

## Tensorflow-Image-Segmentation-Augmented-ALCAPA (2025/02/15)

This is the first experiment of Image Segmentation for **ALCAPA (Anomalous Left Coronary Artery from Pulmonary Artery)** based on the latest [Tensorflow-Image-Segmentation-API](#), and [ALCAPA-ImageMask-Dataset.zip](#), which was derived by us from the dataset ImageALCAPA\_BIBM\_Publish.change2zip in [ImageALCAPA](#)

### Data Augmentation Strategy:

To address the limited size of ImageALCAPA dataset, we employed [an online augmentation tool](#) to augment the dataset, which supports the following augmentation methods.

- Vertical flip
- Horizontal flip
- Rotation
- Shrinks
- Shears
- Deformation
- Distortion
- Barrel distortion
- Pincushion distortion

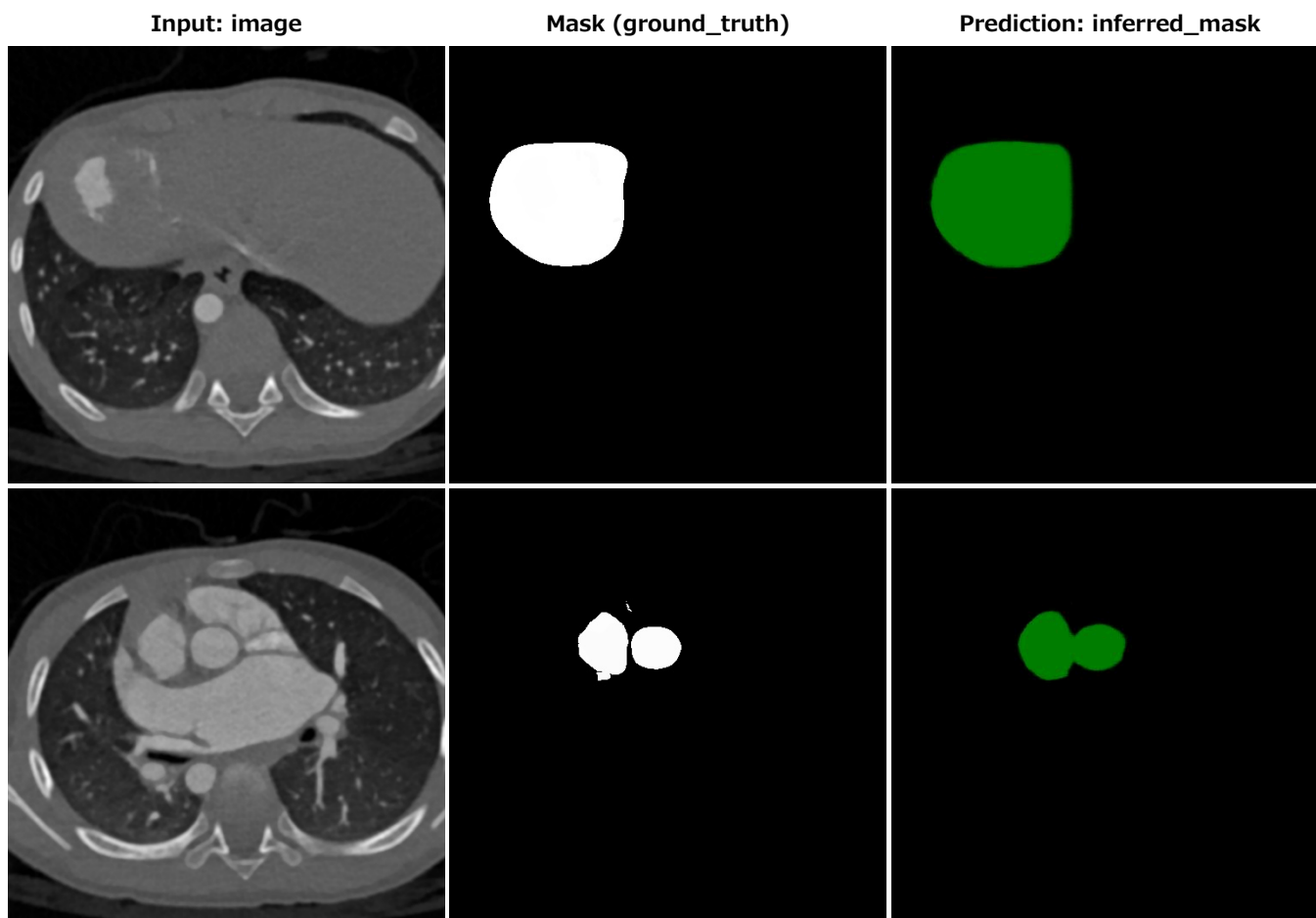
Please see also the following tools

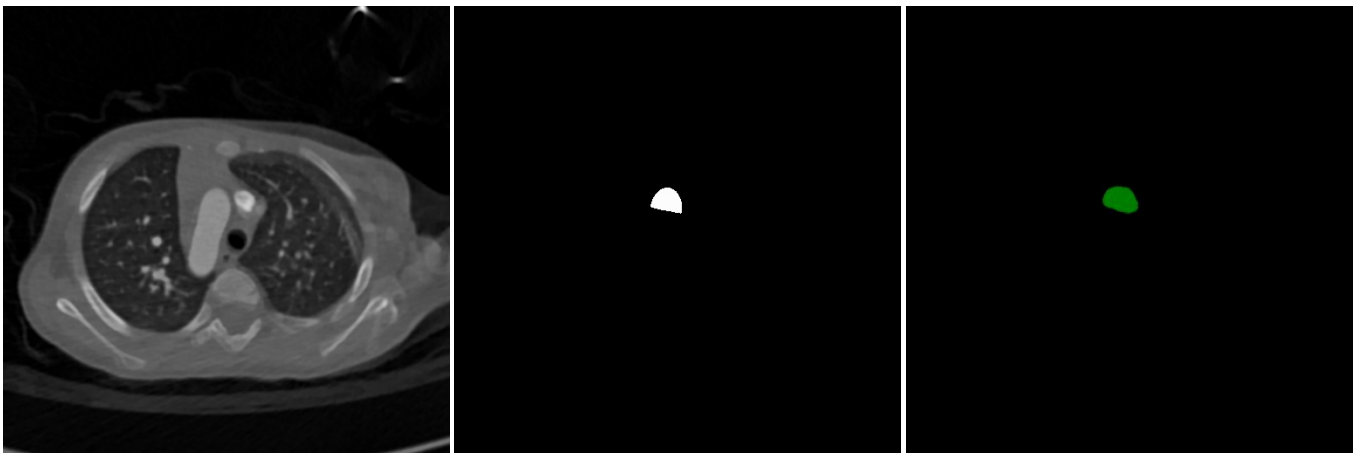
- [Image-Deformation-Tool](#)
- [Image-Distortion-Tool](#)
- [Barrel-Image-Distortion-Tool](#)

---

### Actual Image Segmentation for Images of 512x512 pixels

As shown below, the inferred masks look similar to the ground truth masks.





In this experiment, we used the simple UNet Model [TensorflowSlightlyFlexibleUNet](#) for this ALCAPA Segmentation Model. As shown in [Tensorflow-Image-Segmentation-API](#), you may try other Tensorflow UNet Models:

- [TensorflowSwinUNet.py](#)
- [TensorflowMultiResUNet.py](#)
- [TensorflowAttentionUNet.py](#)
- [TensorflowEfficientUNet.py](#)
- [TensorflowUNet3Plus.py](#)
- [TensorflowDeepLabV3Plus.py](#)

## 1. Dataset Citation

The dataset used here has been taken from **ImageALCAPA\_BIBM\_Publish.change2zip** in the kaggle website [ImageALCAPA](#)

### About Dataset

The ImageALCAPA dataset totally consists of 30 3D CTA images gathered from Guangdong Provincial Peoples' Hospital from June 17, 2016, to August 8, 2021. These images are acquired by a SOMATOM Definition Flash CT machine. All the images are pre-operative ALCAPA CTA images whose top and bottom are around the neck and the brachiocephalic vessels, respectively, in the axial view. The segmentation labeling is performed by a team of two cardiovascular radiologists who have extensive experience in ALCAPA. For each image, one radiologist fulfills the labelling while the other verifies it afterwards. The segmentation includes seven substructures: Myo, LV, RV, PA, Ao, LCA, and RCA, and the labelling of each image goes by 1-1.5 hours.

Please see also:

[ImageALCAPA: A 3D Computed Tomography Image Dataset for Automatic Segmentation of Anomalous Left Coronary Artery from Pulmonary Artery](#)

### License

[Apache 2.0](#)

## 2 ALCAPA ImageMask Dataset

If you would like to train this ALCAPA Segmentation model by yourself, please download our 512x512 pixels JPG dataset from the google drive [ALCAPA-ImageMask-Dataset.zip](#), expand the downloaded ImageMaskDataset and put it under **./dataset** folder to be

```

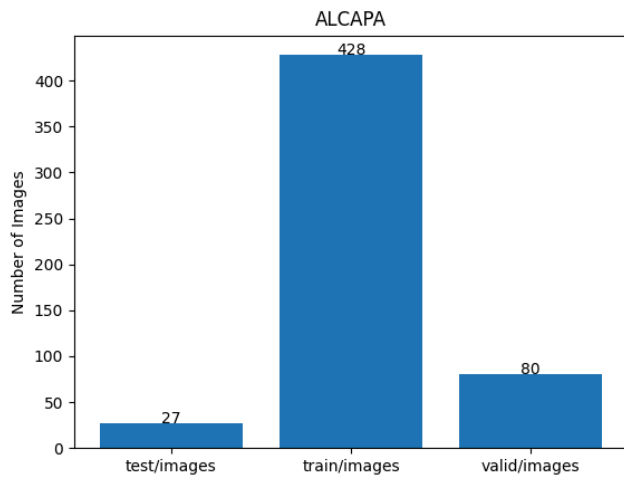
./dataset
├── ALCAPA
│   ├── test
│   │   ├── images
│   │   └── masks
│   ├── train
│   │   ├── images
│   │   └── masks
│   └── valid
│       ├── images
│       └── masks

```

On the derivation of this ImageMask Dataset, please refer to the following Python scripts.

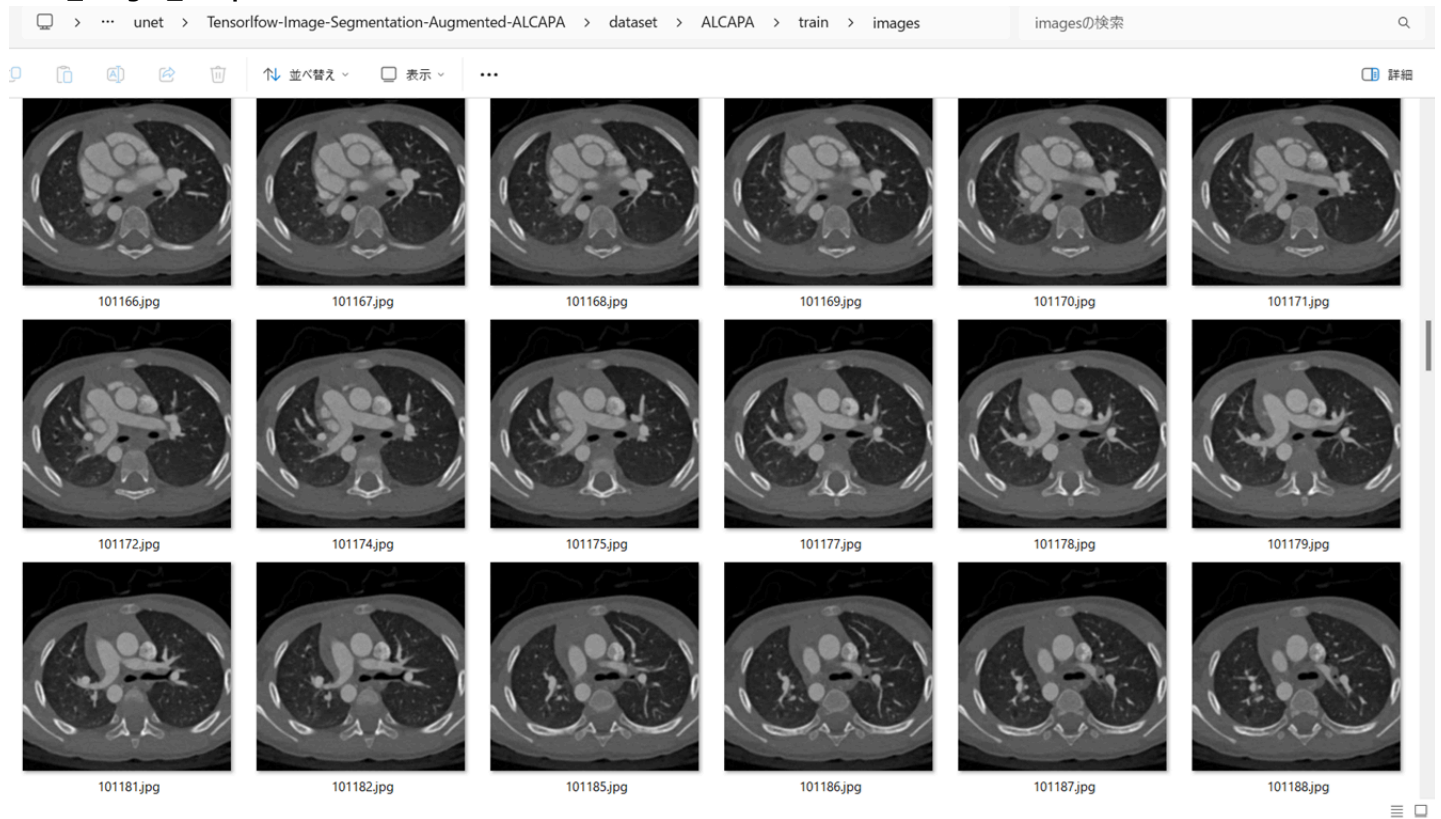
- [ImageMaskDatasetGenerator.py](#)
- [split\\_master.py](#)

## ALCAPA Dataset Statistics

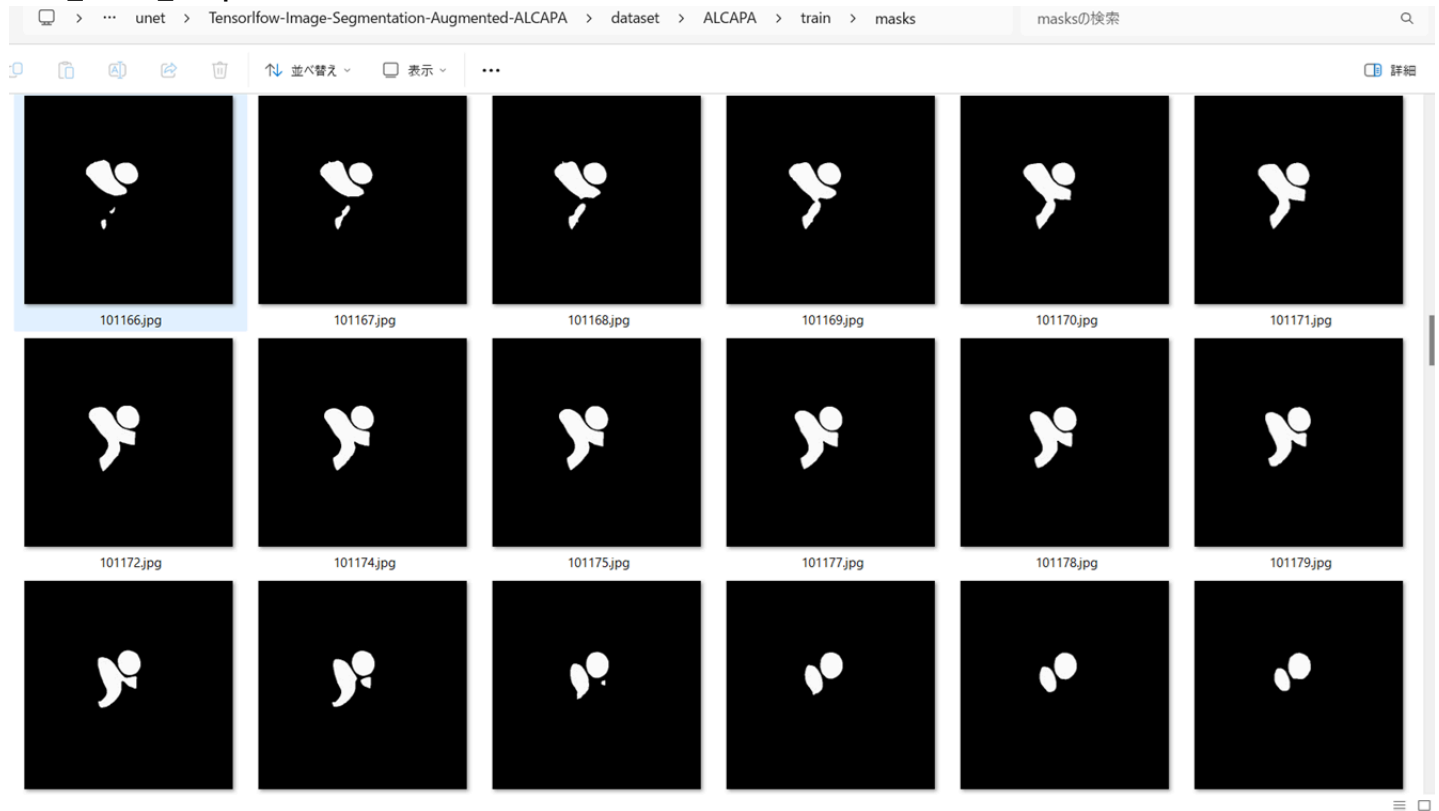


As shown above, the number of images of train and valid datasets is not large enough to use for a training set of our segmentation model. therefore we used an online augmentation tool [ImageMaskAugmentor.py](#) to improve generalization performance.

## Train\_images\_sample



## Train\_masks\_sample



### 3 Train TensorflowUNet Model

We have trained ALCAPATensorflowUNet Model by using the following [train\\_eval\\_infer.config](#) file. Please move to `./projects/TensorflowSlightlyFlexibleUNet/ALCAPA` and run the following bat file.

>1. train.bat

, which simply runs the following command.

```
>python ../../src/TensorflowUNetTrainer.py ./train_eval_infer.config
```

#### Model parameters

Defined a small **base\_filters** and large **base\_kernels** for the first Conv Layer of Encoder Block of [TensorflowUNet.py](#) and a large num\_layers (including a bridge between Encoder and Decoder Blocks).

```
[model]
base_filters = 16
base_kernels = (9,9)
num_layers = 8
```

#### Learning rate

Defined a small learning rate.

```
[model]
learning_rate = 0.0001
```

#### Online augmentation

Enabled our online augmentation tool.

```
[model]
model = "TensorflowUNet"
generator = True
```

#### Loss and metrics functions

Specified "bce\_dice\_loss" and "dice\_coef".

```
[model]
loss = "bce_dice_loss"
metrics = ["dice_coef"]
```

#### Learning rate reducer callback

Enabled learing\_rate\_reducer callback, and a small reducer\_patience.

```
[train]
learning_rate_reducer = True
reducer_factor = 0.3
reducer_patience = 4
```

#### Early stopping callback

Enabled early stopping callback with patience parameter.

```
[train]
patience      = 10
```

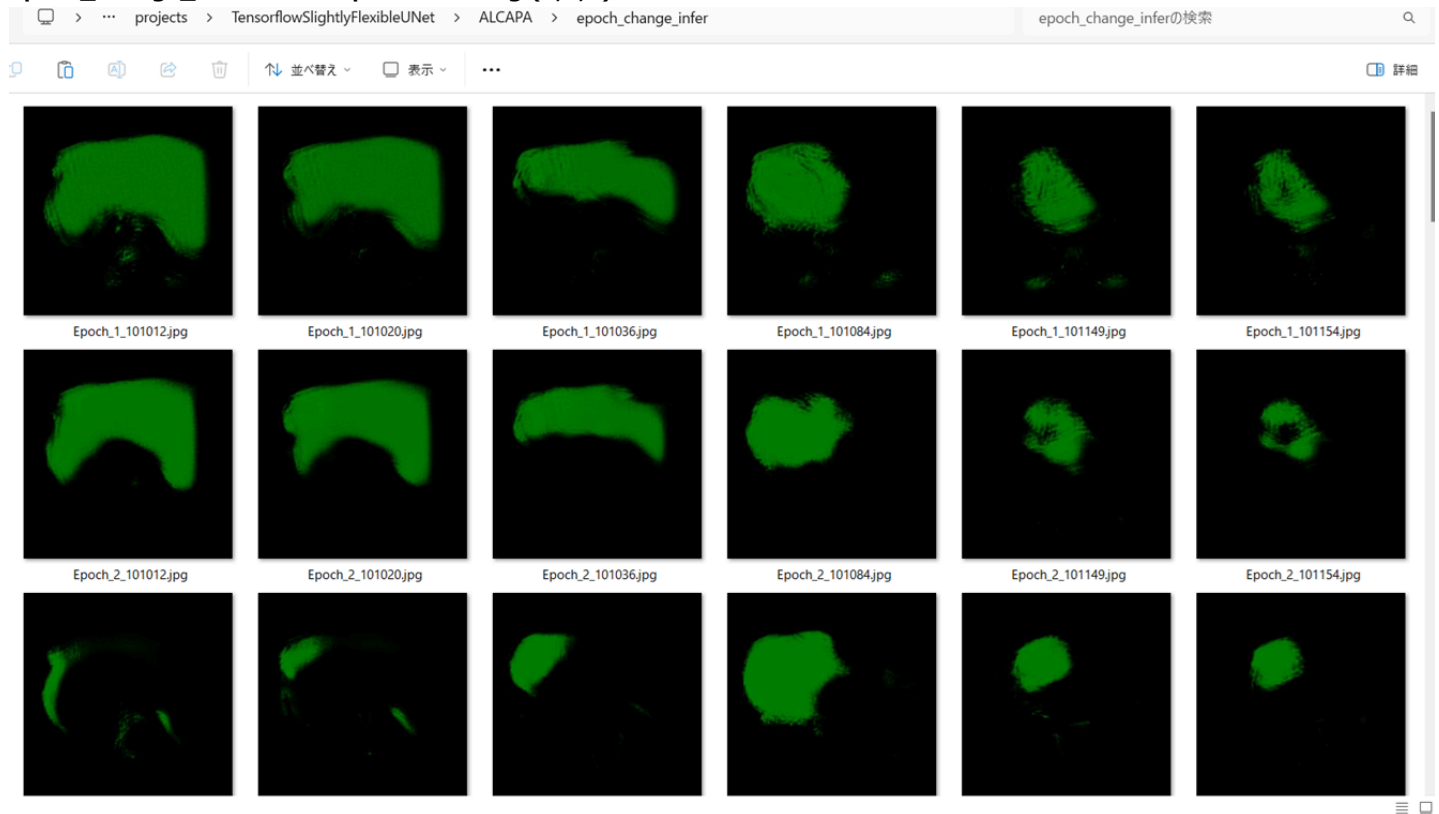
### Epoch change inference callbacks

Enabled epoch\_change\_infer callback.

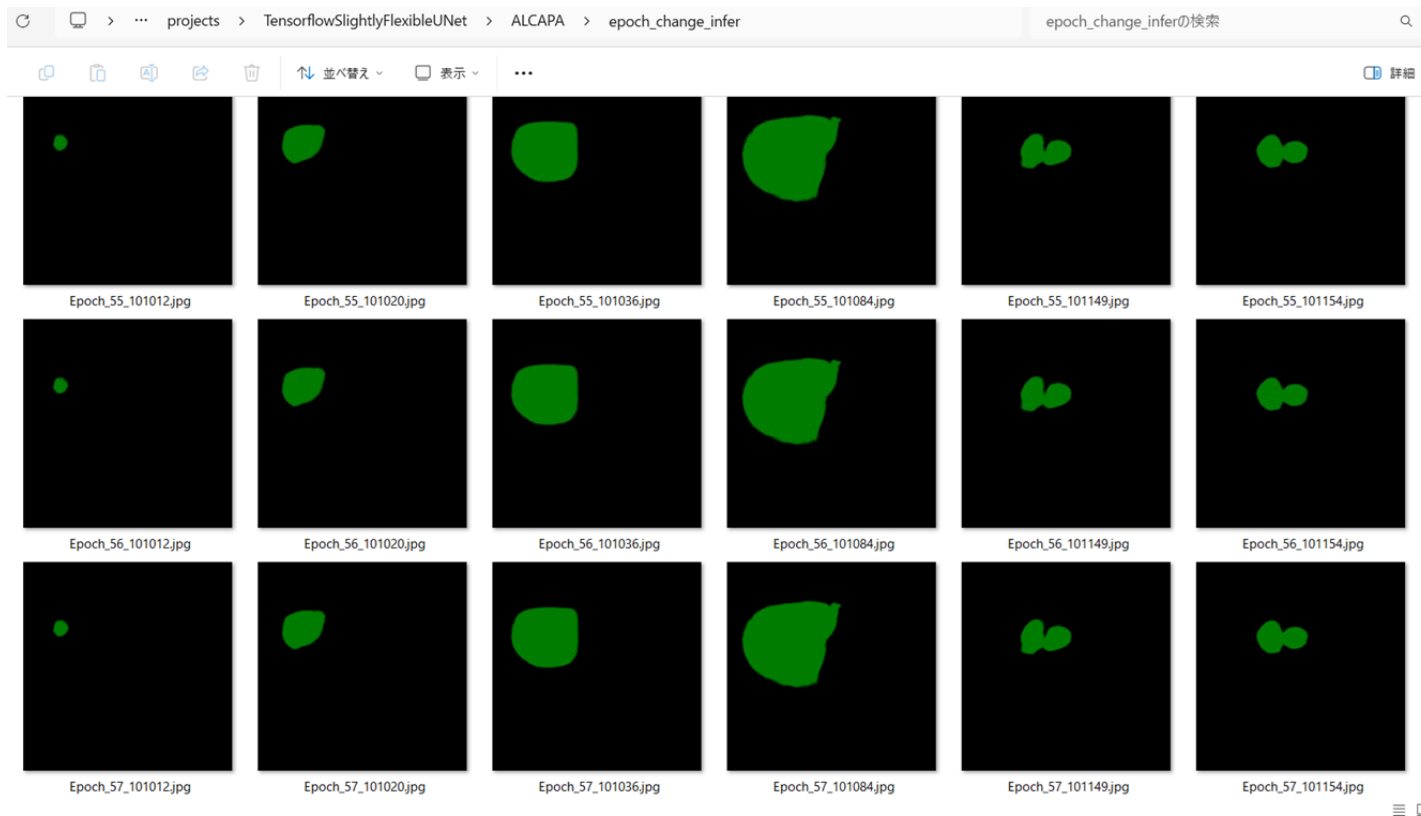
```
[train]
epoch_change_infer      = True
epoch_change_infer_dir  = "/epoch_change_infer"
epoch_changeinfer       = False
epoch_changeinfer_dir   = "/epoch_changeinfer"
num_infer_images        = 6
```

By using this callback, on every epoch\_change, the inference procedure can be called for an image in **mini\_test** folder. This will help you confirm how the predicted mask changes at each epoch during your training process.

### Epoch\_change\_inference output at starting (1,2,3)



### Epoch\_change\_inference output at ending (55,56,57)



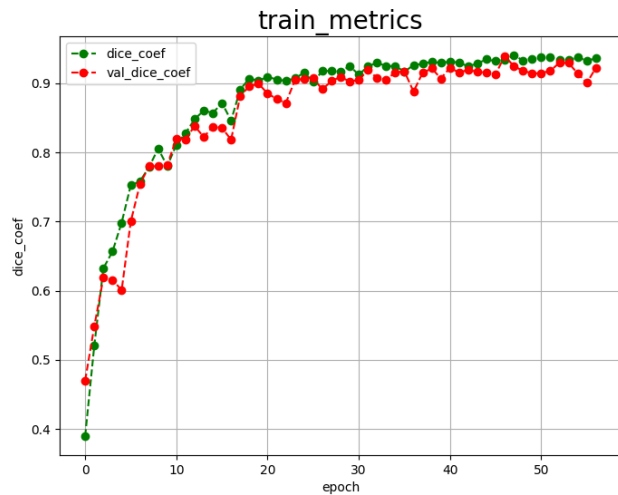
In this experiment, the training process was stopped at epoch 57 by EarlyStopping Callback.

```

PowerShell 7 (x64)
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0441 - dice_coef: 0.9318 - val_loss: 0.0528 - val_dice_coef: 0.9131 - lr: 2.7000e-06
Epoch 47/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9335
Epoch 47: val_loss improved from 0.0487 to 0.0400, saving model to ./models/best_model.h5
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9335 - val_loss: 0.0400 - val_dice_coef: 0.9394 - lr: 2.7000e-06
Epoch 48/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0401 - dice_coef: 0.9403
Epoch 48: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0401 - dice_coef: 0.9403 - val_loss: 0.0467 - val_dice_coef: 0.9245 - lr: 2.7000e-06
Epoch 49/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9330
Epoch 49: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9330 - val_loss: 0.0507 - val_dice_coef: 0.9181 - lr: 2.7000e-06
Epoch 50/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0422 - dice_coef: 0.9355
Epoch 50: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0422 - dice_coef: 0.9355 - val_loss: 0.0522 - val_dice_coef: 0.9142 - lr: 2.7000e-06
Epoch 51/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0410 - dice_coef: 0.9378
Epoch 51: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0410 - dice_coef: 0.9378 - val_loss: 0.0522 - val_dice_coef: 0.9144 - lr: 2.7000e-06
Epoch 52/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0411 - dice_coef: 0.9375
Epoch 52: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0411 - dice_coef: 0.9375 - val_loss: 0.0496 - val_dice_coef: 0.9181 - lr: 8.1000e-07
Epoch 53/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0430 - dice_coef: 0.9332
Epoch 53: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0430 - dice_coef: 0.9332 - val_loss: 0.0447 - val_dice_coef: 0.9295 - lr: 8.1000e-07
Epoch 54/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9332
Epoch 54: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9332 - val_loss: 0.0450 - val_dice_coef: 0.9292 - lr: 8.1000e-07
Epoch 55/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0412 - dice_coef: 0.9376
Epoch 55: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0412 - dice_coef: 0.9376 - val_loss: 0.0519 - val_dice_coef: 0.9140 - lr: 8.1000e-07
Epoch 56/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9326
Epoch 56: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0435 - dice_coef: 0.9326 - val_loss: 0.0587 - val_dice_coef: 0.9004 - lr: 2.4300e-07
Epoch 57/100
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0418 - dice_coef: 0.9359
Epoch 57: val_loss did not improve from 0.0400
400/400 [=====] - ETA: 0s - batch: 199.5000 - size: 2.0000 - loss: 0.0418 - dice_coef: 0.9359 - val_loss: 0.0482 - val_dice_coef: 0.9224 - lr: 2.4300e-07
Epoch 57: early stopping
=== Save history.json

```

[train\\_metrics.csv](#)



[train\\_losses.csv](#)



## 4 Evaluation

Please move to a `./projects/TensorflowSlightlyFlexibleUNet/ALCAPA` folder, and run the following bat file to evaluate TensorflowUNet model for ALCAPA.

`./2. evaluate.bat`

This bat file simply runs the following command.

`python ../../src/TensorflowUNetEvaluator.py ./train_eval_infer_aug.config`

Evaluation console output:

```
PowerShell 7 (x64)
WARNING: Not found [model] optimizer, return default value Adam
Optimizer Adam (learning_rate 0.0001 clipvalue 0.5)
loss <function bce_dice_loss at 0x000002735F20CF70>
metrics <function dice_coef at 0x000002735F20CE50>
WARNING: Not found [train] show_history, return default value False
==== ConfigParser ./train_eval_infer.config
WARNING: Not found [train] best_model_file, return default value best_model.h5
Loaded a weight file: models/best_model.h5
Dataset(class <class 'BaseImageMaskDataset.BaseImageMaskDataset'>)
BaseImageMaskDataset.constructor
==== ConfigParser ./train_eval_infer.config
WARNING: Not found [mask] algorithm, return default value None
WARNING: Not found [mask] blur_size, return default value (3, 3)
WARNING: Not found [dataset] image_format, return default value rgb
WARNING: Not found [dataset] input_normalize, return default value True
WARNING: Not found [dataset] debug, return default value True
WARNING: Not found [dataset] rgb_mask, return default value False
WARNING: Not found [dataset] color_order, return default value bgr
WARNING: Not found [dataset] mask_format, return default value gray
WARNING: Not found [mask] grayscale, return default value True
WARNING: Not found [dataset] image_normalize, return default value False
WARNING: Not found [dataset] debug, return default value False
WARNING: Not found [mask] mask_colors, return default value None
mask_colors None
num_classes 1
image_normalize False
binarize_algorithm None
WARNING: Not found [model] evaluation, return default value test
BaseImageMaskDataset.create_dataset test
create ../../dataset/ALCAPA/test/images/ ../../dataset/ALCAPA/test/masks/
WARNING: Not found [mask] mask_channels, return default value 1
num_classes 1 image_data_type <class 'numpy.uint8'>
num_images 27 512 512
====
X: shape (27, 512, 512, 3) type uint8
Y: shape (27, 512, 512, 1) type bool
create X: len: 27 Y: len: 27
WARNING: Not found [eval] batch_size, return default value 4
evaluate batch size 4
C:\Python310-eficient\lib\site-packages\keras\engine\taining_v1.py:2332: UserWarning: 'Model.state_updates' will be removed in a future version. This property should not be used in Tensor
Flow 2.0, as updates are applied automatically.
updates = self.state_updates
Test loss: 0.0568
Test accuracy: 0.9129
Evaluation metric: loss score: 0.0568
Evaluation metric: dice_coef score: 0.9129
Saved ./evaluation.csv
```



[evaluation.csv](#)

The loss (bce\_dice\_loss) to this ALCAPA/test was low, and dice\_coef high as shown below.

loss, 0.0568  
dice\_coef, 0.9129

## 5 Inference

Please move to a **./projects/TensorflowSlightlyFlexibleUNet/ALCAPA** folder

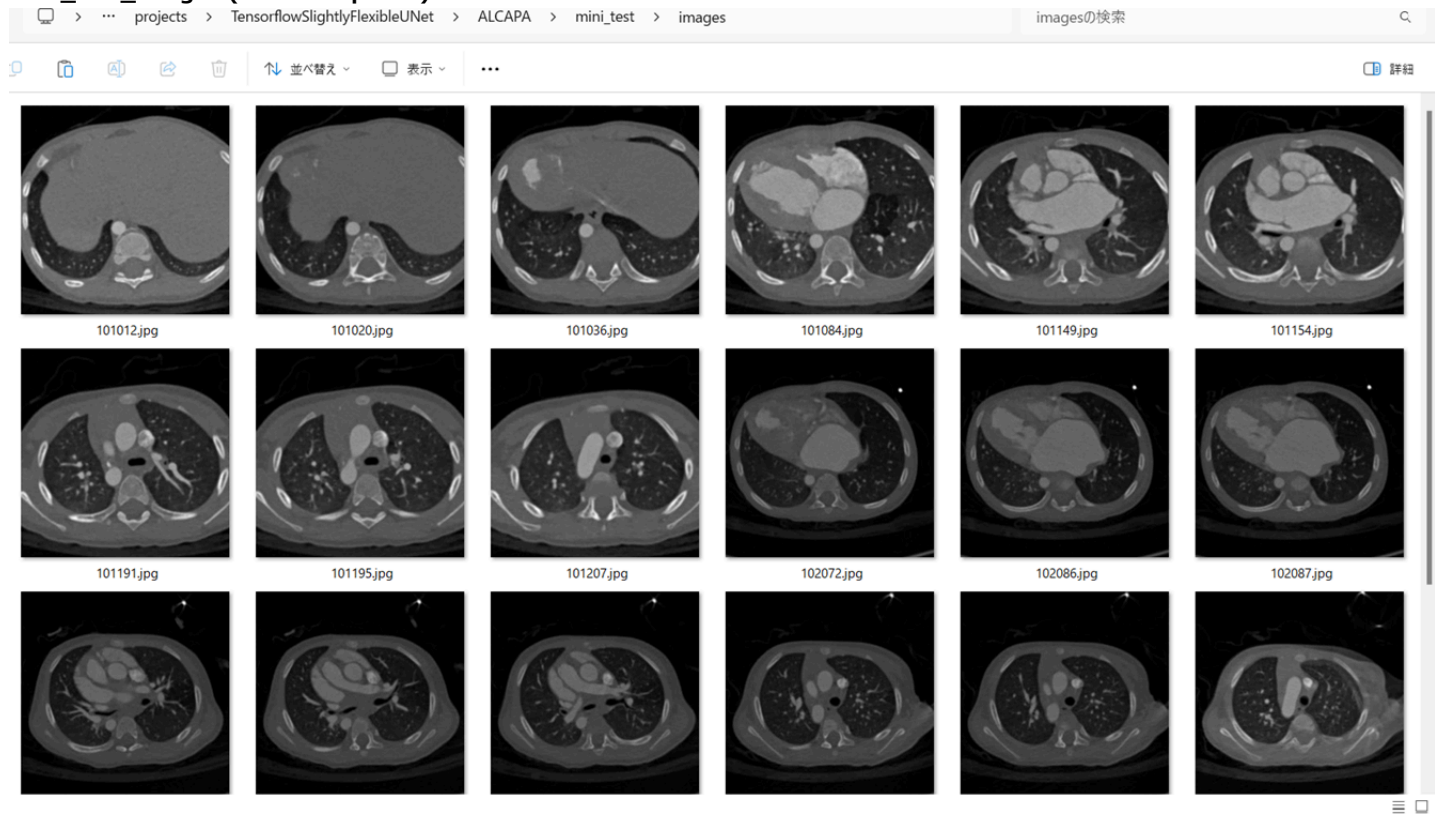
,and run the following bat file to infer segmentation regions for images by the Trained-TensorflowUNet model for ALCAPA.

./3. infer.bat

This simply runs the following command.

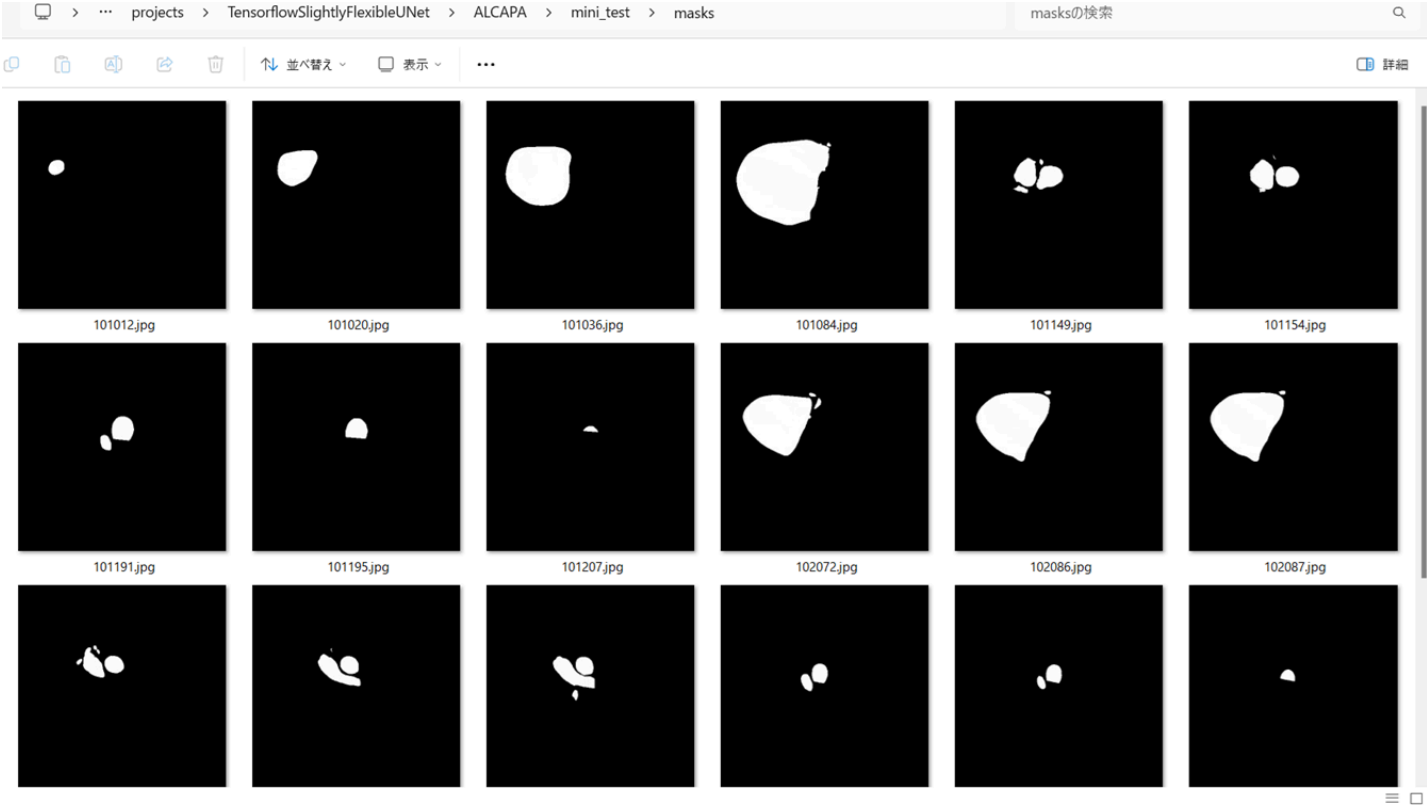
```
python ../../src/TensorflowUNetInferencer.py ./train_eval_infer_aug.config
```

### mini\_test\_images (512x512 pixels)

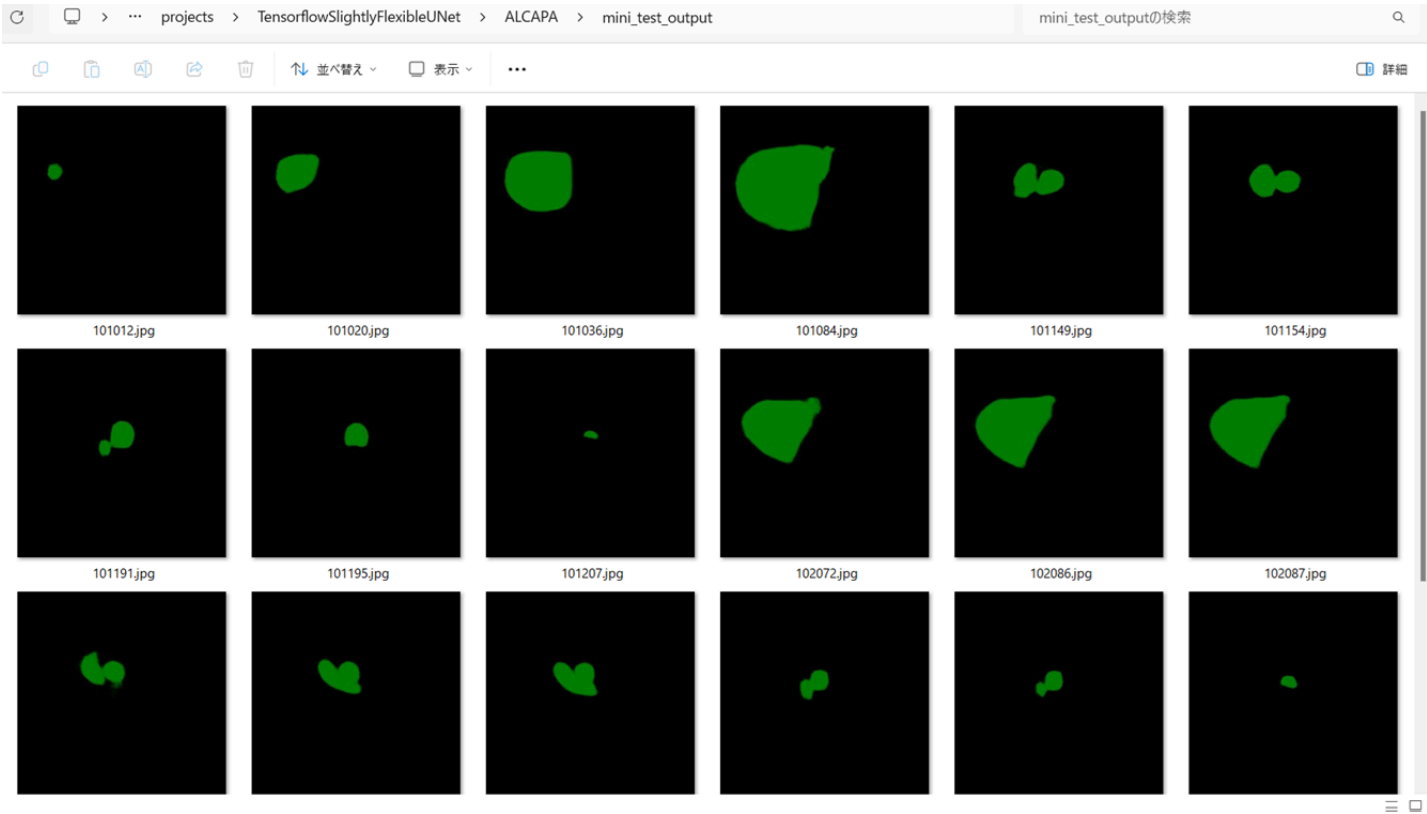




mini\_test\_mask(ground\_truth)



Inferred test masks

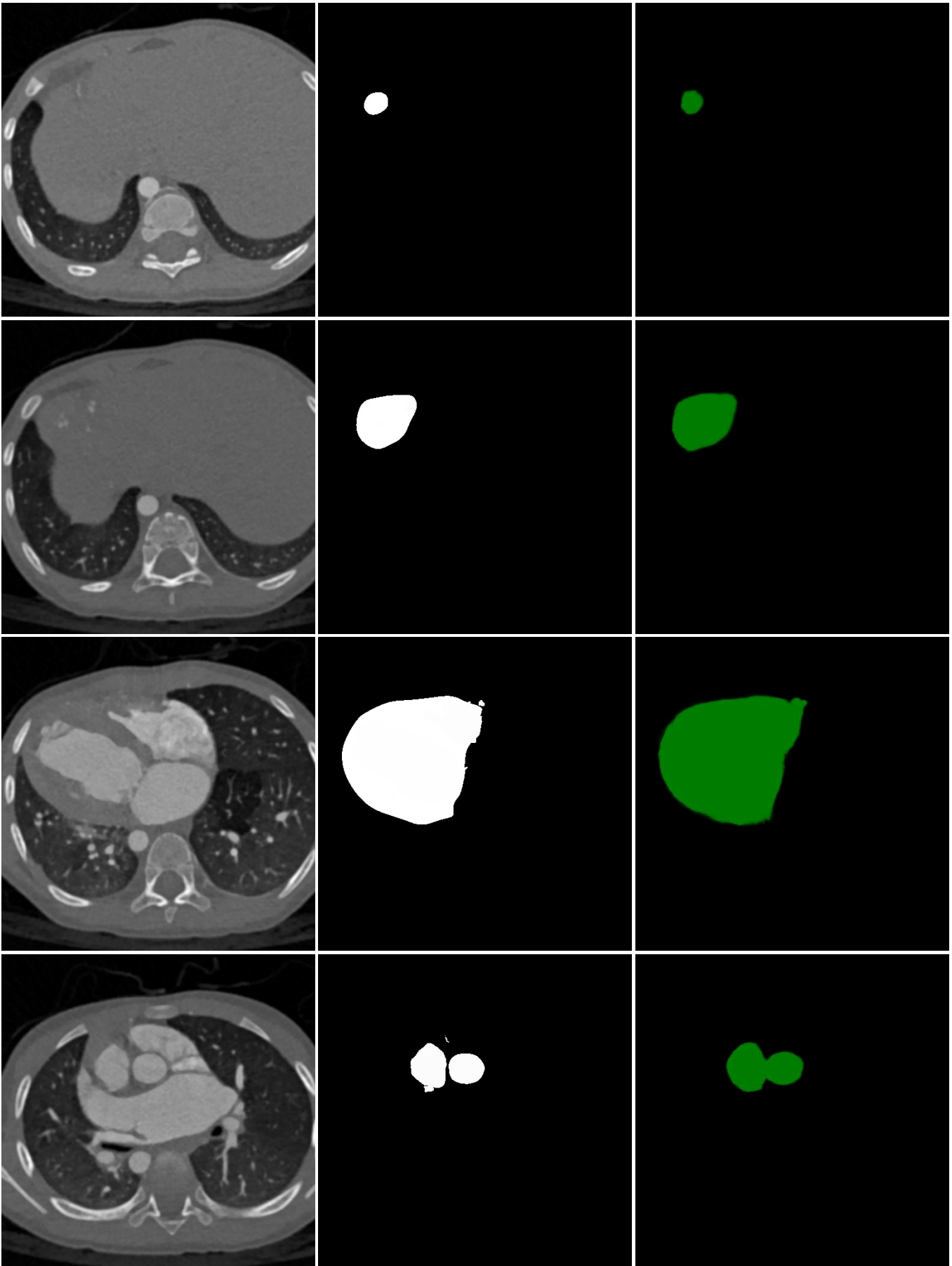


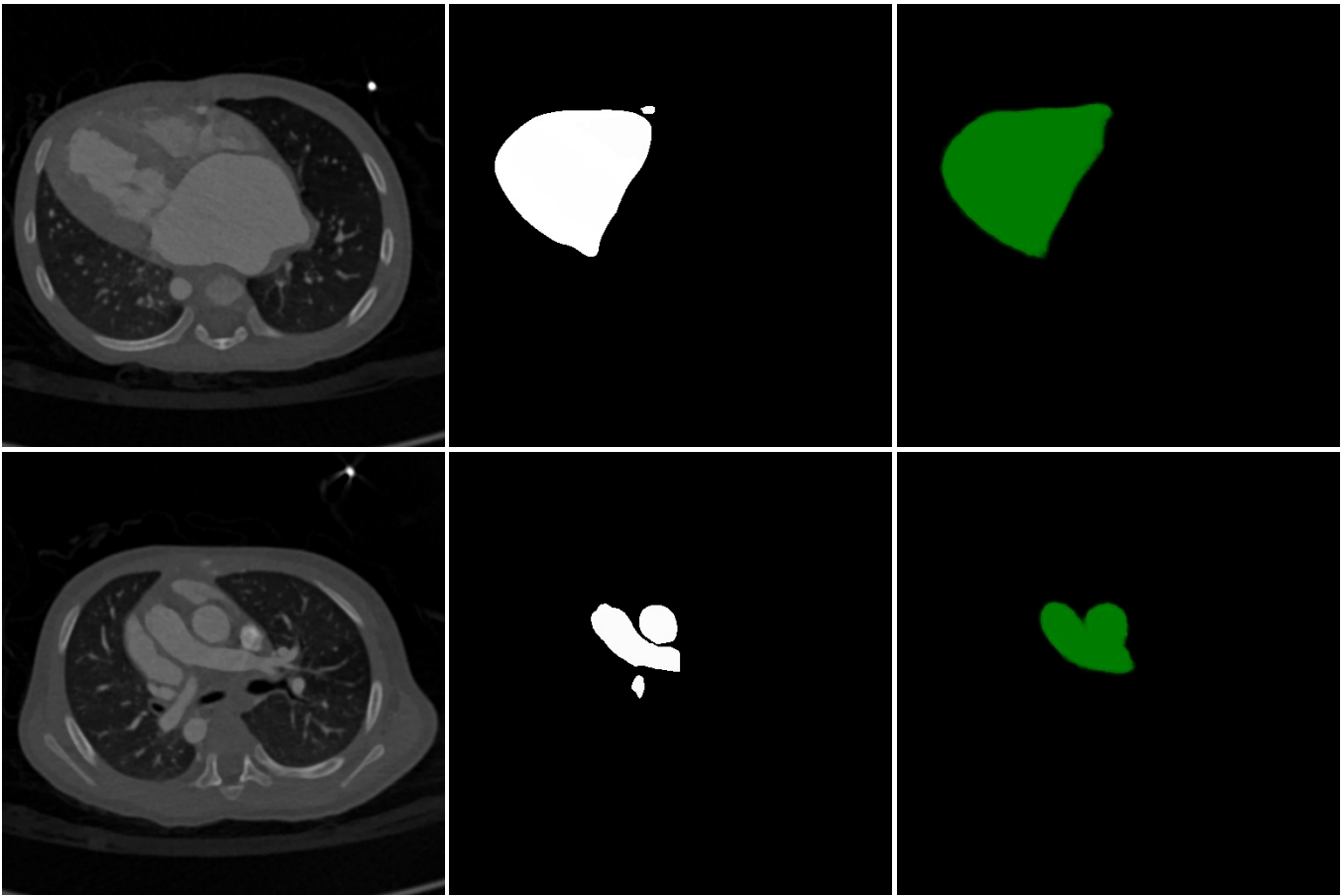
Enlarged images and masks (512x512 pixels)

Image

Mask (ground\_truth)

Inferred-mask





## References

### 1. ImageALCAPA: A 3D Computed Tomography Image Dataset for Automatic Segmentation of Anomalous Left Coronary Artery from Pulmonary Artery

A. Zeng, C. Mi, D. Pan, Q. Lu and X. Xu,  
2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM),  
Las Vegas, NV, USA, 2022, pp. 1800-1803,  
doi: 10.1109/BIBM55620.2022.9994951.  
<https://ieeexplore.ieee.org/document/9994951>

### 2. Anomalous Left Coronary Artery From the Pulmonary Artery

<https://www.chop.edu/conditions-diseases/anomalous-left-coronary-artery-pulmonary-artery>