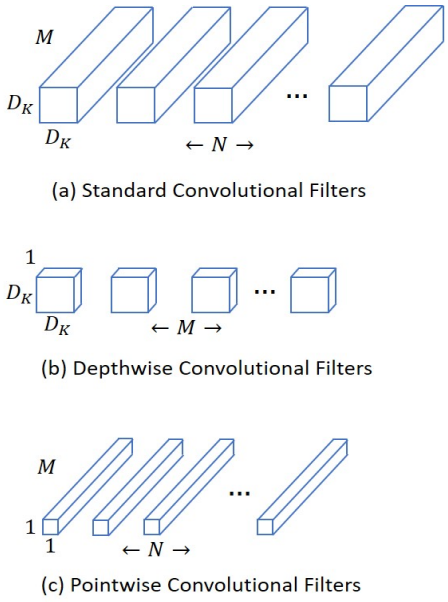

Figure 2: Depthwise convolutional vs. standard convolutional
The classic convolution filters in (a) have been decomposed to depthwise convolution in (b) and pointwise convolution in (c).

Inverted residual with linear bottleneck Inverted residual with linear bottleneck was proposed in [13] and described in Fig 3. In deep neural network research, there is a notorious degradation problem: with more stacked layers to a deep model, the acuracy becomes saturated and then degrades rapidly. To solve this problem and create deeper models

with promising accuracy, [5] proposed ResNet. ResNet includes residual learning blocks to learn residual of desired underlying feature mapping instead of the feature mapping itself, then adds the input feature map to the end of the block. This residual block dramatically relieves the degradation problem that leads to deeper and deeper networks such as ResNet152, which included 152 layers. Inverted residual builds a similar residual block with bottleneck compared with ResNet. The difference is that residual blocks in ResNet are connecting two layers with higher number of channels while inverted residual blocks are connecting two bottleneck layers with low number of channels. Thus, residual blocks have an hourglass-shape while inverted residual blocks are spindle-shaped. The intuition of inverted residual block is: non-linear function such as ReLU does not work well in low-dimensional space compared with linear functions [13]. Instead of connecting two high-dimensional layers, inverted residual blocks are connecting two low-dimensional linear bottleneck layers while the intermediate high-dimensional expansion layers are more efficient to use non-linear activation functions for information retrieval. With this inverted residual block, deep neural networks can be deeper without explosion on number of parameters and relieve the degradation problem.

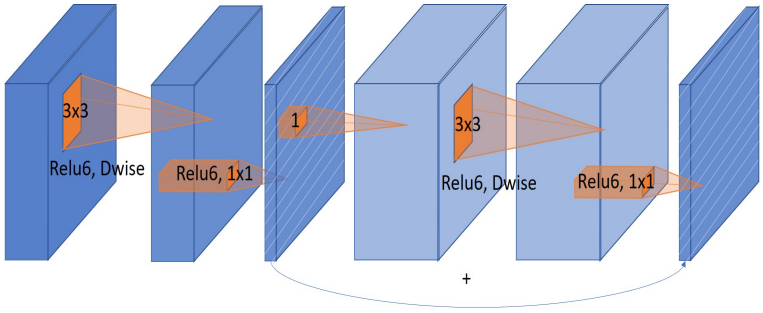

Figure 3: Inverted residual block with bottleneck layer
The inverted residual block inserts a bottle neck layer (diagonally batched layers) between pointwise convolutional layers and output feature map. Then, a inverted residual block is considered as components between two bottleneck layers shown with last 4 layers.

Squeeze-and-Excitation(SE) Squeeze-and-Excitation(SE) was proposed in [14] and is described in Fig 4. SE is a powerful tool to build a unit to recalibrate any feature maps. The goal of SE is to selectively emphasize informative features and suppress less useful ones. In Fig 4, an input feature map X with dimension $H \times W \times C$ is passed to a transformation operation F_{tr} and generates an output feature map U with dimension $H^{'} \times W^{'} \times C^{'}$. Then, a unit built by SE is described in the following steps:

1. U has been squeezed channel-wise by F_{sq}, i.e. calculate the mean of each $H^{'} \times W^{'}$ feature map which resulted in a $1 \times 1 \times C^{'}$ tensor;
2. the squeezed feature map is passed to a self-gating function F_{ex} i.e. a sigmoid activation $s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2\delta(W_1 z))$, where δ refers to the ReLU function, $W_1 \in \Re^{\frac{C'}{r} \times C'}$ and $W_2 \in C' \times \Re^{\frac{C'}{r}}$. C' is the number of channel of U and r is a prespecified reduction ratio used to build the self-gating mechanism with detailed discussion in [14];
3. The final output of the block is obtained by rescaling U with the operation F_{scale} i.e. $\tilde{U} = F_{scale}(u_c, s_c) = s_c u_c$, where F_{scale} refers to channel-wise multiplication between the scalar s_c from excitation operation and each channel-wise 2d feature map in U.

ACU-Net convolutional layer block is then built based on above techniques and described in Fig 5. Fig 5.(a) shows ACU-Net convolutional layer block without Squeeze-and-Excitation which is the same block built in [13]. Fig 5.(b) shows ACU-Net convolutional layer block with Squeeze-and-Excitation which is the same block built in [16].

2.2 ACU-Net architecture

ACU-Net architecture is established based on following two ideas to relieve heavy parameterization problem of U-Net to avoid overfitting. The first idea is *Light-Coder-and-Heavy-Bottleneck* and the second is Asymmetric-Auto-Encoder.

Light-Coder-and-Heavy-Bottleneck In original U-Net described in Fig 1, number of channels of encoders (left part of U-Net) and decoders (right part of U-Net) are doubled in next level layer block. This is a heavy design where the trade off between computation cost and accuracy might not be well-optimized. Ignited by MobileNetV2 in [13], low-dimensional bottleneck layer can well preserve the information. Thus, ACU-Net demonstrated in Fig 6 uses a light encoder and decoder design with much fewer channels compared with U-Net while still keep channel concatenation at the beginning of each decoder block.

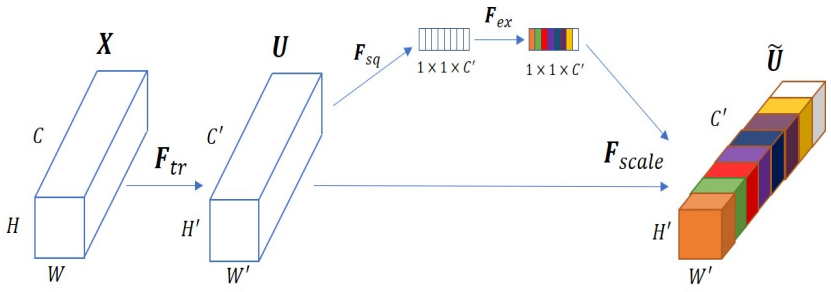

Figure 4: Squeeze-and-Excitation block

An output feature map U is first squeezed by a function F_{sq} and followed by an excitation operation with a self-gating function F_{ex}. Output weights from excitation will used to recalibrate U and generate final output feature map $\tilde{U}$ with operation F_{scale}.

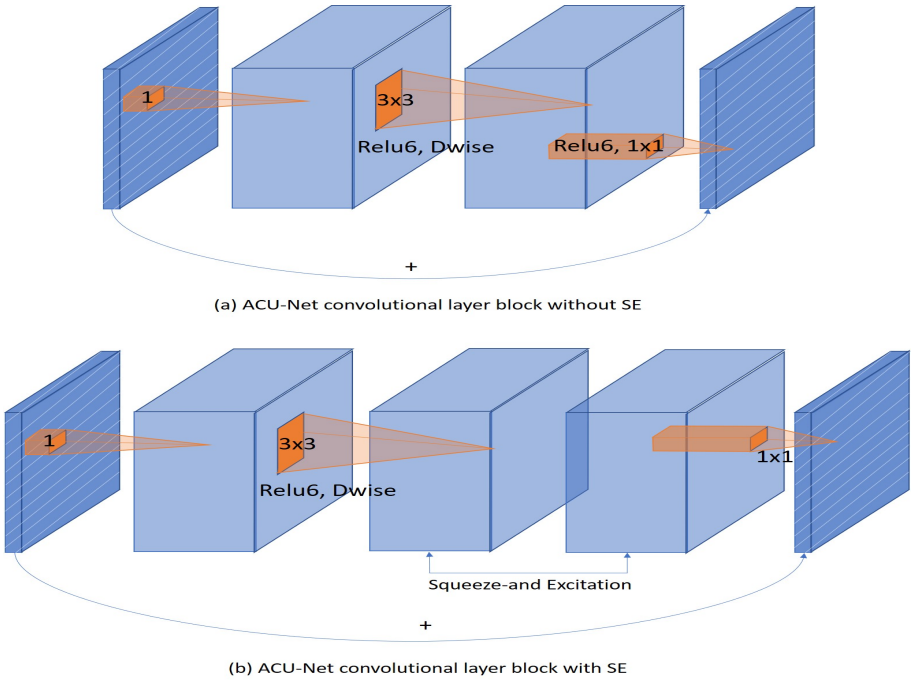

Figure 5: ACU-Net convolutional layer blocks

ACU-Net convolutional layer block without Squeeze-and-Excitation in (a) and with Squeeze-and-Excitation in (b).

Asymmetric-Auto-Encoder In U-Net, each decoder block has the same operation compared with its corresponding encoder block i.e. two convolutional layers operation. Although U-Net has a U-shape symmetric architecture, it is still in sequential order. The double convolutional layer operations in encoder block might be helpful for information retrieval while the corresponding decoder blocks with the same operation might not be able to keep the same efficacy compared with their encoder counterpart. Thus, decoder parts in ACU-Net in Fig 6 with green color have fewer operations compared with their corresponding encoder parts with blue color.

Before introducing the details of components of ACU-Net, we first introduce several definitions that are used in ACU-Net. *Batch Normalization* was proposed in [17], which was used to relieve internal covariate shift i.e. different inputs of each layers slowed down the training by requiring lower learning rates and careful parameter initialization. Batch Normalization (BN) normalizes a part of the model architecture and performing the normalization for each training mini-batch. BN is used in ACU-Net inverted block after each convolutional operation. *Hard swish* activation function is defined as: $h\text{-}swish(x) = x\frac{ReLU6(x+3)}{6}$ where $ReLU6(x) = min(max(x, 0), 6)$ is the clipped version of ReLU. This activation function is well validated in [16] to avoid gradient vanishing/exploration problem while reduce the number of memory accesses by used in deeper layers of the model. In Table 1, we list the details of layers in ACU-Net.

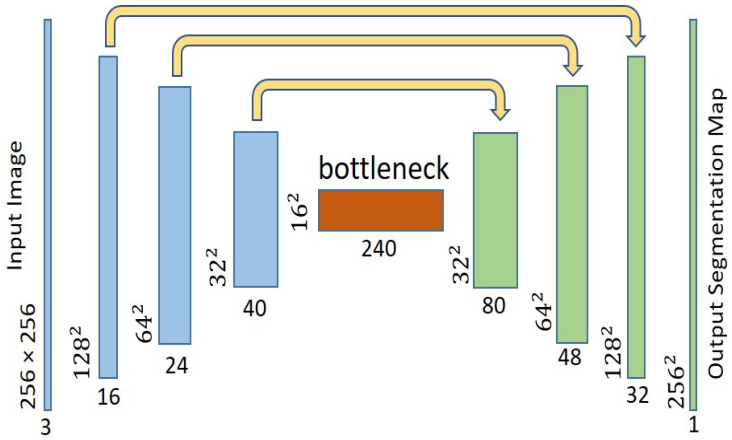

Figure 6: ACU-Net architecture

Table 1: Details for ACU-Net

Input	Operator	exp size	#out channel	SE	NL	s
Encoder						
$256^2 \times 3$	conv2d	-	16	-	HS	2
$128^2 \times 16$	InvRes-B, 3×3	16	16	-	RE	1
$128^2 \times 16$	InvRes-B, 3×3	64	24	-	RE	2
$64^2 \times 24$	InvRes-B, 3×3	72	24	-	RE	1
$64^2 \times 24$	InvRes-B, 5×5	96	40	$\surd$	HS	2
$32^2 \times 40$	InvRes-B, 5×5	240	40	$\surd$	HS	1
$32^2 \times 40$	InvRes-B, 5×5	240	40	$\surd$	HS	1
$32^2 \times 40$	InvRes-B, 5×5	240	240	$\surd$	HS	2
Decoder						
$16^2 \times 240$	Upconv2d	-	40	-	-	2
$32^2 \times 40$	Up-InvRes-B, 5×5	240	40	$\surd$	HS	1
$32^2 \times 40$	Upconv2d	-	24	-	-	2
$64^2 \times 24$	Up-InvRes-B, 5×5	72	24	$\surd$	HS	1
$64^2 \times 24$	Upconv2d	-	16	-	-	2
$128^2 \times 24$	Up-InvRes-B, 3×3	16	16	$\surd$	HS	1
$128^2 \times 16$	Upconv2d	-	16	-	-	2
$256^2 \times 16$	conv2d	-	1	-	Sig	1

exp size is expansion layer channel size in ACU-Net convolutional layer block. **InvRes-B**, 3×3 refers to ACU-Net convolutional layer block with 3×3 filter size. **Upconv2d** is up-convolutional layer as same as in U-Net to double the height and width of input feature map while change the number of channels in decoder part. **Up-InvRes-B** is operation which first concatenates encoder part to decoder then followed by InvRes-B operation. **SE** refers to whether uses Squeeze-and-Excitation in a speicific block. **NL** refers to non-linear activation function. **HS** refers use hard-swish acitivation function, **RE** refers to ReLU and **Sig** refers to Sigmoid. s refers to stride.

3 Experiments

In this section, we present our experimental results to show the effectiveness of ACU-Net. We report segmentation results on the ongoing normal aging study previously described in first project.

Our models are trained with data augmentation. As described in [18], data augmentation techniques have been widely used and validated in the application to medical image analysis to avoid over-fitting problem of heavy models. For an image object, data augmentation techniques includes: flipping, rotation, shearing, cropping and etc. In Fig 7, we show an example of data augmentation application to our medical image data. In addition, we use online data augmentation in training models. Compared with offline data augmentation which generates a fixed size of augmented dataset, online data augmentation generates an augmented training dataset in each training iteration step based on different augmentation settings. Thus, online data augmentation can generate infinite training samples if the training

iteration number grows. In practice, we include rotation, random horizontal flipping and scaling in our online data augmentation step which can generate augmented data with less heterogeneity.

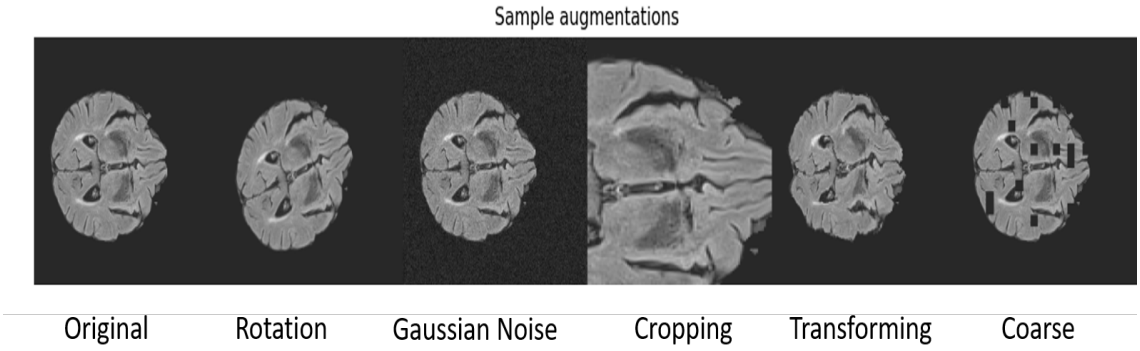

Figure 7: Data augmentation case study example

3.1 Normal aging dataset

Dataset We compared ACU-NET with U-Net on the same normal aging cohort used in [19]. Data included 20 subjects, which were randomly split into training (15 study participants) and testing (5 study participants) sets. Models were trained on training data set and compared in terms of their performance on the testing data set.

Training setup We trained our models on a 8GB GTX 1080 GPU. We use the standard Adam optimizer [20] with initial learning rate of 0.01. The mini-batch size is set to 15. We use dropout [11] with rate as 0.5 to last output layer.

Measurement setup Results were compared with manual segmentation performed by an experienced neuroradiologist, which provided the gold standard. The manual tracings of WMH were performed on 5 contiguous slices on the T2-FLAIR scans, the same for each subject. Models were compared in terms of the following metrics: (1) number of true positive voxels (TP), (2) number of false-positive voxels (FP), (3) number of true negative voxels (FN), and (4) number of false negative voxels (FN). We computed four additional combined metrics commonly used for prediction performance evaluation [21]: (1) accuracy, defined as ACC = (TP+TN)/ (TP+FP+FN+TN), (2) positive predictive value, defined as PPV = TP/(TP+FP); (3) true positive rate, defined as TPR = TP/(TP+FN), (4) false positive rate, defined as FPR = FP/(FP+TN), and (5) dice similarity coefficient, defined as DSC = 2TP/(2TP+FP+FN) as well as 95% confidence interval (CI) computed using bootstrap. We also included the receiver operating characteristic curve (ROC curve), the precision-recall curve (PRC), and the area under these two curves (AUC) [22].

In addition, we use number of parameters and FLOPs to measure the efficiency of models. FLOPs is the floating point operations which measures the complexity of the model.

Results The performance comparison is described in Table 2. We can find ACU-Net only loses 2% DSC on original testing dataset and 1% DSC augmented testing dataste while achieves around $1/20$ model size and $1/40$ complexity compared with U-Net-small.

Table 2: Performance comparison

	ACC	PPV	TPR	FPR	DSC	AUC	Params(M)	FLOPs(G)
U-Net	0.985(0.003)	0.876(0.037)	0.767(0.068)	0.005(0.001)	0.817(0.049)	0.881(0.034)	7.76	13.72
ACU-Net	0.985(0.002)	0.866(0.055)	0.753(0.092)	0.005(0.002)	0.801(0.06)	0.874(0.045)	**0.37**	**0.39**

3.2 White matter hyper-intensity challenge dataset

Dataset 60 MRI scans(T1 and FLAIR) from WMH 2017 Challenge [15] were used in this experiment. We have used 5-fold cross-validation to finetune the hyperparameters such as learning rate and batch size. The cross-validation is at subject level, not at slice level. ROBEX [23] is used for brain extraction. We used images after brain extraction as our model inputs.

Training setup Since our model is a model for 2D inputs, we expand a 3D MRI scan on axial view. For example, if a scan is with dimension $256 \times 256 \times 60$, we processed it as 60 slices with dimension 256×256. In addition, we have

applied following image processing steps sequentially: (1) cropping to remove background slices, (2) padding all slices to square image, (3) resizing all slices to 256×256 dimension and (4) intensity normalization.

Measurement setup Evaluation were be done according to the following metrics:DSC(Dice similarity coefficient), H95(modified Hausdorff distance, 95th percentile), AVD(Average volume difference in percentage), Recall(Sensitivity for individual lesions in percentage), F1 (F1 score for individual lesions) [15]. The testing dataset is only visible to the host of this challenge. The pretrained model had been sent to the host. They calculated the metrics value to ensure the results from over 50 models are comparable.

Results The performance comparison is described in Table 3. The testing dataset of WMH challenge were collected from multiple sites multiple MRI scanners. We can find ACU-Net output U-Net in every metric used in the open-source data challenge.

Table 3: Performance comparison

	DSC	H95	AVD	Recall	F1
U-Net	0.69	10.30	26.14	0.63	0.71
ACU-Net	0.69	**8.96**	28.99	**0.65**	0.65

4 Conclusions and future work

The proposed ACU-Net represents a novel compact convolutional neural network based on a well-validated architecture U-Net. The goal of ACU-Net is to build an efficient compact convolutional neural network for biomedical image segmentation. Thus, ACU-Net builds an inverted residual block with linear bottleneck and squeeze-and-excitation for convolutional layers block. In addition, ACU-Net builds a new asymmetric auto-encoder architecture with more weights on encoders part. This architecture decreases computation cost on decoders part while preserves the model performance. Compared with U-Net, ACU-Net focuses more on the information passing to bottleneck layer in the full architecture, thus, ACU-Net decreases the number of channels used in encoders and decoders part while keeps the high channel numbers in the bottom bottleneck layer. ACU-Net achieves competitive model performance compared with U-Net on a normal aging cohort WMH segmentation problem while decreases the model size and model complexity to 1/20 and 1/40 of U-Net respectively. This efficient structure of ACU-Net is favorable since modern CNNs require more and more computation resources while in many research environments, the computation resources are limiting. ACU-Net's compact model size enables researchers to train the model from scratch with their own data instead of using pre-trained models due to limited computation resources. It is even possible to move ACU-Net to mobile devices in the future since its convolutional layers block are based on blocks built in MobileNets which are designed for mobile devices.

Segmentation of WMHs is an important task with biological meanings. Accurate automatic segmentation of WMHs not only help physicians save time on manual tracing of neuroimages of patients, but also ensure the accuracy of automatic tissue segmentation of the brain by filling out WMHs with normal white matter tissues such that those WMH regions will not be wrongly classified as grey matter tissues. In addition, Deep learning approaches can provide an alternative to traditional machine learning models and we continue to investigate the added benefit of these techniques, including convolutional neural networks. So far, we have seen encouraging results, though much remains to be done in terms of increasing the sample size of the training data (not easy to achieve in low resource environments), performance (we have not yet matched OASIS-AD), interpretability (we would like to better understand what features of the data are actually contributing to improved prediction performance), and choices of the many tuning parameters (e.g., neighborhood size and filter types). Thus, it is still very challenging to adopt modern DNNs to medical image segmentation studies. Among those DNNs, U-Net might be the most doable DNN segmentation tool that can work with a small sample size of images. Based on U-Net, we built ACU-Net with modern compression techniques. ACU-Net can dramatically decrease the number of parameters and model complexity compared with U-Net while keep the similar model performance on WMH segmentation tasks.

For future works, instead of considering each 2D slice of a 3D brain as a single sample, proper directions to evolve this could be: (1) treat a single input as a 3D image, reconstruct the neural network for 3D image inputs; or (2) still use 2D image as an input, but build an ensemble model using each of axial, sagittal and cortical slices as one of the ensemble element.

References

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[3] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[4] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[7] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016.

[8] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379, 2017.

[9] Dongsoo Lee, Se Jung Kwon, Byeongwook Kim, and Gu-Yeon Wei. Learning low-rank approximation for cnns. *arXiv preprint arXiv:1905.10145*, 2019.

[10] Jean Kossaifi, Adrian Bulat, Georgios Tzimiropoulos, and Maja Pantic. T-net: Parametrizing fully convolutional nets with a single high-order tensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7822–7831, 2019.

[11] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[13] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[15] Hugo J. Kuijf, J. Matthijs Biesbroek, Jeroen De Bresser, Rutger Heinen, Simon Andermatt, Mariana Bento, Matt Berseth, Mikhail Belyaev, M. Jorge Cardoso, Adrià Casamitjana, D. Louis Collins, Mahsa Dadar, Achilleas Georgiou, Mohsen Ghafoorian, Dakai Jin, April Khademi, Jesse Knight, Hongwei Li, Xavier Lladó, Miguel Luna, Qaiser Mahmood, Richard McKinley, Alireza Mehrtash, Sébastien Ourselin, Bo-Yong Park, Hyunjin Park, Sang Hyun Park, Simon Pezold, Elodie Puybareau, Leticia Rittner, Carole H. Sudre, Sergi Valverde, Verónica Vilaplana, Roland Wiest, Yongchao Xu, Ziyue Xu, Guodong Zeng, Jianguo Zhang, Guoyan Zheng, Christopher Chen, Wiesje van der Flier, Frederik Barkhof, Max A. Viergever, and Geert Jan Biessels. Standardized assessment of automatic segmentation of white matter hyperintensities and results of the wmh segmentation challenge. *IEEE Transactions on Medical Imaging*, 38(11):2556–2568, 2019.

[16] A Howard, M Sandler, G Chu, LC Chen, B Chen, M Tan, W Wang, Y Zhu, R Pang, V Vasudevan, et al. Searching for mobilenetv3. arxiv 2019. *arXiv preprint arXiv:1905.02244*, 2019.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[18] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[19] T Ding, AD Cohen, EE O'Connor, HT Karim, A Crainiceanu, J Muschelli, O Lopez, WE Klunk, HJ Aizenstein, R Krafty, et al. An improved algorithm of white matter hyperintensity detection in elderly adults. *NeuroImage: Clinical*, 25:102151, 2020.

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[21] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European Conference on Information Retrieval*, pages 345–359. Springer, 2005.

[22] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

[23] Juan Eugenio Iglesias, Cheng-Yi Liu, Paul M Thompson, and Zhuowen Tu. Robust brain extraction across datasets and comparison with publicly available methods. *IEEE transactions on medical imaging*, 30(9):1617–1634, 2011.