

---

# SIMPLE PRESENTATION FOR COMPETITION CHOOSE U-NET

MINGZIRUI WU BOTAO JIANG



# LITERATURE REVIEW

- What is U-Net?
  - U-Net: Convolutional Networks for Biomedical Image Segmentation
- What is U-Net++?
  - UNet++: A Nested U-Net Architecture for Medical Image Segmentation
  - UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation

# DIFFICULTIES OF AGGC22

- Huge Dataset: nearly 400GB (Insufficient computing power...)
  - Solution 1: Resize images
    - Will loss too many features.
  - Solution 2: Intercept only the characteristic parts.
    - Difficult to compile automated code (official contest requirements)
- Lack of time...

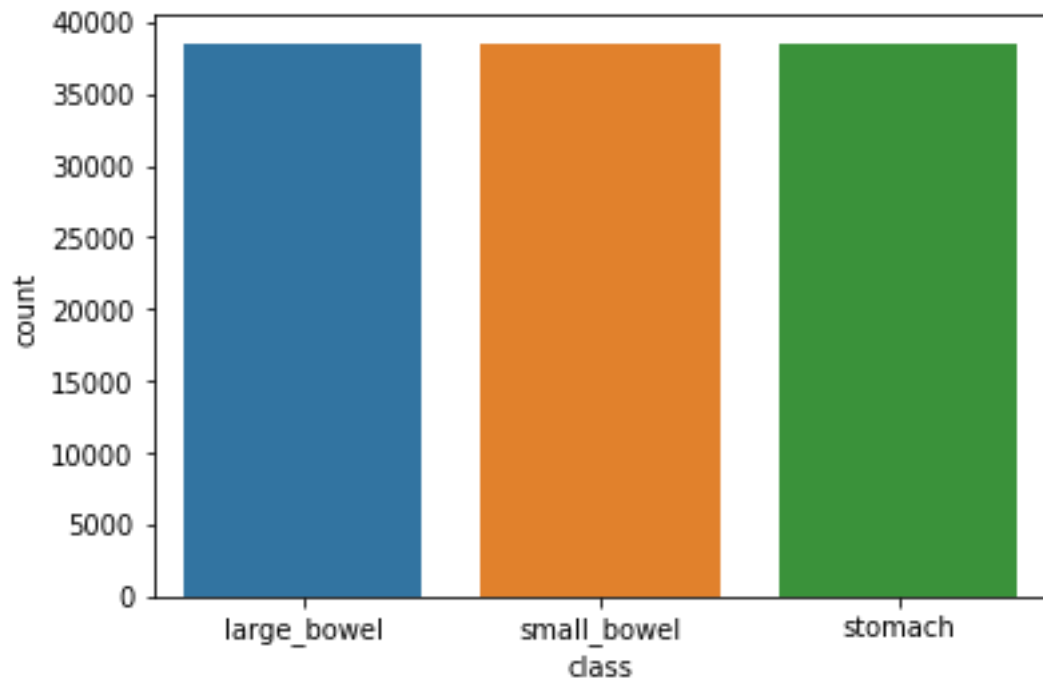
# LEARN FROM AGGC22

- Sigmoid for overlapping mask
  - Use SoftMax to convert logits into a probability distribution, and then take the one with the highest probability value as the classification of the sample. ( One label for one Sample)
  - A sample has multiple labels. (When applying the sigmoid activation function to multi-label classification, its loss function should be set to binary cross entropy.)

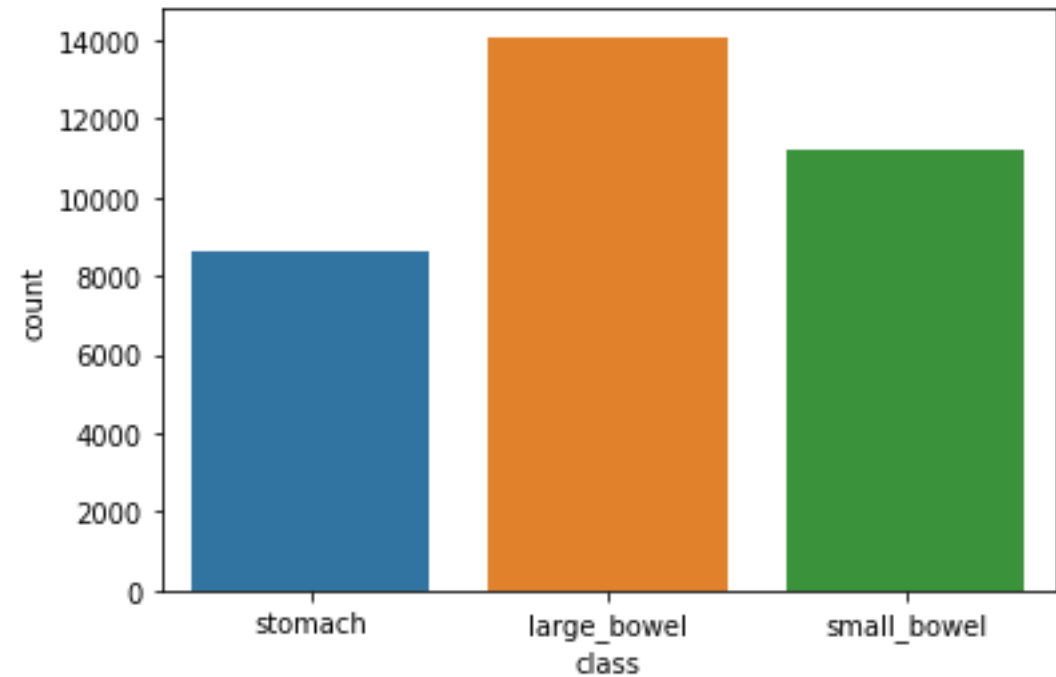
# TWO NEW COMPETITION: UW-MADISON GITRACT IMAGE SEGMENTATION TASK

- Create a model to automatically segment the stomach and intestines on MRI scans.
- The training annotations are provided as RLE-encoded masks, and the images are in 16-bit grayscale PNG format.
- Files
  - train.csv - IDs and masks for all training objects.
  - sample\_submission.csv - a sample submission file in the correct format
  - train - a folder of case/day folders, each containing slice images for a particular case on a given day.

## TWO NEW COMPETITION: UW-MADISON GITRACT IMAGE SEGMENTATION TASK



- As per the above chart we have equal number of slices for each of the three classes.



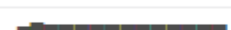
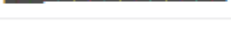



- The order of number of annotation data available for the training can be observed from above.

# U-NET IMPLEMENTATION RESULT

- Details that could be upgraded.
  - Data pre-processing
  - Network model complexity
  - Hyperparameters

Run history:

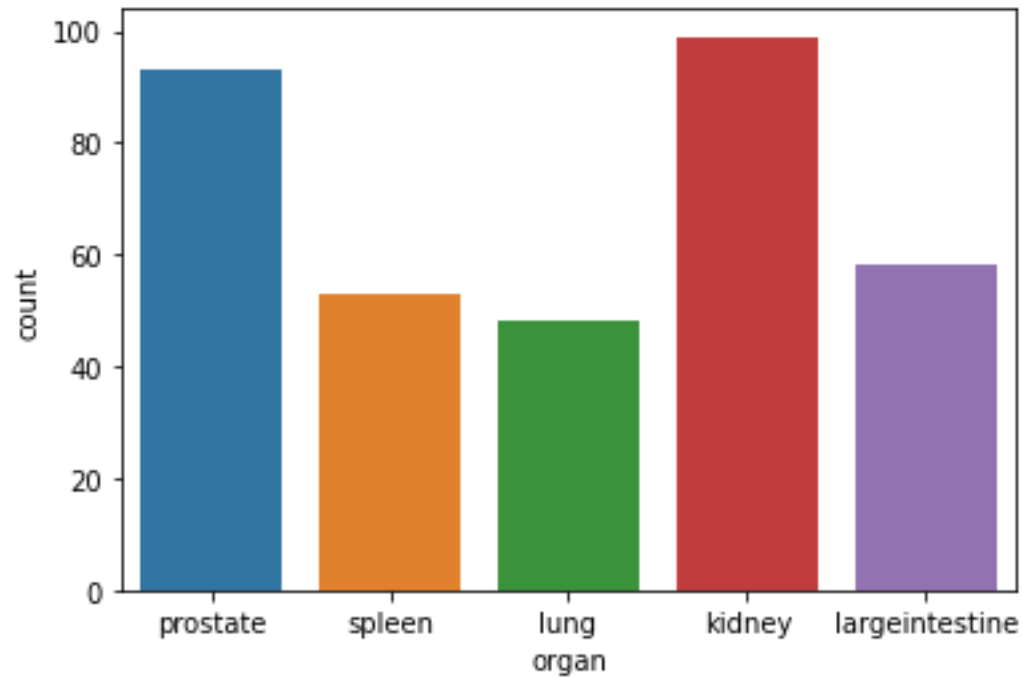
LR	
Train Loss	
Valid Dice	
Valid Jaccard	
Valid Loss	

Run summary:

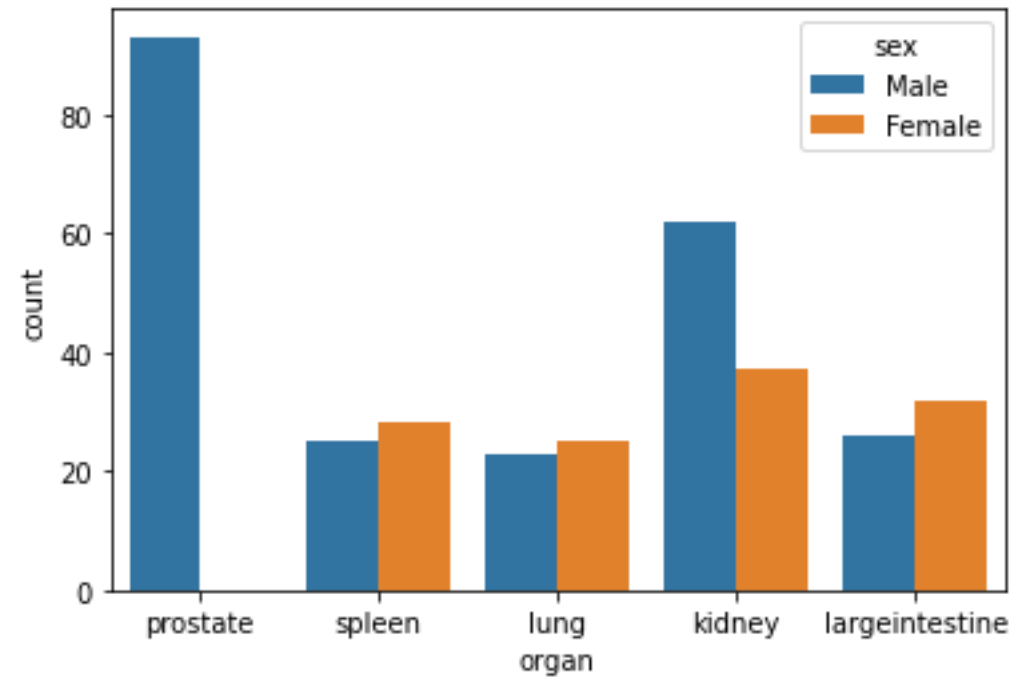
Best Dice	0.90578
Best Epoch	13
Best Jaccard	0.87745
LR	0.0
Train Loss	0.0788
Valid Dice	0.90559
Valid Jaccard	0.87726
Valid Loss	0.12213

# TWO NEW COMPETITION: HUBMAP + HPA - HACKING THE HUMAN BODY TASK

- Identify and segment functional tissue units (FTUs) across five human organs.



- Number of body organizations sample in the dataset.



- Number of body organizations sample in the dataset. (With Gender)



# COMPETITION OVERVIEW



The banner features a dark background with a circuit-like pattern. It displays three axial MRI scans of the abdomen, with the stomach, large bowel, and small bowel segmented in teal and orange. Three circular progress indicators are positioned above the scans. The text 'Research Code Competition' is in the top left. The main title 'UW-Madison GI Tract Image Segmentation' is in large white font, followed by the subtitle 'Track healthy organs in medical scans to improve cancer treatment'. At the bottom left, it says 'UW Madison · 1,466 teams · 7 days to go'. At the bottom right, it shows '\$25,000 Prize Money' and the 'WISCONSIN UNIVERSITY OF WISCONSIN-MADISON' logo.

Research Code Competition

## UW-Madison GI Tract Image Segmentation

Track healthy organs in medical scans to improve cancer treatment

UW Madison · 1,466 teams · 7 days to go

\$25,000  
Prize Money

WISCONSIN  
UNIVERSITY OF WISCONSIN-MADISON

In this competition, we need to come up with a deep learning solution to automatically segment 3 classes of stomach, large bowel and small bowel on MRI scans.

# DATASET

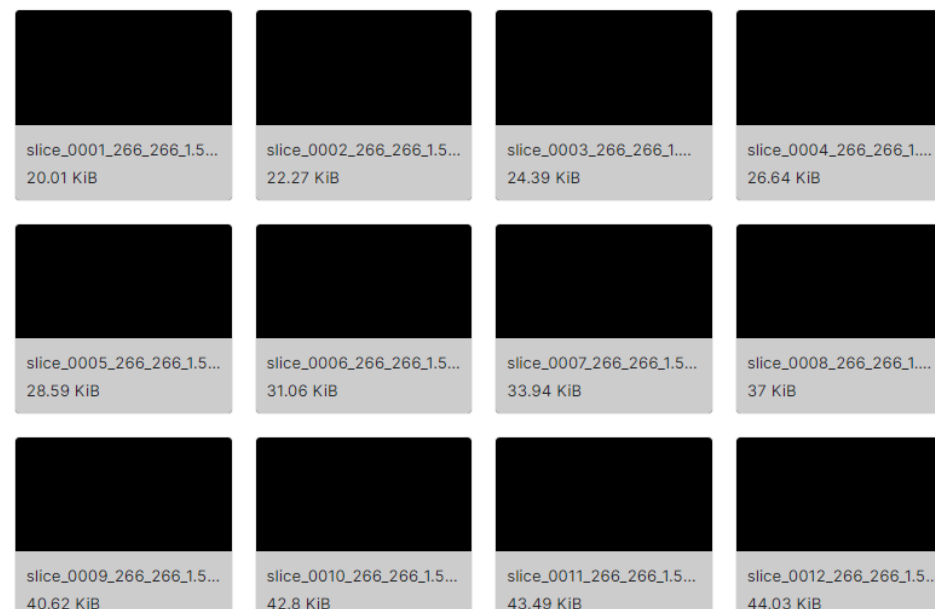
- Each case in this competition is represented by multiple sets of scan slices. Note that some cases are split for training set and test set.
- The training images are provided in 16-bit grayscale PNG format.
- Each images are named with the slice number, image size and pixel size.

## Data Explorer

2.47 GiB

- test
- train
  - case101
    - case101\_day20
      - scans
    - case101\_day22
    - case101\_day26
    - case101\_day32
  - case102
  - case107
  - case108
  - case11
  - case110
  - case111
  - case113
  - case114
  - case115
  - case116
  - case117

## < scans (144 files)



# DATASET

- The training annotations in this dataset are provided as RLE-encoded masks.
- RLE-encoded masks are in the forms of several binary arrays, where the first one in the array is the location of first pixel and the second one is the running length of this pixel.

## Data Explorer

2.47 GiB

- test
- train
  - sample\_submission.csv
  - train.csv

## Summary

- 38.5k files
- 6 columns

## < train.csv (25.35 MiB)

Detail Compact Column

id	class	segmentation
<b>38496</b> unique values	<b>3</b> unique values	[null] 71% 29755 2 30112 8 30... 0% Other (33911) 29%
		12108 26 12373 28 12638 30 12903 32 13168 34 13434 34 13699 36 13...
case123_day20_slice_0083	large_bowel	17480 6 17745 8 18010 11 18275 13 18540 15 18805 18 19071 19 19337 20 19602 22 19868 22 20134 22 203...
case123_day20_slice_0083	small_bowel	22234 6 22499 9 22764 10 23030 11 23296 11 23562 12 23828 13 24094 14 24360 14 24627 14 24893 14 251...
case123_day20_slice_0083	stomach	11049 11 11313 16 11577 20 11842 23 12107 26 12371 30 12636 32 12902 33 13167 34 13433 35 13698 36 1...

2022/7/8

# UNET++ IMPLEMENTATION

```
▼ uwmgj-25d-stride2-dataset
  ▼ images
    ▸ images
  ▼ masks
    ▼ masks
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
      case101_day20_slice_...
```

- Data pre-processing:
  - Convert the names of slices in multi-level folders to the ids consist of original name with the head of cases and days in single level folder.
  - Implement functions to convert the RLE mask to the files in the format of ndarray.

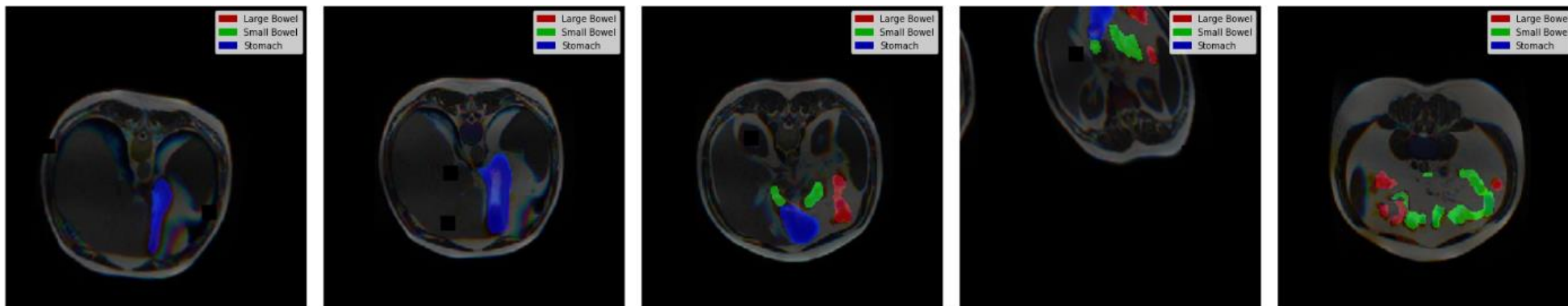
# UNET++ IMPLEMENTATION

## ■ Data Augmentation

```
data_transforms = {
    "train": A.Compose([
        A.Resize(*CFG.img_size, interpolation=cv2.INTER_NEAREST),
        A.HorizontalFlip(p=0.5),
        A.VerticalFlip(p=0.5),
        A.ShiftScaleRotate(shift_limit=0.0625, scale_limit=0.05, rotate_limit=10, p=0.5),
        A.OneOf([
            # A.GridDistortion(num_steps=5, distort_limit=0.05, p=1.0),
            A.OpticalDistortion(distort_limit=0.05, shift_limit=0.05, p=1.0),
            A.ElasticTransform(alpha=1, sigma=50, alpha_affine=50, p=1.0)
        ], p=0.25),
        A.CoarseDropout(max_holes=8, max_height=CFG.img_size[0]//20, max_width=CFG.img_size[1]//20,
                        min_holes=5, fill_value=0, mask_fill_value=0, p=0.5),
        ], p=1.0),
    "valid": A.Compose([
        A.Resize(*CFG.img_size, interpolation=cv2.INTER_NEAREST),
        ], p=1.0)
}
```

# UNET++ IMPLEMENTATION

Batch review



# UNET++ IMPLEMENTATION

```
import segmentation_models_pytorch as smp

def build_model():
    model = smp.UnetPlusPlus(
        encoder_name=CFG.backbone,
        encoder_weights="imagenet",
        in_channels=3,
        classes=CFG.num_classes,
        activation=None,
    )
    model.to(CFG.device)
    return model

def load_model(path):
    model = build_model()
    model.load_state_dict(torch.load(path))
    model.eval()
    return model
```

- Model

```
JaccardLoss = smp.losses.JaccardLoss(mode='multilabel')
DiceLoss = smp.losses.DiceLoss(mode='multilabel')
BCELoss = smp.losses.SoftBCEWithLogitsLoss()
LovaszLoss = smp.losses.LovaszLoss(mode='multilabel', per_image=False)
TverskyLoss = smp.losses.TverskyLoss(mode='multilabel', log_loss=False)

def dice_coef(y_true, y_pred, thr=0.5, dim=(2,3), epsilon=0.001):
    y_true = y_true.to(torch.float32)
    y_pred = (y_pred>thr).to(torch.float32)
    inter = (y_true*y_pred).sum(dim=dim)
    den = y_true.sum(dim=dim) + y_pred.sum(dim=dim)
    dice = ((2*inter+epsilon)/(den+epsilon)).mean(dim=(1,0))
    return dice

def iou_coef(y_true, y_pred, thr=0.5, dim=(2,3), epsilon=0.001):
    y_true = y_true.to(torch.float32)
    y_pred = (y_pred>thr).to(torch.float32)
    inter = (y_true*y_pred).sum(dim=dim)
    union = (y_true + y_pred - y_true*y_pred).sum(dim=dim)
    iou = ((inter+epsilon)/(union+epsilon)).mean(dim=(1,0))
    return iou

def criterion(y_pred, y_true):
    return 0.5*BCELoss(y_pred, y_true) + 0.5*DiceLoss(y_pred, y_true)
```

- Loss

# UNET++ IMPLEMENTATION

## ■ Hyper parameters:

```
class CFG:
    seed = 101
    debug = False # set debug=False for Full Training
    exp_name = '2.5D'
    comment = 'unetplusplus - regnet_008'
    model_name = 'UNetPlusPlus'
    backbone = 'timm-regnety_008'
    train_bs = 32
    valid_bs = train_bs*2
    img_size = [224, 224]
    epochs = 15
    lr = 2e-3
    scheduler = 'CosineAnnealingWarmRestarts'
    min_lr = 1e-6
    T_max = int(30000/train_bs*epochs)+50
    T_0 = 25
    warmup_epochs = 0
    wd = 1e-6
    n_accumulate = max(1,64/train_bs)
    n_fold = 5
    folds = [0]
    num_classes = 3
    device = torch.device("cuda:0" if torch.cuda.is_available()
```

```
class CFG:
    seed = 101
    debug = False # set debug=False for Full Training
    exp_name = '2.5D'
    comment = 'unetplusplus - resnet34'
    model_name = 'UNetPlusPlus'
    # backbone = 'timm-regnety_008'
    backbone = 'resnet34'
    train_bs = 32
    valid_bs = train_bs*2
    img_size = [224, 224]
    epochs = 15
    lr = 2e-3
    scheduler = 'CosineAnnealingWarmRestarts'
    min_lr = 1e-6
    T_max = int(30000/train_bs*epochs)+50
    T_0 = 25
    warmup_epochs = 0
    wd = 1e-6
    n_accumulate = max(1,64/train_bs)
    n_fold = 5
    folds = [0]
    num_classes = 3
    device = torch.device("cuda:0" if torch.cuda.is_available()
```



# UNET++ IMPLEMENTATION

Training complete in 6h 30m 57s

Best Score: 0.8683

Waiting for W&B process to finish... **(success)**.

[Links: fold-0|dim-224x224|model-UNetPlusPlus | uw-maddison-gi-tract – Weights & Biases \(wandb.ai\)](#)

## Run history:



## Run summary:

Best Dice	0.8991
Best Epoch	15
Best Jaccard	0.86834
LR	0.00069
Train Loss	0.05347
Valid Dice	0.8991
Valid Jaccard	0.86834
Valid Loss	0.10597

Source code with different loss function, scheduler and data augmentation.

# UNET++ IMPLEMENTATION

Links: [unetplusplus - resnet34](#) | [anony-moose-348813 Workspace – Weights & Biases \(wandb.ai\)](#)

Run history:



Run summary:

Best Dice	0.88545
Best Epoch	11
Best Jaccard	0.85445
LR	0.00069
Train Loss	0.05773
Valid Dice	0.88079
Valid Jaccard	0.85112
Valid Loss	0.12554

Unet++ with Resnet34



Thank you!