Overview
Congrats on achieving a prize winning rank in the SpaceNet-8, Flood Detection Using Multiclass Segmentation challenge. As part of your final submission and in order to receive payment for this match, please complete the following document.

1. Introduction
Tell us a bit about yourself, and why you have decided to participate in the contest.

- Name: Doyoung Jeong, Hakjin Lee, Junhwa Song, Yongjin Jeon, Yooseung Wang
- Handle: SIAnalytics
- Placement you achieved: -
- About you: Doyoung Jeong and Yongjin Jeon are research scientists at SI Analytics, Hakjin Lee and Junhwa Song are ML engineers at SI Analytics. Yooseung Wang is currently a graduate student at KAIST.
- Why you participated in the challenge: We are developing AI Change Detection solution for disaster damage assessment. Flood Detection Using Multiclass Segmentation challenge is exactly in line with our goals.

2. Solution Development
How did you solve the problem? What approaches did you try and what choices did you make, and why? Also, what alternative approaches did you consider?

In order to achieve the goal, three independent models were trained and used for inference: building segmentation model, road segmentation model, and flood segmentation model.

2.1. Modeling:

- Model Framework: MMSegmentation
  - MMSegmentation, which is a powerful Semantic Segmentation framework, provides SOTA model implementations, various augmentations with guaranteed performance, and training and inference pipeline with various options. In order to eliminate the confusion between members to integrate their code and share their intermediate implementation, we choose MMSegmentation as our baseline framework.
- Backbone Network: Swin-Transformer
  - Swin transformer is the winner solution in “LUAI Challenge 2021 on Learning to Understand Aerial Images”, it demonstrates that Swin trasnformer is a powerful backbone network in the remote sensing domain. we started making solutions from using swin-transformer as the backbone network.
  - MMSegmentation provides pre-trained backbone models. Swin transformer pre-trained with ImageNet-22k surpasses the other pre-trained with ImageNet-1k by a margin of + 2.32 mIoU on ADE20K val. We chose Swin transformer backbone pre-trained with ImageNet-22K since we believe representation extracted by the better backbone network makes a downstream task better.
  - We used swin-transformer with window12 for building segmentation. In swin-transformer structure, self-attention is computed within local windows which contain MxM patches. The computational cost is quadratic to window size M, and it is set to

7 by default. By increasing the window size M to 12 from 7, we took advantage of the satellite imagery domain in that the size of the receptive field would be increased in a transformer network. The performance using Swin-transformer with window 12 got better performance than the model with window 7 by a margin of + 0.83 building IoU on SpaceNet8 custom validation dataset.

- Decode Head: UperNet for building and flood, Segformer for road.
  - Upernet is a segmentation network using Feature Pyramid Network with a Pyramid Pooling Module from PSPNet. We have traditionally used UperNet as the default decoder in segmentation models for satellite images.
  - SegformerHead is one of the lightest decoders in mmsegmentation library, using only MLP Layer for mixing multi-level features extracted by the encoder. The inference result was predicted in a fragmented shape. when using UperNet for road segmentation. Since the connectivity must be guaranteed for better road graph extraction, we selected SegformerHead specifically for road segmentation.
  - In addition, we applied several models for large-scale image segementation(MSPNet, SSL), road network segmentation(NL-LinkNet) or change detection(SNUNet, FloodNet), but we couldn't see any good results.

- Additional auxiliary head (FCN branch) only for training
  - Auxiliary head is an architectural component that seeks to improve the convergence of deep network. We found that the segmentation performance varies with the presence of the auxiliary head. We put the additional classifier head behind the encoder network. The loss ratio of the main decode head to the auxiliary head was set to 1:0.4.

2.2. Training

- Dataset:
  - Because the final score is validated for a mysterious site, it is important to improve the generalization performance of the flood detection model. Therefore, in addition to the past SpaceNet datasets, SpaceNet2(Building) and SpaceNet3(Road), various datasets labeled with buildings and roads such as Massachusetts dataset or Inria dataset have been used to train the building segmentation and the road segmentation model. Using various datasets has had a particularly important impact on improvements of road segmentation performance.

- Loss Function: Focal Loss + Dice Loss + Lovasz Loss
  - We observed that the combination of Focal loss and Dice loss performed well in several paper or segmentation challenges. Since both the road and building segmentation are binary segmentation tasks, we used Lovasz Softmax Loss, which is more advantageous with fewer classes. However, when pretraining the model for building segmentation, we selected Cross Entropy Loss instead of Focal Loss.
  - We also tested topology-based loss such as Edge Loss, CLDice loss for road segmentation, but failed to show improvement.

- Augmentation: RandomShadow, RandomFog, PhotometricDistortion, Resize, RandomRotate, RandomFlip, RandomCrop

- RandomFog and RandomShadow: Unlike other satellite image datasets, SpaceNet8 have many images corrupted by clouds or clouds shadows. To simulate interventional data, we used RandomFog and RandomShadow from albumentations library. In order to simulate clouds more plausible, for RandomFog, we implemented it without gaussian blur.
- PhotometricDistortion: When using photometric distortion, the strength must be lowered than default parameters. The default photometric distortion setting creates unrealistic images and rather negatively affects performance. For this reason, the default hyperparameter for saturation, contrast_range, delta_hue were all lowered to about half of the defaults.
- We didn't use Test Time Augmentation (TTA) for inference because we observed that TTA with RandomFlip and RondomRatate showed poor performance.

2.3. Additional Features

- Reflect Padding: As demonstrated from other SpaceNet winner solutions, Reflect Padding is important when extracting roads or building near image boundaries. We properly implemented in mmsegmentation library and set the pad as 32.
- Input Size: According to some experiments, the optimal input_size is 1024x1024 for building and 1300x1300 for road and flood segmentations. However, due to GPU memory limitations, we use 1024x1024 instead of 1300x1300 for all models.
- Activation Checkpointing: In order to fit to 16gb GPU memory limit, Activation Checkpointing, which takes 30% slower to train but uses 30% less memory. was applied for Siamese flood segmentation.
- Postprocessing: Our post-processing was based on baseline, but hyperparameters for post-processing were adjusted to be generously predicted as flooded.

3. Final Approach

Please provide a bulleted description of your final approach. What ideas/decisions/features have been found to be the most important for your solution performance.

3.1. Road Segmentation

- Model: swin-transformer-Base(window7, ImageNet22k pretrained) + segformer
- Additional auxiliary head (FCN branch)
- Augmentation: PhotometricDistortion, RandomShadow, RandomFog, Resize, RandomRotate, RandomFlip, RandomCrop
- Reflect padding
- Number of classes: 2(Road and background)
- 2-stage training:
- Stage 1. pretraining
  - Loss = 1.0 * Focal loss + 1.0 * Dice loss + 1.0 * Lovasz loss
  - Dataset: Massachusetts road, Deepglobe, SpaceNet3
  - Input_size = (1024,1024)

  - batch size 8, 120 epoch, AdamW(lr=3e-05, weight_decay=0.01

- Stage 2. SpaceNet8
  - Loss = 1.0 * Focal loss + 1.0 * Dice loss + 1.0 * Lovasz loss
  - Dataset: SpaceNet8
  - Input_size = (1024, 1024)
  - Batch_size 8, 60 epoch, AdamW(lr=3e-0.5, weight_decay=0.01)
- Postprocessing
  - Applying Logit threshold 100 for binary classification
  - Apply cannab's solution at SpaceNet3 for extracting road network

3.2. Building Segmentation

- Model: swin-transformer-Base(window12, ResNet22k pretrained) + Upernet
- Additional auxiliary head (FCN branch)
- Augmentation: PhotometricDistortion, RandomShadow, RandomFog, Resize, RandomRotate, RandomFlip, RandomCrop
- Reflect padding
- Number of classes: 2 (Building and background)
- 2-stage training:
- Stage 1. pretraining
  - Loss = 1.0 * CrossEntropy loss + 1.0 * Dice loss + 1.0 * Lovasz loss
  - Dataset: SpaceNet2, Inria, Massachusetts building, xView2
  - Input_size = (650, 650)
  - Batch_size 16, 36 epoch, AdamW(lr=6e-0.5, weight_decay=0.01)
- Stage 2. SpaceNet8
  - Loss = 1.0 * Focal loss + 1.0 * Dice loss + 1.0 * Lovasz loss
  - Dataset: SpaceNet8
  - Input_size = (1024, 1024)
  - Batch_size 4, 60 epoch, AdamW(lr=1.5e-0.5, weight_decay=0.01)
- Post-processing is the same as the baseline

3.3. Flood Segmentation

- Siamese encoder network with swin-transformer-Base(window7, ImageNet22k pretrained) + Upernet
- Additional auxiliary head (FCN branch)
- Augmentation: Resize, RandomRotate, RandomFlip, RandomCrop
- Reflect padding
- Activation checkpointing
- Number of classes: 5 (background, Non-flooded building, flooded building, non-flooded road, flooded road)
- 1-stage training:
- Stage 1.
  - Loss: 1.0 * Focalloss + 1.0 * Dice loss + 1.0 * Lovasz loss
  - Dataset: SpaceNet8
  - Input_size = (1024, 1024)
  - Batch_size 4, 120 epoch, AdamW(lr=1.5e-0.5, weight_decay=0.01)
- Post-processing is the same as the baseline

4. Open Source Resources, Frameworks and Libraries
Please specify the name of the open source resource, the URL to where it can be found, and it's license type:

- mmcv, https://github.com/open-mmlab/mmcv, Apache-2.0
- mmsegmentation, https://github.com/open-mmlab/mmsegmentation, Apache-2.0
- pytorch, https://github.com/open-mmlab/mmsegmentation, BSD-Style
- Fiona, https://github.com/Toblerity/Fiona, BSD-3-Clause
- Shapely, https://github.com/shapely/shapely, BSD-3-Clause
- GeoJSON, https://github.com/jazzband/geojson, BSD-3-Clause
- utm, https://github.com/Turbo87/utm, MIT
- osmnx, https://github.com/gboeing/osmnx, MIT
- statsmodels, https://github.com/statsmodels/statsmodels, BSD-3-Clause
- geopandas, https://github.com/geopandas/geopandas, BSD-3-Clause
- networkx, https://github.com/networkx/networkx, BSD-3-Clause
- rasterio, https://github.com/rasterio/rasterio, MapBox customized
- scikit-image, https://github.com/scikit-image/scikit-image, Modified BSD
- scikit-learn, https://github.com/scikit-learn/scikit-learn, BSD-3-Clause
- numpy, https://github.com/numpy/numpy, BSD-3-Clause
- numba, https://github.com/numba/numba, BSD-2-Clause
- pygdal, https://github.com/nextgis/pygdal, Customized MIT/X style
- opencv, https://github.com/opencv/opencv, Apache-2.0

5. Potential Algorithm Improvements
Please specify any potential improvements that can be made to the algorithm:

- Additional datasets and more training epochs: There was a limitation when making model configs due to the 48-hour limitation of training time. If we would use more data to allow the model to learn enough the intra-class variance of building or road class, we expect better performance.
- Increase the input size for road and flood segmentation: Input size 1024x1024 is optimal for buildings, but1300x1300 performs better than 1024x1024 for roads and floods segmentation. However, the input size of 1300x1300 is limited to 16gb of GPU memory for validation, so we trained all models with 1024x1024. If the individual GPU memory is large, it would be better to increase the input size for road and floods segmentation.
- Applying SpaceNet 5 solution for road extraction: We used the cannab's solution in SpaceNet 3 to extract the road network, but wouldn't it be better to use the SpaceNet 5 solution…?
- Occlusion in roads: In many cases, roads are occluded by trees or vehicles, so there are some cases where good performance is achieved by extracting the roads from both pre-image and post-image respectively and merging them. However, fatal results occur when applied in areas such as Germany where satellite images are not properly co-registered.

6. Algorithm Limitations
Please specify any potential limitations with the algorithm:

- Road prediction at road junctions: Due to sidewalk, there was a limitation for model to predict the road at the road junctions. Whether or not the junction is extracted has a fatal effect on the APLS score, a metric that measures the connectivity of the road network.
- Generalization of the model for flood segmentation: Since xBD Building Damage Assessment dataset was not used to train the flood segmentation model, only flood in advanced countries was learned by the model. Generalized performance may decrease in applying the model in developing countries.

7. Deployment Guide
Please provide the exact steps required to build and deploy the code:

- Build docker image:

```
docker build -t <id> .
```

- Run docker:

```
docker run -v <local_data_path>:/data -v <local_writable_area_path>:/wdata -it <id>
```

Please see https://github.com/topcoderinc/marathon-docker-template/tree/master/data-plus-code-style

8. Final Verification
Please provide instructions that explain how to train the algorithm and have it execute against sample data:

- Train:

```
./train.sh /data/train
```

- Test:

```
./test.sh /data/test/ solution.csv
```

9. Feedback

Please provide feedback on the following - what worked, and what could have been done better or differently?

- Problem Statement -
- Data -
- Contest – It will be better to be able to check the validation for final testing in each submission.
- Scoring – Showing the weights for each score and intermediate scores (such as building IoU, APLS metric) could help the participants analyze their solutions and improve their models.