DGTerritório

Active Learning Implementation Documentation

Technical document

[Manvel Khudinyan]

Outline

- Training data selection
- Active learning procedure
- Uncertainty map post-processing
- New training data resampling

All the scripts and toolboxes for Active Learning performance can be found in the GitHub repository with the link below:

https://github.com/armkhudinyan/active-learning/tree/master/AL_T29SND

1. Training data selection

For the initial Active Learning (AL) implementation the extent area of Sentinel 2 Tile Number 29SND was selected. The class nomenclature consists of 30 classes (table 1), which will be added with one more class ("burnt areas") from the second iteration. Initially there was a 540044 training sample labeled using 2018COSsim map, which was randomly down-sampled to 6705 sample (250 samples per class if available).

id	class_name	size
0	shrubland	250
1	build_up	250
2	baresoil	250
3	natural_grassland	250
4	sparse_vegetation	103
5	wetland	250
6	water	250
7	bare_rock	25
8	agricultural_grassland	250
9	ryegrass	250
10	lupin	250
11	wheat	250
12	oatmeal	250
13	sorghum	250
14	rice	250
15	maize	250
16	potato	250
17	barley	250
18	tomato	250
19	orchards	250
20	olive_trees	250
21	vineyard	250
22	sobreiro_corkoak	250
23	eucalyptus	250
24	other_broadleaf	250
25	azinheiro_holmoak	43
26	pbravo_maritimepine	250
27	pmanso_stonepine	250
28	openpbravo_openmaritimepine	250
29	other_coniferous	34

Table 1. Initial training dataset

Based on a prior experiment on feature importance using relief method, a subset of 40 features were selected for AL procedure (table 2).

feature	rank
S2X_L2A_composite_median_20180301_20180331_Portugal.3	0
S2X_L2A_composite_median_20180701_20180731_Portugal.1	1
S2X_L2A_composite_median_20180601_20180630_Portugal_NDBI	2
S2X_L2A_composite_median_20171201_20171231_Portugal_NDBI	3
S2X_L2A_composite_median_20180101_20180131_Portugal_NDBI	4
S2X_L2A_composite_median_20180301_20180331_Portugal_NDWIF	5
S2X_L2A_metric_20171001_20180930_Portugal_b3_q90	6
S2X_L2A_composite_median_20180101_20180131_Portugal_NDWIF	7
S2X_L2A_composite_median_20180801_20180831_Portugal.1	8
S2X_L2A_composite_median_20180901_20180930_Portugal.1	9
S2X_L2A_composite_median_20180201_20180228_Portugal.3	10
S2X_L2A_composite_median_20180501_20180531_Portugal.8	11
S2X_L2A_composite_median_20180201_20180228_Portugal_NDBI	12
S2X_L2A_composite_median_20180801_20180831_Portugal.2	13
S2X_L2A_composite_median_20171101_20171130_Portugal.4	14
S2X_L2A_composite_median_20171101_20171130_Portugal.5	15
S2X_L2A_composite_median_20171001_20171031_Portugal.7	16
S2X_L2A_composite_median_20171001_20171031_Portugal.4	17
S2X_L2A_composite_median_20171001_20171031_Portugal.1	18
S2X_L2A_composite_median_20171101_20171130_Portugal.7	19
S2X_L2A_composite_median_20180101_20180131_Portugal.8	20
S2X_L2A_composite_median_20180401_20180430_Portugal.4	21
S2X_L2A_composite_median_20180901_20180930_Portugal.8	22
S2X_L2A_composite_median_20180601_20180630_Portugal.2	23
S2X_L2A_composite_median_20180301_20180331_Portugal_NDVI	24
S2X_L2A_composite_median_20180801_20180831_Portugal_NDBI	25
S2X_L2A_metric_20171001_20180930_Portugal_b5_q75-q25	26
S2X_L2A_composite_median_20180401_20180430_Portugal_NDBI	27
S2X_L2A_composite_median_20171201_20171231_Portugal.1	28
S2X_L2A_composite_median_20171201_20171231_Portugal.6	29
S2X_L2A_composite_median_20171201_20171231_Portugal.8	30
S2X_L2A_composite_median_20180301_20180331_Portugal.1	31
S2X_L2A_composite_median_20180501_20180531_Portugal.3	32
S2X_L2A_composite_median_20180501_20180531_Portugal.7	33
S2X_L2A_composite_median_20171001_20171031_Portugal_NDWIF	34
S2X_L2A_composite_median_20171201_20171231_Portugal_NBR	35
S2X_L2A_composite_median_20180701_20180731_Portugal_NDVI	36
S2X_L2A_composite_median_20180801_20180831_Portugal.7	37
S2X_L2A_composite_median_20180701_20180731_Portugal.10	38
S2X_L2A_composite_median_20180501_20180531_Portugal.4	39

Table 2. The ranking of the best 40 fetures by the relief feature selection method

Both data subsetting and feature selection processes are implemented in Python and the code can be found in the repository with the name **“data_subsetting.py”**. The code requires only to define the path to the proper .csv file containing the dataset and set up the number of samples per class and can be re-run for a new output.

2. Active learning procedure

AL main procedure is implemented in Python environment and can be found in the repository with a filename of “**AL_procedure.py**”. The code requires to define paths for the chosen Sentinel 2 image tile in the database of DGT, for training data in .csv and also for feature rankings .csv. The code is designed the way that it can accept new list of best features and read the proper bands for DGT image repository. The code also takes care of the feature’s name order and can handle any order of feature naming in the training data.

```
n_run = 0
#====================
# Defining paths
#====================
#sys.path.append(join(dirname(__file__), '..', '..'))
PATH = r'C:\Users\arman\Documents\GitHub\active-learning\AL_T29SND'

SENTINEL_PATH = r'\\dgt-759\S2_2018\Theia_S2process\T29SND\composites'
METRICS_PATH = join(SENTINEL_PATH, 'metrics')
INDICES_PATH = join(SENTINEL_PATH, 'indices')
#INDICES_METRICS_PATH = join(SENTINEL_PATH, 'indices', 'metrics')
TRAIN_PATH = join(PATH, 'train_data','train_best_40.csv')
FEATURE_NAME_PATH = join(PATH, 'feature_importance', 'feature_rankings.csv')
OUT_PATH = join(PATH, 'results')
```

Note: The names of features in training dataset should be matching with the ones from DGT repository.

As for the classification Random Forest classifier is set up, number of trees are set up equal to 500 and it occupies 34 logistic processors (“n_estimators=500”, “n_jobs=34”).

The uncertainty is calculated used the probability output of Fandom Forest classifier. In this case the difference of probability of the 2 highest classes for each pixel is calculated, which is known as Margin Sampling strategy. The uncertainty has values between 0 and 1, while 0 is very high uncertainty and 1 is absence of uncertainty:

$$M = argmin\ P_\theta(y_1|x) - P_\theta(y_2|x),$$

where P stands for the probability output of θ model for a sample x and y_1, y_2 are the first and second most probable classes.

In order to handle with classification of such huge areas and avoid from memory error, the dataset is split into **40 parts**. This is adjustable form the code and should be increased if memory error occurs (like if the server is already busy partially or if the experiment is run on a weaker machine).

The outputs are 2 rasters: tile classification map and the uncertainty map.

Note: In the beginning of the code it should be specified the n-th run of the AL procedure which will be expressed also in the output raster's names.

This will allow to keep the order in AL procedure and track the progress. The first initial run had the number of "0": "lulc_0.tif" as classification output and "uncertainty_0.tif" as uncertainty output.

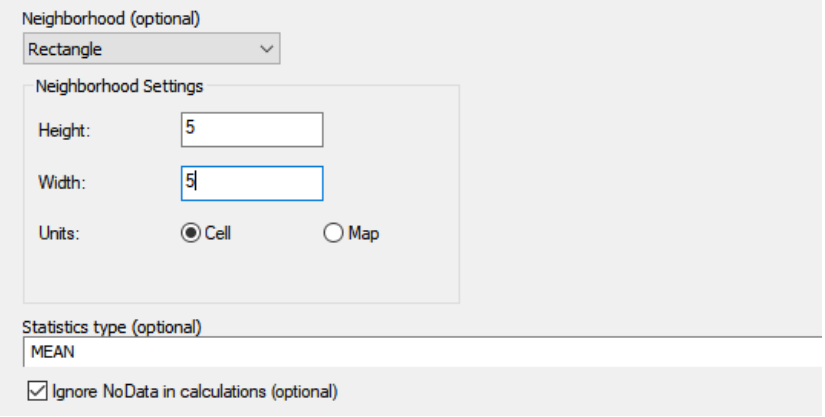
3. Uncertainty map post-processing

Once the uncertainty map is produced, several post-processing steps should be held in order to make it ready for the most uncertain pixel's selection by photointerpretation. Those steps are

- Neighborhood analysis
- Thresholding (binarization)
- Vectorization
- Intersection
- Large uncertainty patches selection

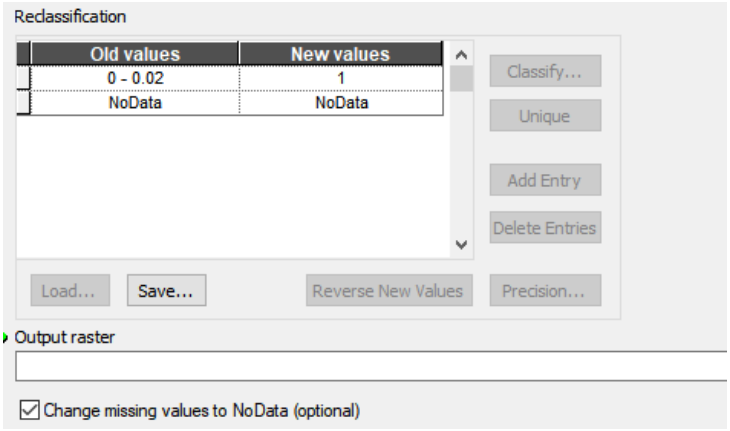
Neighborhood analysis

A focal moving window function ArcMAP (toolboxes\spatial analyst tools\neighborhood\focal statistics) is applied on the uncertainty raster map using 5x5 pixel window size. The purpose of applying this function is to avoid selecting a separated pixel with high uncertainty and select areas (patches) of high uncertainty instead. The "Statistics type" is set up to "MEAN".



Thresholding

In order to select only the areas with high uncertainty we put a low threshold on the uncertainty value and produce binary map containing only high uncertain patches. This threshold is a heuristic and the first experimental values was set up equal to 0.02. For this purpose, "Reclassify" tool of ArcMAP was used (toolboxes\spatial analyst tools\reclass\reclassify).



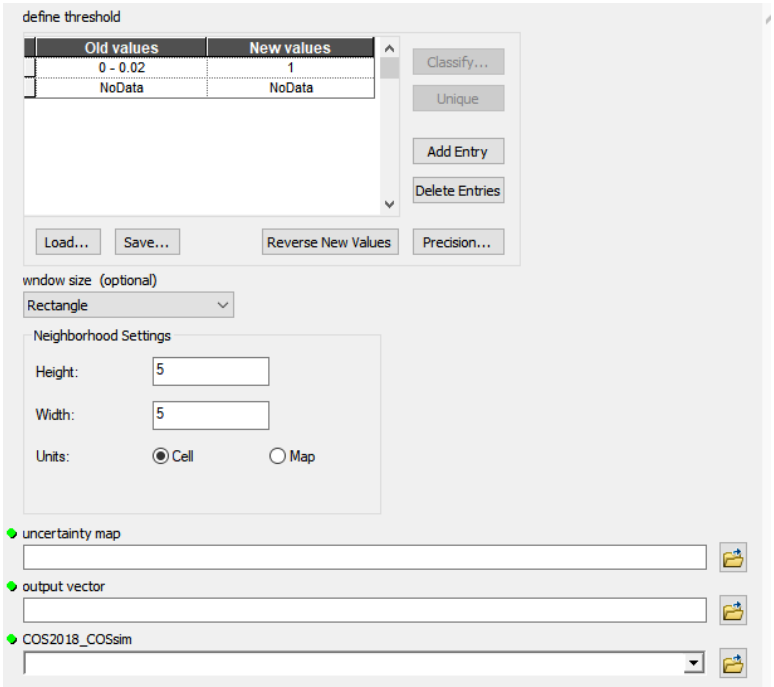
Vectorization

Once the binary map is produces, the raster map is converted into vector map in order to intersect with COS2018 vector map. For this "Raster to polygon" tool from ArcMAP was used (toolboxes\conversion tools\from raster\raster to polygon).

Intersection

We intersect vectorized uncertainty map with COS2018 product to identify the class belongness of uncertainty based on COS2018 map. For this "Intersect" tool of ArcMAP was used (toolboxes\analysis tools\overlay\intersect).

All the previous steps are combined into one tool with "ModelBuilder" function in ArcMAP and can be found in the GitHub repository as **"TBox.zip",** the tool name is "AL_post_process". It requires to set up the paths to the uncertainty raster, path to the 2018COSsim and the output directory. It also allows to change the parameters of window size for neighborhood analysis and threshold value for binarization.



Large uncertainty patches selection

Once the intersection is performed, we select the N largest uncertainty patches per class. For this we run the Python script from the GitHub repository called "**large_patch_selection.py".** It requires to set up only the path to the intersected vector file and the number of patches to be selected per class. Also

```
12 PATH = r'C:\Users\arman\Desktop\ActiveLearning\Experiment\dgt_T29SND\results'
13 gdf_uncert = gpd.read_file(join(PATH,'uncert_intersect.shp'))
14
15 # define the amount of patches to be selected per class
16 n_patches = 10
```

The output of the run is the N largest patches per class of interest (excluded "irrigated" and "rainfed" classes).

Note: In the beginning of the code it should be specified the n-th run of the AL procedure which will be expressed also in the output csv's names.

4. New training data resampling

The last step of preparation before running the Active Learning procedure is the new training data selection and addition to the one from previous iteration. For this we run the Python script from the GitHub repository called **"train_data_resampling.py**". For this we specify the path to the folder containing the initial training data in .csv files. Each successive iteration will select N number of randomly selected samples from the ground truth areas and will add same number of samples from agricultural classes. For the later one we use the large amount of initial training data sampled for the study area.

This script saves the selected agricultural data, the actively selected data and the combination of all samples for the next iteration in separate .csv's. Their names also contain the sequential number of iterations, which should be defined before running the script. This will allow us to get back to any step of the Active learning procedure and re-do or take another path with new hyperparameters if needed.