Redefining U-Net

Pasawee Wirojwatanakul
New York University
pw1103@nyu.edu

Alex Sung-Min Hong
New York University
hong@nyu.edu

Abstract

In this study, we thoroughly studied U-Net and tweaked its architecture. Empirically, the ensemble of our top models is able to outperform the plain U-net by 4.66% on Pixel Accuracy, and 1.76% on the Jaccard Index. We studied different techniques to fuse the feature-maps of the encoder of Residual U-Net with the feature-maps of the decoder of Residual U-Net. We empirically show that concatenation of feature-maps is not the optimal way to fuse the feature-maps. To fuse the feature-maps, we experimented with addition, concatenation, Feature-wise Linear Modulations, Gate, and weighted combinations of feature-maps. Furthermore, we studied how changing the encoder can affect the performance, as well as how pre-training can affect the performance, in which for our case, choosing a strong encoder affects the score more than the choice of fusion techniques.

1 Introduction

Using Deep Learning techniques to segment images is a well studied task[1, 2]. The goal of image segmentation is to perform pixel-wise classification for each input image. One of the problems of using Convolutional Neural Networks (CNN) to segment images is that high-resolution feature maps and high-level feature maps are distributed in the opposite side of the network, where high-level feature maps are distributed in the later layers, and high-resolution feature maps are distributed in the earlier layers. Due to this, there have been attempts to reuse the earlier layers by adding or concatenating the feature-maps. Li et al. [3] argued that naively fusing feature maps result in drowning useful information with useless information. The majority of our work will be devoted to finding better ways to pass information from the feature-maps in the earlier layers to feature-maps in the further layers.

2 Related Works

Following the success on image classification tasks, Convolutional Neural Network (CNN) based architecture has first been introduced to image segmentation by Long *et al.* through Fully Convolutional Networks (FCN) [2]. In order to solve image segmentation tasks, FCN adopts up-sampling operators at the final layers for calculating pixel-wise output, whereas CNN for image classification adopts fully connected layers. FCN and U-Net [1] embrace the key idea of skip connections, but unlike FCN that fuses earlier layers with deeper layers via element-wise addition, U-Net uses concatenation. In addition, it uses convolutions and non-linearities between each up-sampling block. Skip connection was the key idea for both papers to reconstruct the full spatial resolution at the network output. Feature Pyramid Network (FPN) [4] uses the structure of U-Net with predictions made on each level of the feature pyramid. Nonetheless, FPN, concatenation, and addition all suffer from the problem of drowning useful information with useless information, since all of these methods do not consider usefulness of feature-maps. Gated Fully Fusion (GFF) [3] uses gates to measure the importance of information in different feature maps. Dilated Convolutions was tested for convolution network module that mixes multi-scale context information without loss of resolution [5], and proved adding a context module can improve the accuracy. There has been an attempt to embrace DenseNet [6] to

downsampling and upsampling blocks [7]. Recently, Takikawa *et al.* proposed a two-stream CNN architecture [8]. In this architecture, shape information is processed as a separate branch and it processes only boundary related information. This is enforced by the model's Gated Convolutional Layer (GCL) and local supervision.

3 Dataset

For our final project, we focused our attention on medical images, where our main task is to segment skin-lesion. We got our data from the ISIC-2017 competition[9], which contains only 2000 images in the training set, 150 images in the validation set, and 600 images in the test set. We believe this is an interesting dataset given the fact that skin cancer contributes to over 9,000 deaths each year with 5 million newly diagnosed cases in the United States[9]. A good segmentation map can reduce the time it takes for physicians to evaluate an image, reducing physicians burn-out, while allowing physicians to spend more time with the patients.

4 Proposed Models

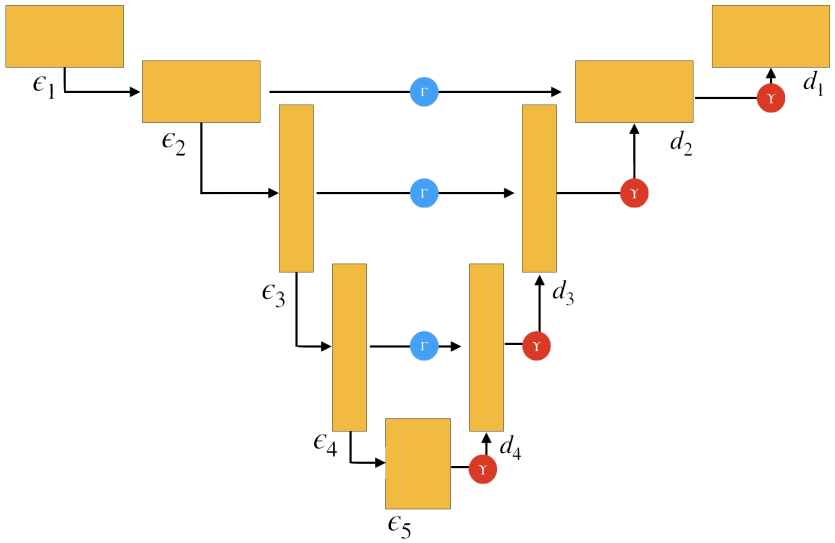

Figure 1: Overall illustration of model architecture. Each encoder (ϵ_i) adopts residual networks and decoders take feature-maps from the encoder and decoder from lower layer, after going through Γ and Υ respectively. Γ and Υ are functions defined by the user and tested with multiple approaches in this paper.

All of our models follow the strucutre as in Fig. 1. The

$$d_i = F_i(\Upsilon(d_{i+1}), \Gamma(\epsilon_i)),$$

where we denote $\Gamma(e_i)$ and $\Upsilon(d_{i+1})$ as the output of the encoder and decoder at some layer i. F represents the fusion policy or fusion function, and denote c_i as the number of channels of $\Upsilon(d_{i+1})$ and $\Gamma(e_i)$. In a Vanilla U-net, F is represented as concatenation. In U-net F_i is the same for all layers since we only concatenate the feature-maps, but this is not necessarily the case since for other choices of F_i, F_i can have its own parameters; as an example, a CNN.

4.1 Residual U-Net

Residual U-net maintains the U-net structure, but removes the contracting network or the encoder with a Resnet[10]. Here, we chose Resnet18 from Pytorch as the encoder. We expect this tweak to significantly improve upon the network for two reasons: firstly, Resnet performs well on many standard classification tasks, and secondly, its pretrained weights on Imagenet[11] are available.

Furthermore, we experimented with both addition and concatenation to merge feature-maps from the encoder with feature-maps from the decoder.

4.2 Residual FiLM U-Net

In many tasks, NN receive multiple inputs from different modalities. In such cases, it can be useful to process one source given the other sources of information as context. Feature-wise Linear Modulation(FiLM) layers influence neural network computation via a simple, feature-wise affine transformation based on conditioning [12]. FiLM could be applied to many problem settings, from visual reasoning to reinforcement learning, and in this paper, we implemented the module into our model. FiLM can be presented mathematically as

$$FiLM(F_{i,c}) = f_c(x_i)F_{i,c} + h_c(x_i) \tag{1}$$

where subscripts of i, c refers to the i^{th} input's c^{th} feature or feature map. f and h can be arbitrary functions such as neural networks [12]. While there are other ways to perform feature-wise transformations, such as addition, element-wise scaling, multiplication, and concatenation, FiLM combines both scaling and translation into a conditional affine transformation.

Residual FiLM U-Net inherits the structure of the Residual U-Net, where we use FiLM as the fusion function. In other words, the decoder integrates the fusion step which is represented as FiLM. Formally, FiLM fusion part for this model can be:

$$FiLM_i = f(\Gamma(e_i)) \odot \Upsilon(d_{i+1}) + h(\Gamma(e_i)), \tag{2}$$

where we view $\Gamma(e_i)$ as the conditioning input and Γ and Υ are $1 \times 1 \times c_i$ CNNs.

4.3 Residual Combination U-Net

Residual Combination U-Net (ResCUN) follows the structure of Residual U-Net, in which we define the fusion function to be:

$$CUN_i = a(\Upsilon(d_{i+1})) + b(\Gamma(e_i)), \tag{3}$$

where a and b are both $1 \times 1 \times c_i$ CNN, which acts to weight the feature maps of the encoder and decoder. This is a simple tweak, in which the main idea is we should not weight the encoder and decoder equally. Merging via concatenation or addition becomes really useful only when the feature-maps in the encoder and decoder should contribute the same amount; however if this is not the case, then we should weight them differently. The CNNs, a and b, are designed to weight the encoder and decoder. If the optimal way to merge the feature-maps is indeed via $\Upsilon(d_{i+1})$ + $\Gamma(e_i)$, then a and b can learn the identity function. Weighing the decoder and the encoder can also be extended to concatenation, but here we focus on weighted combinations of feature-maps.

4.4 Gated Residual U-Net

The Gated Fully Fusion (GFF) module was designed to perform multi-level feature fusion via gates. Here the author notes three common ways to merge feature-maps: addition, concatenation, and Feature Pyramid Network (FPN). These three merge methods suffer from the problem of drowning useful information with useless information. The motivation behind the use of gates is to measure the usefulness of each feature vector in a feature map and aggregate the feature-maps accordingly. GFF can be formally expressed as:

$$\tilde{X}_l = (1 + G_l) \odot X_l + (1 - G_l) \odot \sum_{i=1, i \neq l}^{L} (G_i \odot X_i), \tag{4}$$

where L is the total number of layers, G_l is a gate map for feature maps at layer l, and $G_l \in [0, 1]^{H_l \times W_l \times C_l}$, where H_l , W_l, and C_l are the dimension of X_l, and $\odot$ represents element-wise multiplication. G_l is approximated by using $sigmoid(w_l * X_l)$, where $w_l * X_l$ refers to a 1 x 1 x C_l convolution. By taking the sigmoid, G_l lies between 0 and 1, where 1 indicates that the information is useful. Here we can see that feature-maps at layer l can only be fused with other layers if the current information in layer l is useless, when G_l contains small values, and if information in other layers is useful, when $G_i, \forall i \neq l$, contains large values, see Eq. 4 . This GFF module can be plug into any

network that merges feature-maps from different layers, specifically where feature-maps at layer l takes in feature-maps from all other layers. Note that the proposed model in [3] contains many other modules that we do not consider in the paper; for this paper we focused our attention only on the GFF module.

Gated Residual U-Net follows the structure of Residual U-Net exactly. The only difference is in the fusion function. Here, we simplified GFF module, and define merge similarly to GFF. The main idea here is that the encoder should only receive information from the decoder if the information in the decoder is useless and information in the encoder is useful. Thus, our merge function using gates can be defined as:

$$gate_i = (1 + G_d) \odot \Upsilon(d_{i+1}) + (1 - G_d) \odot (G_e \odot \Gamma(e_i)) \tag{5}$$

where $\odot$ represents element-wise multiplication, G_d is a gate to measure the importance of current information in the decoder, and G_e is a gate to measure the importance of the information from the encoder. From Eq.5 we can see that only when the current information in the decoder is useless,when G_d contains small values, and when information in the encoder is useful,when G_e contains large values, can the two feature-maps be merged. Borrowing our idea from GFF, we define G_d as:

$$sigmoid(w_i * \Upsilon(d_{i+1})),$$

where $w_i * \Upsilon(d_{i+1})$ can be represented using a $1 \times 1 \times c_i$ CNN. It is important to note the subtle difference between our gate and the GFF module. Since we are following the structure of U-Net, we do not merge feature-maps from layer l, with all the other layers. We only merge feature maps from the encoder with feature maps from the decoder at the same layer.

Furthermore, we perform two variants of model ensembles for Residual Gated U-Net: deterministic ensembling and probabilistic ensembling. For deterministic ensembling of Residual Gated U-Net, we train M independent networks. We can obtain the predictions via the mean of:

$$p(y^*|x^*) = \frac{1}{M} \sum_{m=1}^{M} p(y^*|x^*, \hat{w}_m), \tag{6}$$

where $\hat{w}_m$ is the weights which minimized the cross-entropy loss for model m, and $p(y^*|x^*, \hat{w}_m)$ is the softmax of the predictions of the CNN.

For probabilistic ensembling we **turn on** Dropout during test time, in which probabilisitc ensembling can be represented as:

$$p(y^*|x^*) = \frac{1}{M} \sum_{m=1}^{M} \frac{1}{T} \sum_{t=1}^{T} p(y^*|x^*, w_{mt}) \tag{7}$$

Note that $\frac{1}{T} \sum_{t=1}^{T} p(y^*|x^*, w_{mt})$ means we pass the image through the model m with Dropout turned on T times, and average out the predictions. Since Dropout is turned on for model m, for each pass through model m, the result will be different. After we averaged out the predictions for all the M models, we average out the averaged predictions for each model.

It is well known that ensembling will improve upon the performance, mainly because model combinations enrich the hypothesis space. However, it is worth noting that for deterministic ensembling, inference time scales linearly with the number of models. In addition we also have to store the output of the M models. For probabilistic ensembling, the inference time is $M \times T$ times more than the inference time of using one model. Ensembles may not be suitable for applications that require real-time decisions, unless you have adequate computing resources.

For Residual Gated U-Net, we tried Standard Dropout and Gaussian Dropout. Let $a(Wx)$ be the output of some layer of a Neural Network with some non-linearity, we can define Gaussian Dropout as:

$$GO = M \odot a(Wx), M \sim Gaussian(1, \frac{p}{1-p}), \tag{8}$$

where $\odot$ is element wise multiplication and p is the dropout rate. The dropout rate and location of the Dropout for Standard and Gaussian Dropout significantly affects the performance. Based on previous research, [13], we only apply Dropout after fusing the encoder and decoder. By doing so, we drop half from the encoder, and half from the decoder.

5 Experiments

For all experiments, we trained the models by minimizing the cross-entropy loss for 30 epochs. Unless stated otherwise, all networks have been trained with Standard Dropout. We only experimented with Gaussian Dropout on Residual Gated U-Net. We fixed the learning rate at 1e-4 and the weight decay at 1e-5. For pretrained models, we subtracted subtracted and divided the test images with the mean and standard deviation of the imagnet images. For models that were not pretrained, we subtracted and divided the test images with the mean and standard deviation of the training images. All images and ground truth maps are resized to 512 and 1024 for computational reasons. In addition, all images are gray-scaled during training, validation and testing. Only during the validation and training do we randomly flip the images and groundtruth. In addition, we also performed model ensembles for our top performing model. We evaluated our models using Pixel Accuracy and the Jaccard Index, also referred to as mean IOU.

Model	Merge Type	Pixel Accuracy	Jaccard
Unet	Concat	0.8707	0.5324
Res U-Net PT = True	Concat	0.9102	0.691
Res U-Net PT = True	FiLM	0.91264	0.6924
Res U-NetPT = True	Add	0.9053	0.6718
Res U-Net PT = False	Concat	0.8728	0.51238
Res U-Net PT = True	Combination	**0.9128**	**0.7**
Res U-Net PT = True	Gate	**0.9144**	**0.702**
Res U-Net Gaussian	Gate	0.9125	0.6846
Res U-Net Ensemble	Gate	0.9164	0.705
Res U-Net Prob. Ensemble	Gate	**0.9173**	**0.7086**

Table 1: Experimental Results of different U-Net Variants. Note that PT = True, means we used pretrained Imagenet weights, and Merge Type = Combination, refers to ResCUN. Res U-Net Prob.Ensemble refers to probabilistic ensembling, and Res U-Net Gaussian refers to U-Net with Gaussian Dropout.

We present the results of Residual U-Net, Gated Residual U-Net, ResCUN, and normal U-Net for baseline, see Table. 1. In addition, we also perform model ensembling for the top performing model. We present U-Net and Residual U-Net as the baseline models, which we improved upon. We performed two ablation studies: the first is how changing the encoder affects the performance, and the second is to see how pretraining affects the performance of the network. The results show that using pretrained Resnet to replace the encoder of U-Net leads to significant gains in performance. Without pretraining, there doesn't seem to be much help.

Contrary to our expectations, our FiLM model did not perform a lot better than Residual U-Net with concatenation. We suspect this is because FiLM was designed to deal with multi-modal inputs, but the encoder and decoder are essentially same modal. However, the result for FiLM still shows that naively concatenating or adding feature-maps is not optimal since FiLM outperforms the two. In addition, we show that using simple combination of the feature-maps from the encoder and decoder, like ResCUN model, can outperform concatenation and addition of feature-maps. This was expected given that addition and concatenation weighs both feature-maps equally, which may not be the optimal case. Finally, our proposed Residual Gated U-Net outperforms all of the baseline models and other Residual U-Net variants proposed by us. In a way, ResCUN, FiLM, and Gated Residual U-Net are similar in a sense that they all assume the feature-maps in the decoder and encoder should not be equally important.

As expected, deterministic ensembles of Gated Residual U-Net outperformed one Gated Residual U-Net. Interestingly, probabilistic ensembling, outperforms deterministic ensembling. Note that for our experiment, we set T in Eq.7 to 5. We hypothesize that by increasing T, the accuracy for probabilistic ensembles will continue to increase until a certain point, then it will stagnate.

Gaussian Dropout seems to hurt the result a little bit. However, as mentioned before, the location and dropout rate matters. In the future, one can find better ways to tune the Dropout rate; as an example, Concrete Dropout[14].

For reference, the top submission in ISIC-2017 has a Pixel Accuracy score of 0.934 and Jaccard Index score of 0.765, which uses additional post-processing techniques. In our work, we did not use any post-processing techniques.

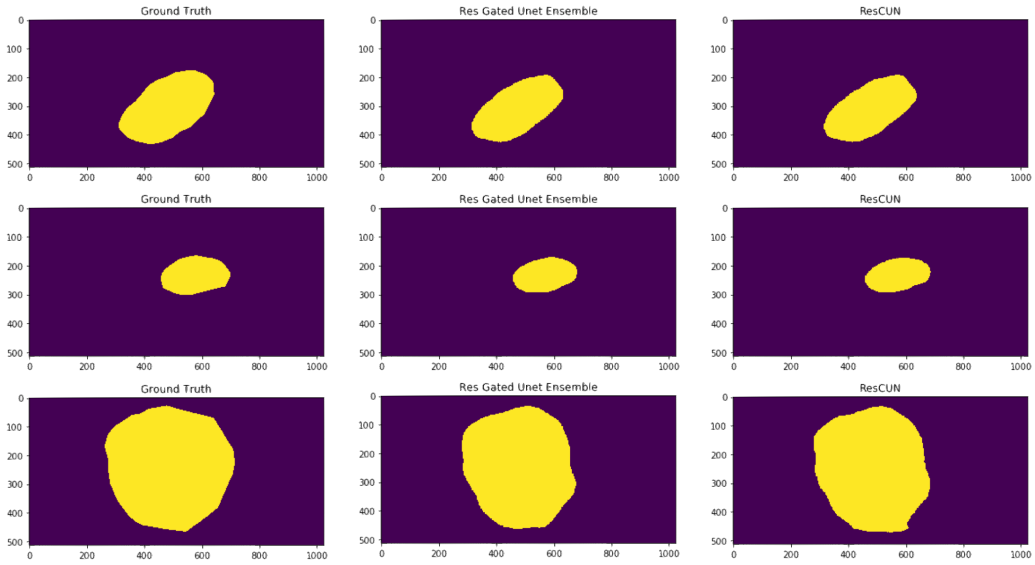

Figure 2: Sample

We present three samples from two of our top performing models. Given that Residual Gated U-Net marginally outperforms ResCUN, the resulting segmented maps are expected to be similar. In the third sample, we can see slight differences, in which Residual Gated U-Net seems to perform better that ResCUN in the boundary area. Even so, both models seem to underestimate the width of the lesion for the third sample.

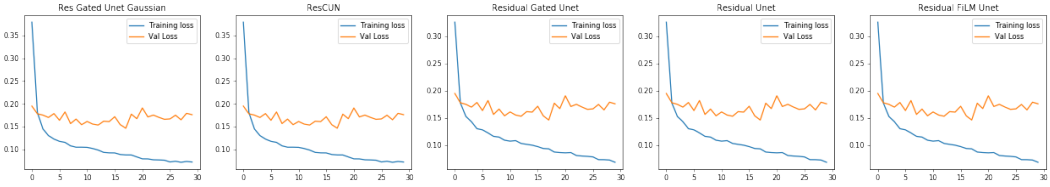

Figure 3: Loss Plots of U-Net variants. Res Gated Unet Gaussian refers to Residual Gated U-Net using Gaussian Dropout

The loss plot presented here are the loss plots for pretrained models with Dropout applied during training and turned off during validation stage. From the plots, Residual Gated U-Net and ResCUN seems to perform slightly better than Residual U-Net and FiLM U-Net. All models seem to overfit. Given the pretrained weights were trained on a significantly larger dataset, we believe training on 2000 images was not enough for the model to learn new useful representation. Given the size of the training set, we believe our models can benefit from Bayesian method and self-supervised learning.

6 Conclusion

In conclusion, the choice of the encoder seems to have the greatest effect on U-Net's performance. Due to this, for future work, we can try using newer models such as Efficient-Net[15], which outperforms ResNet in the ImageNet challenge, as our encoder. In addition, we can study how the choice of the decoder affects the performance of the model. In this work, we show that concatenation is not the optimal way to fuse feature-maps of the encoder and the decoder, in which a simple weighted combination can already outperform concatenation and addition. In addition, the use of gates is effective in controlling information propagation between the encoder and decoder. Given

limited training data, the models were not able to learn much. We believe tying self-supervised task to our U-Net structure, similar to [16], will be useful. Since we only have 2000 images in the training set, it is easy for the test images to be out of the training distribution, and thus the CNN is more likely to predict incorrectly of such images. Finally, given that a strong performing image classifier can be used as the encoder, we can try classifying and segmenting lesions via multi-task learning by sharing the features in the encoders.

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2014.

[3] Xiangtai Li, Houlong Zhao, Lei Han, Yunhai Tong, and Kuiyuan Yang. Gff: Gated fully fusion for semantic segmentation. *arXiv preprint arXiv:1904.01803*, 2019.

[4] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016.

[5] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[6] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[7] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 11–19, 2017.

[8] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5229–5238, 2019.

[9] Matt Berseth. Isic 2017-skin lesion analysis towards melanoma detection. *arXiv preprint arXiv:1703.00523*, 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[12] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. *CoRR*, abs/1709.07871, 2017.

[13] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding, 2015.

[14] Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout, 2017.

[15] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019.

[16] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training for out-of-distribution generalization, 2019.