**Marathon Match - Solution Description**

**Overview**

Congrats on winning this marathon match. As part of your final submission and in order to receive payment for this marathon 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: Konstantin Maksimov (official Ukrainian transliteration is Kostiantyn Maksymov, I just prefer my own variant of transliteration)
* Handle: MaksimovKA
* Placement you achieved in the MM: TBA
* About you: I am Computer Vision / Machine Learning Engineer in VirtualControl company (virtualcontrol.io), previously worked as Data Scientist and Computer Vision Engineer for different different companies for different domains (check my linkedin for more information <https://www.linkedin.com/in/konstantin-maksimov/>)
* Why you participated in the MM: I like to participate in different DS competitions and especially in geo oriented competitions. Also I got third place on spacenet 4 year ago and then failed on spacenet 5 and decided to recover this fail on new spacenet challenge.

1. **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?

* I solved problem as semantic segmentation problem. I made 3 classes as target masks - building instance, building border and separation between buildings as it was done in second place of Spacenet 4 contest. As an input data I just used raw 4 channels input from SAR images that we have in training data (my experiments showed that there was no any difference how to preprocess data so I decide not to do any preprocessing).
* After that almost all experiments I done with Unet model with ResNet-34 encoder from this great repo - <https://github.com/qubvel/segmentation_models.pytorch>
* Most of my experiments was about to find proper random crop size and augmentations setup. The best crop size was 320 (higher sizes gave zero increase in quality). The best augmentations setup was combination of crop, random flips and small rotates and small zooming (see code for more details). Also experiments showed that some augmentations like 90 degrees rotate was very harmful for training).
* Also I experimented with parameters for watershed transforms but initial params was the best.

1. **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:

* Divide data randomly to 8 folds (so it will be able to under training time limitations).
* Train on these folds 8 Unet with SENet-154 encoder model from this great repo - <https://github.com/qubvel/segmentation_models.pytorch>
* Add TTA (Test Time Augmentations) Vertical and Horizontal Flips for inference stage.
* Make inference using 8 models and TTA and make mean averaging of predicted segmentation masks.
* After segmentations maks perform the same watershed approach as in my solution from Spacenet 6 contest (<https://cutt.ly/AyAPSzp>) and get buildings instance prediction.

1. **Open Source Resources, Frameworks and Libraries**

Please specify the name of the open source resource along with a URL to where it’s housed and it’s license type:

* segmentation\_models.pytorch,<https://github.com/qubvel/segmentation_models.pytorch>, MIT
* catalyst, <https://github.com/catalyst-team/catalyst>, Apache-2.0
* albumentations, <https://github.com/albumentations-team/albumentations>, MIT
* fire,<https://github.com/google/python-fire>,Apache-2.0
* rasterio,<https://github.com/mapbox/rasterio>, <https://github.com/mapbox/rasterio/blob/master/LICENSE.txt>
* pytorch-toolbelt, <https://github.com/BloodAxe/pytorch-toolbelt>, MIT

1. **Potential Algorithm Improvements**

Please specify any potential improvements that can be made to the algorithm:

* I realized that data has a lot of overlaps very late, so instead of random folds generation it is better to perform folds split on tiles that does not separate with each other - I mean the best way is to make validation train split with tiles that does not separate with each other.

1. **Algorithm Limitations**

Please specify any potential limitations with the algorithm:

* SENet-154 is very and very big encoder and if you want to make fast predictions you should select some other encoder.

1. **Deployment Guide**

Please provide the exact steps required to build and deploy the code:

1. **Final Verification**

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

1. **Feedback**

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

* Problem Statement - Statement was very great with all needed details, the one thing could improve user interaction - If you can add bookmarks at the begging, sometimes it is hard to scroll all the text.
* Data - everything was great, but I think it will be great if spacenet hosts will make possible by rules to take info from other tiles on inference stage, it could increase accuracy of the models.
* Contest - It will be great to see next spacenet challenge.
* Scoring - Add possibility to run docker containers from the begging on 1 gpu as you did in the end in the validation tool.

**NOTE**: Please save a copy of this template in word format. Please do not submit a .pdf