

Tensorflow-Tiled-Image-Segmentation-Pre-Augmented-IDRiD-HardExudates (2025/01/28)

Sarah T. Arai
Software Laboratory antillia.com

This is the second experiment of Tiled Image Segmentation for IDRiD-HardExudates based on the latest [Tensorflow-Image-Segmentation-API](#), and a **pre-augmented tiled dataset** [Augmented-Tiled-IDRiD-HardExudates-ImageMask-Dataset.zip](#), which was derived by us from Segmentation dataset of [Indian Diabetic Retinopathy Image Dataset \(IDRiD\)](#).

Please see also our first experiment [Tensorflow-Tiled-Image-Segmentation-IDRiD-HardExudates](#)

Experiment Strategies

In this experiment, we employed the following strategies.

1. Tiled ImageMask Dataset

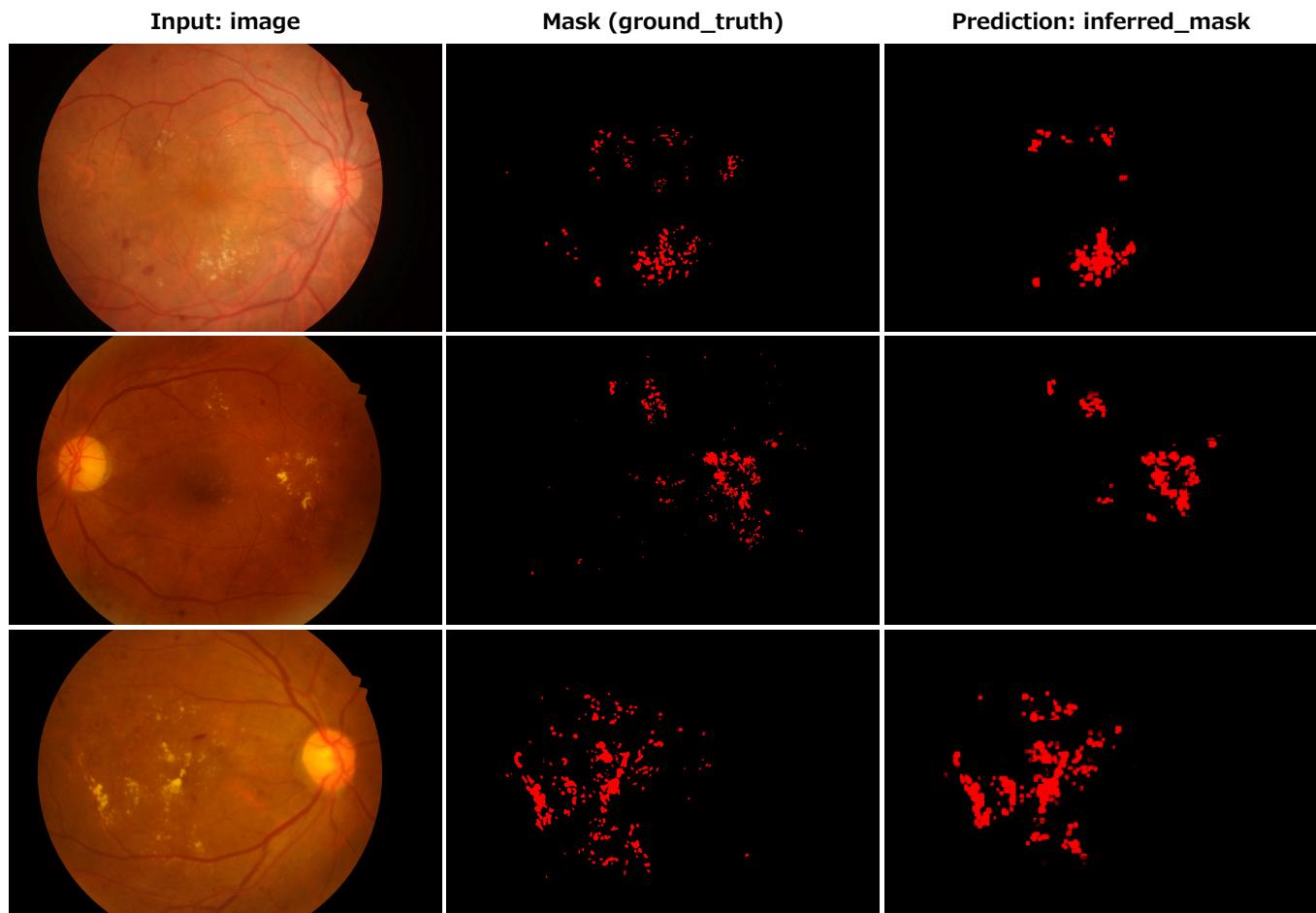
We trained and validated a TensorFlow UNet model using the **Pre-Augmented-Tiled-IDRiD-HardExudates-ImageMask-Dataset**, which was tiledly-split to 512x512 pixels and reduced to 512x512 pixels image and mask dataset from the original 4288x2848 pixels images and mask files.

2. Tiled Image Segmentation

We applied our Tiled-Image Segmentation inference method to predict the HardExudates regions for the mini_test images with a resolution of 4288x2848 pixels.

Actual Tiled Image Segmentation for 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 IDRiD-HardExudatesSegmentation 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 **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,

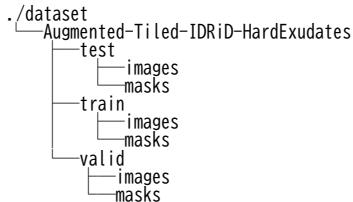
DOI: <https://dx.doi.org/10.21227/H25W98>

License:

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

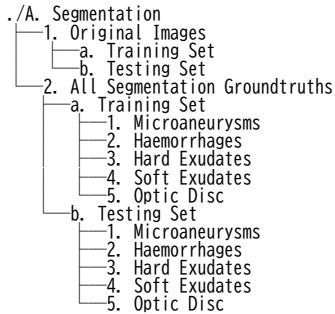
2 Augmented-Tiled-IDRiD-HardExudates ImageMask Dataset

If you would like to train this IDRiD-HardExudates Segmentation model by yourself, please download the pre-augmented dataset from the google drive [Augmented-Tiled-IDRiD-HardExudates-ImageMask-Dataset.zip](#), expand the downloaded ImageMaskDataset and put it under **./dataset** folder to be



This is a 512x512 pixels pre augmented tiles dataset generated from 4288x2848 pixels 54 **Original Images** and their corresponding **Hard Exudates GroundTruths** in Training Set.

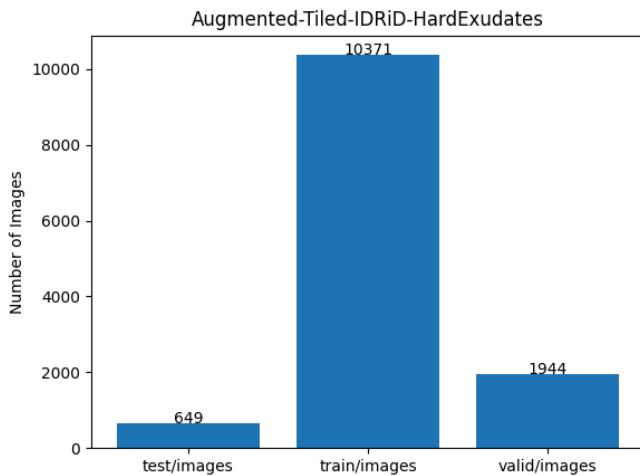
We excluded all black (empty) masks and their corresponding images to generate our dataset from the original one.
The folder structure of the original Segmentation data is the following.



On the derivation of this tiled dataset, please refer to the following Python scripts.

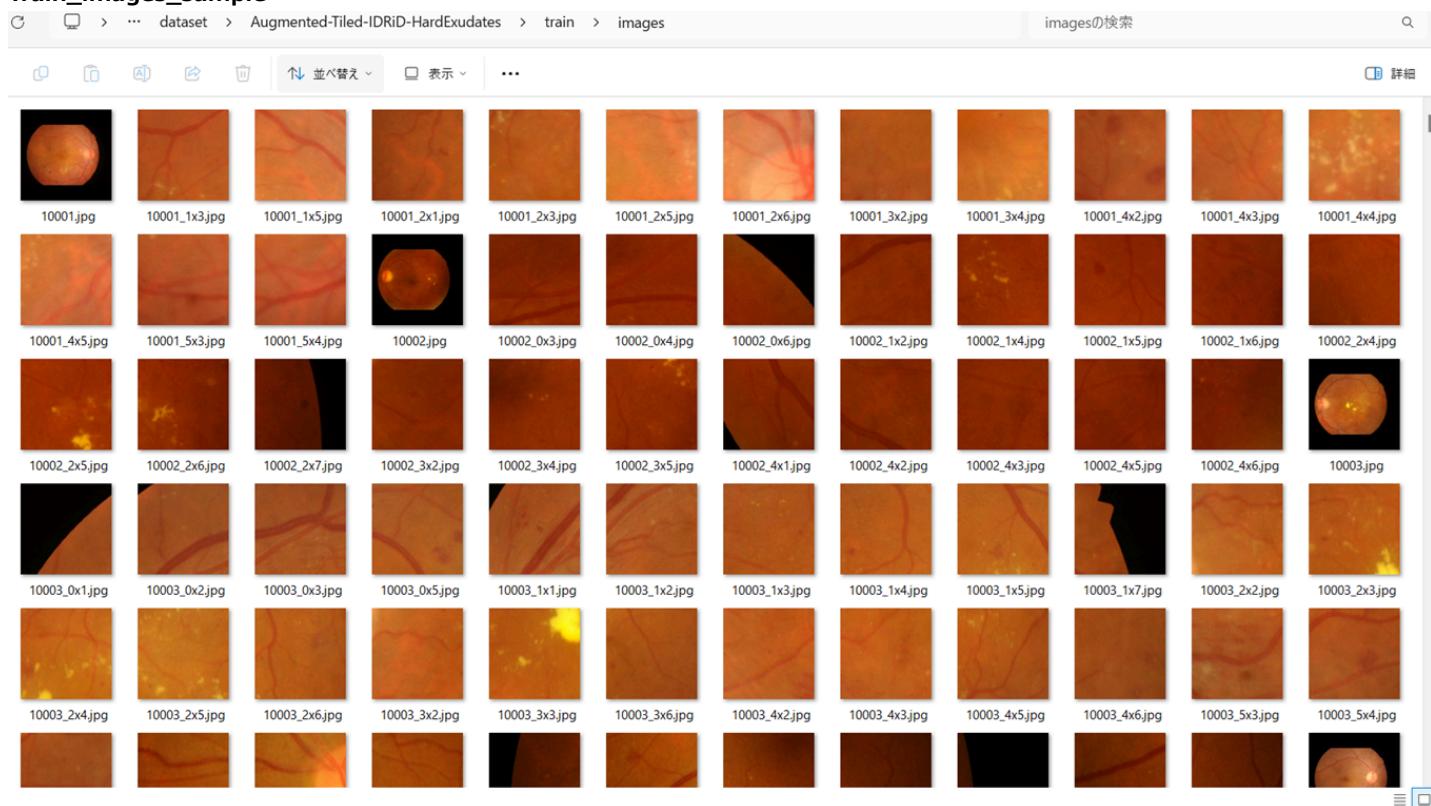
- [TiledImageMaskDatasetGenerator.py](#)
- [split_tiled_master.py](#)

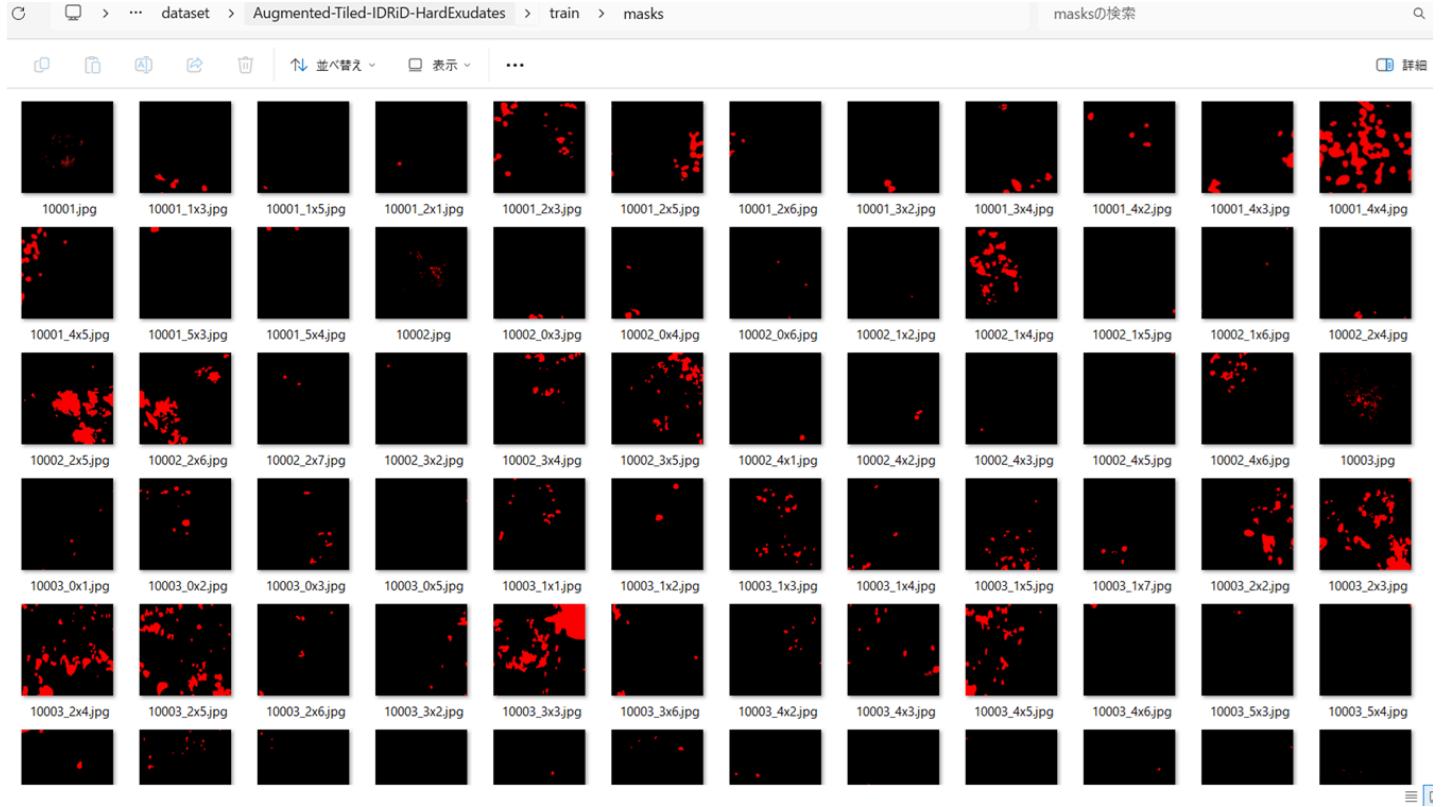
Augmented-Tiled-IDRiD-HardExudates Statistics



As shown above, the number of images of train and valid datasets is enough to use for a training set of our segmentation model.

Train_images_sample



Train_masks_sample**3 Train TensorflowUNet Model**

We have trained IDRiD-HardExudates TensorflowUNet Model by using the following [train_eval_infer.config](#) file.
Please move to ./projects/TensorflowSlightlyFlexibleUNet/IDRiD-HardExudates 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=(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

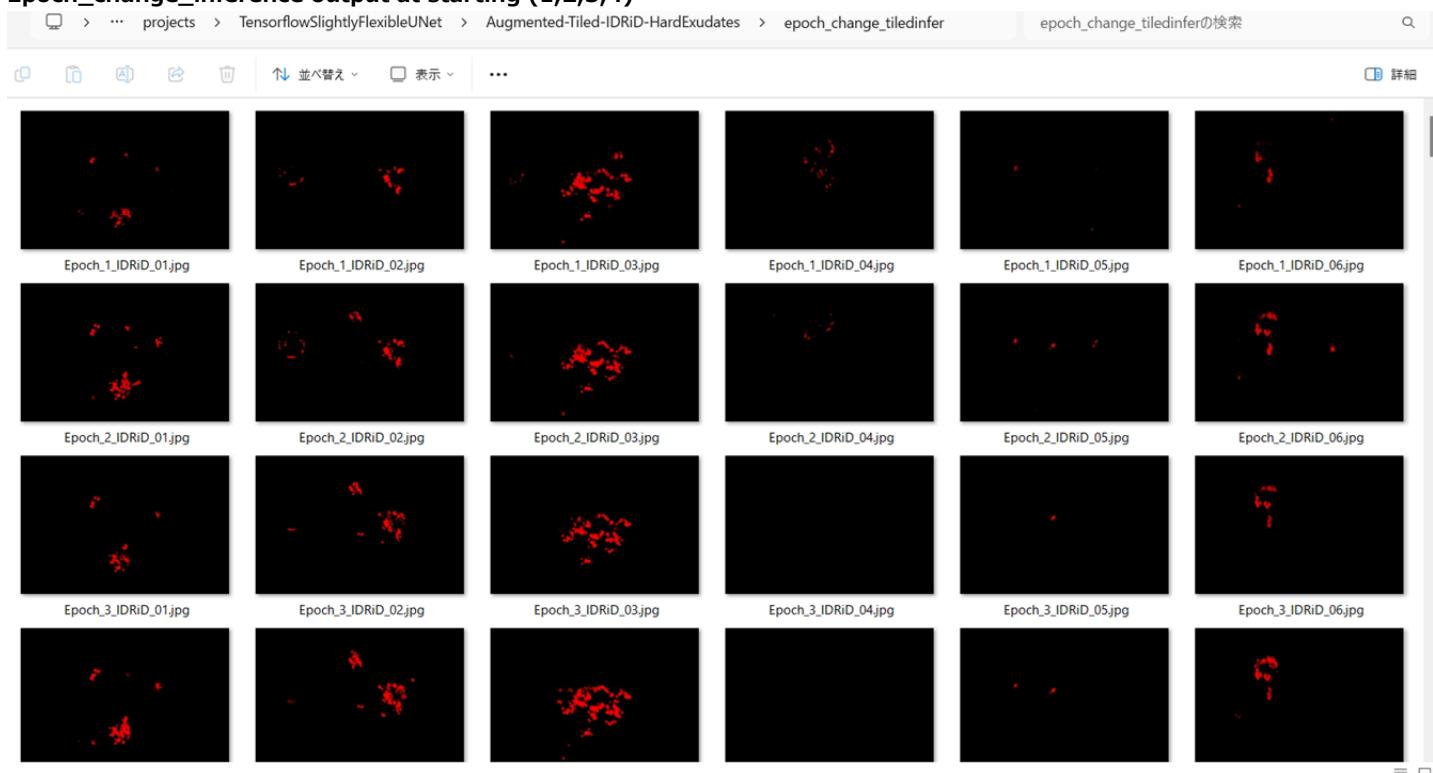
Enabled epoch_change_infer callback.

```
[train]
```

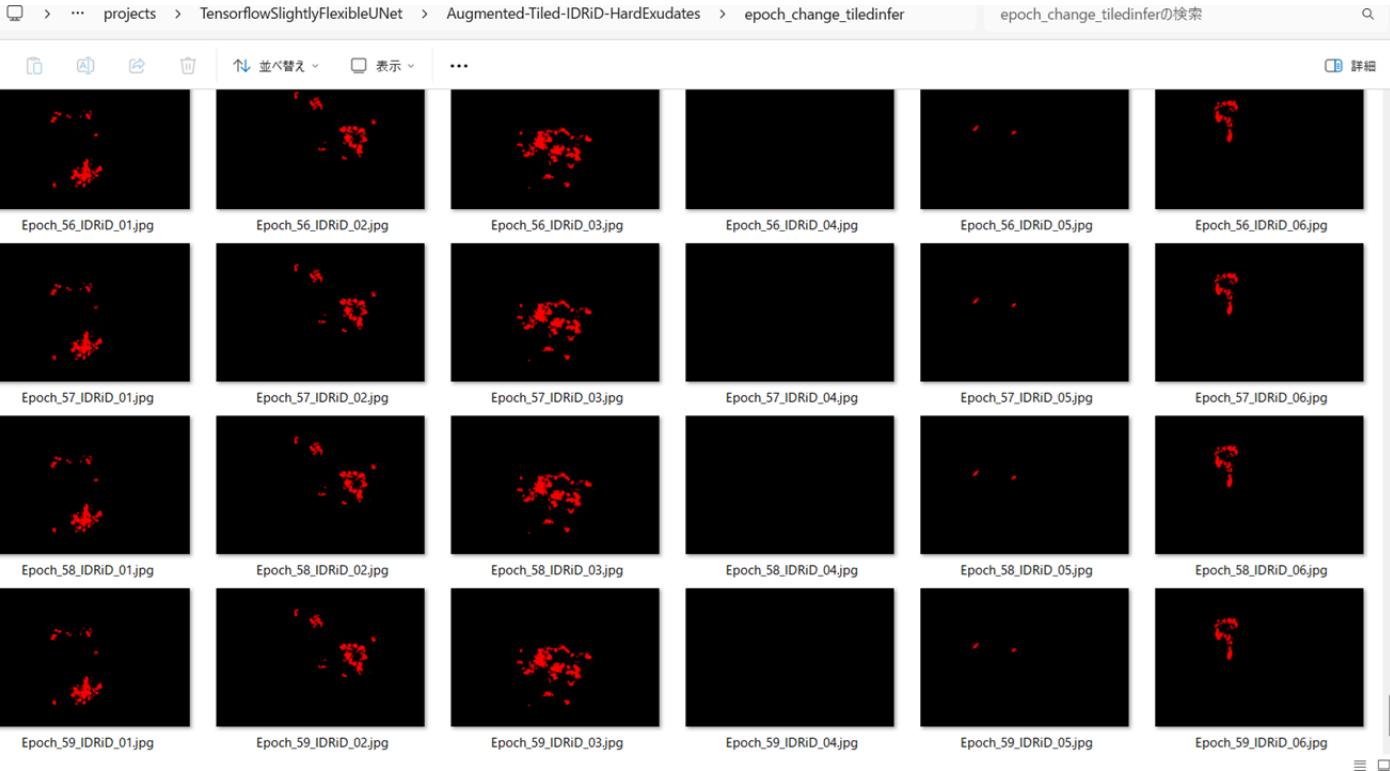
```
epoch_change_infer = False
epoch_change_infer_dir = "./epoch_change_infer"
epoch_change_tiledinfer = True
epoch_change_tiledinfer_dir = "./epoch_change_tiledinfer"
num_infer_images = 6
```

By using this callback, on every epoch_change, the epoch change 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 (1,2,3,4)



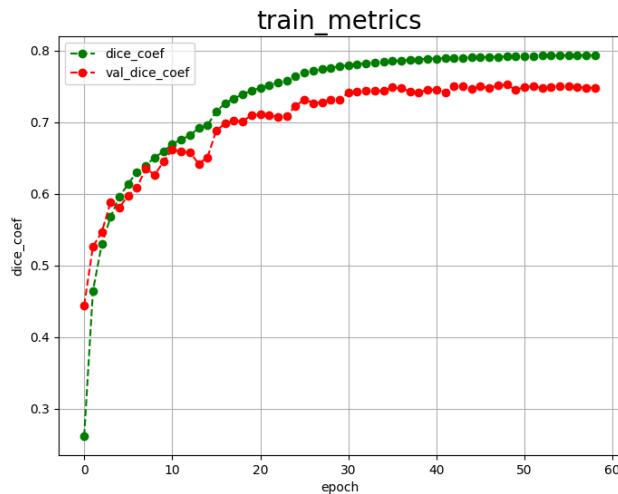
Epoch_change_inference output at ending (56,57,58,59)



In this experiment, the training process was stopped at epoch 59 by EarlyStopping Callback.

```
PowerShell 7 (x64) - + - x
10371/10371 [=====] - ETA: 0s - loss: 0.1314 - dice_coef: 0.7918
Epoch 49: val_loss improved from 0.15649 to 0.15634, saving model to ./models/best_model.h5
10371/10371 [=====] - 2667s 257ms/sample - loss: 0.1314 - dice_coef: 0.7918 - val_loss: 0.1563 - val_dice_coef: 0.7523 - lr: 2.5600e-06
Epoch 50/100
10371/10371 [=====] - ETA: 0s - loss: 0.1312 - dice_coef: 0.7920
Epoch 50: val_loss did not improve from 0.15634
10371/10371 [=====] - 2701s 260ms/sample - loss: 0.1312 - dice_coef: 0.7920 - val_loss: 0.1598 - val_dice_coef: 0.7452 - lr: 2.5600e-06
Epoch 51/100
10371/10371 [=====] - ETA: 0s - loss: 0.1310 - dice_coef: 0.7924
Epoch 51: val_loss did not improve from 0.15634
10371/10371 [=====] - 2730s 263ms/sample - loss: 0.1310 - dice_coef: 0.7924 - val_loss: 0.1578 - val_dice_coef: 0.7490 - lr: 2.5600e-06
Epoch 52/100
10371/10371 [=====] - ETA: 0s - loss: 0.1308 - dice_coef: 0.7927
Epoch 52: val_loss did not improve from 0.15634
10371/10371 [=====] - 2725s 263ms/sample - loss: 0.1308 - dice_coef: 0.7927 - val_loss: 0.1570 - val_dice_coef: 0.7505 - lr: 2.5600e-06
Epoch 53/100
10371/10371 [=====] - ETA: 0s - loss: 0.1306 - dice_coef: 0.7931
Epoch 53: val_loss did not improve from 0.15634
10371/10371 [=====] - 2748s 265ms/sample - loss: 0.1306 - dice_coef: 0.7931 - val_loss: 0.1585 - val_dice_coef: 0.7475 - lr: 2.5600e-06
Epoch 54/100
10371/10371 [=====] - ETA: 0s - loss: 0.1308 - dice_coef: 0.7928
Epoch 54: val_loss did not improve from 0.15634
10371/10371 [=====] - 2748s 265ms/sample - loss: 0.1308 - dice_coef: 0.7928 - val_loss: 0.1575 - val_dice_coef: 0.7486 - lr: 1.0240e-06
Epoch 55/100
10371/10371 [=====] - ETA: 0s - loss: 0.1307 - dice_coef: 0.7929
Epoch 55: val_loss did not improve from 0.15634
10371/10371 [=====] - 2755s 266ms/sample - loss: 0.1307 - dice_coef: 0.7929 - val_loss: 0.1564 - val_dice_coef: 0.7508 - lr: 1.0240e-06
Epoch 56/100
10371/10371 [=====] - ETA: 0s - loss: 0.1305 - dice_coef: 0.7934
Epoch 56: val_loss did not improve from 0.15634
10371/10371 [=====] - 2781s 268ms/sample - loss: 0.1305 - dice_coef: 0.7934 - val_loss: 0.1568 - val_dice_coef: 0.7504 - lr: 1.0240e-06
Epoch 57/100
10371/10371 [=====] - ETA: 0s - loss: 0.1305 - dice_coef: 0.7933
Epoch 57: val_loss did not improve from 0.15634
10371/10371 [=====] - 2835s 273ms/sample - loss: 0.1305 - dice_coef: 0.7933 - val_loss: 0.1571 - val_dice_coef: 0.7494 - lr: 1.0240e-06
Epoch 58/100
10371/10371 [=====] - ETA: 0s - loss: 0.1307 - dice_coef: 0.7930
Epoch 58: val_loss did not improve from 0.15634
10371/10371 [=====] - 2776s 268ms/sample - loss: 0.1307 - dice_coef: 0.7930 - val_loss: 0.1576 - val_dice_coef: 0.7480 - lr: 4.0960e-07
Epoch 59/100
10371/10371 [=====] - ETA: 0s - loss: 0.1306 - dice_coef: 0.7932
Epoch 59: val_loss did not improve from 0.15634
10371/10371 [=====] - 2819s 272ms/sample - loss: 0.1306 - dice_coef: 0.7932 - val_loss: 0.1574 - val_dice_coef: 0.7483 - lr: 4.0960e-07
Epoch 59: early stopping
Save history.json
```

[train_metrics.csv](#)



[train_losses.csv](#)



4 Evaluation

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

`./2.evaluate.bat`

This bat file simply runs the following command.

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

Evaluation console output:

```

metrics [function dice_coef at 0x000001900BCACAF0>]
--- 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
--- DatasetClass <class 'ImageMaskDataset.ImageMaskDataset'>
--- ConfigParser ./train_eval_infer.config
--- WARNING: Not found [mask] blur_size, 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 False
--- WARNING: Not found [dataset] 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] 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_classes 1
--- image_normalize False
--- binarize algorithm None
--- ImageMaskDataset.constructor
--- self.resize_interpolation 2
--- WARNING: Not found [model] evaluation, return default value test
--- BinaryMaskDataset.create_dataset test
--- create
--- BinaryMaskDataset.create_dataset /Augmented-Tiled-IDRiD-HardExudates/test/images/ ../../dataset/Augmented-Tiled-IDRiD-HardExudates/test/masks/
--- WARNING: Not found [mask] mask_channels, return default value 1
--- num_classes 1 image data_type <class 'numpy.uint8'>
num images 649 512 512
100% | 649/649 [00:04<00:00, 152.97it/s]
--- X: shape (649, 512, 512, 3) type uint8
--- Y: shape (649, 512, 512, 1) type bool
--- Create X: len: 649 Y: len 649
--- WARNING: Not found [eval] batch_size, return default value 4
--- evaluate 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 accuracy:0.7631
    Evaluation metric:loss score:0.1514
    Evaluation metric:dice_coef score:0.7631
    Saved ./evaluation.csv

```

Image-Segmentation-IDRiD-HardExudates [evaluation.csv](#)

The loss (bce_dice_loss) to this IDRiD-HardExudates/test was not low, and dice_coef not high as shown below.

```

loss,0.1514
dice_coef,0.7631

```

But these are better than those of the first experiment [Tensorflow-Tiled-Image-Segmentation-IDRiD-HardExudates](#)

```

loss,0.2541
dice_coef,0.6416

```

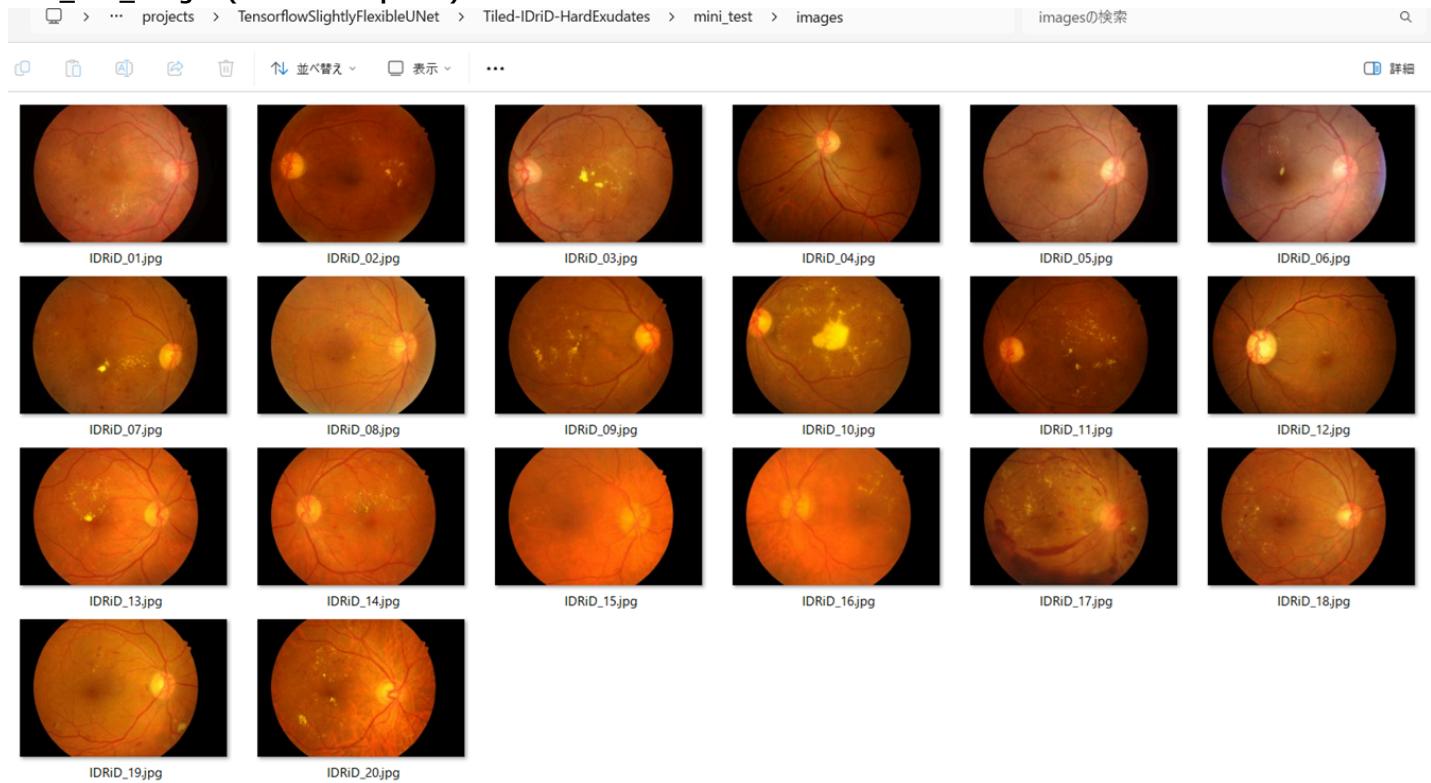
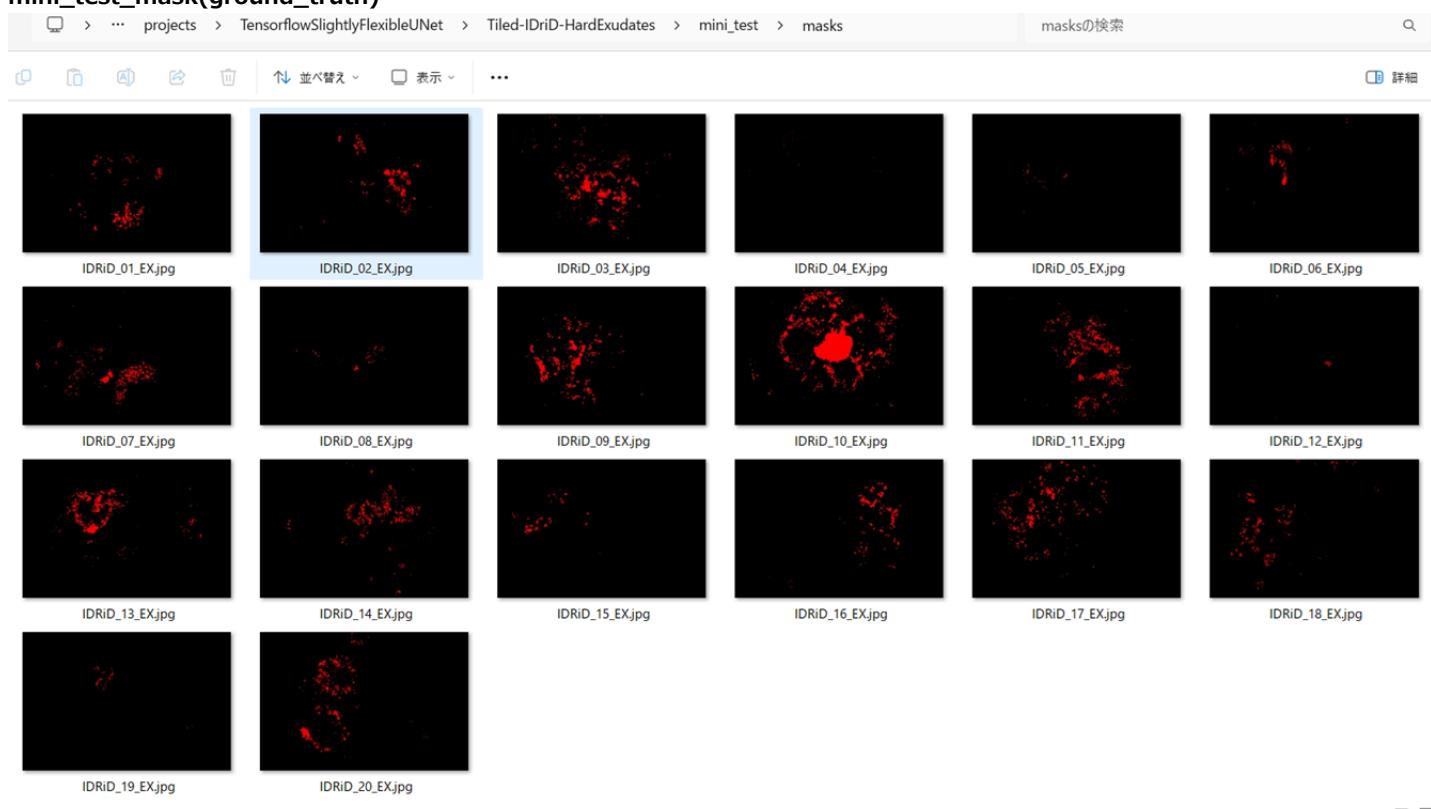
5 Tiled inference

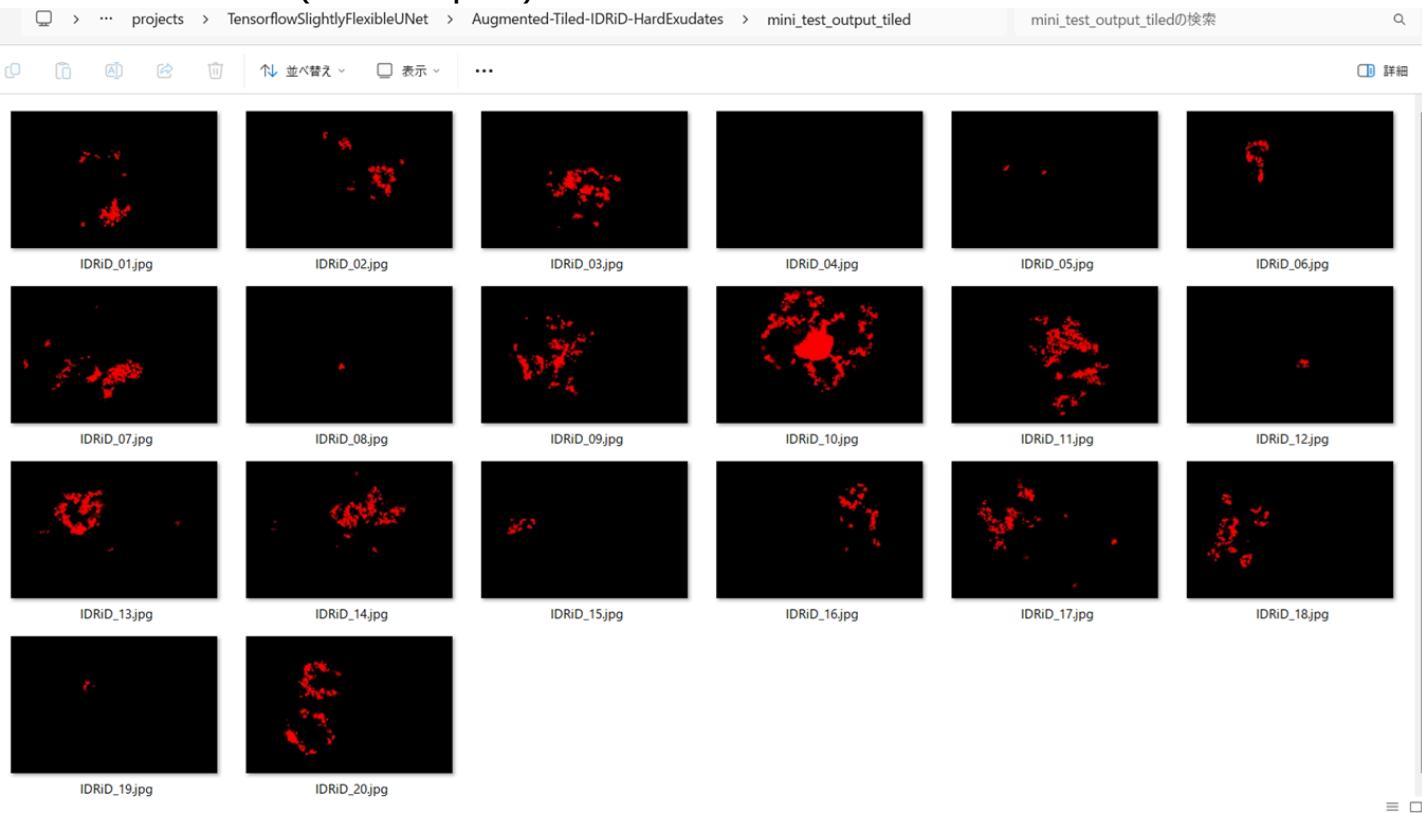
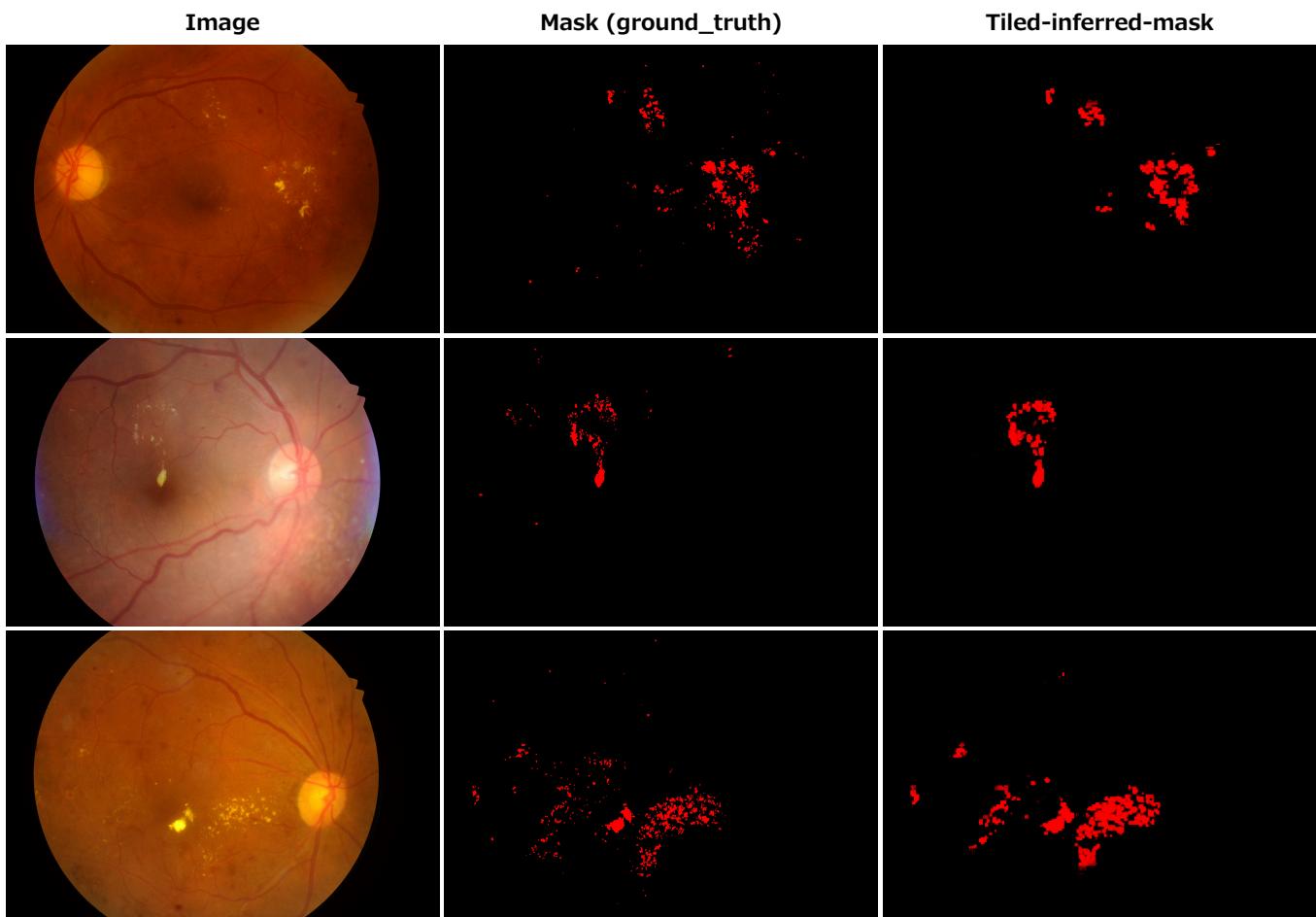
Please move to a **./projects/TensorflowSlightlyFlexibleUNet/IDRiD-HardExudates** folder
and run the following bat file to infer segmentation regions for images by the Trained-TensorflowUNet model for IDRiD-HardExudates.

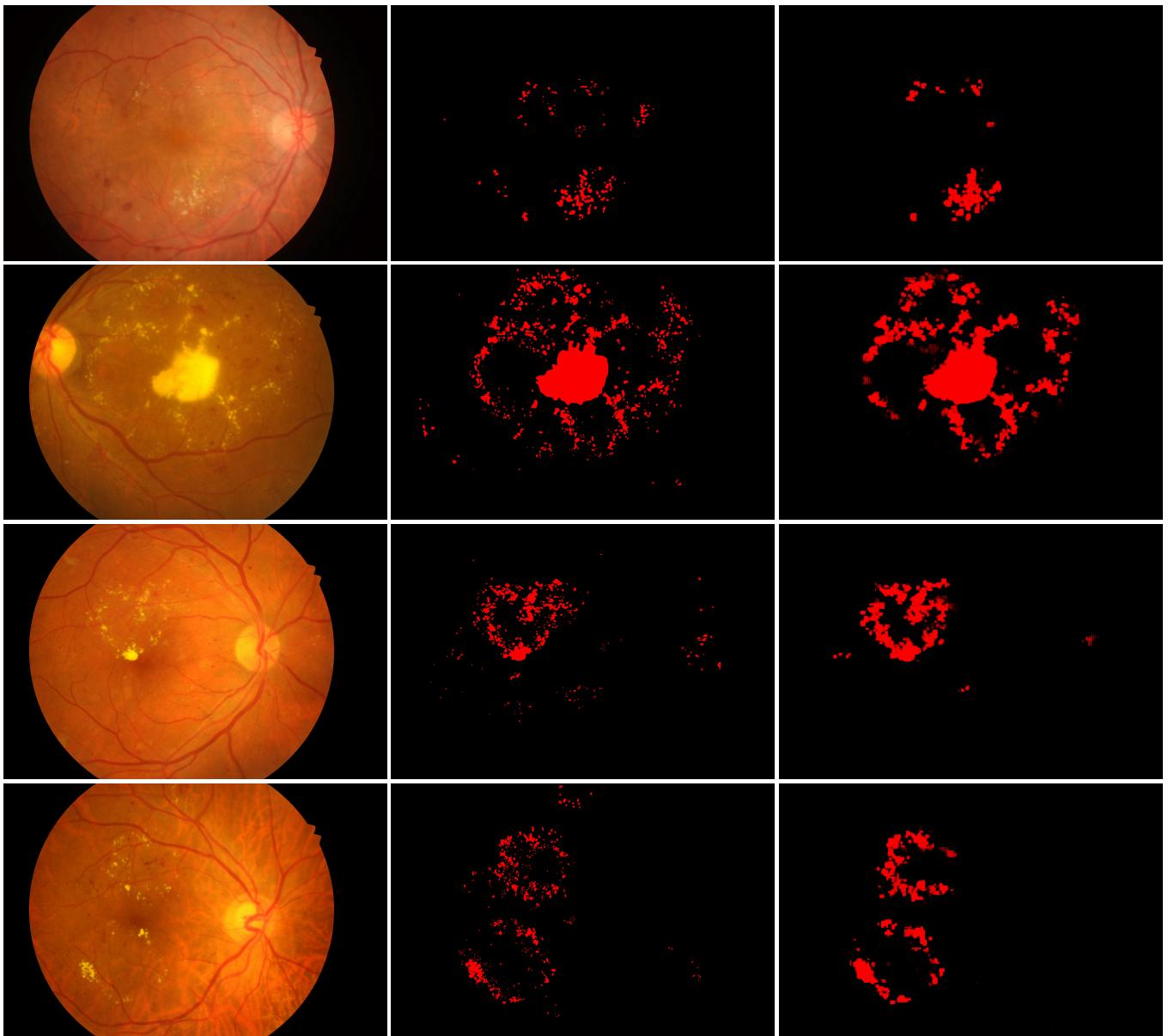
```
./4.tiled_infer.bat
```

This simply runs the following command.

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

mini_test_images (4288x2848 pixels)**mini_test_mask(ground_truth)**

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



References

1. IDRiD: Diabetic Retinopathy – Segmentation and Grading Challenge

Prasanna Porwal, Samiksha Pachade, Manesh Kokare, Girish Deshmukh, Jaemin Son, Woong Bae, Lihong Liu, Jianzong Wang, Xinhui Liu, Liangxin Gao, TianBo Wu, Jing Xiao, Fengyan Wang, Sang-Hyuk Jung, Fabrice Mériadeau

DOI: <https://doi.org/10.1016/j.media.2019.101561>

<https://www.sciencedirect.com/science/article/abs/pii/S1361841519301033>

2. RMCA U-net: Hard exudates segmentation for retinal fundus images

Yinghua Fu, Ge Zhang, Xin Lu, Honghan Wu, Dawei Zhang

<https://doi.org/10.1016/j.eswa.2023.120987>

<https://www.sciencedirect.com/science/article/abs/pii/S0957417423014896>

3. Tensorflow-Tiled-Image-Segmentation-IDRiD-HardExudates

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tensorflow-Tiled-Image-Segmentation-IDRiD-HardExudates>

4. Tensorflow-Tiled-Image-Segmentation-Augmented-Skin-Cancer

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tensorflow-Tiled-Image-Segmentation-Augmented-Skin-Cancer>

5. Tensorflow-Tiled-Image-Segmentation-Augmented-MultipleMyeloma

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tensorflow-Tiled-Image-Segmentation-Augmented-MultipleMyeloma>

6. Tiled-ImageMask-Dataset-Breast-Cancer

Toshiyuki Arai @antillia.com

<https://github.com/sarah-antillia/Tiled-ImageMask-Dataset-Breast-Cancer>