Scene Segmentation Project

Supervisor
Prof. Abdul Qayyum

Students

Belal Hmedan
Deng Jianning
Thien Bao Bui

Outline

I. Overview
II. Implementation and result
    - Original Unet
    - Modified Unet
III. Difficulties
IV. Future improvements
V. Demo

Overview of the Project

Objective:

2D Multiclass Segmentation using Deep Learning Model

for COVID-19 CT scans Dataset.

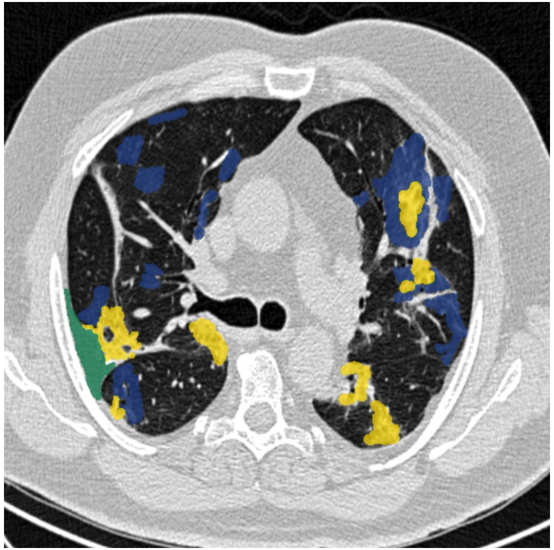


Image source:
http://medicalsegmentation.com/covid19/

Dataset:

20 lung CT scans Nifti images,

Annotations include left lung, right lung and infections.

Images vary in size, intensity, number of slices, orientation, and bit depth.

PreProcessing

1. Image Resize 256x256.

2. Normalization (min-max).

3. One Hot Encoding.

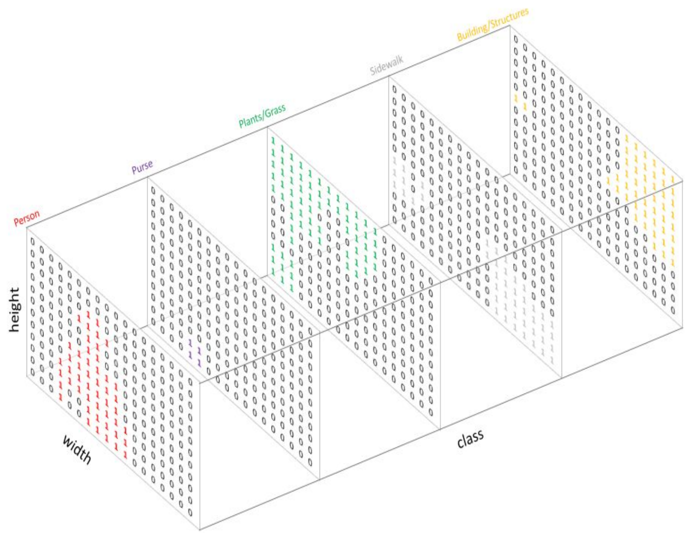


Image source:
https://www.jeremyjordan.me/content/images/2018/05/Screen-Shot-2018-05-16-at-9.36.00-PM.png

Implementation and Result

Implementation of 3 models:

- Original Unet
- Unet with Backbone EfficientNet-B7
- Unet with Backbone MobileNet

Implementation and Result

Evaluation:

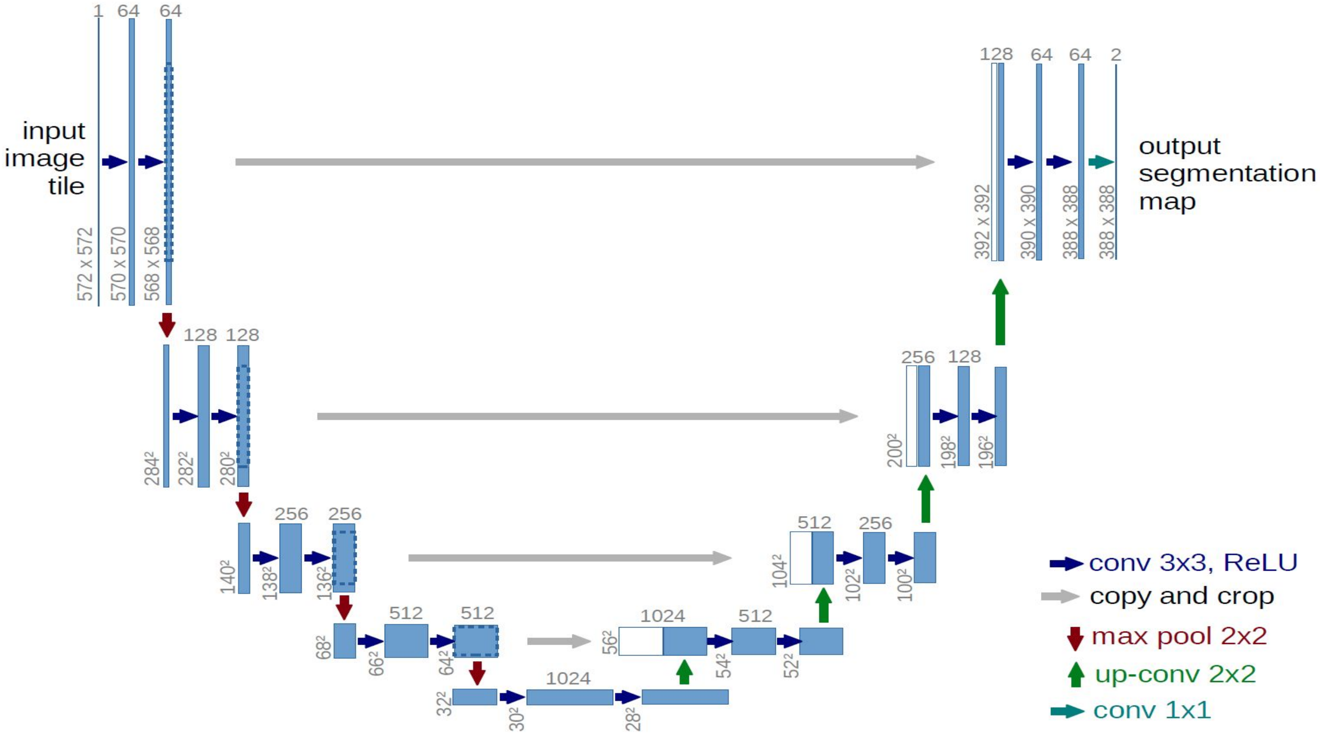
- Dice coefficient
- Hausdorff distance
- P-value
- Pearson correlation coefficients
- Bland-Altman plot
- ROC/AUC curve

Dataset

Dataset is splitted with the following ratio: 0.8 - 0.1 - 0.1

for training, validating, testing respectively

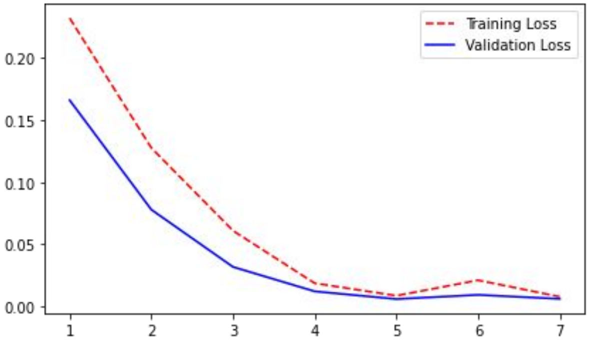
Original Unet model

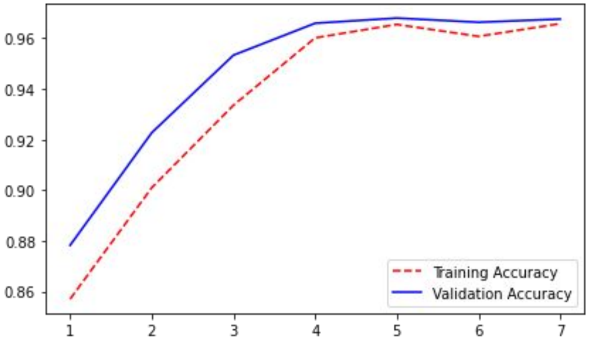


First Settings

Parameter	Value
Number of initial feature maps	64
Batch size	16
Optimizers	Adam (lr = 1e-3)
Loss function	Categorical cross entropy
Metrics	Dice coefficient

First Result





First Result

Evaluation on test dataset

Test loss	Test accuracy	Pearson coefficient	P_value
0.0052	0.96	0.86	~0

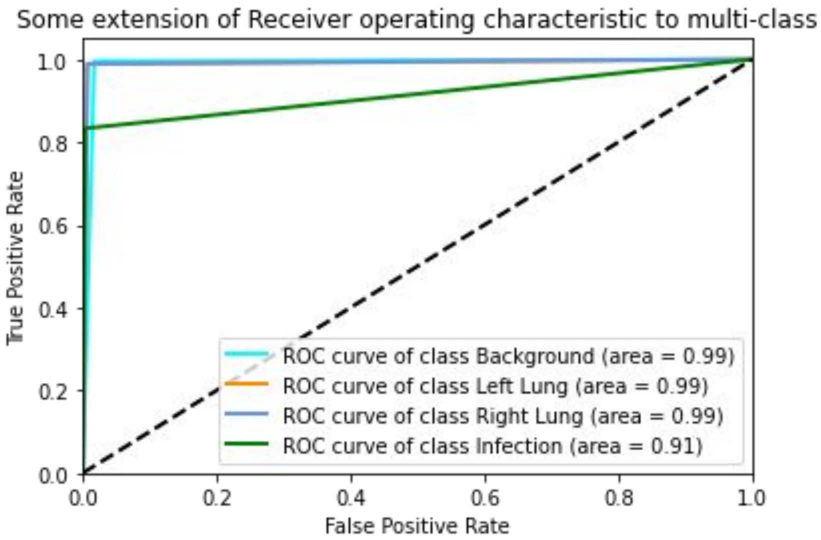
First Result

Evaluation on each class of the test dataset

	Background	Left Lung	Right Lung	Infection
Dice coef	0.88	0.88	0.89	0.570
Hausdorff distance	51.31	126.31	99.7	69.34
Pearson correlation	0.88	0.88	0.89	0.59

First Result

ROC/AUC curve



Modifications

- Change the number of filters/feature maps
- Add Batch Normalization layers
- Add Drop out layers
- Try different optimizers, learning rate, batch_size,...

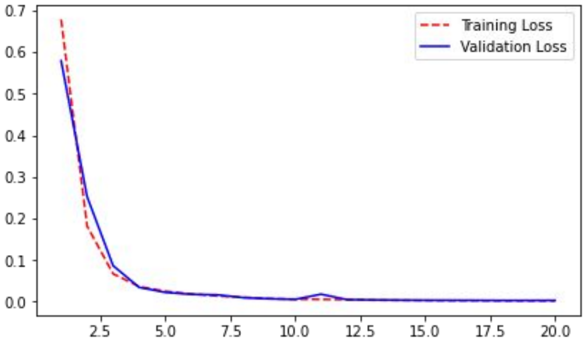
Add 2 batch normalization layers for the first 3 blocks

Add 1 batch normalization layer for the rest of the blocks

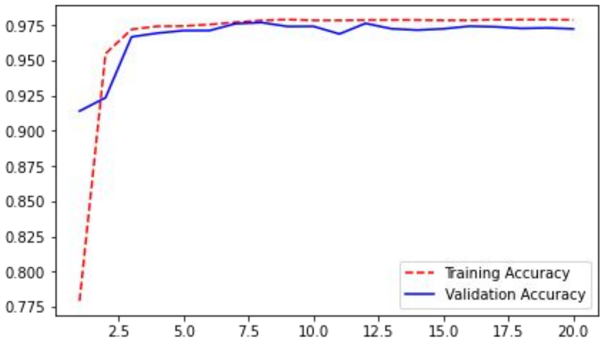
Final settings

Parameter	Value
Number of initial feature maps	16
Batch size	16
Optimizers	Adam (lr = 1e-4)
Loss function	Categorical cross entropy
Metrics	Accuracy

Result



Training Loss
Validation Loss
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Training Accuracy
Validation Accuracy
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
0.775
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

Result

Evaluation on test dataset

Test loss	Test accuracy	Pearson coefficient	P_value
0.0044	0.98	0.89	~0

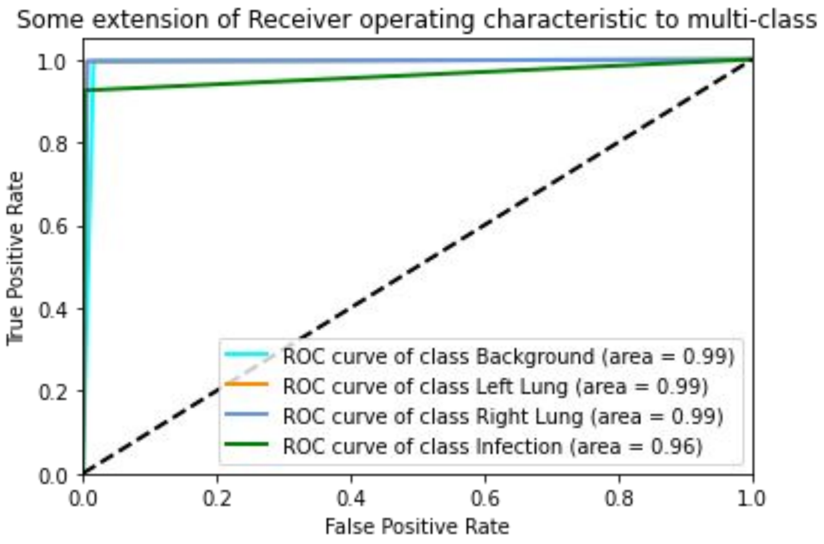
Result

Evaluation on each class of the test dataset

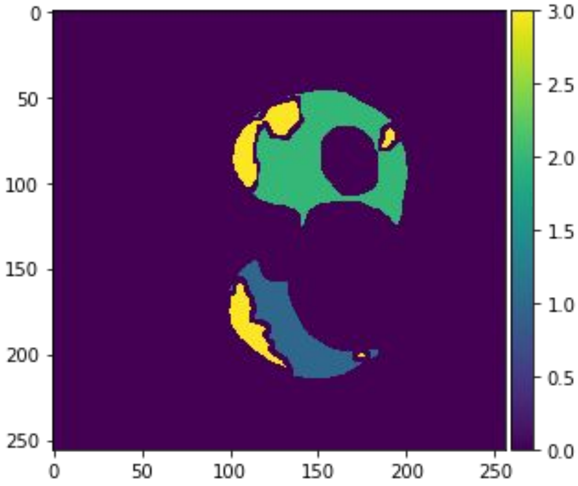
	Background	Left Lung	Right Lung	Infection
Dice coef	0.91	0.92	0.92	0.71
Hausdorff distance	54.48	106.04	88.82	65.40
Pearson correlation	0.91	0.92	0.92	0.73

Result

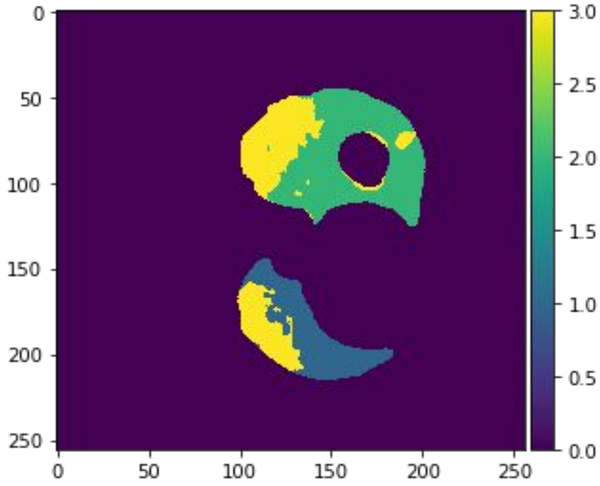
ROC/AUC curve



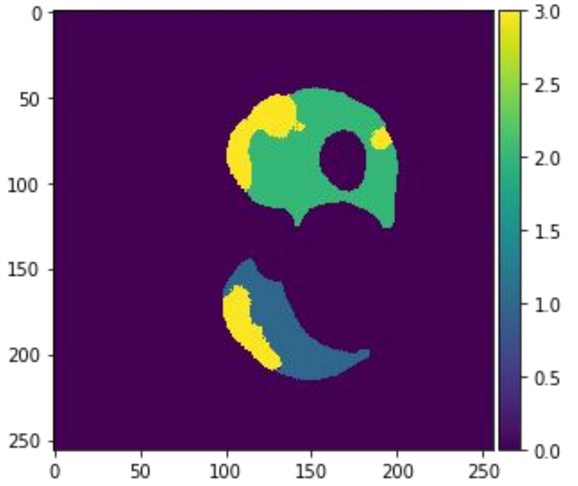
Result



Ground truth

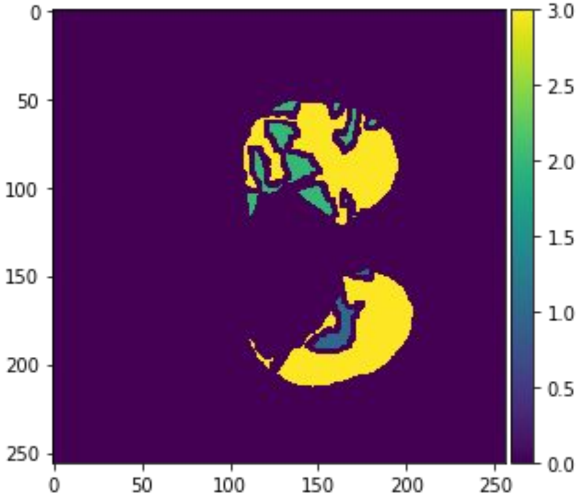


First model prediction

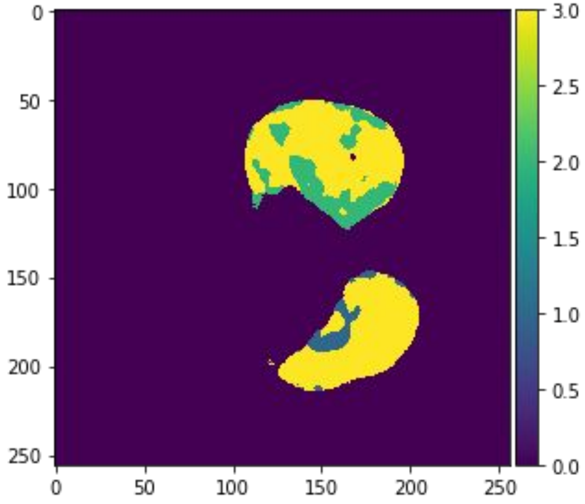


Final model prediction

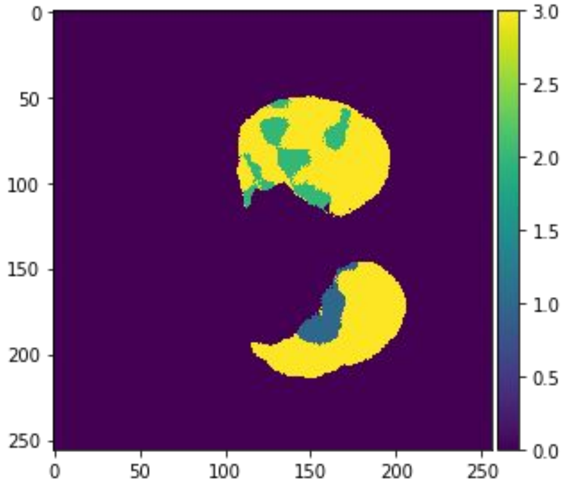
Result



Ground truth



First model prediction



Final model prediction

Modified Unet Encoder

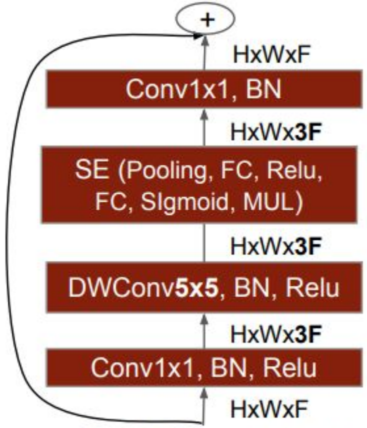
Using pre-trained model as encoder:

EfficientNet-b7

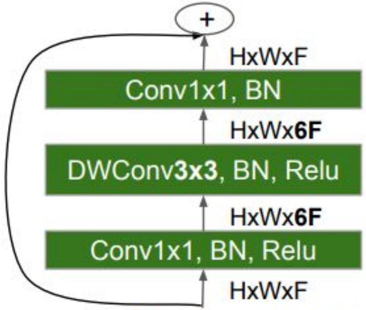
Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Removed

Scale up the base model with a factor of 7



(b) MBConv3 (k5x5)



(c) MBConv6 (k3x3)

Reference:
Tan M, Le Q V. Efficientnet: Rethinking model scaling for convolutional neural networks[J]. arXiv preprint arXiv:1905.11946, 2019.

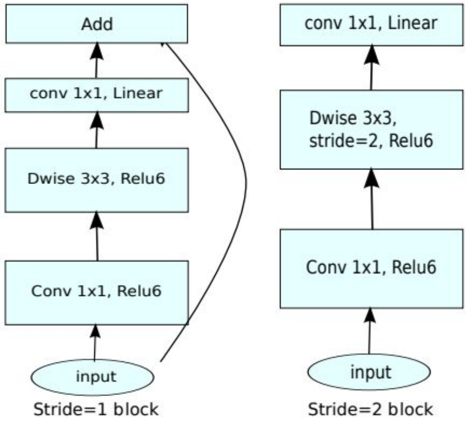
Modified Unet **Encoder**

Using pre-trained model as encoder:

MobileNet-v2 (baseline model)

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Removed

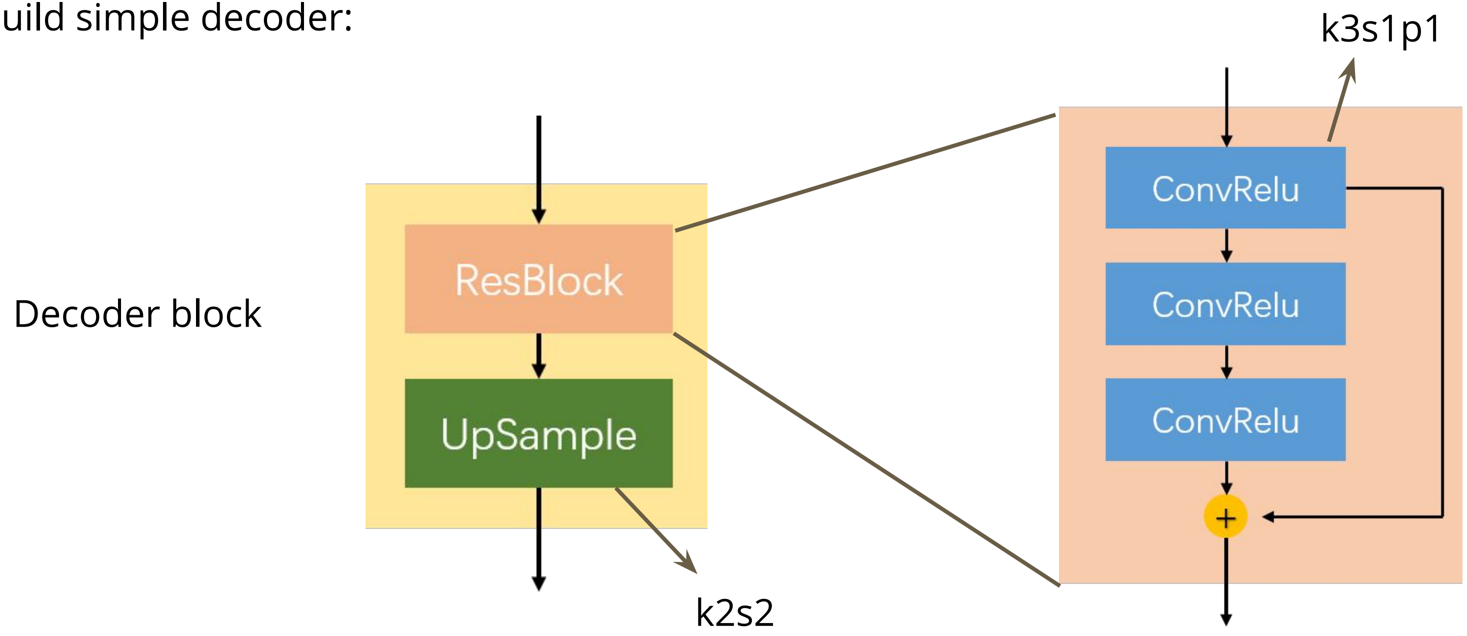


Reference:

Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.

Modified Unet Decoder

Self-build simple decoder:



Reference:
Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.

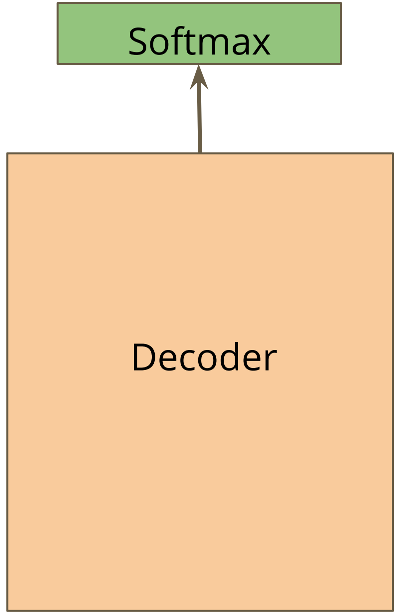
Modified Unet Connection

Self-build simple decoder:

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Connection
Concat channel information
proportion: 1/1.5

Bottom
Connected with ConvRelu



Modified Unet Training

- Transfer training:
  freeze the weights in encoder
- Epoch:
  50
- Learning rate:
  1e-3, adjusted with lr = lr * 0.97^epoch
- Loss function:
  BCE, DICE and their equally weighted combination
- Optimizer:
  Adam, beta = (0.9, 0.999)

For DICE metric:

- Global DICE

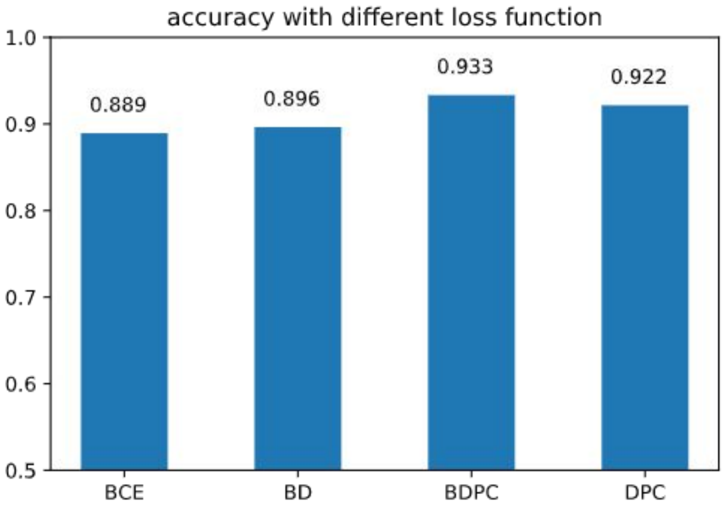
  Calculated by flatten pred and mask

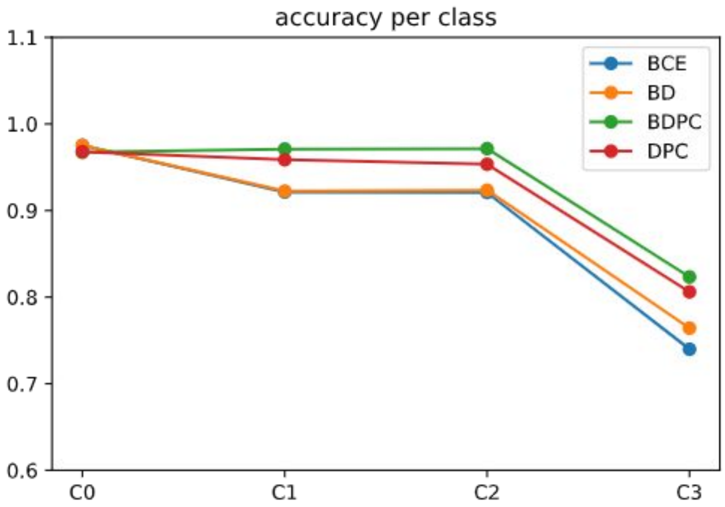
- Avg Local DICE

  Calculate class-wise dice metric, then use the mean

Modified Unet   Results

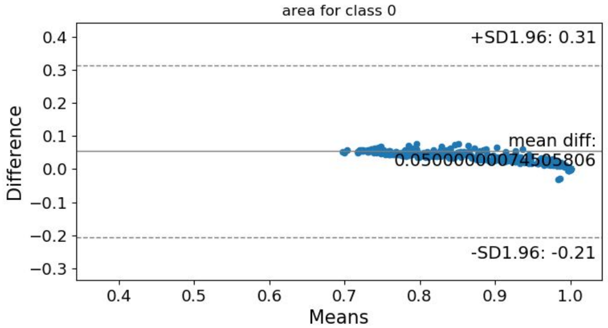
MobileNet backbone with different loss function

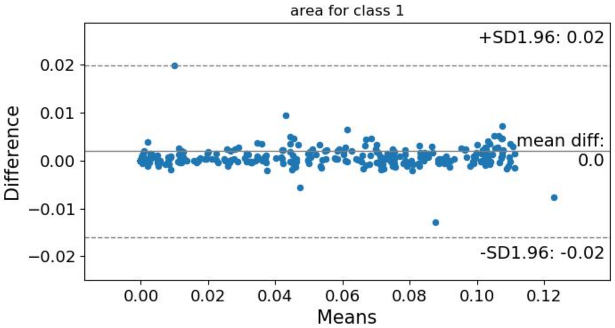


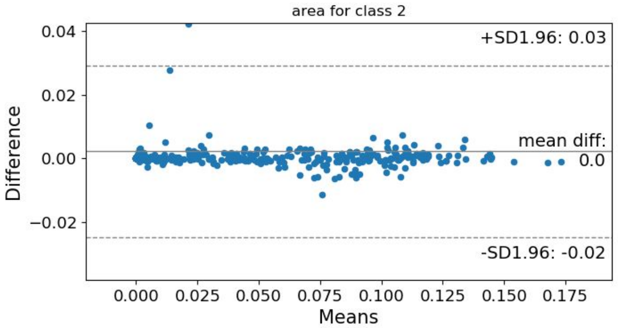


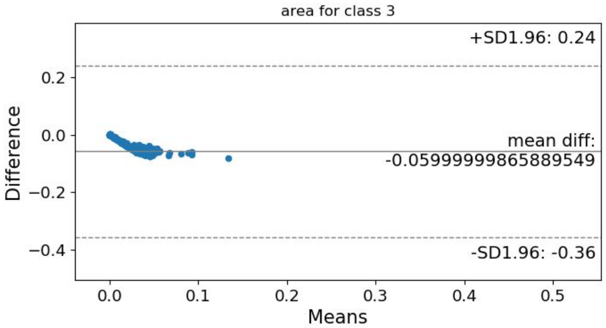
Modified Unet Results

Statistical results with BCE and Dice loss: Area









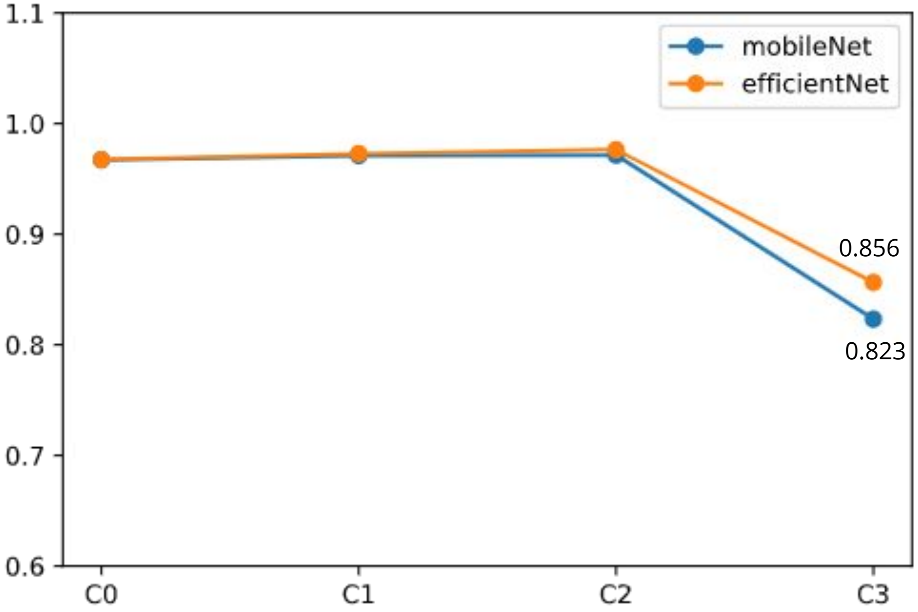
Modified Unet Results

Statistical results with different loss function on MobileNet backbone

	Loss Function				
	BCE	G_Dice	L_Dice	BCE&G_Dice	BCE&L_Dice
G_Dice	0.9682	0.9683	0.9658	0.9686	0.9669
L_Dice	(0.928, 0.804, 0.823, 0.528)	(0.979, 0.897, 0.895, 0.718)	(0.968, 0.954, 0.952, 0.793)	(0.975, 0.922, 0.925 , 0.762)	(0.967, 0.971, 0.971, 0.823)
Area Correlation	(0.5285, 0) (0.9934, 0) (0.9843, 0) (0.0655, 0.2206)	(0.6811, 0) (0.9489, 0) (0.9438, 0) (0.2173, 0)	(0.6656, 0) (0.9603, 0) (0.9538, 0) (0.2881, 0)	(0.4304, 0) (0.9961, 0) (0.9969, 0) (0.0579, 0.2788)	(0.4444, 0) (0.9988, 0) (0.9990, 0) (0.1197, 0.0247)
TP	(1, 0.973, 0.938, 0.073)	(1, 0.979, 0.993, 0.155)	(1, 0.972, 0.941, 0.085)	(1, 0.985, 0.932, 0.088)	(1, 0.962, 0.973, 0.124)
PPV	(0.928, 0.804, 0.823, 0.528)	(0.988, 0.877, 0.876, 0.573)	(0.978, 0.95, 0.915, 0.491)	(0.932, 0.79, 0.819, 0.468)	(0.977, 0.943, 0.918, 0.515)

Modified Unet Results

Comparison between mobileNet and efficientNet-b7



Avg local dice:

- mobileNet: 93.3%
- efficientNet: 94.3%

Model Size:

- mobileNet: ~27MB
- efficientNet: ~270MB

Modified Unet **Results**

Comparison with other statistical results

	G_Dice	L_Dice	Area Correlation	TP	PPV
MobileNet	0.9669	0.967, 0.971, 0.971, 0.823)	(0.4444, 0) (0.9988, 0) (0.9990, 0) (0.1197, 0.0247)	(1, 0.962, 0.973, 0.124)	(0.977, 0.943, 0.918, 0.515)
EfficientNet	0.9690	(0.969, 0.971, 0.972, .0854)	(0.6033, 0) (0.9539, 0) (0.9927, 0) (0.0888, 0.0963)	(1, 0.979, 0.965, 0.107)	(0.978, 0.968, 0.962, 0.552)

Modified Unet Conclusion

- Increasing model complexity won't do much help on this task
- Normal 'average' and 'global' performance is not suitable to assess the result
- Loss function is important
  - More criteria should be introduced: TP(Sensitivity), PPV(Precision), area(surface), etc.

    To correctly handle the naturally 'null' mask
  - The evaluation should be measured channel-wise/class-wise

    To correctly handle the small volume effect in the infection area
- Try training with smaller LR with regularization on loss function

Difficulties

1. Hardware limitations

2. Time limitations.

3. Dataset Size.

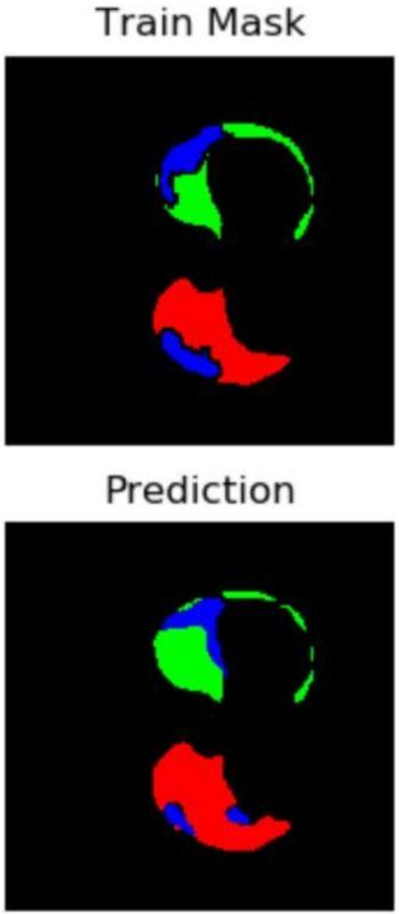
Image credit:
https://images.anandtech.com/doci/12673/nvidia-titanv-technical-front-3qtr-left_1512609636_678x452.jpg

Future Improvements

1. Pre-Processing.

2. Approach.

3. Post-Processing.



Thank you for your attention!