

Tensorflow-Tiled-Image-Segmentation-IDRiD-Retinal-Vessel (2025/03/09)

Sarah T. Arai
Software Laboratory antillia.com

This is the first experiment of Image Segmentation for [IDRiD Retinal Vessel](#) based on Pretrained HRF Retinal Vessel Model, which was trained by the latest [Tensorflow-Image-Segmentation-API](#), and a **pre-augmented tiled dataset** [Augmented-Tiled-HRF-ImageMask-Dataset.zip](#), which was derived by us from the following dataset:

[Download the whole dataset \(~73 Mb\)](#) in [High-Resolution Fundus \(HRF\) Image Database](#).

Please see also our experiments:

- [Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-HRF-Retinal-Vessel](#) based on [High-Resolution Fundus \(HRF\) Image Database](#).
- [Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-DRIVE-Retinal-Vessel](#) based on [DRIVE: Digital Retinal Images for Vessel Extraction](#)
- [Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-STARE-Retinal-Vessel](#) based on [STructured Analysis of the Retina](#).
- [Tensorflow-Image-Segmentation-Retinal-Vessel](#) based on [CHASE DB1 dataset](#).

Experiment Strategies

1. Masks (antillia ground truth) for IDRiD master Original Images

The IDRiD dataset contains no masks (ground truths) data for Retinal Vessel. Therefore, we created our own master masks (ground truths) from the IDRiD Original Images by using Tiled Image Segmentation method and Pretrained-HRF-Retinal-Vessel UNet Model which was trained by the latest [Tensorflow-Image-Segmentation-API](#), and a **pre-augmented tiled dataset** [Augmented-Tiled-HRF-ImageMask-Dataset.zip](#).

2. Augment IDRiD Dataset

We augmented the IDRiD-master images and masks by using ImageMaskDatasetGenerator.

3. Split Augmented-IDRiD-master

We splitted the Augmented-IDRiD-master images and masks into test, train and valid subsets.

4. Train IDRiD Retinal Vessel Segmentation Model

We trained and validated a TensorFlow UNet model by using the **Augmented-IDRiD train and valid subsets**

5. Evaluate IDRiD Segmentation Model

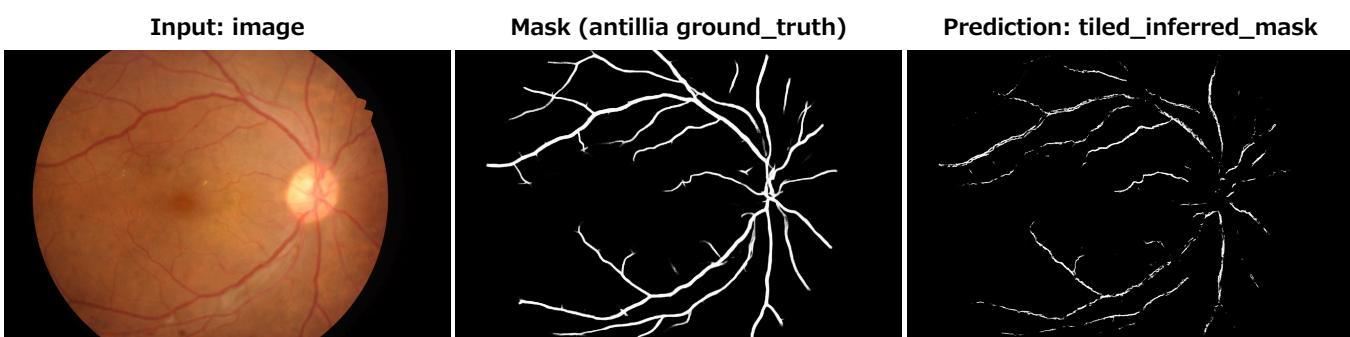
We evaluated the performance of the trained UNet model by using the **Augmented-IDRiD test** dataset by computing the **bce_dice_loss** and **dice_coef**.

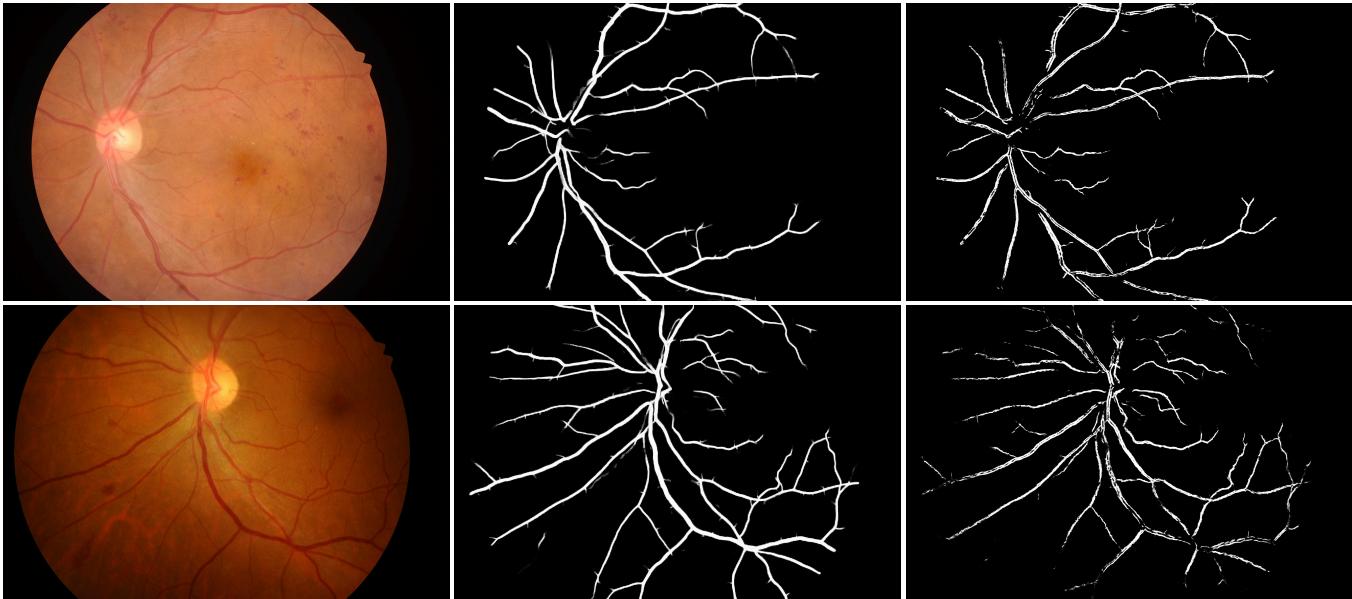
5. Tiled Inference

We applied our Tiled Image Segmentation method to infer the Retinal Vessel for the mini_test images of the IDRiD images of 4288x2848 pixels.

Actual Tiled Image Segmentation for IDRiD images of 4288x2848 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 HRFSegmentation 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

1.1 High-Resolution Fundus (HRF) Image Database

The dataset used here has been taken from the dataset [Download the whole dataset \(~73 Mb\)](#) in [High-Resolution Fundus \(HRF\) Image Databaset](#).

Introduction

This database has been established by a collaborative research group to support comparative studies on automatic segmentation algorithms on retinal fundus images. The database will be iteratively extended and the webpage will be improved. We would like to help researchers in the evaluation of segmentation algorithms. We encourage anyone working with segmentation algorithms who found our database useful to send us their evaluation results with a reference to a paper where it is described. This way we can extend our database of algorithms with the given results to keep it always up-to-date. The database can be used freely for research purposes. We release it under Creative Commons 4.0 Attribution License.

Citation

[Robust Vessel Segmentation in Fundus Images](#)

Budai, Attila; Bock, Rüdiger; Maier, Andreas; Hornegger, Joachim; Michelson, Georg.

International Journal of Biomedical Imaging, vol. 2013, 2013

1.2 IDRiD Dataset

The dataset used here has been take from the following **IEEE DataPort** web site

[Indian Diabetic Retinopathy Image Dataset \(IDRiD\)](#)

Please see also [DIABETIC RETINOPATHY: SEGMENTATION AND GRAND CHALLENGE](#)

Citation Author(s):

Prasanna Porwal, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe, Fabrice Meriaudeau, April 24, 2018, "Indian Diabetic Retinopathy Image Dataset (IDRiD)", IEEE Dataport,

DOI: <https://dx.doi.org/10.21227/H25W98>

License:

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

2. Create Masks for IDRiD Retinal Vessel Images

2.1 Download IDRiD IMAGES Dataset

Please download [Indian Diabetic Retinopathy Image Dataset \(IDRiD\)](#). The folder structure of the original Segmentation data is the following.

```
./A. Segmentation
  1. Original Images
    └── a. Training Set
    └── b. Testing Set
  2. All Segmentation Groundtruths
    └── a. Training Set
      └── 1. Microaneurysms
      └── 2. Haemorrhages
      └── 3. Hard Exudates
      └── 4. Soft Exudates
      └── 5. Optic Disc
    └── b. Testing Set
      └── 1. Microaneurysms
      └── 2. Haemorrhages
      └── 3. Hard Exudates
      └── 4. Soft Exudates
      └── 5. Optic Disc
```

As shown above, this dataset contains no masks (ground truths) data for **IDRiD Retinal Vessel**. Therefore, we created our own mask files for the IDRiD Retinal Vessel.

At first, please copy all image files in **1. Original Images** under the following folder **images**.

```
./projects
└── generator
  └── IDRiD-master
    └── images
```

2.2 Download Augmented-Tiled-HRF-Pretrained-Model

Please download our [Augmented-Tiled-HRF-Pretrained-Model.zip](#), expand it and place **best_model.h5** under models folder as shown below.

```
./projects
└── TensorflowSlightlyFlexibleUNet
  └── Augmented-Tiled-HRF
    └── models
      └── best_model.h5
```

2.3 Run Tiled Inference method

Please move the folder `./projects/TensorflowSlightlyFlexibleUNet/Augmented-Tiled-HRF`. and run the following bat file.

`5_tiled_infer-idrid.bat`

This will generate our own masks (ground truths) for Retinal Vessel of IDRiD images by using the HRF-Pretrained-Model, without any human experts.

```
./projects
└── generator
  └── IDRiD-master
    └── images
      └── masks
```

The number of images and their corresponding masks in IDRiD-master is 81 respectively, which is too small to use for a training set of our Segmentation Model.

2.4 Augment IDRiD-master

Please move to `./projects/generator` folder and run the following Python script.

`python ImageMaskDatasetGenerator.py`

, by which the following Augmented-IDRiD-master will be generated.

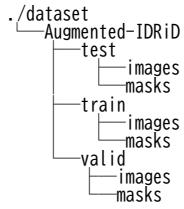
```
./projects
└── generator
  └── Augmented-IDRiD-master
    └── images
      └── mask
```

2.5 Split Augmented-IDRiD-master

Please move to `./projects/generator` folder and run the following Python script.

`python spit_augmented_master.py`

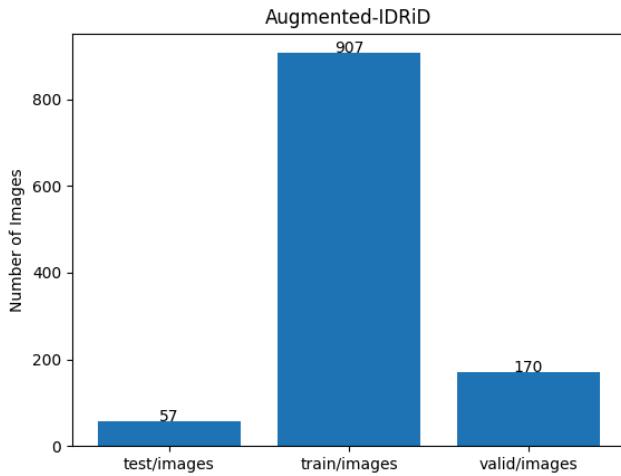
, by which the following Augmented-IDRiD dataset will be created.



This is a Non-Tiled 4288x2848 pixels images and their corresponding masks dataset for IDRiD Retinal Vessel.

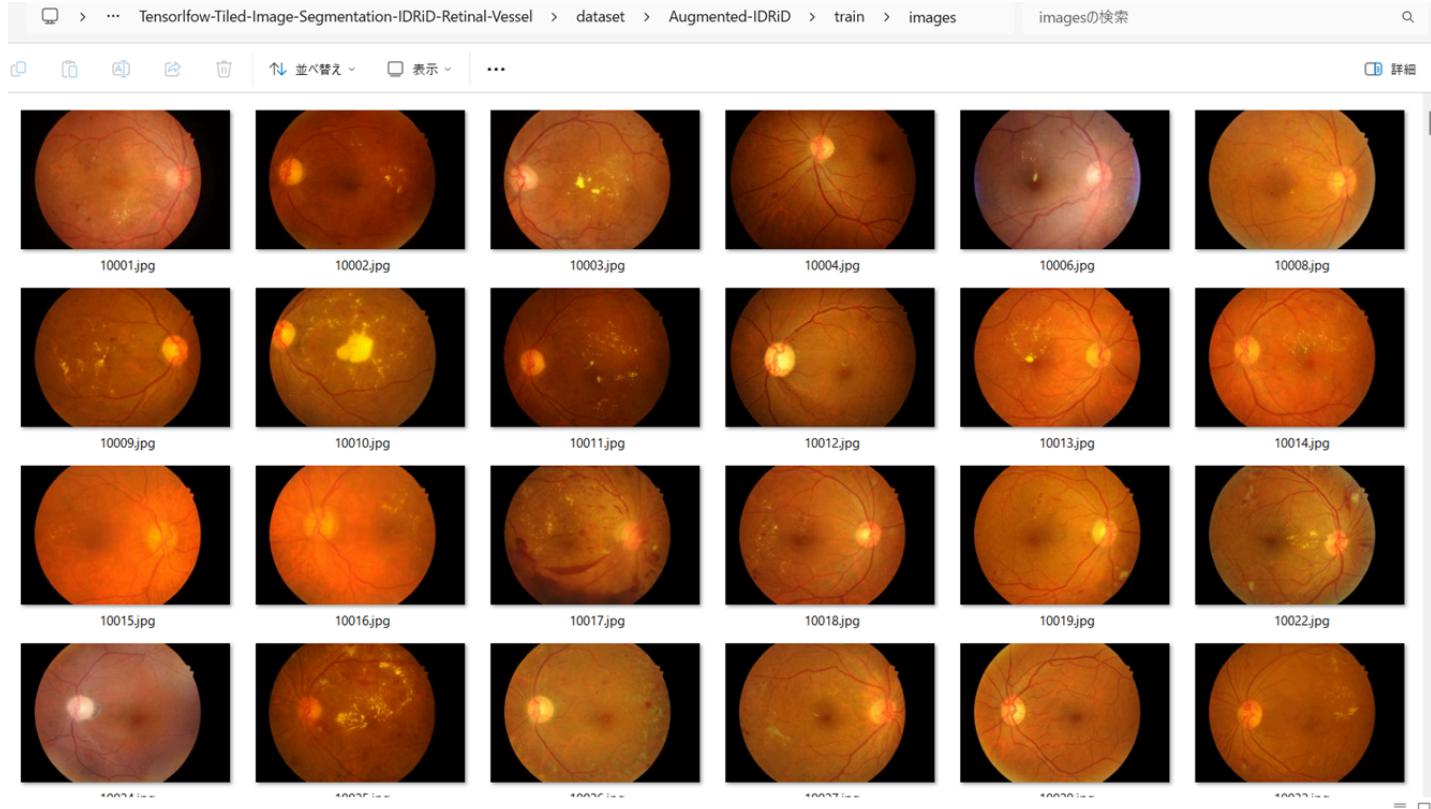
You may also download this dataset from the google drive [Antillia-Augmented-IDRiD-Dataset.zip \(2.06G\)](#).

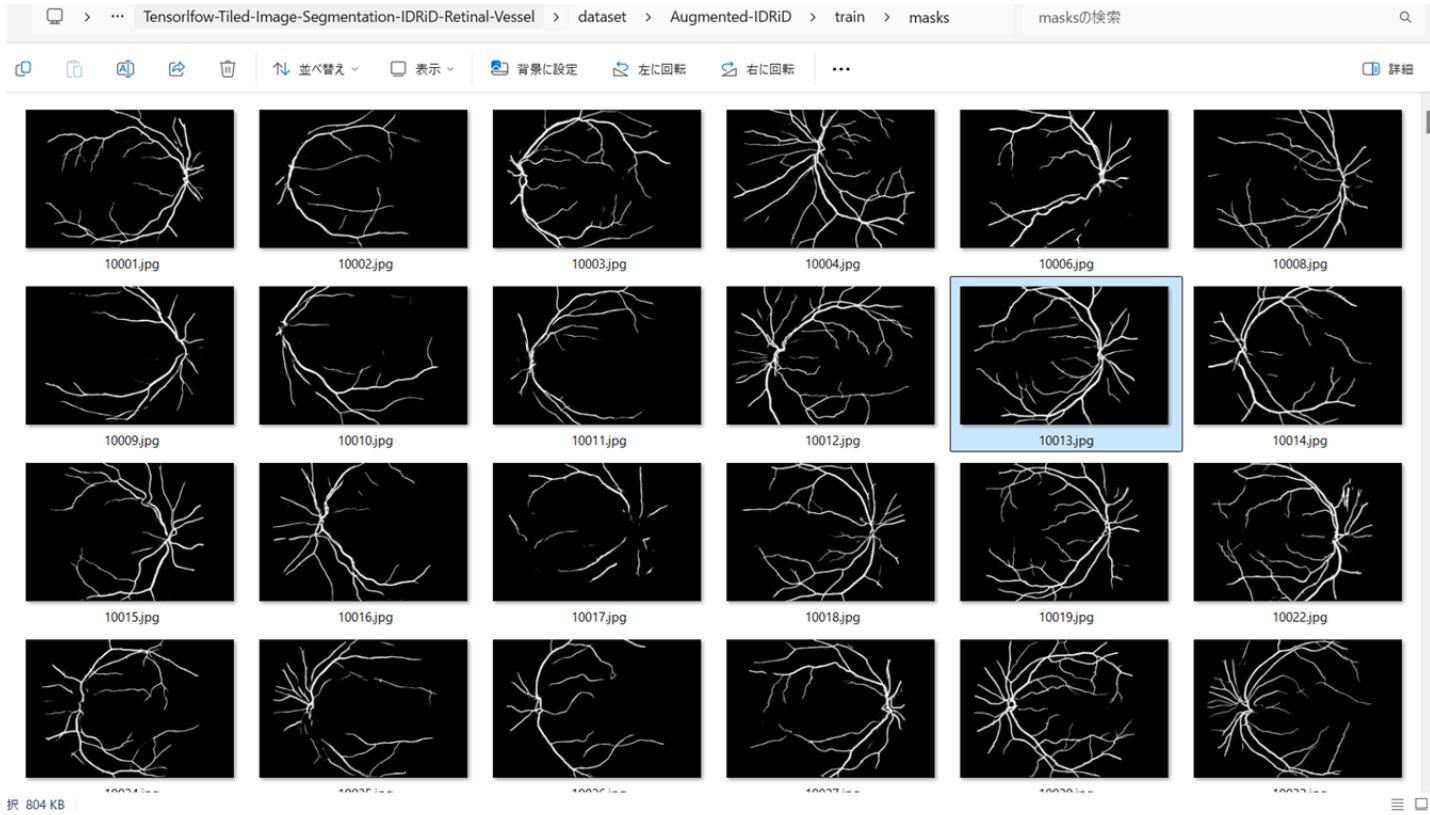
Augmented-IDRiD Statistics



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

Train_images_sample



Train_masks_sample**3 Train TensorflowUNet Model**

We have trained HRF TensorflowUNet Model by using the following [train_eval_infer.config](#) file.

Please move to ./projects/TensorflowSlightlyFlexibleUNet/IDRiD 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

Enabled Batch Normalization.

Defined a small **base_filters=16** and large **base_kernels=(11,11)** 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 = (11,11)
num_layers = 8
dilation = (1,1)
```

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_LINEAR"
```

Early stopping callback

Enabled early stopping callback with patience parameter.

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

Tiled Inference

Used the original IDRiD IMAGES as a mini_test dataset for our inference images.

```
[tiledinfer]
images_dir = "./mini_test/images"
output_dir = "./mini_test_output_tiled"
```

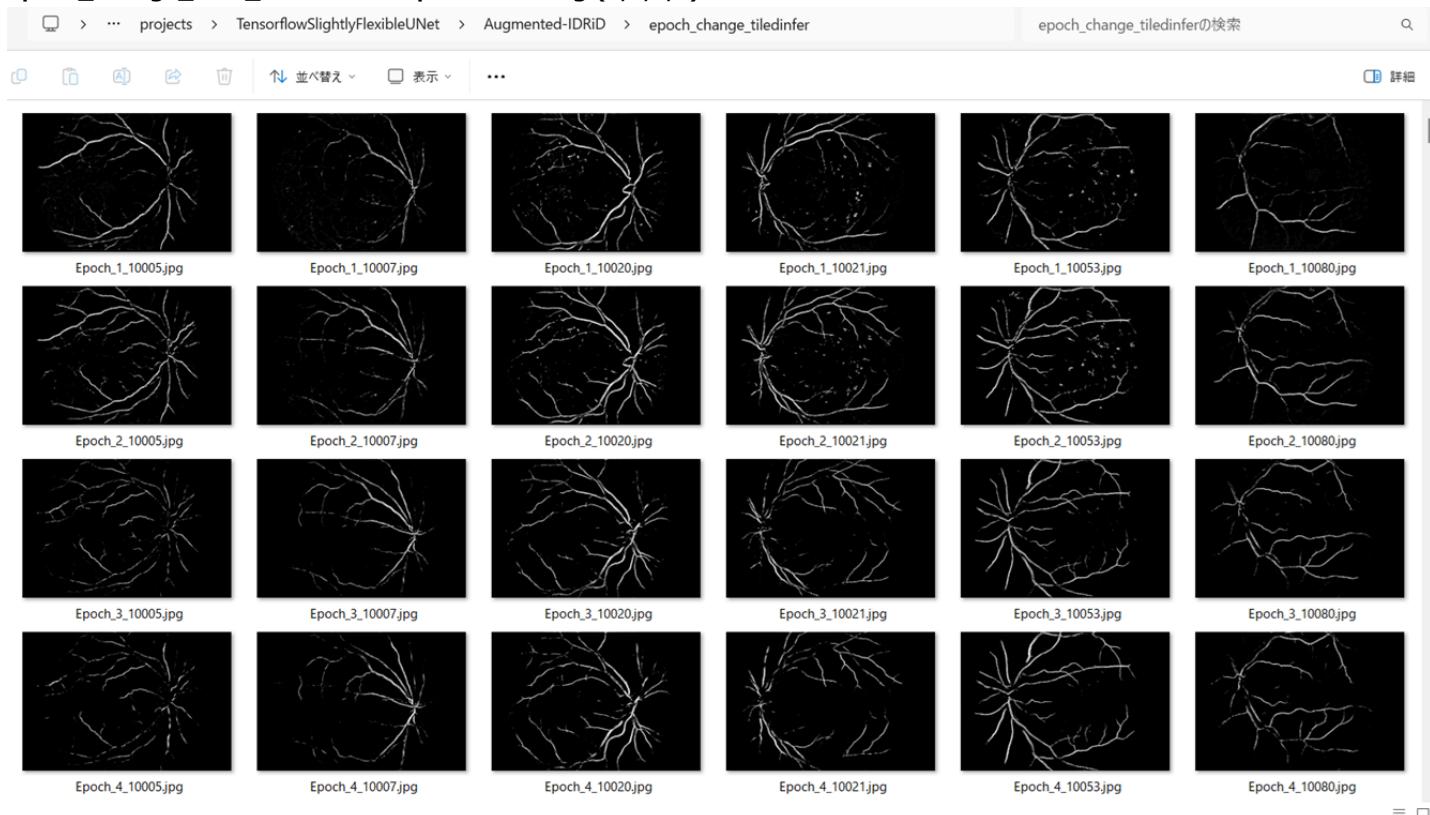
Epoch change inference callbacks

Enabled epoch_change_infer callback.

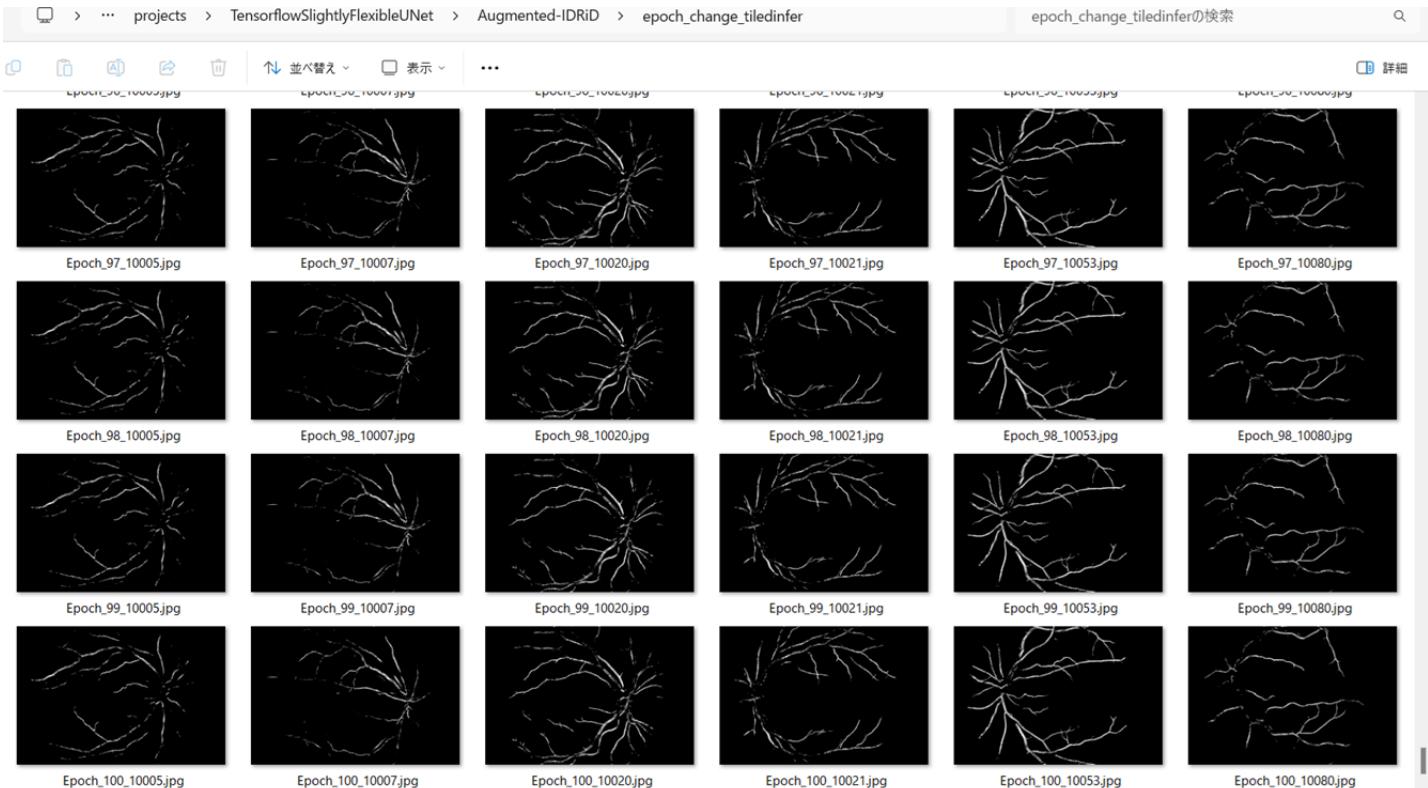
```
[train]
epoch_change_infer = False
epoch_change_infer_dir = "./epoch_change_infer"
epoch_change_tiledinfer = True
epoch_change_tiledinfer_dir = "./epoch_change_tiledinfer"
num_infer_images = 6
```

By using this callback, on every epoch_change, the epoch change tiled inference procedure can be called for 6 images in **mini_test** folder. This will help you confirm how the predicted mask changes at each epoch during your training process.

Epoch_change_tiled_inference output at starting (1,2,3,4)



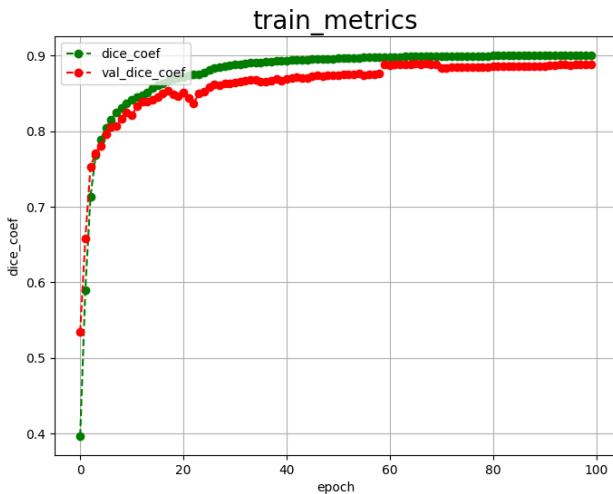
Epoch_change_tiled_inference output at ending (75,76,77,78)



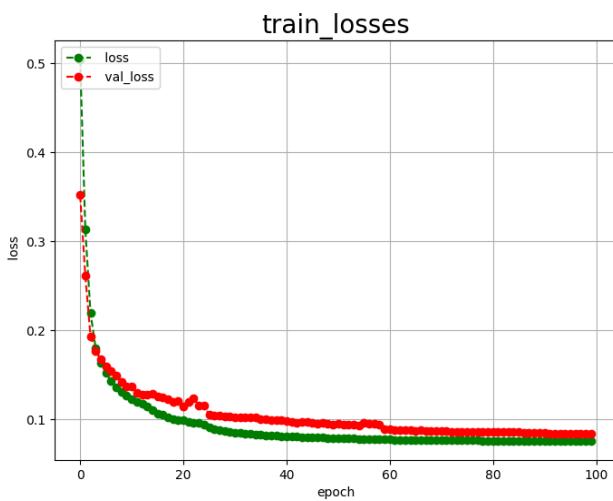
In this experiment, the training process was terminated at epoch 100.

```
PowerShell 7 (x64) + x - x
Epoch 89: val_loss improved from 0.08529 to 0.08528, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0853 - val_dice_coef: 0.8862 - lr: 2.5600e-06
Epoch 90/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001
Epoch 90: val_loss did not improve from 0.08528
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0853 - val_dice_coef: 0.8861 - lr: 2.5600e-06
Epoch 91/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001
Epoch 91: val_loss improved from 0.08528 to 0.08527, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0853 - val_dice_coef: 0.8862 - lr: 2.5600e-06
Epoch 92/100
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001
Epoch 92: val_loss improved from 0.08527 to 0.08400, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001 - val_loss: 0.0840 - val_dice_coef: 0.8875 - lr: 1.0240e-06
Epoch 93/100
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001
Epoch 93: val_loss did not improve from 0.08400
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001 - val_loss: 0.0840 - val_dice_coef: 0.8875 - lr: 1.0240e-06
Epoch 94/100
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001
Epoch 94: val_loss improved from 0.08400 to 0.08392, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0760 - dice_coef: 0.9001 - val_loss: 0.0839 - val_dice_coef: 0.8877 - lr: 1.0240e-06
Epoch 95/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001
Epoch 95: val_loss improved from 0.08392 to 0.08388, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0839 - val_dice_coef: 0.8878 - lr: 1.0240e-06
Epoch 96/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001
Epoch 96: val_loss did not improve from 0.08388
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0841 - val_dice_coef: 0.8874 - lr: 1.0240e-06
Epoch 97/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001
Epoch 97: val_loss improved from 0.08388 to 0.08385, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9001 - val_loss: 0.0839 - val_dice_coef: 0.8878 - lr: 1.0240e-06
Epoch 98/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9002
Epoch 98: val_loss improved from 0.08385 to 0.08374, saving model to ./models/best_model.h5
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9002 - val_loss: 0.0837 - val_dice_coef: 0.8881 - lr: 1.0240e-06
Epoch 99/100
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9002
Epoch 99: val_loss did not improve from 0.08374
907/907 [=====] - ETA: 0s - loss: 0.0759 - dice_coef: 0.9002 - val_loss: 0.0838 - val_dice_coef: 0.8879 - lr: 1.0240e-06
Epoch 100/100
907/907 [=====] - ETA: 0s - loss: 0.0758 - dice_coef: 0.9002 - val_loss: 0.0838 - val_dice_coef: 0.8878 - lr: 1.0240e-06
907/907 [=====] - ETA: 0s - loss: 0.0758 - dice_coef: 0.9002 - val_loss: 0.0838 - val_dice_coef: 0.8878 - lr: 1.0240e-06
Save history.json
```

[train_metrics.csv](#)



[train_losses.csv](#)



4 Evaluation

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

`./2.evaluate.bat`

This bat file simply runs the following command.

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

Evaluation console output:

```

PowerShell 7 (x64) + - x
--- ConfigParser ./train_eval_infer.config
--- WARNING: Not found [train] best model file, return default value best_model.h5
--- Loaded config file: /models/best_model.h5
--- DatasetClass <class 'ImageMaskDataset.ImageMaskDataset'>
--- BaseImageMaskDataset.constructor
--- ConfigParser ./train_eval_infer.config
--- WARNING: Not found [dataset] image, return default value bgr
--- 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 False
--- WARNING: Not found [dataset] rgb mask, return default value False
--- WARNING: Not found [dataset] color.order, return default value bgr
--- contrast adjuster False
--- WARNING: Not found [image] contrast alpha, return default value 1.5
--- WARNING: Not found [image] contrast test, return default value 40
--- WARNING: Not found [dataset] mask format, return default value gray
--- WARNING: Not found [mask] binarize, return default value False
--- WARNING: Not found [mask] grayscaling, 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 3
--- image normalize False
--- binarize algorithm None
--- ImageMaskDataset.constructor
--- self.resize interpolation 1
--- WARNING: Not found [model] evaluation, return default value test
--- BaseImageMaskDataset.create_dataset test
--- create
--- E:\py310-efficientdet\lib\site-packages\keras\engine\training_v1.py:2332: 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 :0.0845
Test accuracy:0.8859
--- Evaluation metric:loss score:0.0845
--- Evaluation metric:dice_coef score:0.8859
--- Saved ./evaluation.csv
100% |-----| 57/57 [00:11<00:00, 4.83it/s]
--- X: shape (57, 512, 512, 3) type uint8
--- Y: shape (57, 512, 512, 1) type bool
--- Create X: len: 57 Y: len: 57
--- WARNING: Not found [val] batch_size, return default value 4
--- WARNING: Not found [val] batch_size 4
E:\py310-efficientdet\lib\site-packages\keras\engine\training_v1.py:2332: 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 :0.0845
Test accuracy:0.8859
--- Evaluation metric:loss score:0.0845
--- Evaluation metric:dice_coef score:0.8859
--- Saved ./evaluation.csv

```

[evaluation.csv](#)

The loss (bce_dice_loss) to this IDRiD/test was low, and dice_coef high as shown below.

```

loss,0.0845
dice_coef,0.8859

```

5 Tiled Inference

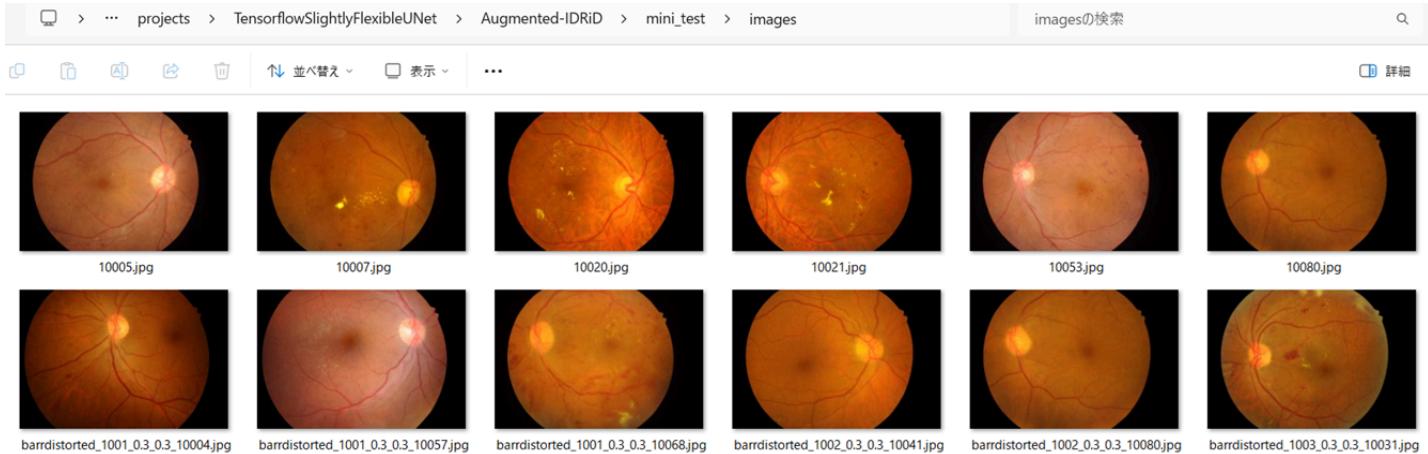
Please move to a **./projects/TensorflowSlightlyFlexibleUNet/IDRiD** folder
and run the following bat file to infer segmentation regions for images by the Trained-TensorflowUNet model for IDRiD.

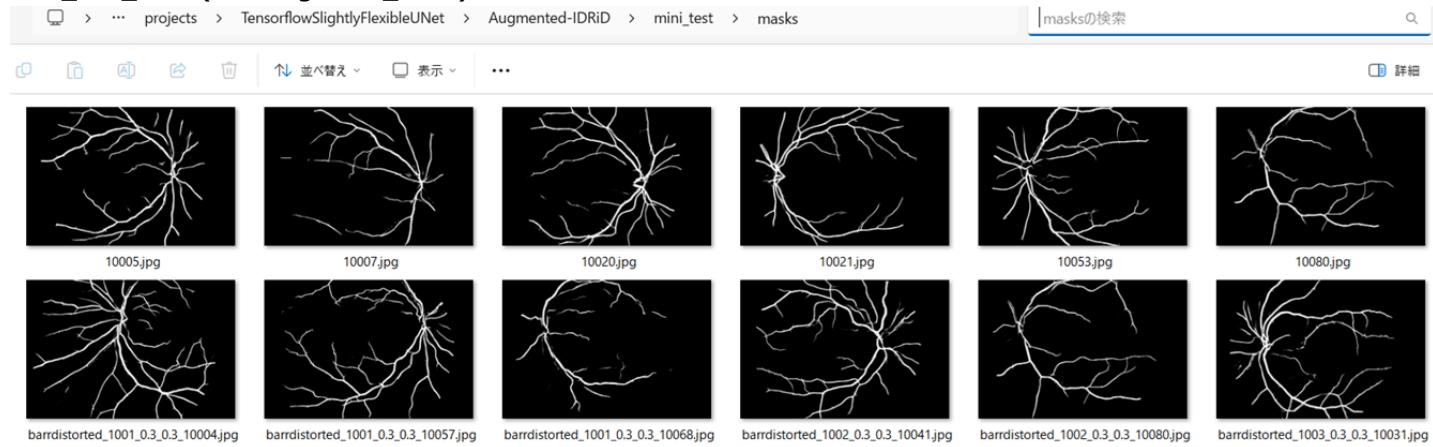
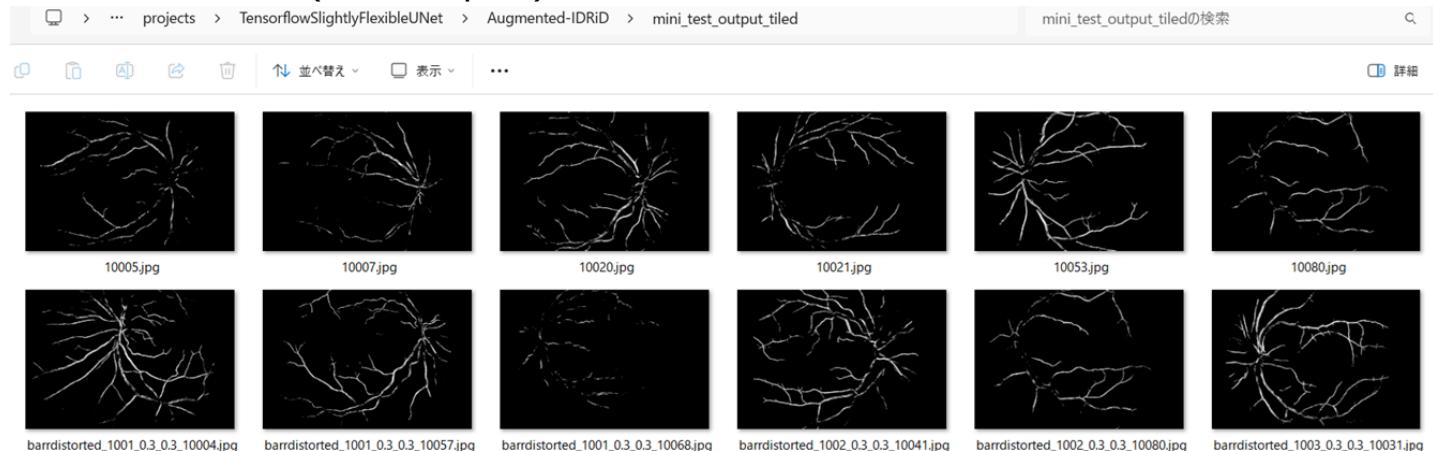
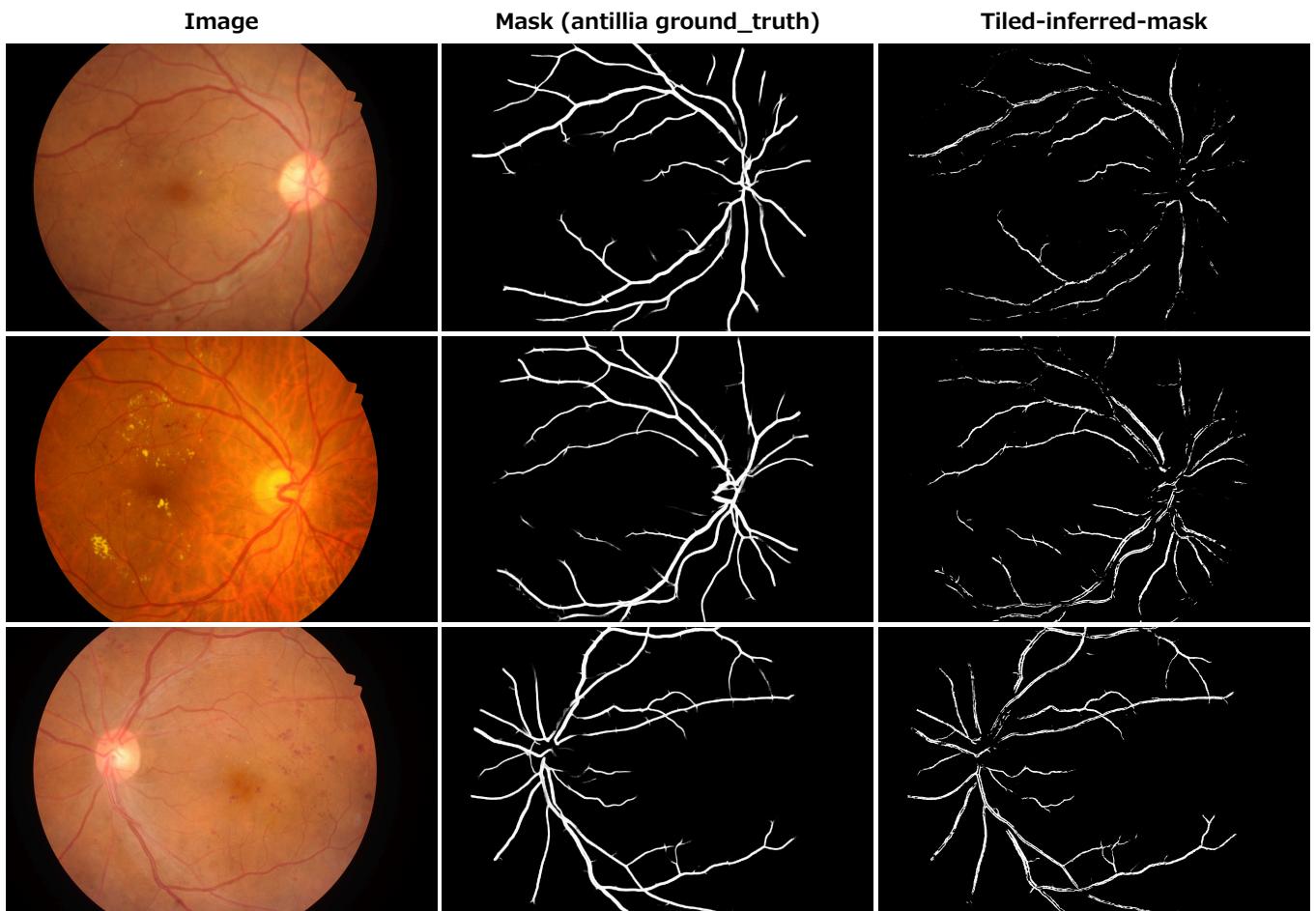
```
./4.tiledinfer.bat
```

This simply runs the following command.

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

mini_test_images (4288x2848 pixels)



mini_test_mask(antillia ground_truth)**Tiled inferred test masks (4288x2848 pixels)****Enlarged images and masks of 4288x2848 pixels**



References

1. Locating Blood Vessels in Retinal Images by Piecewise Threshold Probing of a Matched Filter Response

Adam Hoover, Valentina Kouznetsova, and Michael Goldbaum

<https://www.uhu.es/retinopathy/General/000301IEEETransMedImag.pdf>

2. High-Resolution Fundus (HRF) Image Database

Budai, Attila; Bock, Rüdiger; Maier, Andreas; Hornegger, Joachim; Michelson, Georg.

<https://www5.cs.fau.de/research/data/fundus-images/>.

3. Robust Vessel Segmentation in Fundus Images

Budai, Attila; Bock, Rüdiger; Maier, Andreas; Hornegger, Joachim; Michelson, Georg.

<https://onlinelibrary.wiley.com/doi/10.1155/2013/154860>

4. State-of-the-art retinal vessel segmentation with minimalistic models

Adrian Galdran, André Anjos, José Dolz, Hadi Chakor, Hervé Lombaert & Ismail Ben Ayed

<https://www.nature.com/articles/s41598-022-09675-y>

5. Retinal blood vessel segmentation using a deep learning method based on modified U-NET model

Sanjeeewani, Arun Kumar Yadav, Mohd Akbar, Mohit Kumar, Divakar Yadav

<https://www.semanticscholar.org/reader/f5cb3b1c69a2a7e97d1935be9d706017af8cc1a3>

6. Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-STARE-Retinal-Vessel

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-STARE-Retinal-Vessel>

7. Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-HRF-Retinal-Vessel

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-HRF-Retinal-Vessel>