Denoising of Amyloid PET Images Using U-net with L1-loss

Yu-Chen Lin
Department of Electrical Engineering
National Central University
Taoyuan, Taiwan
linyc.thomas@gmail.com

Meng-Hen Liu
Department of Electrical Engineering
National Central University
Taoyuan, Taiwan
rockliu1999@gmail.com

Jang-Zern Tsai
Department of Electrical Engineering
National Central University
Taoyuan, Taiwan
jztsai@ee.ncu.edu.tw

***Abstract*–**

***Purpose:* PET is a medical imaging process which records the location and quantity of photons emitted by the radioactive tracer injected within patients. Such continuous recording process takes around 70 minutes to finish. Long acquisition time of PET results in motion blur and other problems that downgrades image quality. We tried to transfer noisy images to higher quality images for more precise diagnosis.**

***Materials and Methods:* Three hundred and fifty-five amyloid PET datasets from Open Access Series of Imaging Studies (OASIS) [1] project were used in our experiment. As for the prediction method, 2D and 3D U-net [2] (one type of CNN architecture) and L1 loss (mean absolute error loss) were used. The purpose of the model is to map the given input to the corresponding target. Here, the input to the CNN was the PET in the 50th~55th min, while the target was the PET in the 50th~50th min. During training, it should minimize the L1 loss between the prediction and the target. In order to evaluate the quality of deep learning model's prediction. PSNR, SSIM and RMSE are calculated between input–target pairs and prediction–target pairs.**

***Results:* Given input images, predictions were synthesized and compared between two deep learning models. For 2D U-net, PSNR of input–target pair was 34.02dB while PSNR of prediction–target pair improved to 37.71dB. The same goes for other evaluation metric. SSIM improved from 0.945 to 0.9667, RMSE improved from 0.02015 to 0.0133. As for 3D U-net, PSNR improved from 34.02dB to 37.79dB, SSIM improved from 0.945 to 0.9676, RMSE improved from 0.02015 to 0.01344. All metrics showed improvement, while 3D U-net slightly outperformed 2D U-net.**

***Conclusion:* U-net with L1 loss is capable of learning the mapping from 50th~55th minutes measurement to 50th~70th minutes measurement for PET. Since we can synthesize 50th~70th minutes images based on 50th~55th minutes images, shorter acquisition time may be feasible. 3D U-net has similar performance compared with 2D U-net.**

***Keywords*–Deep learning, amyloidosis, PET, Alzheimer's disease, dementia, U-net, AV-45**

I. Introduction

Acquisition of high-quality images has always been an objective within many fields, it's no exception when it comes to medical imaging. It's especially crucial for the medical images to be as sharp as possible, given that they're one of the most important information of a precise diagnosis.

Of all different kinds of medical imaging, some may be harder to acquire then the others. Consider amyloid PET imaging, when a patient is subject to Florbetapir F18 tracer when taking a PET scan. It's more likely the subject has some indication for Alzheimer's disease (AD) [3]. Such that, further study needs to be conducted to track the progress of amyloid accumulation. One of the methods to track it is through amyloid PET scan. The short period segments from PET scans have to be aligned with each other to form so called frames before passing them to the doctors for diagnosis. However, PET scans usually take a considerably long time to finish (70 minutes in this study). Thus, unwanted dislocation needs to be corrected. Patients with AD are especially more likely to introduce more unwanted minor movements during the long acquisition of the image. These dislocation makes it harder for radiological technologist to relocate each segment of the record, which tends to result in noisier images.

Convolutional neural networks (CNNs) opens up different opportunities to solve many problems previous solutions cannot resolve due to the fact that they have the ability to learn the mapping from one domain to another. Thus, it's a great idea to apply it in some medical imaging problems.

In this study, we made hypothetical situation that 55th~70thmins image was absent or significantly distorted to the point that discarding it would be a better idea. There are two conditions to support this assumption. First, patients are more likely to have AD which tends to leads to more movements that downgrades the image quality. These movements hypothetically take place late at the acquisition. The other is again because of AD, patients may even leave the PET scanner in the middle of the acquisition, which results in missing data.

This is when CNNs comes in handy. In theory, after learning the mapping from 50~55mins single-time-frame images to their corresponding 50~70mins target images, predictions can be made given STF. Predictions need to be as close to target as possible comparing to STF.

II. Materials and Methods

This study utilized the amyloid PET images with Florbetapir F18 tracer granted by Open Access Series of Imaging Studies (OASIS). There are two acceptable procedures for obtaining the florbetapir F18 PET scans referred to OASIS-3 Data Dictionary from [1]. We choose to only use the data containing full 70-minute dynamic scan. Post-Processing method was also elaborated in OASIS-3 Data Dictionary.

A. PET Data Acquisition

All PET images scanned using Florbetapir F18 tracer was first downloaded then filtered with the following given condition. First, it needs to be exactly 26 time-frames, 23rd frame should be 50th~55thmins, 24th frame should be 55th~60thmins, 25th frame should be 60th~65thmins and 26th frame should be 65th~70thmins. Second, image volume should be 127 in z axis, 256 in y axis and 256 in x axis.

These requirements were decided because most of the data meets conditions from above. After filtering the original data pool, 355 sets of images were left. 325 sets of images were used for training, 30 sets were used for testing.

B. *Image Preprocessing*

Training pairs consisted of the 23rd single time-frame, STF23zyx, as the input and the average of the 23rd~26th frames, averageSTFzyx, as the target. The intensity of each voxel in each training pair was normalized according to the following formula:

$$z = \frac{y}{\max(\max STF23_{xyz}\,,\ \max averageSTF_{xyz}) * 0.5} - 1$$

, where y represents the original intensity of a voxel in the training pair, and z is the intensity of the voxel after the normalization.

After normalization, center cropping was conducted to crop out backgrounds which don’t contain any useful information. Padding was also conducted to form cube shaped images with length of sides to be the power of 2. By doing so, one can prevent mismatching between convolution layers during upsampling. This results in the final shape of 128 by 128 by 128, voxel value between -1 and 1 per image.

C. *CNN Implementation*

U-net [2] architecture is a commonly used solution in image to image synthesis problems. The architecture consists of “A contracting path to capture context and a symmetric expanding path that enables precise localization.” quote from U-net’s paper. From basic image segmentation to GAN’s variations such as pix2pix [4] to realizing amazing ideas such as noise2noise [5]. The main generator part of these architectures all based upon U-net. As such, it is reasonable for us to build upon this widely accepted architecture for image to image translation.

D. *Network architecture*

As mentioned above, U-net was used as base model. Since we’re dealing with 3D volume data. It’s intuitive for us to modify its each and every layer to accept and process 3D data. Other similar study [6] has used 2D layers model to conduct prediction on each x-y plane then stack them together. However, in fear of introducing unwanted discontinuity on Z axis. Also, the waste of not utilizing information between different sections on z axis. These two reasons make us choose to alter the original U-net to fit 3D data. Standard 2D model was also used in this experiment.

Since the publish of batch normalization [7] in 2015, this method has been widely used in different kinds of model due to its ability to normalize different feature. Since different features are on the same scale, their gradient will also be on the same scale. As such, we can stabilize the gradient descend even if we apply same learning rate to different feature. This is extremely helpful for models with many convolution layers. For reasons mentioned above, batch normalization was applied for faster and more stable convergence throughout the training process.

More detail of the model structure is elaborated in the appendix section. Including placement and number of convolution layers, pooling layers, activation function, batch normalization layers and contracting path.

For loss function, we choose L1 loss for less blurry effect. As such, the prime objective for the model is to minimize the L1 distance between prediction and target. Which can be expressed as:

$$\arg\min E_{STF23,averageSTF}[|averageSTF - f(STF23)|]$$

, where f denotes the trained model. Thus, f(STF23) is the predicted image.

As for optimizer, Adam was used with learning rate of 0.0002. Batch size was set to 1 because of the GPU memory limit. One hundred epochs were trained. 3D U-net’s training loss plateaued at around 8th epoch, 2D U-net’s training loss plateaued at around 8th epoch as well.

For 3D U-net, 128 by 128 by 128 STF23 images were passed in as input per subject, 128 by 128 by 128 averageSTF images were passed in as target per subject. As for 2D U-net, because each training pairs actually contains about 38 layers of useless axial view which don’t contain any information from the brain. So we decided not to pass in those as input. As such, nighty 128 by 128 STF23 images were passed in as input per subject, nighty 128 by 128 averageSTF images were passed in as target per subject.

E. *Evaluation*

Comparison were made with STF23 – averageSTF pairs versus prediction – averageSTF pairs.

Before the evaluation of image quality, every images were first rescaled from [-1,1] to [0,1], because the dynamic range of an image should not be negative. Assessment of image quality was conducted with peak signal-to-noise ratio (PSNR), structural similarity index (SSIM) [8] and root-mean-square error (RMSE). These evaluation metrics were calculated as follow, first calculate the mean-square error (MSE):

$$MSE = \frac{1}{x * y * z}\sum_{i=0}^{x-1}\sum_{j=0}^{y-1}\sum_{k=0}^{z-1}[img1(i,j,k) - img2(i,j,k)]^2$$

, where (x,y,z) is the shape of img1 and img2. While calculating MSE of STF23 – averageSTF pairs, img1 was STF23, img2 was averageSTF. While calculating MSE of prediction was averageSTF pairs, img1 was prediction, img2 was averageSTF. Then, we can calculate RMSE and PSNR with MSE:

$$\mathrm{RMSE} = \sqrt{MSE}$$

$$\mathrm{PSNR} = 20 * \log_{10}\frac{1}{\sqrt{MSE}}$$

, SSIM was calculated as follow:

$$\mathrm{SSIM} = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

, while calculating SSIM of STF23 – averageSTF pairs, x denotes STF23, y denotes averageSTF. While calculating SSIM of prediction – averageSTF pairs, x denotes prediction, y denotes averageSTF.

(μ_x= average of x, μ_y= average of y, σ_x= standard deviation of x, σ_y= standard deviation of y, σ_{xy}= covariance of x and y, c_1=0.01^2, c_2=0.03^2)

While calculating MSE, each and every voxel within every image that are below 0.01 were excluded, in order to exclude background voxels.

SSIM was calculated on every axial plane then averaged.

III. Results

For 2D U-net, PSNR of input–target pair is 34.02(dB) while PSNR of prediction–target pair improved to 37.71(dB). The same goes for other evaluation metric. SSIM improved from 0.945 to 0.9667, RMSE improved from 0.02015 to 0.0133. As for 3D U-net, PSNR improved from 34.02(dB) to 37.79(dB), SSIM improved from 0.945 to 0.9676, RMSE improved from 0.02015 to 0.01344. All metrics showed improvement. These values were picked from the best results within 0th~100th epoch.

TABLE I. Best SSIM, PSNR and RMSE of test dataset within 100 epochs

Metric	STF23–averageSTF	2D U-net prediction–averageSTF	3D U-net prediction–averageSTF
PSNR	34.02dB	37.71dB	37.79dB
SSIM	0.945	0.9667	0.9676
RMSE	0.02015	0.0133	0.01344

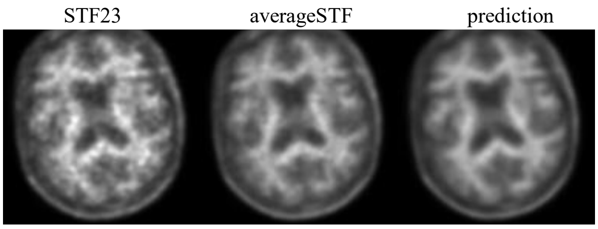

Fig. 1. Example of 2D U-net's 100th epoch, input (STF23), target (averageSTF) and prediction. Prediction was synthesized based on STF23, using 2D U-net.

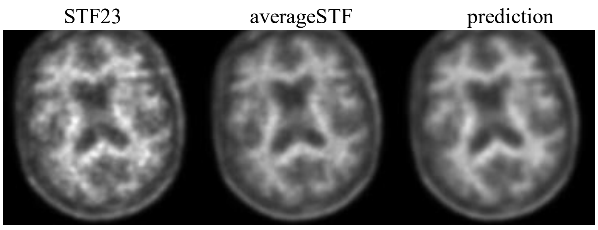

Fig. 2. Example of 3D U-net's 100th epoch, input (STF23), target (averageSTF) and prediction. Prediction was synthesized based on STF23, using 3D U-net.

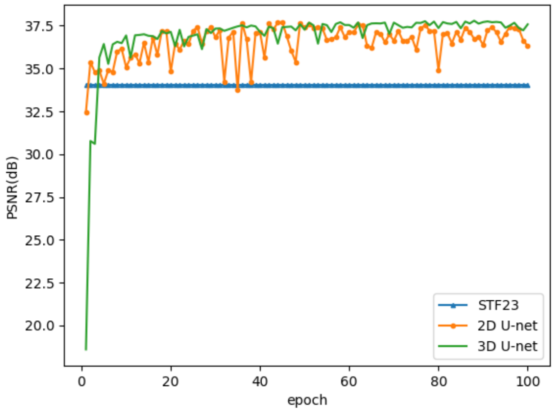

Fig. 3. During training, PSNR of STF23 – averageSTF pairs, prediction – averageSTF pairs generated by 2D U-net and prediction – averageSTF pairs generated by 3D U-net. (The lower, the better)

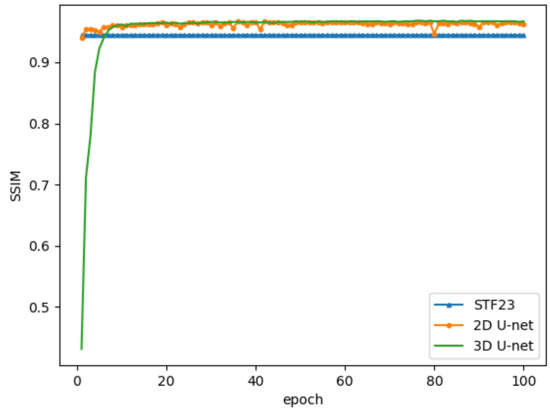

Fig. 4. During training, SSIM of STF23 – averageSTF pairs, prediction – averageSTF pairs generated by 2D U-net and prediction – averageSTF pairs generated by 3D U-net. (The lower, the better)

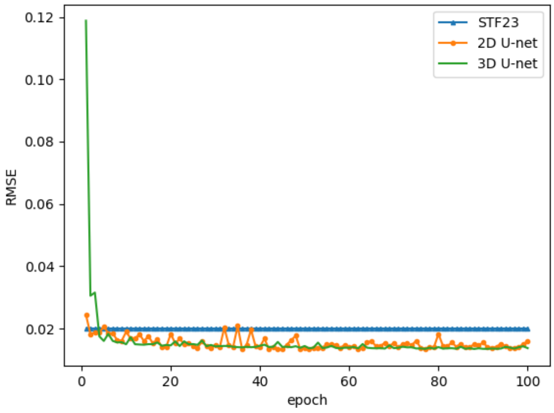

Fig. 5. During training, RMSE of STF23 – averageSTF pairs, prediction – averageSTF pairs generated by 2D U-net and prediction – averageSTF pairs generated by 3D U-net. (The lower, the better)

IV. Discussion

First of all, Prediction – averageSTF pairs all showed improvements comparing with STF23 – averageSTF pairs. Which shows that our model has indeed learned some mapping from input to output. Such that, by reducing the L1 distance from input to targets, it also improved in terms of PSNR, SSIM, RMSE.

From the results, we can see that 3D U-net showed similar performance compared with 2D U-net in terms of PSNR, SSIM and RMSE. 3D training contained 38 layers useless empty data on z axis, just to maintain cube shape with length of sides to be the power of 2 for upscaling problem. 2D training didn't have direct access of information between layers on the z axis and have significant less trainable parameters.

In terms of the limitation of almost every deep learning methods, including this one. Deep learning models in general only knows how to transfer data from one learned domain to another. Which means, we can't apply this model and its weight trained from one dataset to predict data from other dataset that is not similar, and expect it to give out result that makes sense. For example, if your PET image was acquired using other tracer, selected other time frame, time frames have different length or having different resolution comparing to the dataset the model was trained on. Since your data's domain is not the same as the dataset used for training. Conducting convolutions with data that is not acknowledged by the model, the trained model won't be able to find features learned from known data. If not entirely, these features will be somewhat distorted for the model. Which tends to leads to worse result.

In summary, you can't train one model and expect it to fit into every situation, unless your data meets every situation.

Similarly, you can't feed the output of your trained model back into the input and expect it to give out better result. Again, because the input is not from a known domain by the trained model.

If you pay close attention to predictions in Fig. 1. And Fig. 2., you may notice some blurriness. Which is a normal result due to the nature of L1 loss. Some study utilized simultaneously training another CNN neural network called discriminator as part of main model's loss function to synthesize images with more contrast and sharper details, for example, pix2pix. These experiments resulted in remarkable outcome. However, balancing the main model (also called generator in pix2pix) and discriminator is really hard. Often the result will be one model's loss descend while the other doesn't, which leads to one of the model's loss not being able to descend. So, unless you can carefully plan out all the parameters for a successful training. It's easier and more stable to train without discriminator model.

Further discussion should be conducted with professionals, whether these predictions help improve the confidence for diagnosis comparing to only inspecting STF23. Or the model even synthesized features or details can't be observed if only provided with STF23.

V. Conclusion

In this study, we tested the capability of using convolutional neural network (CNN) to synthesize full acquisition time (50^{th}~70^{th} minute) PET images with one fourth acquisition time (50^{th}~55^{th} minute) PET images. This way, when only part of the acquisition is valid, we can still synthesize images that has quality similar to full acquisition time.

Three evaluation metrics, PSNR, SSIM and RMSE were calculated for the comparison between input–target pairs and prediction–target pairs. Our experiment results showed improvement of prediction–target pairs over input–target pairs, in terms of PSNR, SSIM and RMSE. Thus, the CNN model we used in this study indeed has the capability to synthesize full acquisition time images with part of the acquisition.

For future study, whether it's a better idea to train a model that passes in the whole 3D volume data as input, or passes 2D slices one by one as input should be further evaluated. Judging from PSNR, SSIM and RMSE, using 2D kernels and 3D kernels both showed similar performance. Also, it's important to note that, discussions with professionals from department of nuclear medicine should be conducted to show more insight about the details of the images that these models synthesized.

References

[1] P. J.LaMontagne *et al.*, "OASIS-3: Longitudinal Neuroimaging, Clinical, and Cognitive Dataset for Normal Aging and Alzheimer Disease," *medRxiv*, p. 2019.12.13.19014902, Jan.2019, doi: 10.1101/2019.12.13.19014902.

[2] O.Ronneberger, P.Fischer, andT.Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation." 2015.

[3] H.Lan, A. W.Toga, andF.Sepehrband, "SC-GAN: 3D self-attention conditional GAN with spectral normalization for multi-modal neuroimaging synthesis," *bioRxiv*, 2020, doi: 10.1101/2020.06.09.143297.

[4] P.Isola, J.-Y.Zhu, T.Zhou, andA. A.Efros, "Image-to-Image Translation with Conditional Adversarial Networks." 2018.

[5] J.Lehtinen *et al.*, "Noise2Noise: Learning Image Restoration without Clean Data." 2018.

[6] K. T.Chen *et al.*, "Ultra–Low-Dose 18F-Florbetaben Amyloid PET Imaging Using Deep Learning with Multi-Contrast MRI Inputs," *Radiology*, vol. 290, no. 3, pp. 649–656, 2019, doi: 10.1148/radiol.2018180940.

[7] S.Ioffe andC.Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." 2015.

[8] Z.Wang, A. C.Bovik, H. R.Sheikh, andE. P.Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004, doi: 10.1109/TIP.2003.819861.

Appendix

TABLE II. Network architecture used in our experiment. All convolutions use padding mode “same”, kernel-size=3, strides=1. All upsampling repeats 1st, 2nd, 3rd dimensions of the data by 2. For 2D U-net, replace every layers with 2×2 instead of 3×3.

NAME	N_{out}	FUNCTION
INPUT	1	
CONV3D	16	Convolution 3x3
BATCH_NORMALIZATION	16	Batchnorm
RE_LU	16	ReLU activation
CONV3D_1	16	Convolution 3x3
BATCH_NORMALIZATION	16	Batchnorm
RE_LU_1	16	ReLU activation
MAX_POOLING3D	16	Maxpool 3x3
CONV3D_2	32	Convolution 3x3
BATCH_NORMALIZATION_2	32	Batchnorm
RE_LU_2	32	ReLU activation
CONV3D_3	32	Convolution 3x3
BATCH_NORMALIZATION_3	32	Batchnorm
RE_LU_3	32	ReLU activation
CONV3D_4	32	Convolution 3x3
BATCH_NORMALIZATION_4	32	Batchnorm
RE_LU_4	32	ReLU activation
MAX_POOLING3D_1	32	Maxpool 3x3
CONV3D_5	64	Convolution 3x3
BATCH_NORMALIZATION_5	64	Batchnorm
RE_LU_5	64	ReLU activation
CONV3D_6	64	Convolution 3x3
BATCH_NORMALIZATION_6	64	Batchnorm
RE_LU_6	64	ReLU activation
CONV3D_7	64	Convolution 3x3
BATCH_NORMALIZATION_7	64	Batchnorm
RE_LU_7	64	ReLU activation
MAX_POOLING3D_2	64	Maxpool 3x3
CONV3D_8	128	Convolution 3x3
BATCH_NORMALIZATION_8	128	Batchnorm
RE_LU_8	128	ReLU activation
CONV3D_9	128	Convolution 3x3
BATCH_NORMALIZATION_9	128	Batchnorm
RE_LU_9	128	ReLU activation
CONV3D_10	128	Convolution 3x3
BATCH_NORMALIZATION_10	128	Batchnorm
RE_LU_10	128	ReLU activation
UP_SAMPLING3D	128	Upsample
CONCATENATE	192	Concatenate output of RE_LU_7
CONV3D_11	192	Convolution 3x3
BATCH_NORMALIZATION_11	192	Batchnorm
RE_LU_11	192	ReLU activation
CONV3D_12	64	Convolution 3x3
BATCH_NORMALIZATION_12	64	Batchnorm
RE_LU_12	64	ReLU activation
CONV3D_13	64	Convolution 3x3
BATCH_NORMALIZATION_13	64	Batchnorm
RE_LU_13	64	ReLU activation
UP_SAMPLING3D_1	64	Upsample
CONCATENATE_1	96	Concatenate output of RE_LU_4
CONV3D_14	96	Convolution 3x3
BATCH_NORMALIZATION_14	96	Batchnorm
RE_LU_14	96	ReLU activation
CONV3D_15	32	Convolution 3x3
BATCH_NORMALIZATION_15	32	Batchnorm
RE_LU_15	32	ReLU activation
CONV3D_16	32	Convolution 3x3
BATCH_NORMALIZATION_16	32	Batchnorm
RE_LU_16	32	ReLU activation
UP_SAMPLING3D_2	32	Upsample
CONCATENATE_2	48	Concatenate output of RE_LU_1
CONV3D_17	48	Convolution 3x3
BATCH_NORMALIZATION_17	48	Batchnorm
RE_LU_17	48	ReLU activation
CONV3D_18	16	Convolution 3x3
BATCH_NORMALIZATION_18	16	Batchnorm
RE_LU_18	16	ReLU activation
CONV3D_19	16	Convolution 3x3
BATCH_NORMALIZATION_19	16	Batchnorm
RE_LU_19	16	ReLU activation
CONV3D_20	1	Convolution 3x3
TANH	1	Tanh activation

More comprehensive experiment method:
https://github.com/yuchen-lin/amyloid_PET_denoise