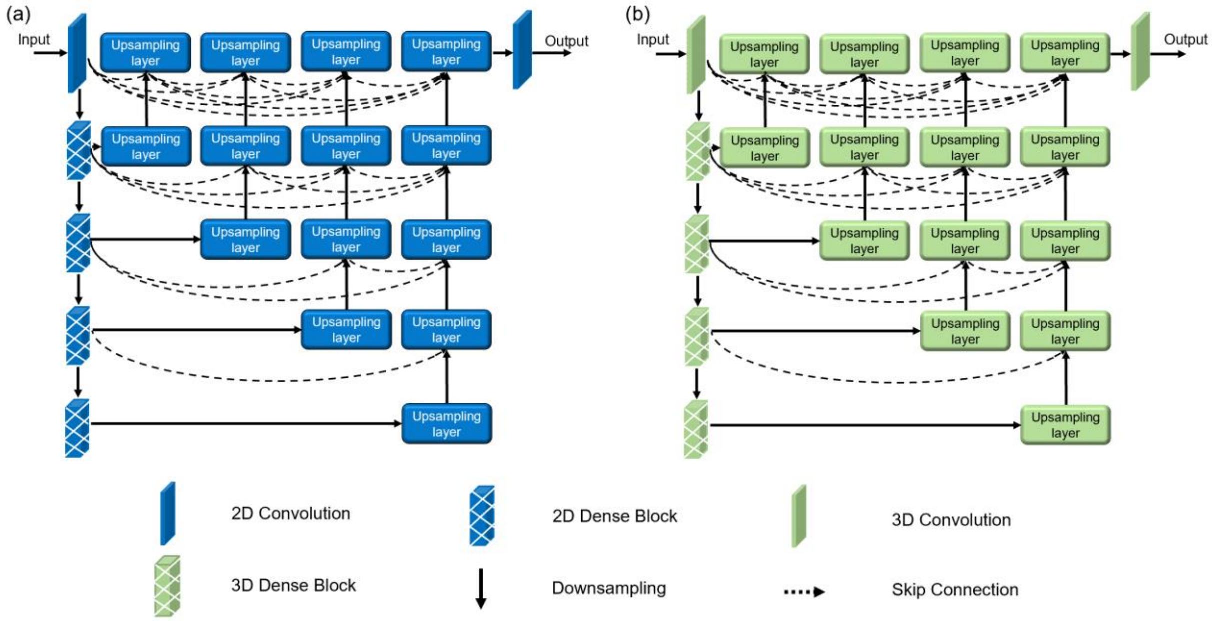
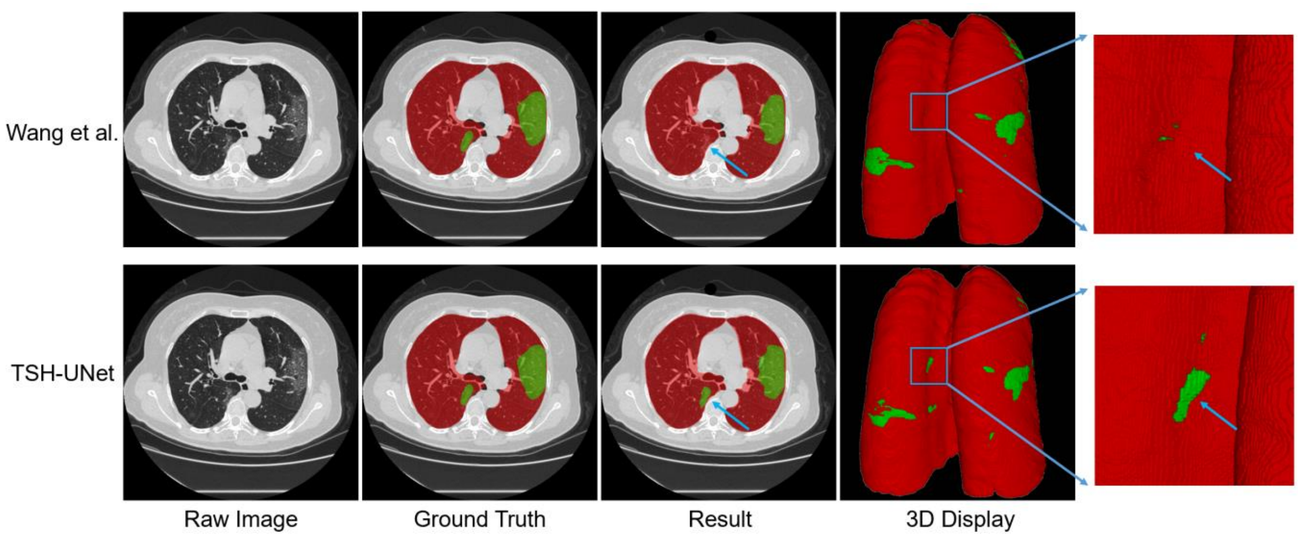
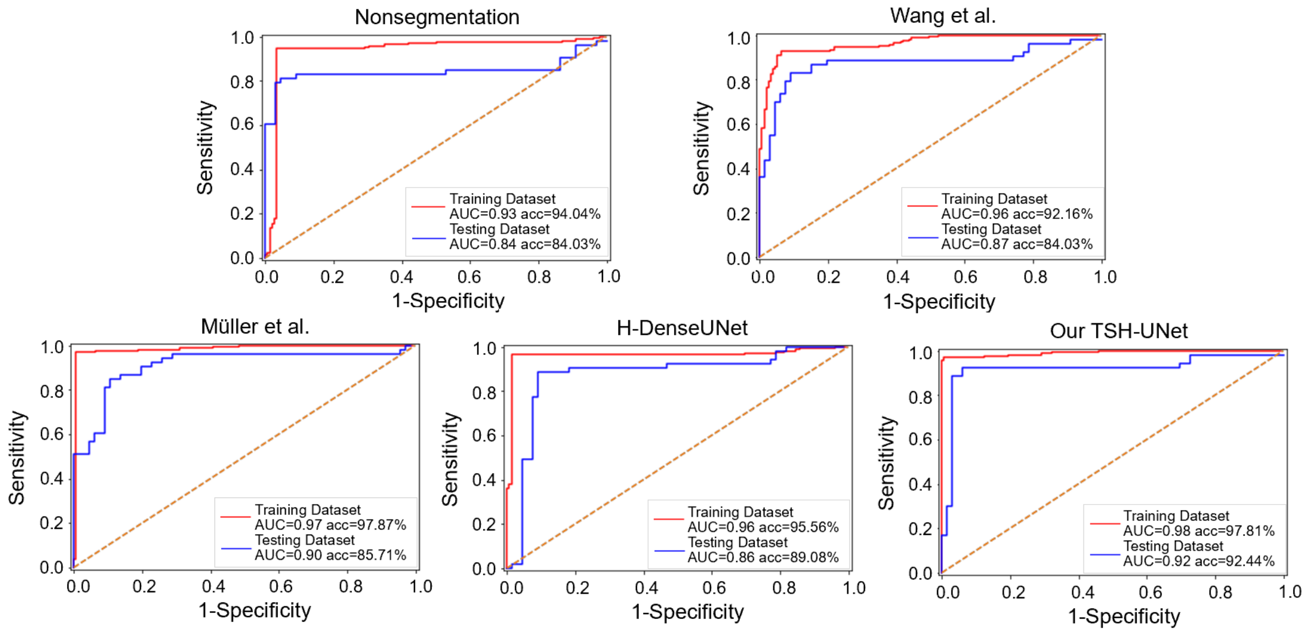
Supplementary Material



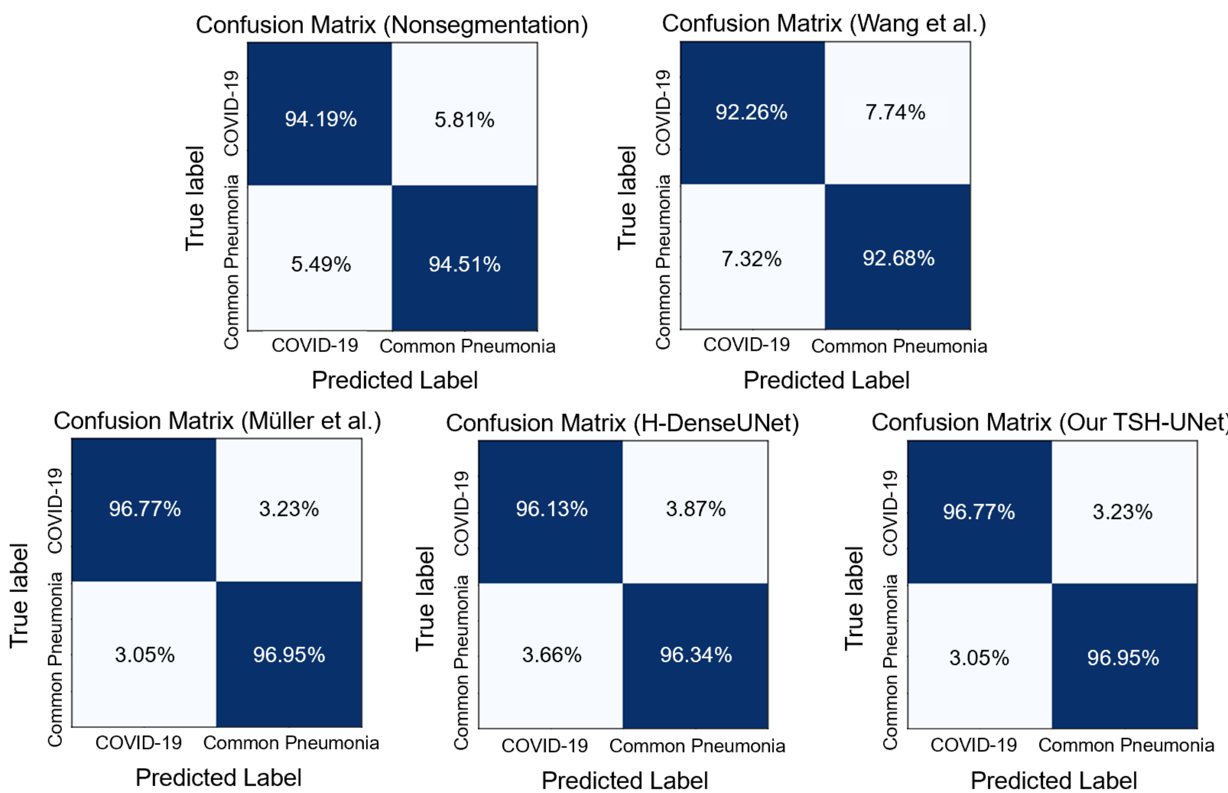
Supplementary Fig. 1. The detailed architecture of the 2D network (a) and 3D network (b) in our proposed TSH-UNet.



Supplementary Fig. 2. The 3D display of the results of Wang et al. (2D UNet++) and our TSH-UNet. The first row is the result of Wang et al. and the second row is the result of our TSH-UNet. The first three columns are a representative slice, its corresponding ground truth, and its corresponding result respectively. The fourth column is the 3D display of a representative case. In the blue frame is the small lesion that cannot be segmented by Wang et al. but can be correctly segmented by our TSH-UNet.



Supplementary Fig. 3. ROC curves of classification tasks using the results of different segmentation networks. Each subgraph represents a segmentation network. In each subgraph, the red curve represents the results on the training dataset, while the blue one represents the results on the test dataset



Supplementary Fig. 4. Confusion matrixes of classification tasks using the results of different segmentation networks on RHWU training dataset.

Supplementary Table 1. Architectures of the proposed TSH-UNet, consisting of the 2D network and 3D network

	Input	Output	2D Network	Input	Output	3D Network
input	512×512×3			512×512×12×2		
convolution 1	512×512×3	256×256×96	7×7, 96 stride 2	512×512×12×2	256×256×6×96	7×7×7, 96 stride 2
max pooling	256×256×96	128×128×96	3×3, stride 2	256×256×6×96	128×128×3×96	3×3×3, stride 2
dense block 1	128×128×96	128×128×384	[1×1, 192 conv 3×3, 48 conv]×6	128×128×3×96	128×128×3×192	[1×1×1, 128 conv 3×3×3, 32 conv]×3
transition layer 1	128×128×384	64×64×192	1×1 conv 2×2 aver pool	128×128×3×192	64×64×3×96	1×1×1 conv 2×2×1 aver pool
dense block 2	64×64×192	64×64×768	[1×1, 192 conv 3×3, 48 conv]×12	64×64×3×96	64×64×3×224	[1×1×1, 128 conv 3×3×3, 32 conv]×4
transition layer 2	64×64×768	32×32×384	1×1 conv 2×2 aver pool	64×64×3×224	32×32×3×112	1×1×1 conv 2×2×1 aver pool
dense block 3	32×32×384	32×32×2112	[1×1, 192 conv 3×3, 48 conv]×36	32×32×3×112	32×32×3×496	[1×1×1, 128 conv 3×3×3, 32 conv]×12
transition layer 3	32×32×2112	16×16×1056	1×1 conv 2×2 aver pool	32×32×3×496	16×16×3×248	1×1×1 conv 2×2×1 aver pool
dense block 4	16×16×1056	16×16×2208	[1×1, 192 conv 3×3, 48 conv]×24	16×16×3×248	16×16×3×504	[1×1×1, 128 conv 3×3×3, 32 conv]×8
upsampling layer1	16×16×2208	32×32×2208	2×2 upsample	16×16×3×504	32×32×3×504	2×2×1 upsample
	--	32×32×2112	sum with dense block 3	--	32×32×3×496	sum with dense block 3
	32×32×2112	32×32×768	3×3 conv, 768	32×32×3×496	32×32×3×504	3×3×3 conv, 504
upsampling layer2	32×32×768	64×64×768	2×2 upsample	32×32×3×504	64×64×3×504	2×2×1 upsample
	--	64×64×768	sum with dense block 2	--	64×64×3×224	sum with dense block 2
	64×64×768	64×64×384	3×3 conv, 384	64×64×3×224	64×64×3×224	3×3×3 conv, 224
upsampling layer3	64×64×384	128×128×384	2×2 upsample	64×64×3×224	128×128×3×224	2×2×1 upsample
	--	128×128×384	sum with dense block 1	--	128×128×3×192	sum with dense block 1
	128×128×384	128×128×96	3×3 conv, 96	128×128×3×192	128×128×3×192	3×3×3 conv, 192
upsampling layer4	128×128×96	256×256×96	2×2 upsample	128×128×3×192	256×256×6×192	2×2×2 upsample
	--	256×256×96	sum with conv 1	--	256×256×6×96	sum with conv 1
	256×256×96	256×256×96	3×3 conv, 96	256×256×6×96	256×256×6×96	3×3×3 conv, 96
upsampling layer5	256×256×96	512×512×96	2×2 upsample	256×256×6×96	512×512×12×96	2×2×2 upsample
	512×512×96	512×512×64	3×3 conv, 64	512×512×12×96	512×512×12×64	3×3×3 conv, 64
convolution 2	512×512×64	512×512×3	1×1 conv, 3	512×512×12×64	512×512×12×3	1×1×1 conv, 3