

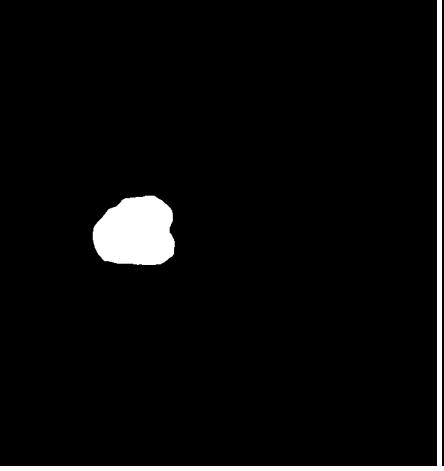
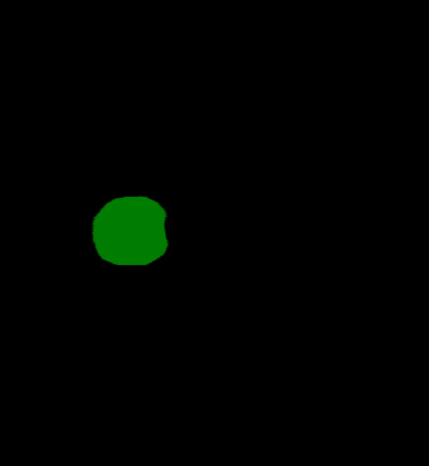
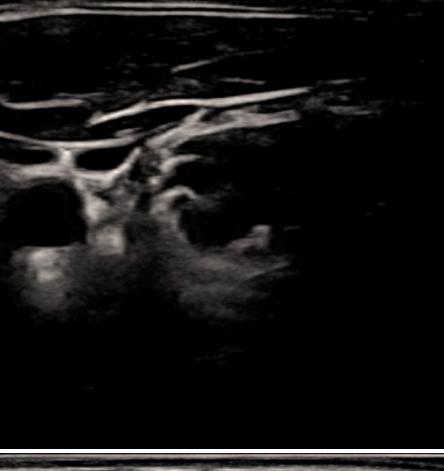
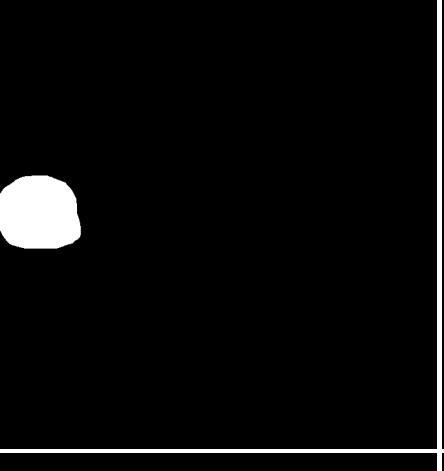
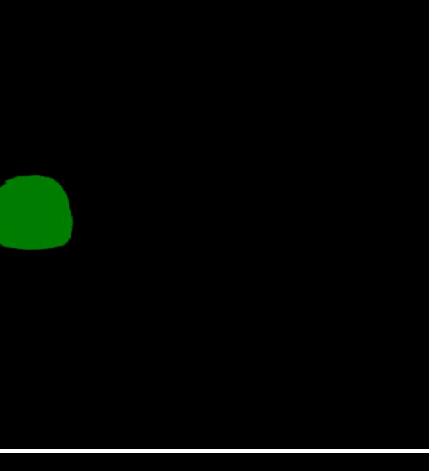
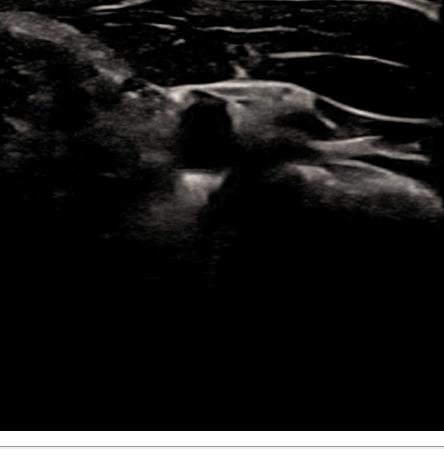
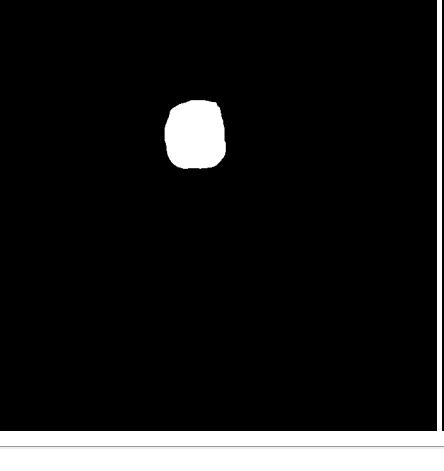
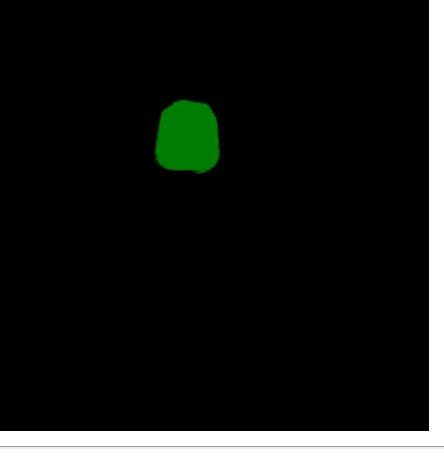
# Tensorflow-Image-Segmentation-Common-Carotid-Artery-UltraSound-Images (2025/02/17)

Sarah T. Arai  
Software Laboratory antillia.com

This is the first experiment of Image Segmentation for **CCAUS (Common Carotid Artery UltraSound) Images** based on the latest [Tensorflow-Image-Segmentation-API](#), and Mendeley [Common Carotid Artery Ultrasound Images](#)

## Actual Tiled Image Segmentation for Images of 709x749 pixels

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

Input: image	Mask (ground_truth)	Prediction: inferred_mask
		
		
		

In this experiment, we used the simple UNet Model [TensorflowSlightlyFlexibleUNet](#) for this CCAUSSegmentation 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 the following Mendeley website:

[Common Carotid Artery Ultrasound Images](#)

Momot, Agata (2022), "Common Carotid Artery Ultrasound Images",

Mendeley Data, V1, doi: 10.17632/d4xt63mgjm.1

### Description

The dataset consists of aquisitioned ultrasound images of the common carotid artery. The images were taken from a Mindray UMT-500Plus ultrasound machine with an L13-3s linear probe. The study group consisted of 11 subjects, with each person examined at least once on the left and right sides. 2 subjects were examined using the vascular modality and 8 using the carotid modality. Time series (DICOM) were converted to png files and cropped appropriately. Each person had 100 images, making a total of 1100 images. The dataset also includes corresponding expert masks (corresponding file name) made by a technician and verified by an expert. The collection can be used for carotid artery segmentation and geometry measurement and evaluation.

Image resolution: 709 x 749 x 3

Number of images: 2200

File format: PNG

### License:

[Creative Commons Attribution 4.0 International License.](#)

## 2 CCAUS ImageMask Dataset

If you would like to train this CCAUS Segmentation model by yourself, please download the dataset [Common Carotid Artery Ultrasound Images](#)

The folder structure of the dataset is the following.

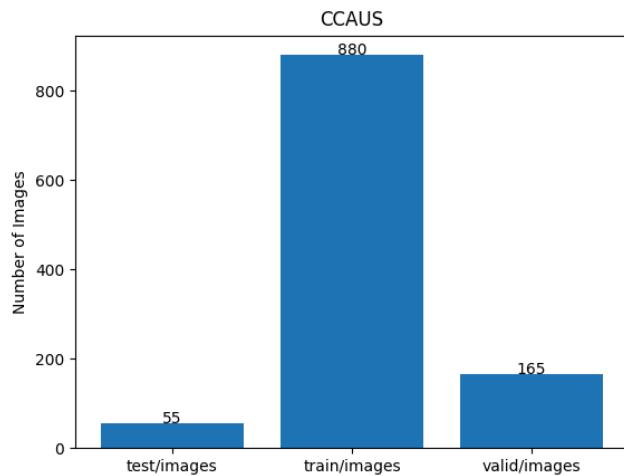
```
./Common Carotid Artery Ultrasound Images
  └── Expert mask images
      ├── 202201121748100022VAS_slice_1069.png
      ├── 202201121748100022VAS_slice_1080.png
      ...
      └── 202202071359200056VAS_slice_258.png
  └── US images
      ├── 202201121748100022VAS_slice_1069.png
      ├── 202201121748100022VAS_slice_1080.png
      ...
      └── 202202071359200056VAS_slice_258.png
```

Please run the following Python script to split the original 709x749 pixels dataset into **test**, **train** and **valid** subsets.

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

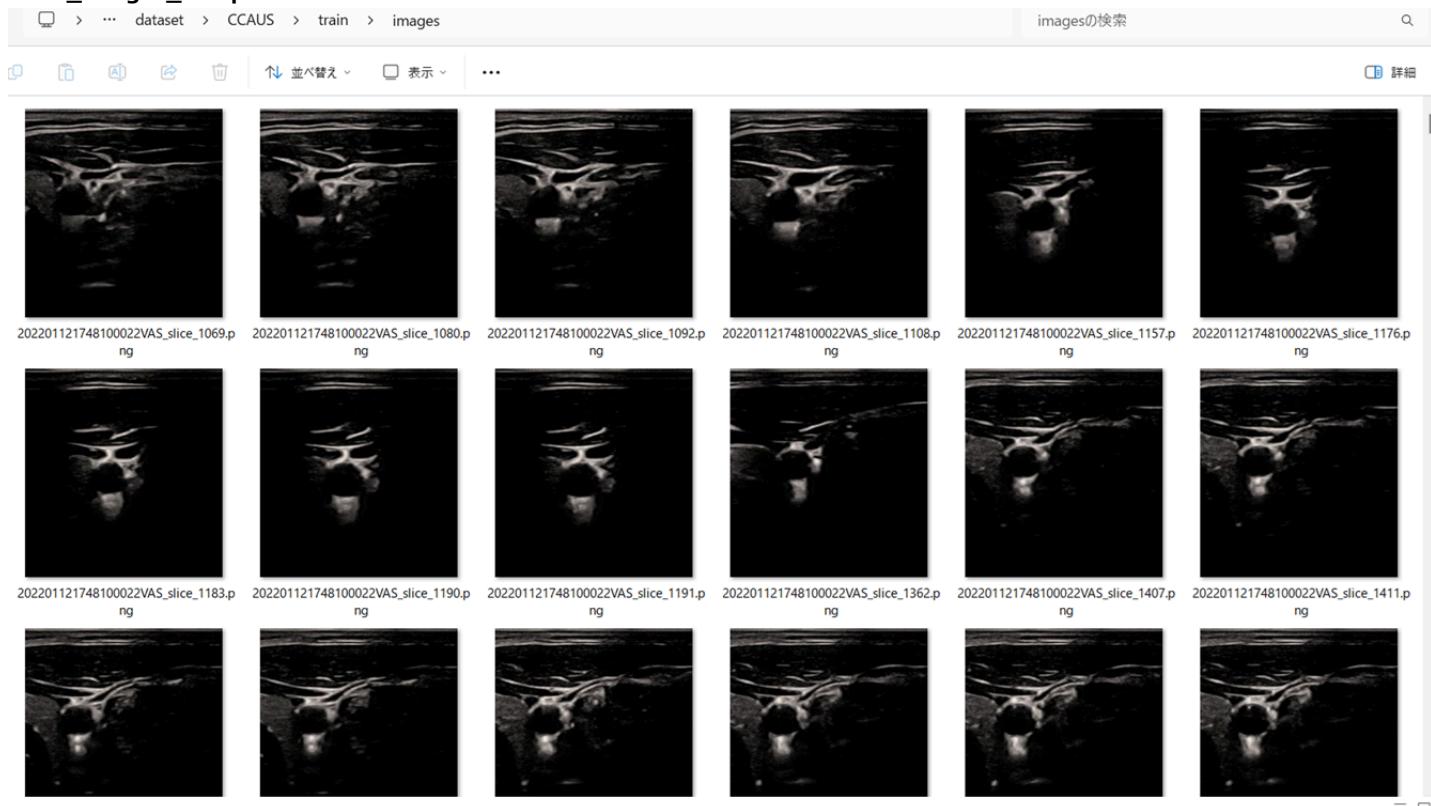
```
./dataset
  └── CCAUS
      ├── test
      │   ├── images
      │   └── masks
      ├── train
      │   ├── images
      │   └── masks
      └── valid
          ├── images
          └── masks
```

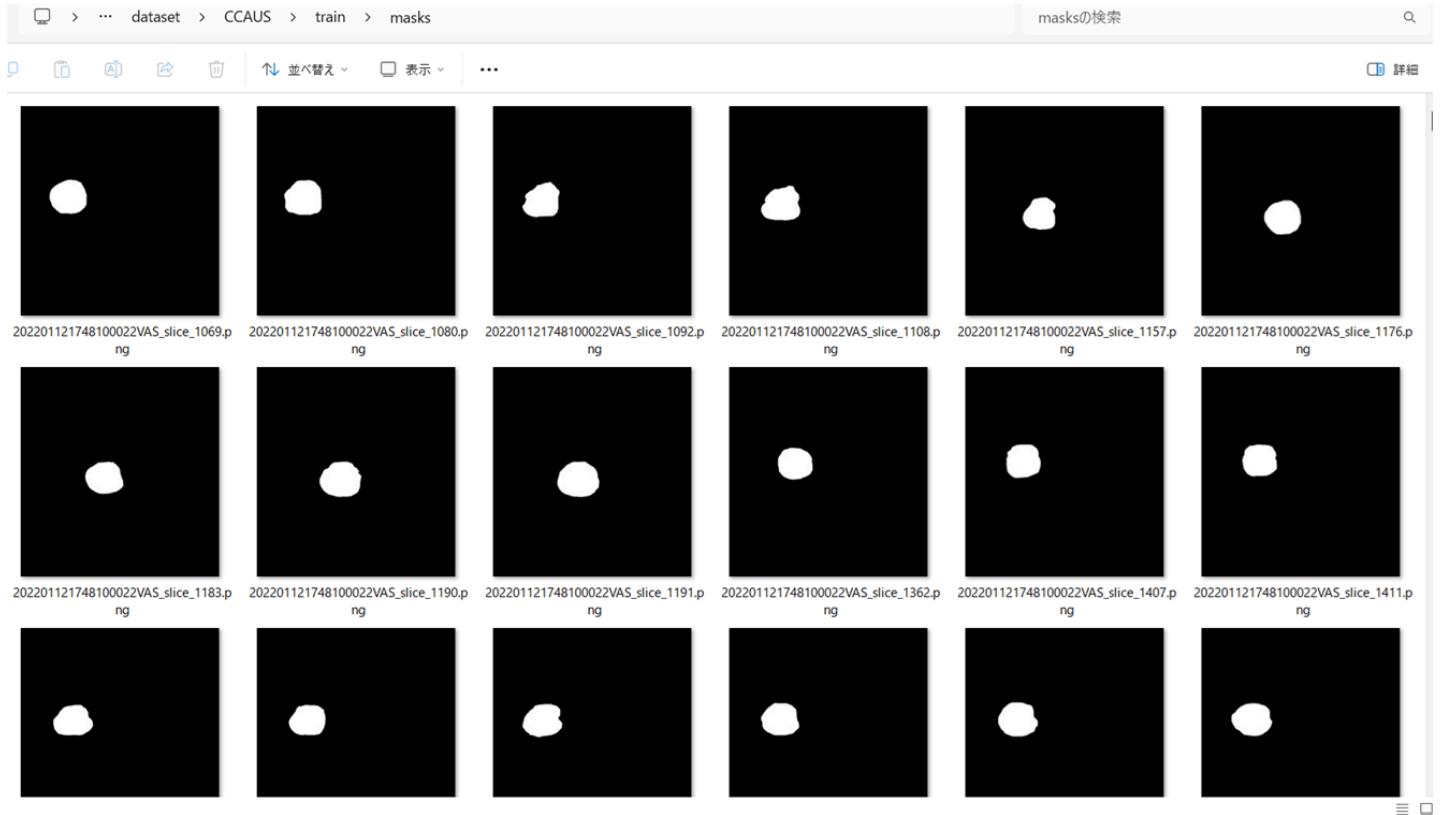
## CCaus Statistics



As shown above, the number of images of train and valid datasets is not so large to use for a training set of our segmentation model.

### Train\_images\_sample



**Train\_masks\_sample****3 Train TensorflowUNet Model**

We have trained CCAUS TensorflowUNet Model by using the following [train\\_eval\\_infer.config](#) file.

Please move to ./projects/TensorflowSlightlyFlexibleUNet/CCAUSSand run the following bat file.

>1. train.bat

, which simply runs the following command.

>python ../../src/TensorflowUNetTrainer.py ./train\_eval\_infer.config

**Model parameters**

Enabled Batch Normalization.

Defined a small **base\_filters=16** and large **base\_kernels=(9,9)** 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
dilation = (3, 3)
```

**Learning rate**

Defined a small learning rate.

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

**Online augmentation**

Disabled our online augmentation tool.

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

**Loss and metrics functions**

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

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

**Learning rate reducer callback**

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

```
[train]
learning_rate_reducer = True
reducer_factor = 0.4
reducer_patience = 4
```

## Dataset class

Specified ImageMaskDataset class.

```
[dataset]
datasetclass = "ImageMaskDataset"
resize_interpolation = "cv2.INTER_CUBIC"
```

## Early stopping callback

Enabled early stopping callback with patience parameter.

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

## Epoch change inference callbacks

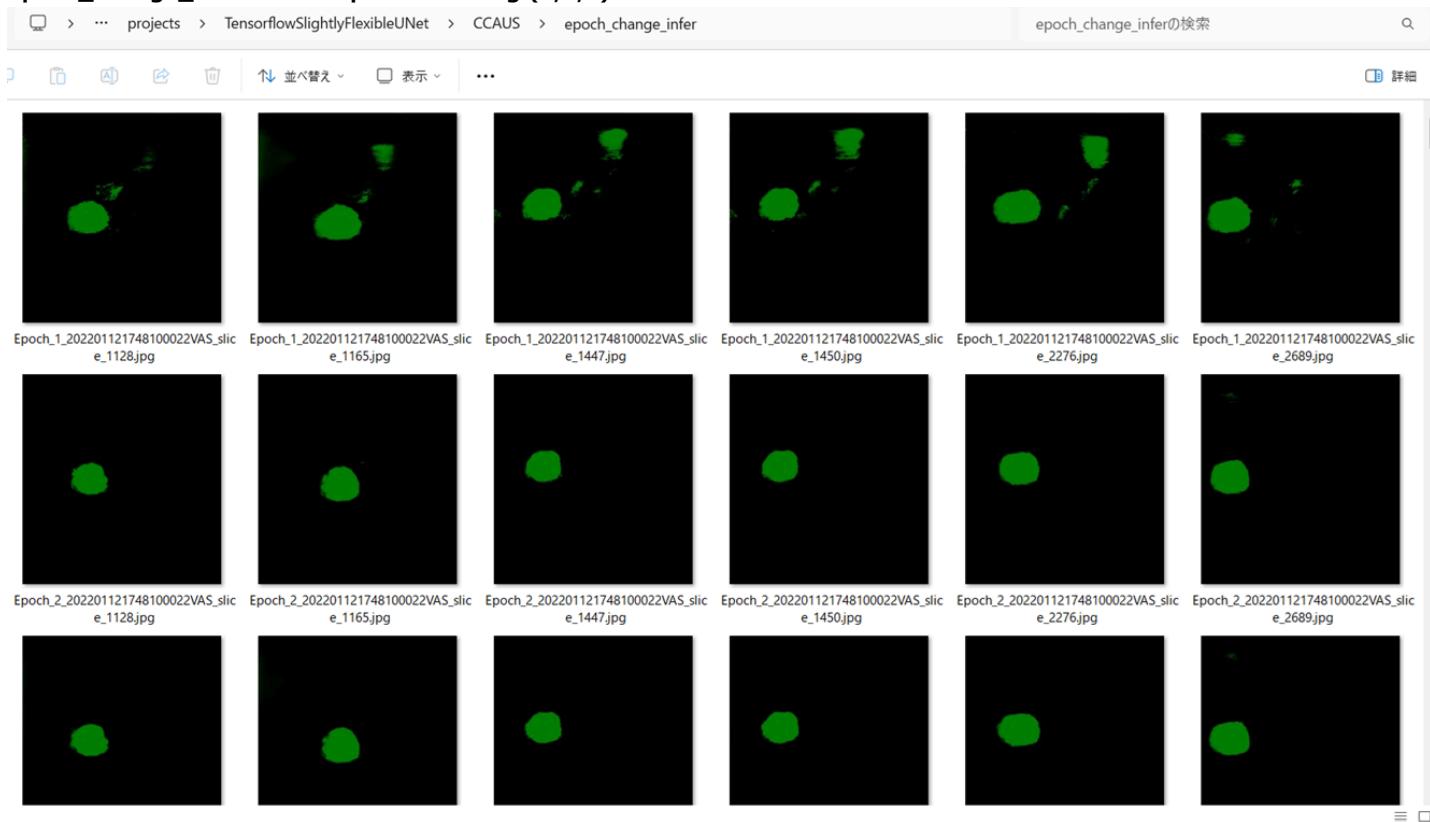
Enabled epoch\_change\_infer callback.

```
[train]
```

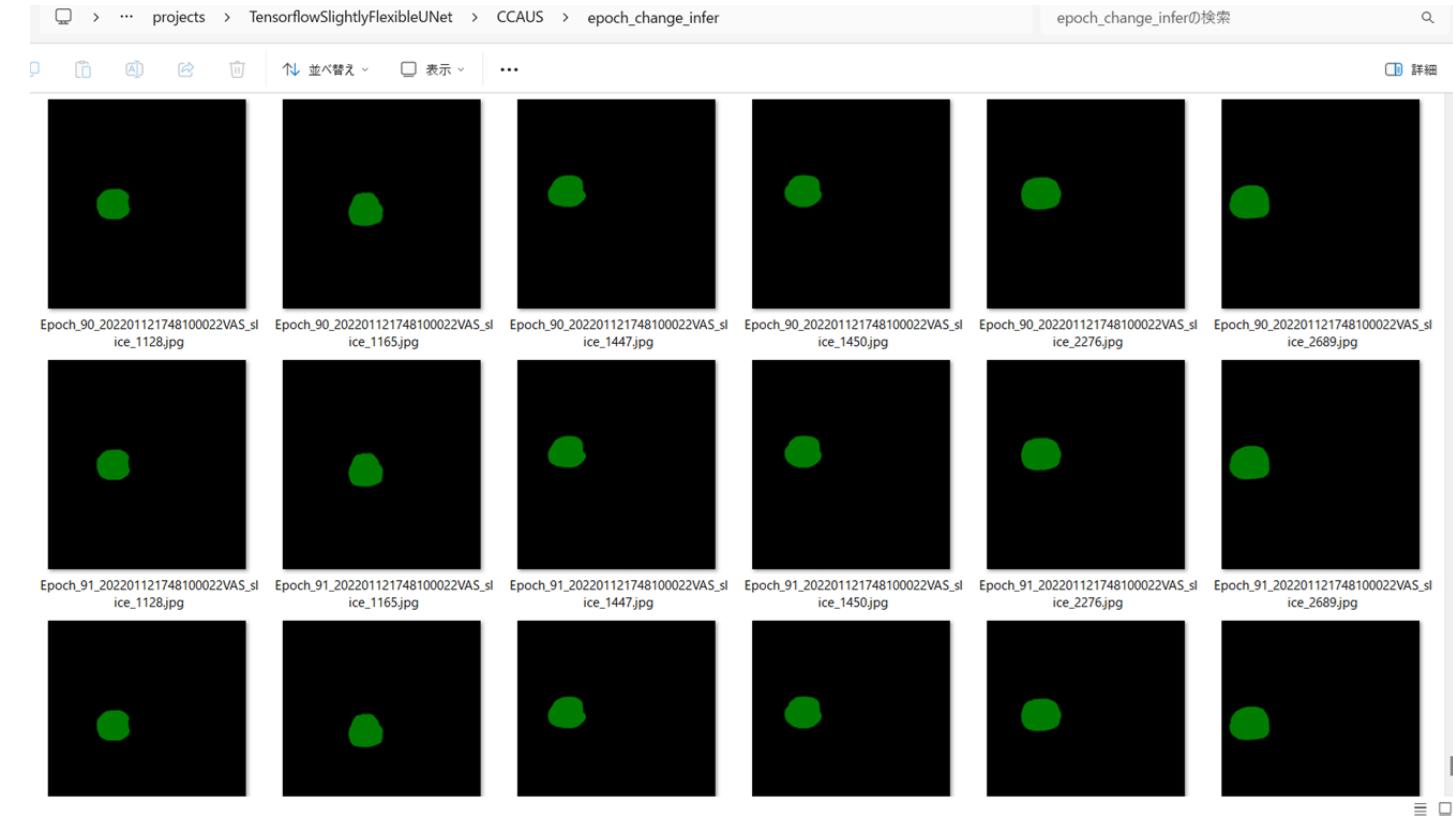
```
epoch_change_infer = True
epoch_change_infer_dir = "./epoch_change_infer"
epoch_change_tiledinfer = False
epoch_change_tiledinfer_dir = "./epoch_change_tiledinfer"
num_infer_images = 6
```

By using this callback, on every epoch\_change, the epoch change tiledinfer procedure can be called for 6 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)



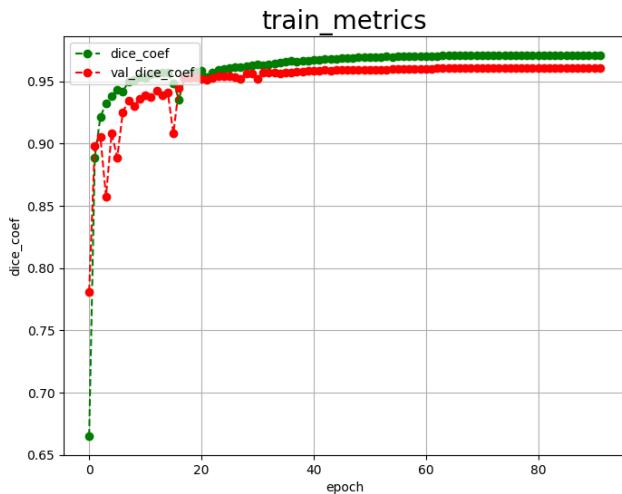
## Epoch\_change\_inference output at ending (90,91,92)



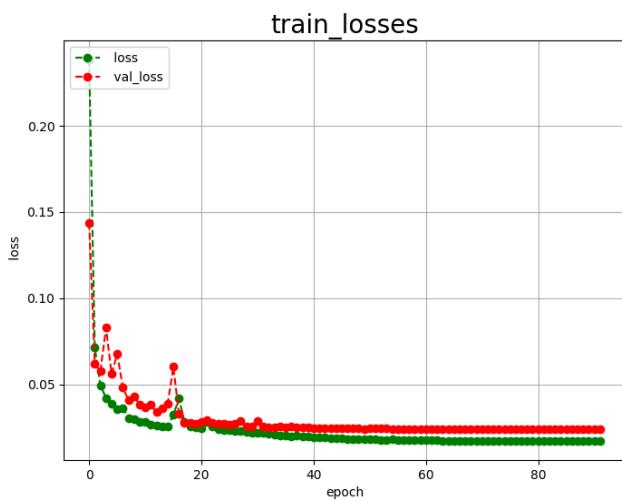
In this experiment, the training process was stopped at epoch 92 by EarlyStopping Callback.

```
PowerShell 7 (x64) + - x
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 82: val_loss improved from 0.02373 to 0.02373, saving model to ./models/best_model.h5
880/880 [=====] - 362s 412ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0237 - val_dice_coef: 0.9608 - lr: 2.6214e-08
Epoch 83/100
880/880 [=====] - ETA: 0s - loss: 0.0172 - dice_coef: 0.9707
Epoch 84/100 [=====] - val_loss did not improve from 0.02373
880/880 [=====] - 361s 410ms/sample - loss: 0.0172 - dice_coef: 0.9707 - val_loss: 0.0238 - val_dice_coef: 0.9605 - lr: 2.6214e-08
Epoch 84/100
Epoch 84: val_loss did not improve from 0.02373
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
880/880 [=====] - 357s 406ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9604 - lr: 2.6214e-08
Epoch 85/100
880/880 [=====] - ETA: 0s - loss: 0.0172 - dice_coef: 0.9707
Epoch 85: val_loss did not improve from 0.02373
880/880 [=====] - 356s 404ms/sample - loss: 0.0172 - dice_coef: 0.9707 - val_loss: 0.0237 - val_dice_coef: 0.9608 - lr: 1.0486e-08
Epoch 86/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 86: val_loss did not improve from 0.02373
880/880 [=====] - 356s 404ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9606 - lr: 1.0486e-08
Epoch 87/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 87: val_loss did not improve from 0.02373
880/880 [=====] - 358s 407ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9606 - lr: 1.0486e-08
Epoch 88/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 88: val_loss did not improve from 0.02373
880/880 [=====] - 358s 407ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0239 - val_dice_coef: 0.9605 - lr: 1.0486e-08
Epoch 89/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 89: val_loss did not improve from 0.02373
880/880 [=====] - 360s 409ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9606 - lr: 4.1943e-09
Epoch 90/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 90: val_loss did not improve from 0.02373
880/880 [=====] - 362s 411ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9606 - lr: 4.1943e-09
Epoch 91/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 91: val_loss did not improve from 0.02373
880/880 [=====] - 361s 411ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9604 - lr: 4.1943e-09
Epoch 92/100
880/880 [=====] - ETA: 0s - loss: 0.0171 - dice_coef: 0.9708
Epoch 92: val_loss did not improve from 0.02373
880/880 [=====] - 363s 413ms/sample - loss: 0.0171 - dice_coef: 0.9708 - val_loss: 0.0238 - val_dice_coef: 0.9605 - lr: 4.1943e-09
Epoch 92: early_stopping
Save history.json
```

[train\\_metrics.csv](#)



[train\\_losses.csv](#)



## 4 Evaluation

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

`./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) + - x
--- WARNING: Not found [train] show.history, return default value False
--- ConfigParser._read_file_infer.config
--- WARNING: Not found [train] best.model_file, return default value best.model.h5
--- Loaded a weight file ./models/best.model.h5
--- DatasetClass <class 'ImageMaskDataset.ImageMaskDataset'>
--- BaseImageMaskDataset.constructor
--- BaseImageMaskDataset._load_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] rgbs_mask, return default value False
--- WARNING: Not found [dataset] color_order, return default value bgr
--- WARNING: Not found [dataset] gray_color, 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
initalize_algorithm None
ImageMaskDataset.constructor
self.resize_interpolation 2
--- WARNING: Not found [model] evaluation, return default value test
BaseImageMaskDataset.create_dataset test
create ../../dataset/CCaus/test/images/ ../../dataset/CCaus/test/masks/
--- WARNING: Not found [mask] mask_channels, return default value 1
num_classes 1 image data type <class 'numpy.uint8'>
num_images 55 512 512
100% | 55/55 [00:00<00:00, 64.04it/s]
X: shape (55, 512, 512, 3) type uint8
--- Y: shape (55, 512, 512, 1) type bool
--- Create X:len: 55 Y:len: 55
--- WARNING: Not found [eval] batch_size, return default value 4
--- eval batch_size 4
E:\Python310\lib\site-packages\keras\engine\training_v1.py:232: UserWarning: 'Model.state_updates' will be removed in a future version. This property should not be used in TensorFlow 2.0, as 'updates' are applied automatically.
  updates = self.state_updates
Test loss :0.0257
Test accuracy:0.9583
Evaluation metric:loss score:0.0257
Evaluation metric:dice_coef score:0.9583
Saved ./evaluation.csv
```

### Image-Segmentation-CCaus [evaluation.csv](#)

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

```
loss,0.0257
dice_coef,0.9583
```

## 5 Inference

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

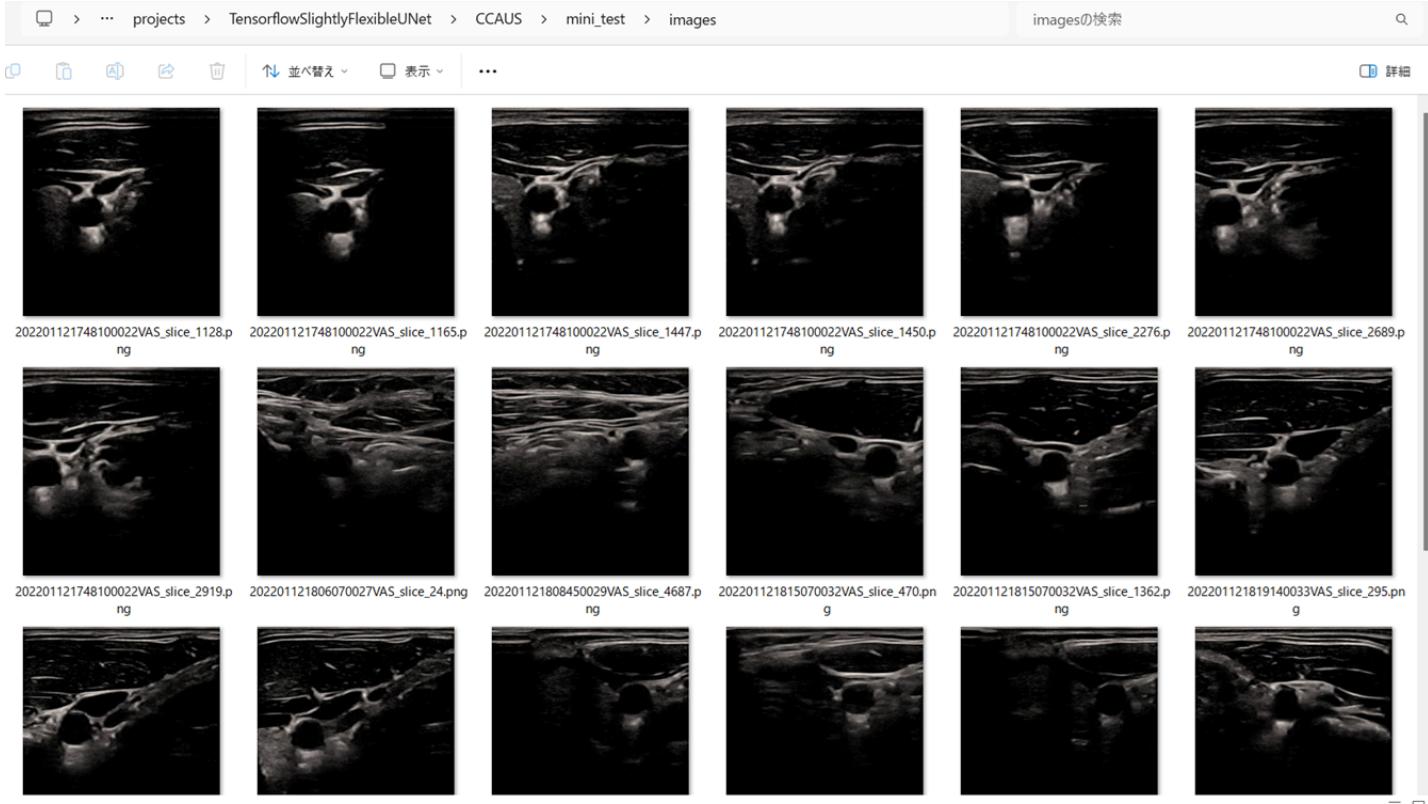
,and run the following bat file to infer segmentation regions for images by the Trained-TensorflowUNet model for CCAUS.

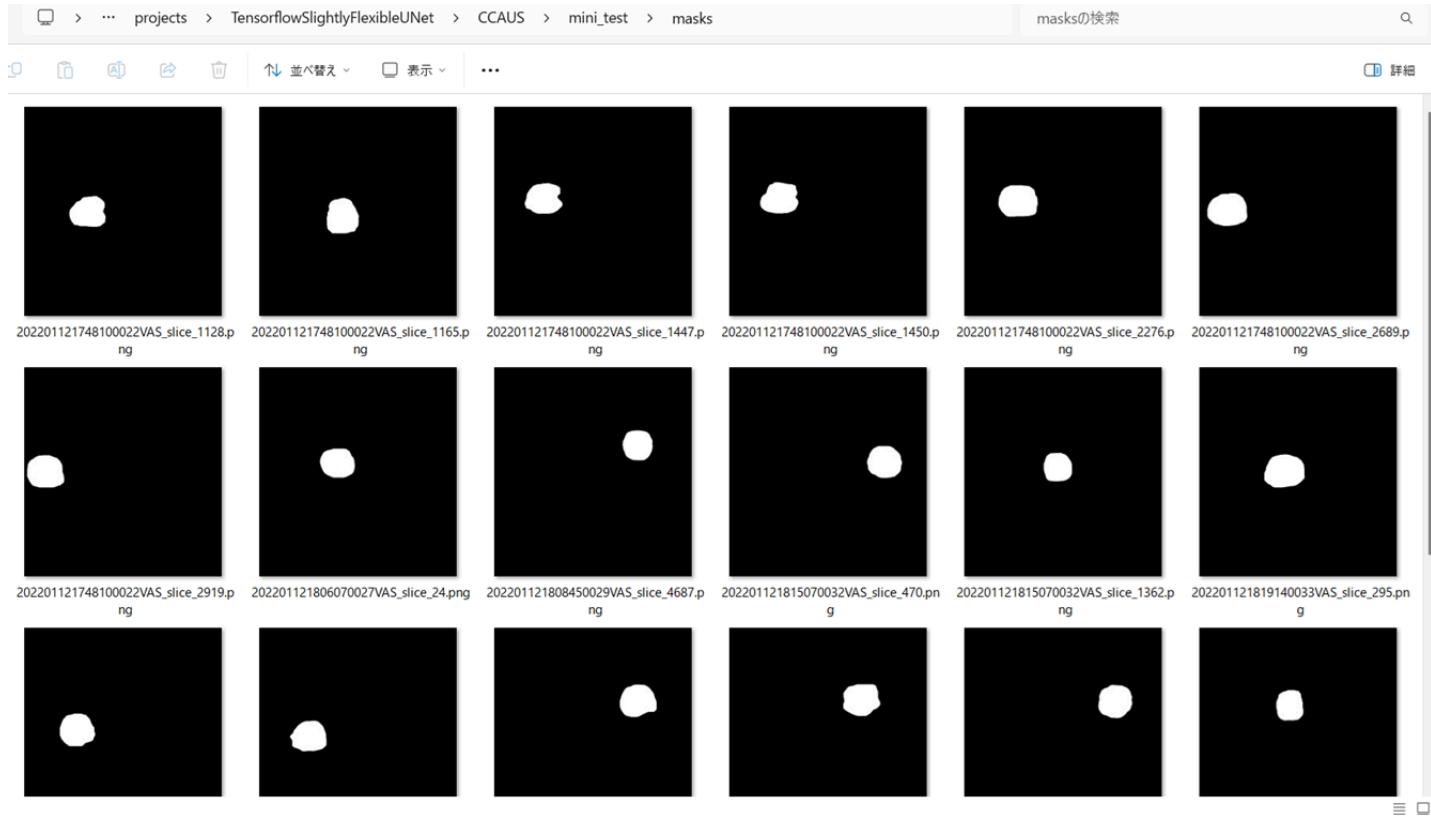
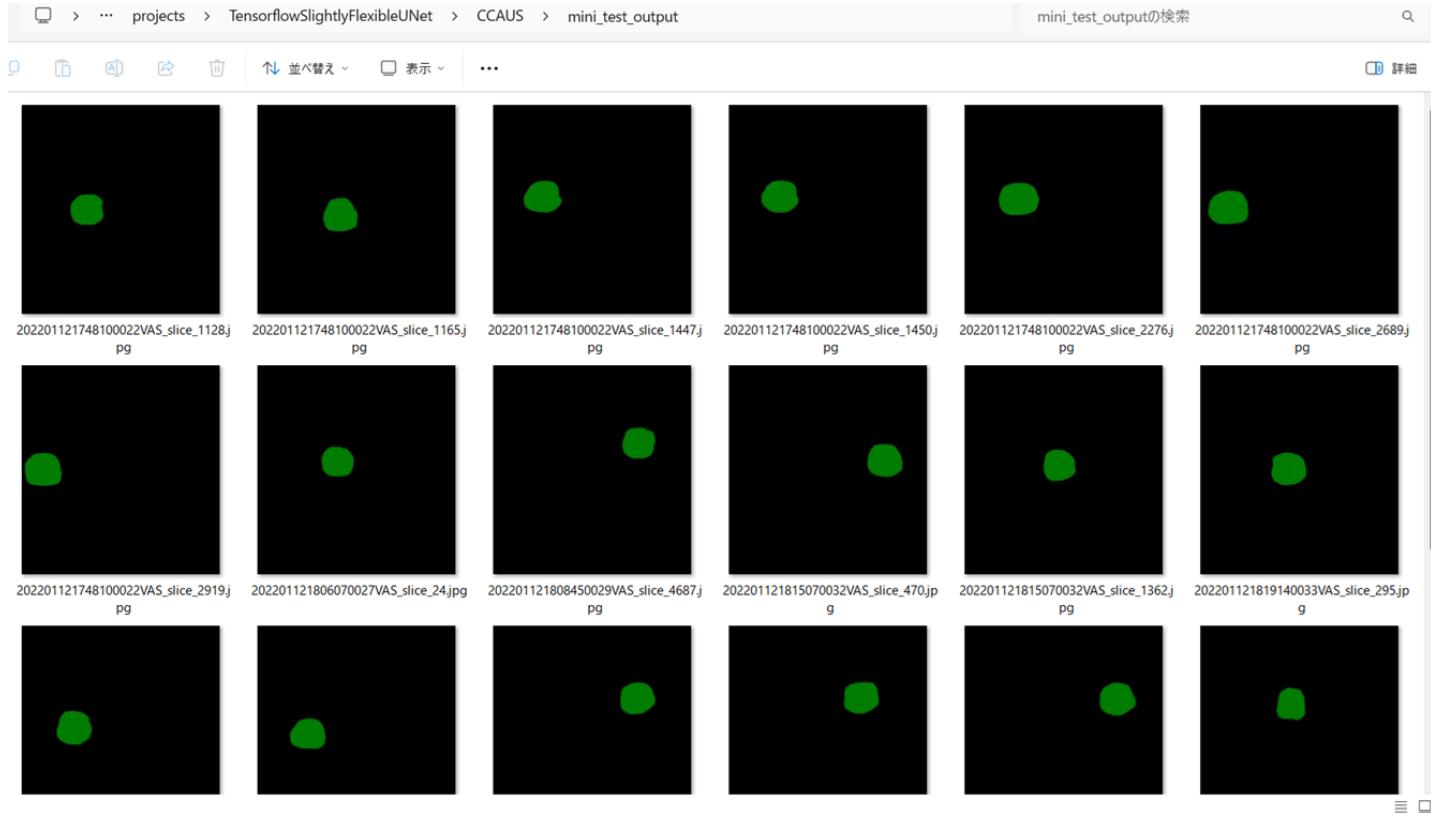
```
./3.infer.bat
```

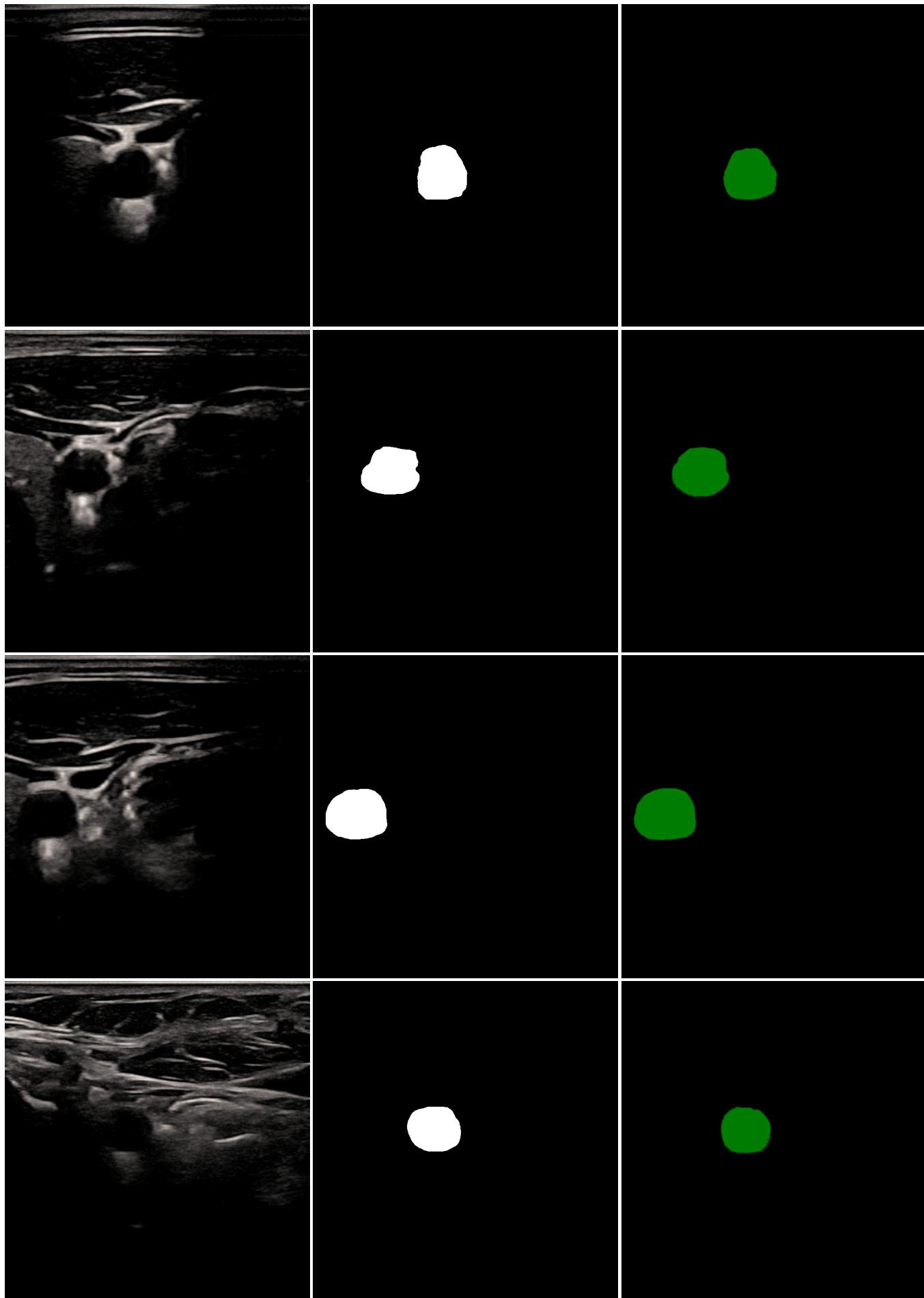
This simply runs the following command.

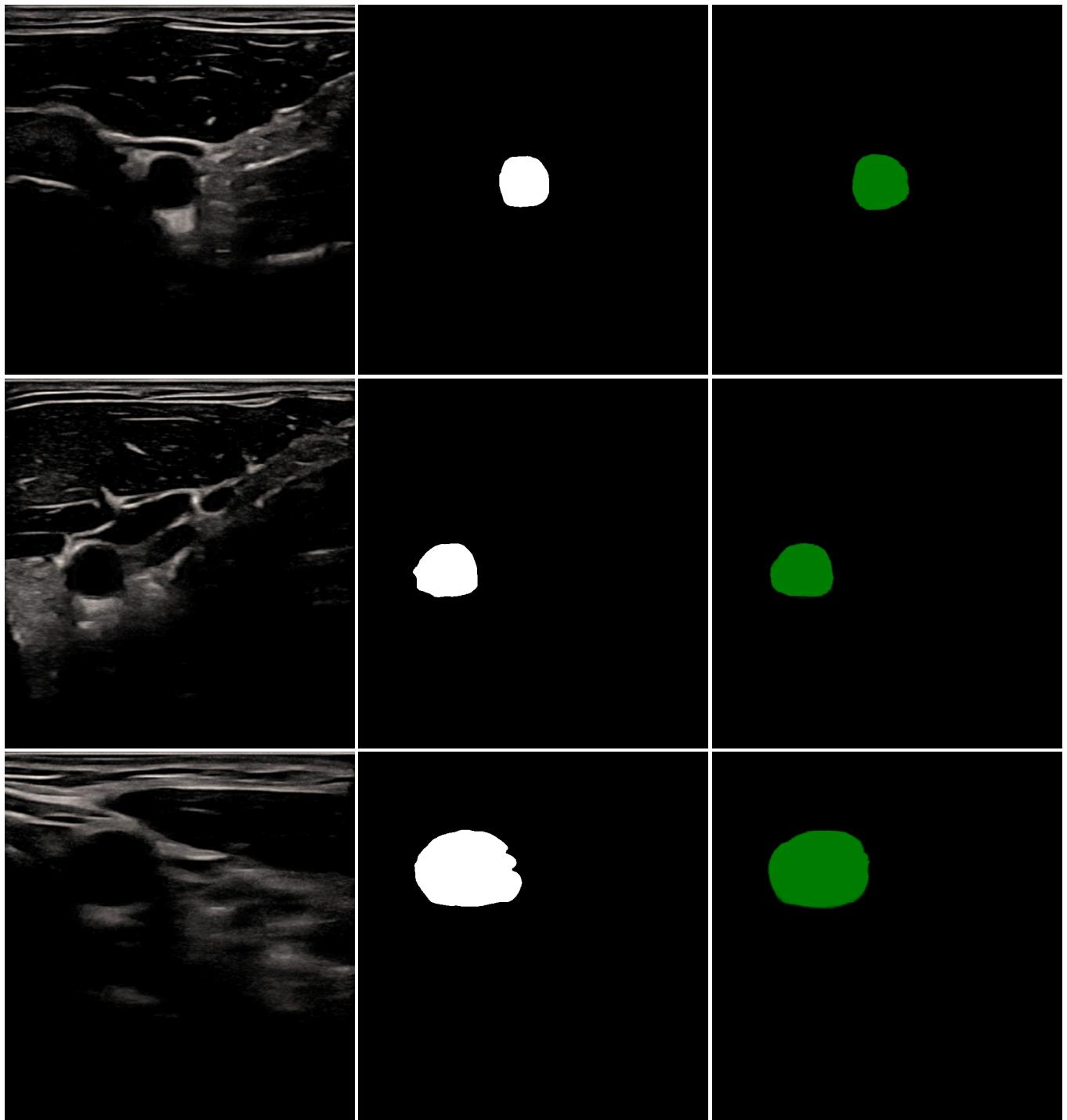
```
python ../../src/TensorflowUNetInferencer.py ./train_eval_infer.config
```

### mini\_test\_images (709x749 pixels)



**mini\_test\_mask(ground\_truth)****Inferred test masks (709x749 pixels)****Enlarged images and masks of 709x749 pixels****Image****Mask (ground\_truth)****Inferred-mask**





## References

### 1. Ultrasound Common Carotid Artery Segmentation Based on Active Shape Model

Xin Yang, Jiaoying Jin, Mengling Xu, Huihui Wu, Wanji He, Ming Yuchi, Mingyue Ding

<https://pmc.ncbi.nlm.nih.gov/articles/PMC3606761/>

### 2. Automated Segmentation of Common Carotid Artery in Ultrasound Images

J. H. Gagan; Harshit S. Shirsat; Grissel P. Mathias; B. Vaibhav Mallya; Jasbon Andrade; K. V. Rajagopal

Published in: IEEE Access ( Volume: 10)

<https://ieeexplore.ieee.org/document/9785785>

### 3. Method for Carotid Artery 3-D Ultrasound Image Segmentation Based on CSWin Transformer

Yanping Lin, Jianhua Huang, Wangjie Xu, Cancan Cui, Wenzhe Xu, Zhaojun Li

<https://www.sciencedirect.com/science/article/abs/pii/S0301562922006342>

### 4. Common Carotid Artery Ultrasound

<https://www.cuh.nhs.uk/patient-information/ultrasound-scan-of-your-carotid-arteries/>