AI6103 Project: U-Net Regularizations

Liu Xuedi
G2202158B
LI0002DI@e.ntu.edu.sg

Pei Yitong
G2202385J
YPEI003@e.ntu.edu.sg

Ye Xinyi
G2202386F
XYE007@e.ntu.edu.sg

Zhang Yuxuan
G2203147K
YZHANG253@e.ntu.edu.sg

Zhou Yingquan
G2202643G
YZHOU046@e.ntu.edu.sg

Introduction

Over-fitting is a very common problem occurring in the field of deep learning. This situation happens when the models fit the training data too well to be generalized on other datasets[2], which limits the performance of the trained networks. Therefore, avoiding the over-fitting is necessary in deep learning studies.

To solve the problem of over-fitting, the scientists proposed a seires of methods, called Regularization, including L1 regularization, L2 regularization, Dropout, data augmentation, early stopping and ect. In this project, we want to discuss and prove how the regularization methods could be effctive to avoid the over-fitting.

We choose the U-Net as our baseline model because it is widely applied in biomedical image segmentation, which is a field lacking public grand datasets and therefore is suitable for generating over-fitting. We also choose two biomedical datasets and do a series of regularization experiments. The code of our project could be find at https://github.com/zoey-shan/AI6104_DL_Project_2022Fall. The contributions of every group member is listed below:

- Liu Xuedi: L1 and L2 regularization
- Pei Yitong: Early stopping regularization
- Ye Xinyi: Data augmentation regularization
- Zhang Yuxuan: Receptive field regularization
- Zhou Yingquan: Dropout regularization

U-Net Model

The U-shaped structure of the U-Net[3] is shown in the figure 1. The network is a classical full convolutional network. The input to the network is a 572*572 image with mirrored edges The left side of the network is a series of downsampling operations consisting of convolution and Max Pooling, resulting in a Feature Map of size 32*32.

The right-hand part of the network is called the extended path. Since the size of the Feature Map of the compressed path on the left and the expanded path on the right are different, U-Net is normalized by cropping the Feature Map of the compressed path to a Feature Map of the same size as the expanded path. Since the task is a binary task, the network has two output Feature Maps.

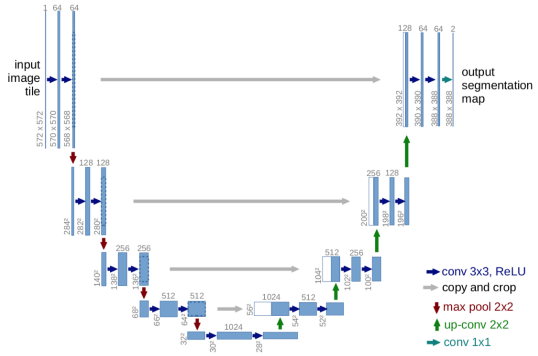

Figure 1: U-Net Architecture

Figure 2: An example from PhC Dataset. From left to right, the first image is the input microscope photo; the second is the segmentation with random positive integer cell labels; and the third is the final target binary mask.

Dataset

In the project, we experimented on two datasets: PhC-C2DH-U373 Cell Segmentation Dataset [7] and Kaggle Skin Lesion Segmentation Dataset [4].

PhC dataset consists of 1-channel black-and-white microscope photos and target segmentations. Both the training dataset and the challenge dataset have 228 samples. Specifically, as Figure 2 shows, in segmentation files, cells are randomly marked as positive integer indexes(shown as different colors with pyplot) while empty space is marked as 0. Due to the randomness of cell indexes, it is meaningless to transform the segmentation task into a multiclass classification task. That is, the actual task is a binary classification task that predicts whether, for each pixel (i, j), it is in a cell,

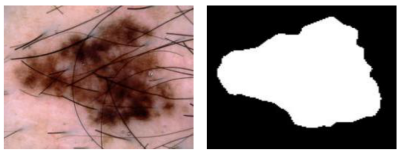

Figure 3: An example from Kaggle Skin Lesion Dataset. The left is the RGB skin lesion photo, and the right is the target mask.

$y_{i,j} > 0$, or in the empty space, $y_{i,j} = 0$.

Kaggle Skin Lesion Segmentation Dataset consists of 3-channel RGB photos and target segmentations. There are 2000 samples for training, 150 samples for validation, and 600 samples for final testing. In the target segmentation files, lesions are marked as 1 while the other parts are marked as 0. Thus, the segmentation task is also a binary classification task. Nevertheless, different from PhC, the skin lesion photos contain many noises, such as human hair, as Figure 3 shows. Moreover, the lesions themselves are in different colors as well. As the result, the segmentation task on this dataset is more complicated than that on Phc.

L1 and L2 regularisation

Method Overview

The main difference between L1 and L2 regularisation is the penalty term. Lasso Regression (L2 regularisation) adds the squared magnitude of the coefficients to the loss function as a penalty term, while Lasso Regression (L1 regularisation) adds the 'absolute magnitude' of the coefficients to the loss function as a penalty term.

Loss function with L1 regularisation is $Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} \|w_i\|$. And loss function with L2 regularisation:$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2$.

Experiments

On the PhC-C2DH-U373 dataset, the L1 and L2 regularization methods were used to tune the model parameters and train 5 epochs, and the variation of the model's training loss and validation dice are shown in Figure 4.

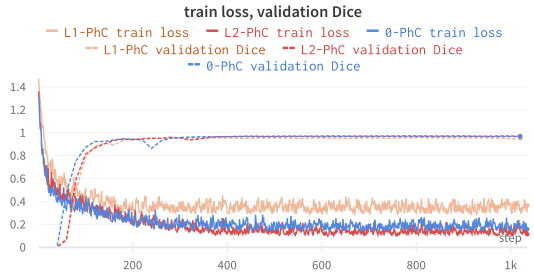

Figure 4: Train Loss and validation dice on PhC-C2DH-U373 dataset

It can be seen that with the addition of regularisation, the train loss of the model decreases more slowly, which is preventing the model from overfitting on the training set. However, due to the simplicity of the PhC dataset, the validation dice did not increase with the number of training sessions, and there was no overfitting. So the effect of regularisation is reflected in the slower rate of decline in train loss.

On the Skin-Lesion-Segmentation dataset, the L1 and L2 regularization methods were used to tune the model parameters and train 100 epochs. The variation of the model's training loss and validation loss are shown in Figure 5 and 6.

It can be seen that on the Skin dataset, the model without added regularisation shows an increase in validation loss after 40 epochs, while the train loss consistently decreases. This situation is due to the overfitting of the model as the number of training sessions increases. After adding the L1 and L2 regularisation methods, the overfitting of the model was reduced.

Figure 5: Train Loss on Skin-Lesion-Segmentation dataset

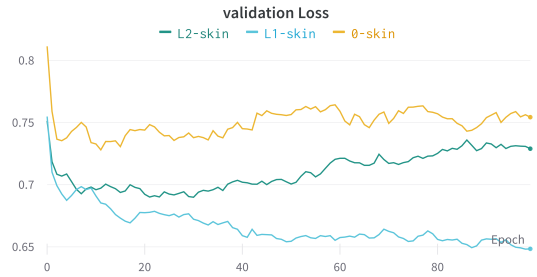

Figure 6: Validation Loss on Skin-Lesion-Segmentation dataset

The Train Loss, Validation Loss and Validation Dice values after regularisation using L1, L2 are shown in Table 1.

	PhC-C2DH-U373			Skin-Lesion-Segmentation		
	No regularisation	L1	L2	No regularisation	L1	L2
Traning Loss	0.3209	0.3392	0.1463	0.2564	0.2465	0.26216
Validation Dice	0.9553	0.9518	0.9719	0.5370	0.6351	0.6082
Validation Loss	0.2783	0.1554	0.1795	0.5376	0.6367	0.6078

Table 1: Training and Validation Performance For PhC and Skin Datasets

Dropout

Method Overview

The Dropout, proposed by Hinton et al.[1] in 2012, is an effective regularization method to avoid the occur of over-fitting. During the front propagation progress, the input will pass through network and be computed with all the neurons, which makes the neural network get a completed learning result from the training data. But when the dataset is small and the model is complex, such computation strategy could reduce the generalization performance of the network. To solve this problem, the Dropout method deactivates a portion of neurons with a probability p, which could make the network get rid of the reliance of local features and become more generic. Another advantage of the Dropout is that this method could reduce the time cost of training, since cutting down the amount of neurons involved in computation could accelerate the propagation. Therefore, we decide to adopt this efficient regularization method in our experiments.

Implementation

The key step is to add the Dropout layers into the U-Net model. Both the contracting path and expansive path of the U-Net use four blocks composed with Convolutional Layers and Max Pooling layers to compute the features[3], which are the main computation units, so these blocks are the best places to be deactivated a portion of nuerons to avoid the happening of over-fitting. Hence, we apply one Dropout Layer to each of these computing blocks and the corresponding network structure is shown in Figure 7. Here, we use the *Dropout* function provided by *Pytorch*[8] as our Dropout Layer implementation.

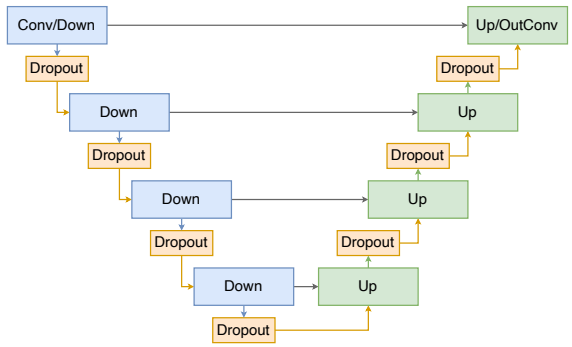

Figure 7: UNet Structure with Dropout Modules

p	Training Loss	Validation Loss	Validation Dice
Baseline	0.1893	**0.1881**	**0.9716**
0.1	**0.1715**	0.2761	0.9526
0.3	0.3041	0.3721	0.7110
0.5	0.2135	0.6368	0.1397

Table 2: Dropout performance on PhC-C2DH-U373 with different dropout probabilities p

Experiments and Results

In the first series of experiments, we apply our Dropout U-net to the PhC-C2DH-U373 dataset. Since the parameter of probability p could heavily influence the effect of Dropout Layers, we explore the network with probability p equals to 0.1, 0.3, 0.5 to find the most suitable parameter value. We train the network for 50 epochs and the corresponding experiment results are shown in Table 2. **When p is 0.1, the Dropout U-net has better performance with training loss 0.1715 and validation loss 0.9607 than other value of p.** The larger the value of p is, the worse the network performs. **We think this is because large p deactivates overmuch nuerons therefore influences the basic learning ability of U-Net.** What's more, compared with the baseline validation loss 0.1881 and validation score 0.9716, **there is no numeric improvement from the Dropout U-net.** The most probable reason is that **the dataset is too simple to generates obvious over-fitting**. PhC-C2DH-U373 only contains 1-channel gray images, whose features are too simple to cause over reliance to the network. Therefore, we decide to adopt the bigger Kaggle Skin Lesion Segmentation dataset and experiment with p equals to 0.1. The tendency of training loss and validation loss is visualized in Figure 8. The plot shows that the over-fitting occurs at the baseline model because the training loss converges ideally while the validation loss is no satisfactory. And comparing with the baseline tendency, **our Dropout-version U-net successfully relives the over-fitting**, since the validation loss becomes obviously smaller and is closer to the training result. The numeric experiment results are shown in Table 3. **We can find that our Dropout-version U-net achieves better validation loss and validation score than the baseline model and the training loss remains the same level with the baseline.**

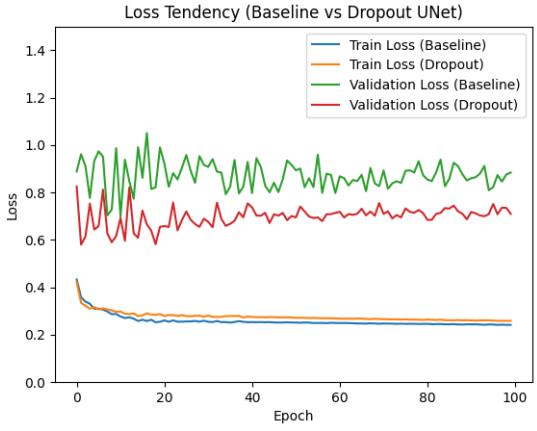

Figure 8: Loss Curves of UNet Baseline and UNet with Dropout on Skin Lesion Segmentation

Model	Training Loss	Validation Loss	Validation Dice
Baseline	0.2419	0.8842	0.2803
Dropout	**0.2585**	**0.7097**	**0.5762**

Table 3: Dropout performance on Skin Lesion Segmentation Dataset

Data Augmentation

Method Overview

Data Augmentation is applied on the training dataset to enrich the diversity and alternations of inputs as one of the regularization technique. The input images of the dataset are varied in terms of its contents, directions and stretching sizes from the original images while the corresponding masks are also being augmented by the same processes. The model, as a result, will be improved to prevent from memorizing and overfitting all images.

Implementation

The gray scale cell images used in this project from the PhC dataset do not have a specific certain direction for reasonable recognition and therefore are ideal to perform **flipping and random rotations**.

The images, with the probability of 0.3, are horizontally flipped to increase the possible directions that the model will learn from the images features. To prevent the model memorizing only some sides of the objects to do the segmentation, the inputs are also being rotated with a by a small angle randomly choosing from (0,10) degree. Pixel values of 0 are filled in the image to resize to the original input size. The model, therefore, learn to be invariant to the object directions for better performance.

The input images are also **cropped** with a random portion selected from (0.8, 1) and are stretch to the size of the original image in (1,1) equal proportion. Cropping allows the model to focus on only a portion of the image which might be the target objects of interest that located partially at the original image.[9]

Gaussian Blur is also used to blur and reduce an image's detail. The degree of blur is positively related to the parameters' value of kernel size and sigma. The kernel size is set to be an odd number of 3 and the sigma is set to 0.1 for only light blurring.

The probability parameter p is set to only 0.1 for rotation, cropping and Guassian blur to largely reserve most of the information from the original data.

Different data augmentations techniques are applied on the first PhC training image as an exampled illustration shown in the figure 9 and 10. The corresponding masks are augmented by the same processes as plotted.

Result

Focusing on the PhC dataset to evaluate the performance of model after data augmentation, as shown in the figure11 and table4, the training and validation loss gradually decrease and reach to the range of (0.13, 0.2). The validation dice score are improved after data augmentation to 0.962 compared with the baseline model running the first 5 epochs.

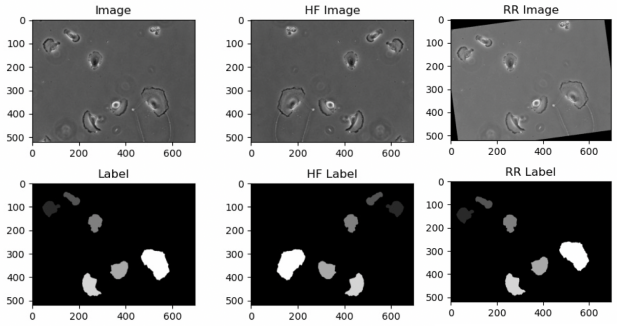

Figure 9: Example plot of data augmentation. The leftmost column are the original image and mask. 'HF', 'RR' indicates horizontal flip and random rotation

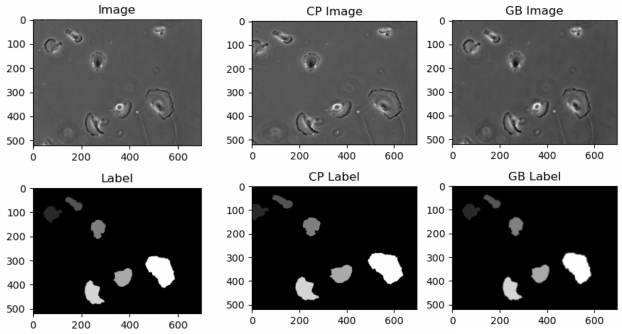

Figure 10: Example plot of data augmentation. The leftmost column are the original image and mask, 'CP' for crop and resize, 'GB' for gaussian blur

Model	Training Loss	Validation Loss	Validation Dice
Base	0.3209	0.2783	0.9553
DA	**0.1781**	**0.1683**	**0.9662**

Table 4: Training and Validation Performance For PhC Data Set-data augmentation

Early stopping

Method Overview

Early stopping (ES) is a method to prevent over-fitting by truncating the iterations based on the specified conditions, i.e., stop the learning process iteration before the model converges to our training data set. Early stopping is done by calculating the validation loss and stop the training process when the loss is moving in an upward trend. This approach is intuitive because it is not beneficial to continue training when the loss is no longer decrease, also further increases the training time. The over-fitting issue will then occur if we keep iterating the model and let the model learn from the training data set repeatedly.

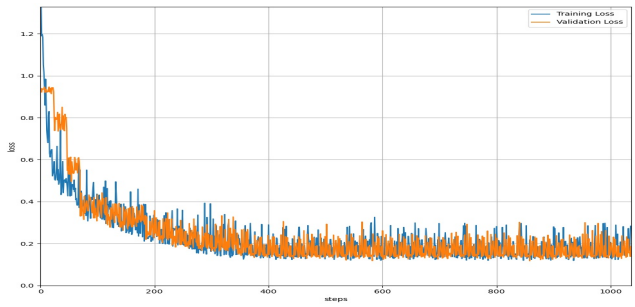

Figure 11: Training and validation loss for PhC Data Set-data augmentation

The working principle of early stopping can be seen visually in the Figure 12. When the loss of the training data set is getting lower, but the loss of the validation data set becomes higher, it can be found that the current model has started to over-fit, which is no longer applicable to the general data set, but has become a model only for the training data set.

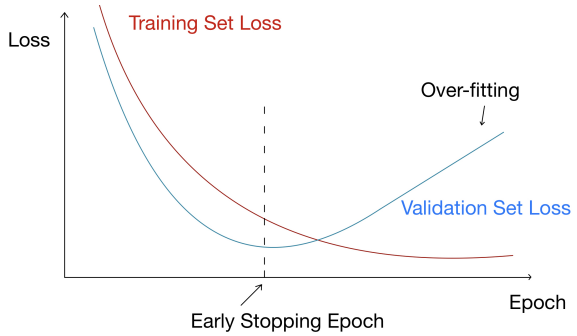

Figure 12: Early Stopping Method

Implementation

In our U-Net semantic segmentation project, we implemented the early stopping regularization method to avoid the over-fitting problem. The We set the conditions that **for the PhC data set/Skin-Lesion data set, if the validation loss is no longer decrease for a consecutive 7/25 times over the total 50/100 epochs of training** [6], then we consider the model as becoming to an over-fit model. Therefore, the early stopping will trigger to stop the training process and save the most accuracy model for further testing. The reason that we set 7/25 times of increasing in validation loss is because there must exist fluctuation during the training and validation. We cannot consider the model is over-fitting because of few losses drop down situations. It is possible that increase in validation loss after a particular epoch, but then the following epoch leads the loss go down again. Thus, we cannot define that it is no longer improving based on a fewer consecutive increase.

Result

In our U-Net semantic segmentation project, We have tested two data sets on it. The result for PhC data set can be found

Model	Training Loss	Validation Loss	Validation Dice
Base	0.1673	0.1752	0.9671
ES	**0.1668**	**0.1784**	**0.9697**

Table 5: Training and Validation Performance For PhC Data Set

in the Figure 13 and Table 5. The result for the Early stopping method is not obvious. There is no clear over-fit issue when we run 50 epoch for the PhC data set. Over the total 50 epochs, we can find that the 49th epoch has the best accuracy result. We directly stop at here because the total number of epoch is 50. **The validation dice for ES model becomes 0.9697 when we stop the iteration at epoch 49**, which higher than the baseline. That is, our early stopping method works. We also adopted the Skin-Lesion-Segmentation data set. The result can be found in the Figure 14 and Table 6. From these two, we could see that the **iteration stopped at epoch 8, and then the validation loss keep increasing in the following 25 epochs. The validation loss when applied ES becomes 0.6256, and the validation dice increases to 0.6416**. Thus, we can see a quite obvious improvements when applied early stopping method on Skin-Lesion data set.

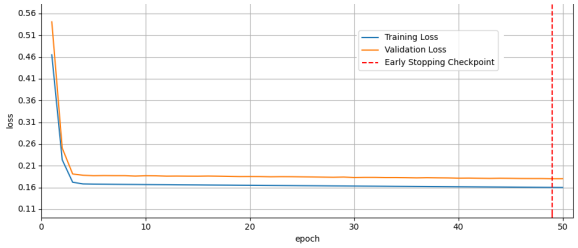

Figure 13: Train Loss vs Validation Loss for PhC Data Set

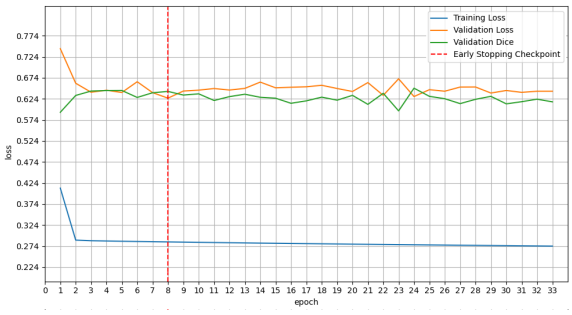

Figure 14: Train Loss vs Validation Loss for Skin-Lesion Data Set

Receptive Field Regularization

This section records the experiments on the Receptive Field Regularization(RFR) for medical segmentation tasks with UNet. This regularization method is inspired by the paper

Model	Training Loss	Validation Loss	Validation Dice
Base	0.2739	0.6921	0.5953
ES	**0.2839**	**0.6256**	**0.6416**

Table 6: Training and Validation Performance For Skin-Lesion Data Set

published in 2021 by Koutini et al. [5], but, instead of setting fixed sizes of the receptive field, here I used the sum/mean of squared values of ReLU outputs as the criterion for the receptive field.

Method Overview

Generally, RFR is a regularization technique that encourages models to have smaller receptive fields. That is, each convolutional layer should perceive areas as small as possible. One possible reason for a model to be overfitting is that the model starts to mesmerize the training data. Then, the model becomes more and more dedicated to the training data instead of the general task. By setting penalties on the size of the receptive field, RFR discourages models from remembering the whole image, and thus helps to avoid overfitting. Here, the penalty for a receptive field $(h \times w)$ is

$$rf\_loss = \frac{\omega}{hw} \sum_{i}^{h} \sum_{j}^{w} x_{i,j}^2 \tag{1}$$

where ω is a hyperparameter which is 0.1 in the implementation.

Implementation

In an UNet model, the RFR method can be implemented on the ReLU outputs of resolution-increment and downsample layers. Each of these layers has several ReLU layers whose outputs are passed to the next layers as inputs. Since the ReLU function reduces all non-positive values to zero, it **filters** the information from previous outputs. Consequently, the fewer non-zero values the ReLU layer generates the smaller areas its next layers can perceive. Thus, implementing receptive field regularization on ReLU outputs can help to encourage the model to use less information from the whole image to predict the result.

Models	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
Base	0.0681	0.7723	0.9829	0.9224
Layer 0	0.0706	0.6081	0.9823	0.9305
Layer 1	0.0656	0.6173	0.9834	0.9299
Layer 2	0.0642	0.6391	0.9838	0.9314
Layer 3	0.0651	**0.5315**	0.9838	**0.9337**
Layer 4	0.0618	0.6689	0.9842	0.9247

Table 7: Training and validation performances. The Receptive Field Regularization helps to reduce overfitting considerably.

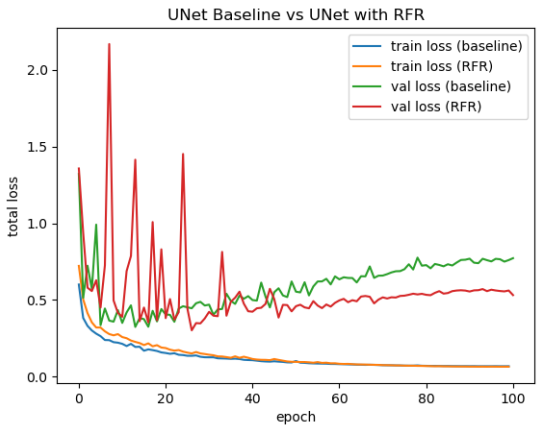

Figure 15: Training and Validation loss of UNet Baseline and UNet with RFR

Experiments

Here, I focus on the Skin Lesion Dataset.

Since the UNet consists of 4 downsample layers, the number of layers to regularize on has been experimented with. To be more specific, here, the resolution-increment layers have an index of 0, and the following indexes $1\tilde{4}$ are the corresponding downsample layers. The RFR loss is cumulative, so the optimizer, like those in ResNets, can maintain sufficient momentum on all the layers even when the RFR loss is computed at Layer 4. As Table **??** shows, no matter which layer the RFR loss is finally computed, it always helps to reduce the validation loss. Particularly, when the RFR is implemented on Layer 3, the model reaches the best performance with over a 30% improvement in validation loss.

Figure 15 compares the loss performance of the Baseline UNet and the UNet with RFR on Layer 3. With RFR, the UNet shows great oscillations at the beginning, but it gradually converges at the end. Moreover, it is obvious that the Baseline starts overfitting from Epoch 20. Its validation loss keeps increasing while the training loss is decreasing stably. In contrast, the RFR model shows a much slighter increment in validation loss. Thus, it is reasonable to conclude that the Receptive Field Regularization method helps to avoid overfitting.

Conclusion

In this project, we apply variant regularization methods to the U-Net model on two biomedical segmentation datasets ,PhC-C2DH-U373 and Kaggle Skin Lesion Dataset, with different size to obtain a better understanding of the overfitting and the impact of regularization. Our experiments including L1 and L2 regularization, data augmentation, the early stopping method, the Dropout method and the receptive field regularization. Our experiment results of all the five methods adpoted prove that the regularization could effectively relieve the over-fitting and therefore improve the performance of neural networks.

References

[1] Geoffrey E Hinton et al. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012).

[2] H Jabbar and Rafiqul Zaman Khan. “Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)”. In: *Computer Science, Communication and Instrumentation Devices* 70 (2015).

[3] 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*. Springer. 2015, pp. 234–241.

[4] Prashant Brahmbhatt. *Skin_lesion_segmentation*. June 2019. URL: www.kaggle.com/datasets/hashbanger/skin-lesion-segmentation.

[5] Khaled Koutini, Hamid Eghbal-zadeh, and Gerhard Widmer. “Receptive Field Regularization Techniques for Audio Classification and Tagging With Deep Convolutional Neural Networks”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 1987–2000. DOI: 10.1109/taslp.2021.3082307. URL: https://doi.org/10.1109%2Ftaslp.2021.3082307.

[6] Jason Brownlee. *Use Early Stopping to Halt the Training of Neural Networks At the Right Time*. URL: https://machinelearningmastery.com/how-to-stop-training-deep-neural-networks-at-the-right-time-using-early-stopping/.

[7] S Kumar. *Glioblastoma-astrocytoma U373 cells on a polyacrylamide substrate*. URL: celltrackingchallenge.net/2d-datasets/.

[8] Pytorch. *DROPOUT*. November 20th, 2022. URL: https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html.

[9] JIM TOL. *Image augmentation: how to overcome small radiology datasets*. URL: https://www.quantib.com/blog/image-augmentation-how-to-overcome-small-radiology-datasets.