



Automatic detection of charcoal kilns in Somalia with a computer vision approach

Living Planet Symposium, Bonn. May 24th, 2022
European Commission - Joint Research Center, Unit D.5

Laura MARTINEZ-SANCHEZ | JRC

Astrid VERHEGGHEN | JRC

Michele BOLOGNESI | FAO

Michele MERONI | JRC

Marijn VAN DER VELDE | JRC

Felix REMBOLD | JRC

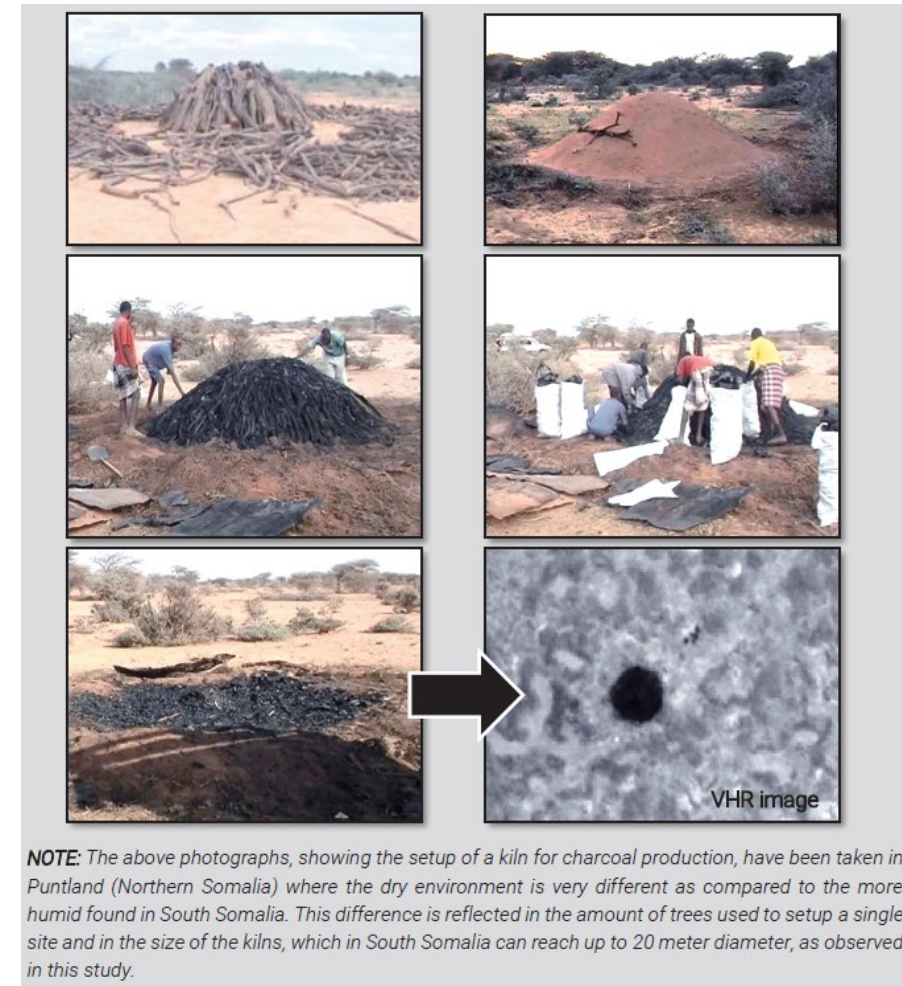
Overview

In sub-Saharan Africa (SSA), woodfuel (charcoal and firewood) is the most common source of energy used by households for cooking and heating.

- Africa accounts for **two third of the global charcoal** production in 2018 (>30 millions tons) (FAO)
- Charcoal is mainly used across urban areas and is a **major source of forest degradation**.

In Somalia, charcoal production is either directly driven by the **domestic demand** for fuel or can be linked to **illegal exports** in support to war regimes.

- In 2012, the UN Security Council passed a resolution which **banned the export of charcoal** from Somalia
- The FAO (SWALIM group) is monitoring the impact of **charcoal production** on the natural vegetation and **its dynamics**.
- Charcoal is made by burning wood in a low-oxygen environment in **type of oven known as “kiln”**.
- **The layer of black ashes left** after the charcoal is formed and loaded into bags is visible on satellite VHR imageries (*Bolognesi et al., 2018*)



Kiln set up in Northern Somalia (Bolognesi et al., 2018)

SWALIM (Somalia Water and Land Information Management)



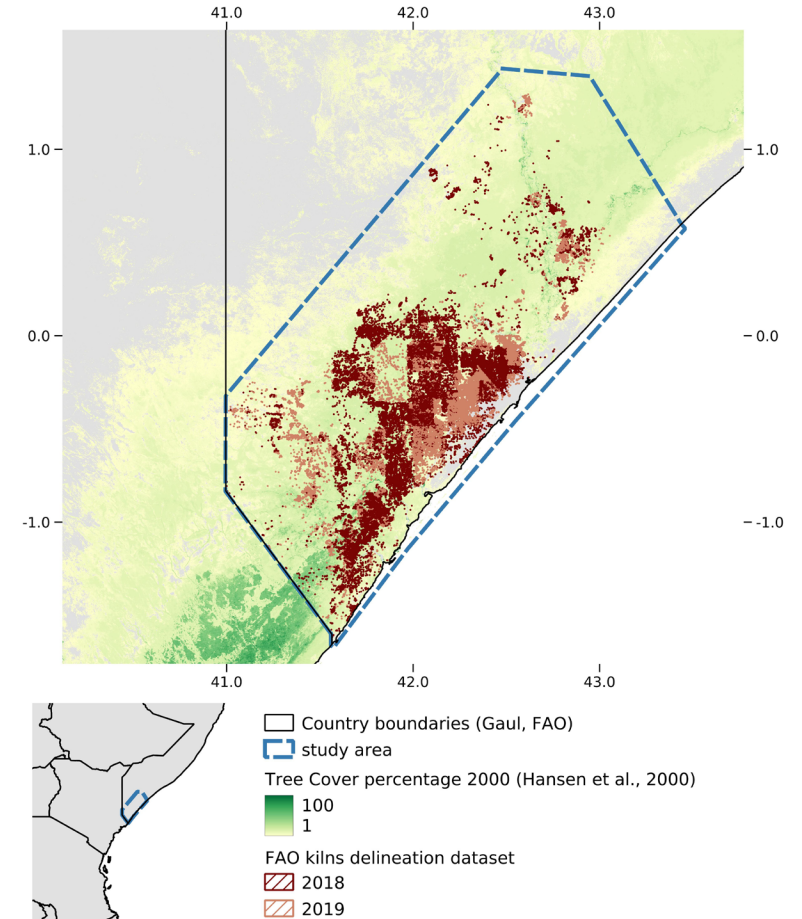
Objectives

- The FAO (SWALIM group) is monitoring forest degradation driven by charcoal production by **visually detecting** the “kilns” sites on VHR images.
- The main purpose of this study is to test whether an **AI approach** can be trained to **automatically identify kilns** related to illegal charcoal production on VHR images.



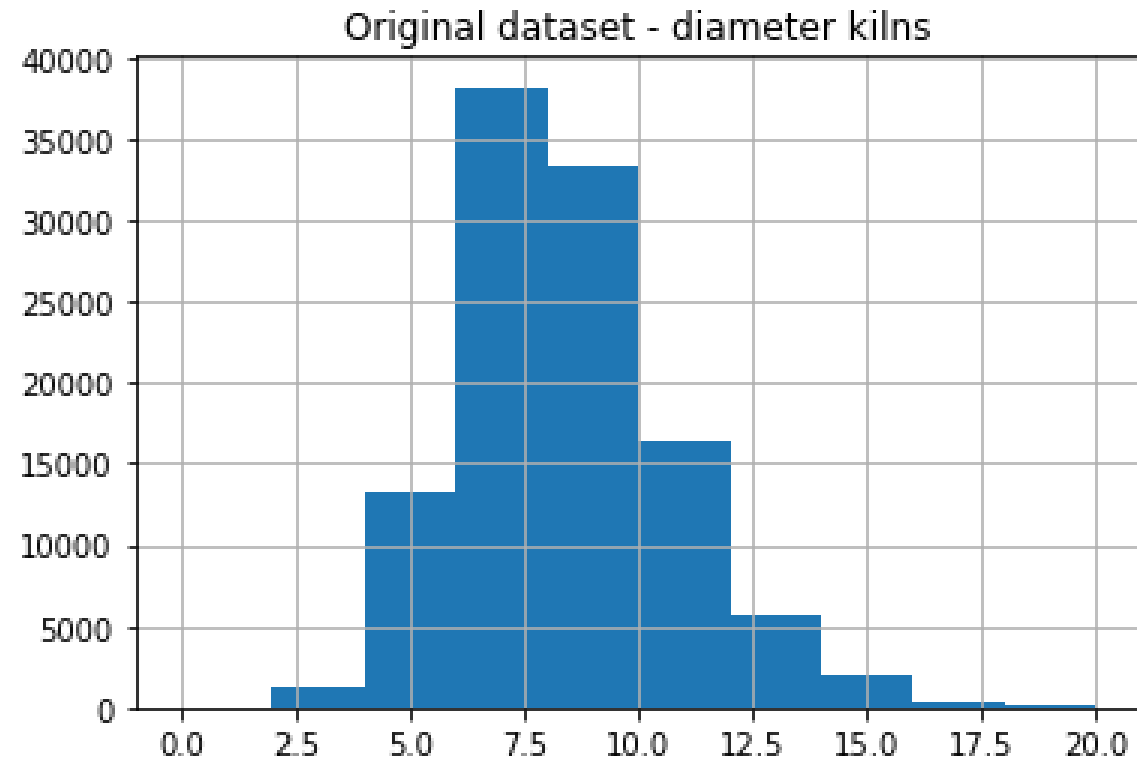
Study area and data availability

- The FAO-SWALIM has created an **expert-based kiln delineation** on VHR imagery that contains more than **69,500 objects** for the **years 2017-2019** over an area of about 37,700 km² in southern Somalia
- We used a **subset** of this large dataset paired with a collection of **WorldView-2** images for the years 2018 and 2019, obtained through the DigitalGlobe NextView license
- 0,5m resolution obtained (Panchromatic/RGB-no NIR), original data:
 - Panchromatic: 0.46 m GSD (0.52 m at 20° off-nadir)
 - RGB **1.8 m** GSD (2.4 m at 20° off-nadir)

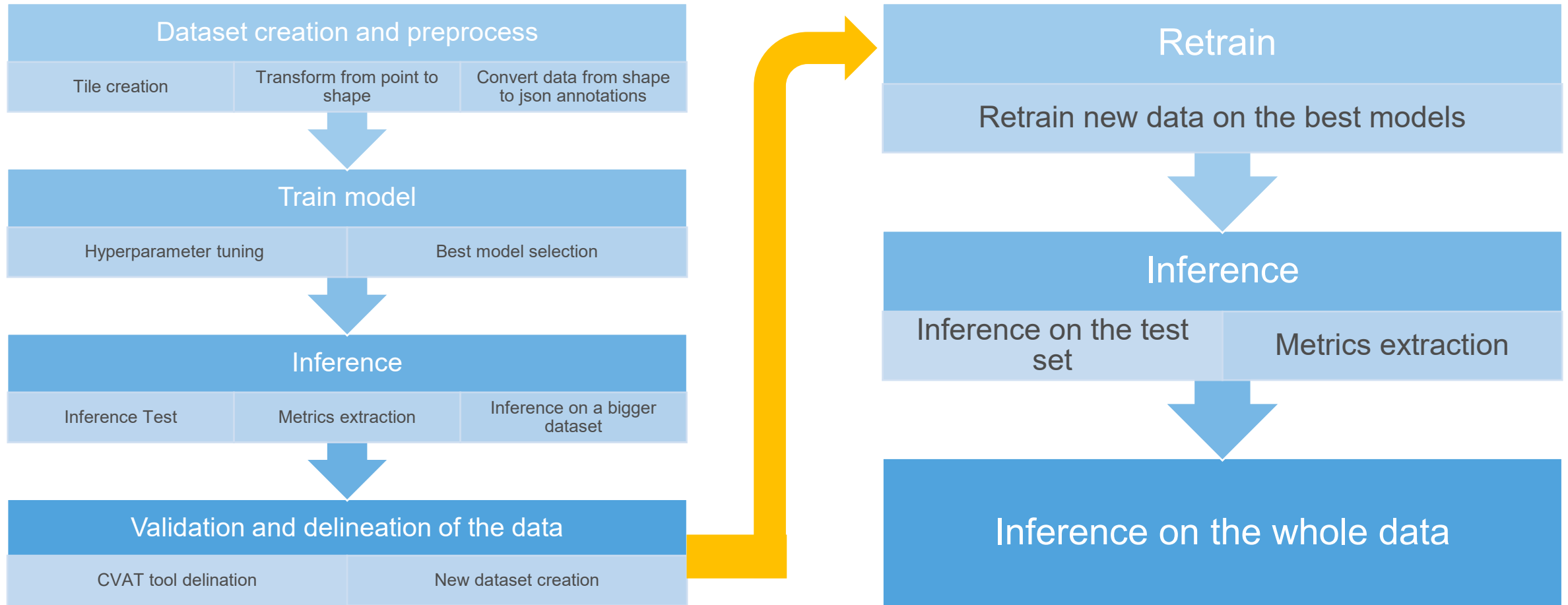


Challenges

- To capture all the different sizes of kilns
- To generalize to the whole area (amount of data, different sensors)
- Acquisitions date do not always match the date of delineation
- Kilns can remain visible for a long time
- Misalignments in some cases

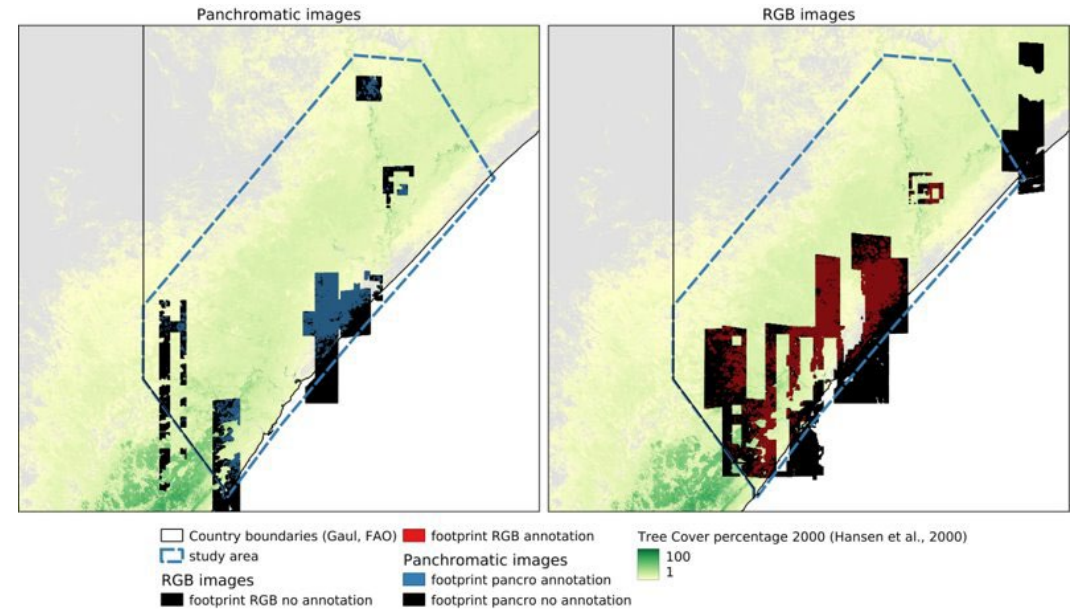


Method pipeline



Dataset creation and pre-process

- Data used:
 - Vector information 2017-2019, circular polygons
 - RGB-Panchromatic images: 2018-2019
- Pre –process:
 - Tiling 1024 px with 20 rows overlap
 - Erase the tiles with no data
 - Split of tiles in tiles with/without kilns



Data type	Nb of tiles	Nb of tiles with delineation	Nb of objects
Panchromatic	33.407	8.586	46.520
RGB	76.651	16.694	81.597
Total	110.058	25.280	127.117

Dataset creation and pre-process

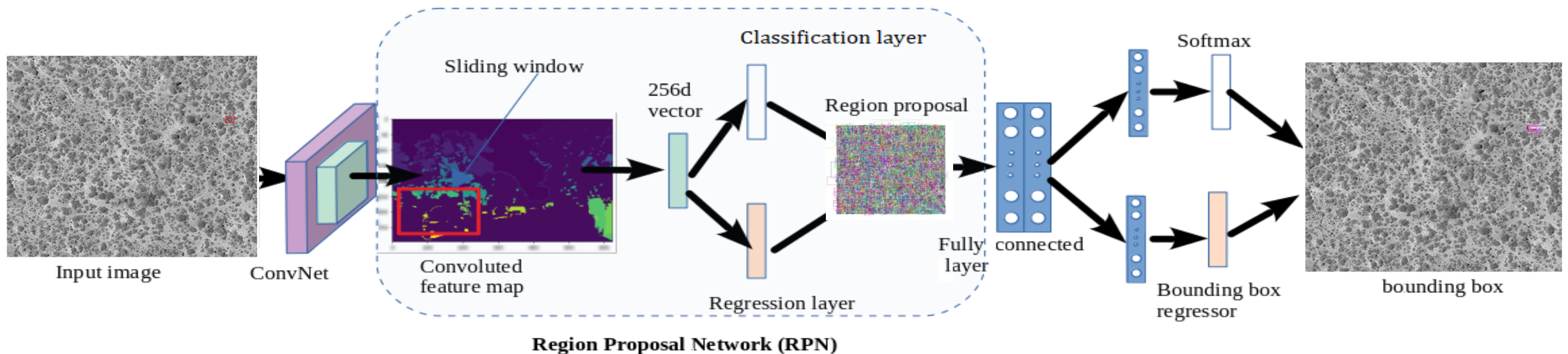
- Date acquisitions do not always match the date of delineation
- Kilns can remain visible for some time
- Misalignments in some cases
- Creation of the sets train/val and test

Image type	Train + 0.1 validation	Test
Panchromatic	70 tiles (387 kilns)	18 tiles (120 kilns)
RGB	110 tiles (426 kilns)	28 tiles (115 kilns)



Faster RCNN

- Object detection and classification



Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. *Faster R-CNN: towards real-time object detection with region proposal networks*. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 91–99.

Metrics extraction and selection best models

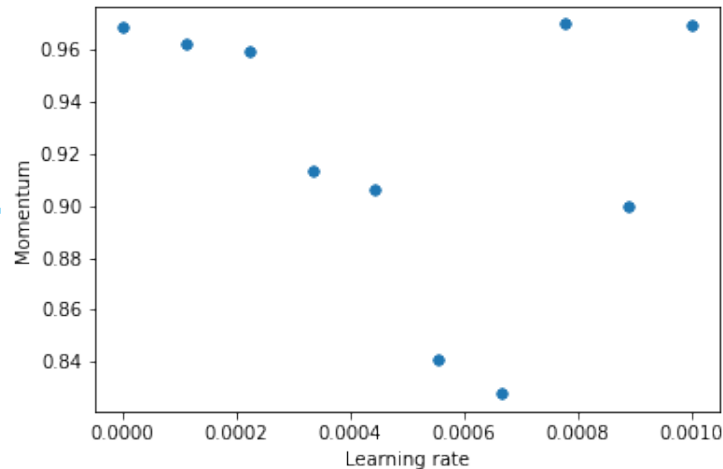
Hyper-parameter tuning

ConvNets/Backbones

- faster_rcnn_R_101
- faster_rcnn_R_50
- faster_rcnn_R_50
- faster_rcnn_X_101

Regions to analyze and classify

- 1024
- 2048



Validation metrics

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

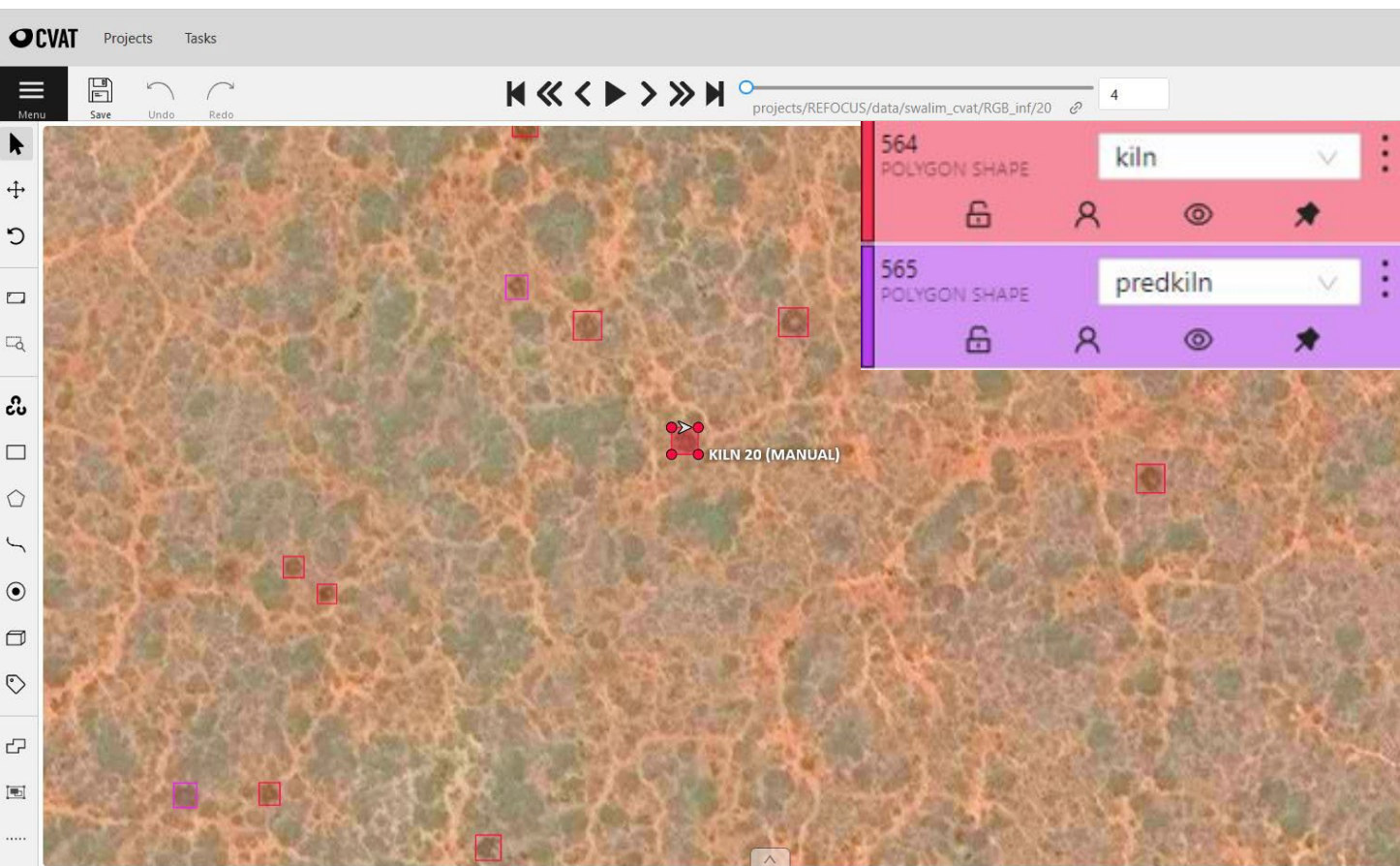
$$mAP_{50} = \frac{1}{N} \sum_{i=1}^N AP_i$$

$$IoU_{50} = \frac{area(Bbox_{pred} \cap Bbox_{gt})}{area(Bbox_{pred} \cup Bbox_{gt})}$$

$$AP = \sum_{i=1}^{n-1} (r_{i+1} - r_i) p_{interp}(r_{i+1})$$

Selection best model and creation final dataset

Image set	TP	FP	FN	AP50	Precis	Recall	F1-score	Backbone/ConvNet	Learning rate	Momentum	# Regions
Panchromatic	116	16	4	93.1	0.87	0.96	0.91	faster_rcnn_X_101	0.0008	0.89	1024
RGB	82	7	33	60.1	0.92	0.71	0.80	faster_rcnn_X_101	0.001	0.96	2048



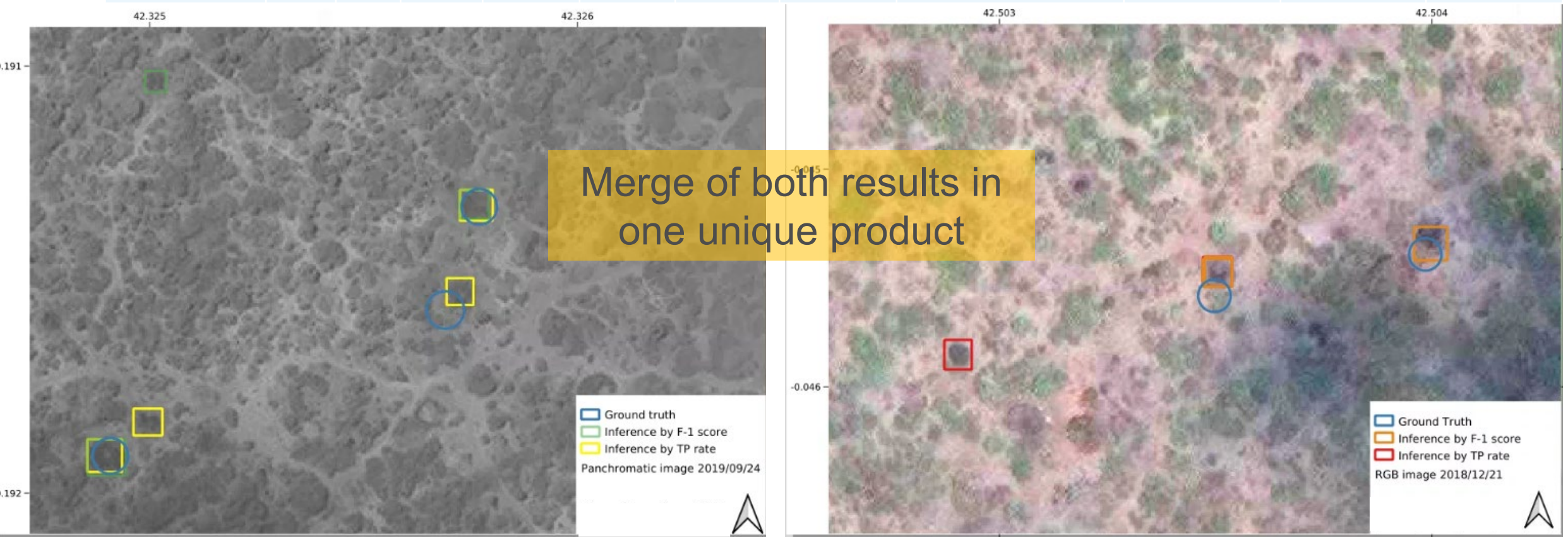
- CVAT tool to validate and delineate new kilns
- Creation of the final datasets

Image type	Train + 0.1 validation	Test
Panchromatic	240 tiles (1.140 kilns)	60 tiles (307 kilns)
RGB	240 tiles (803 kilns)	60 tiles (184 kilns)

- Retrain the **best 10 models** on the new dataset

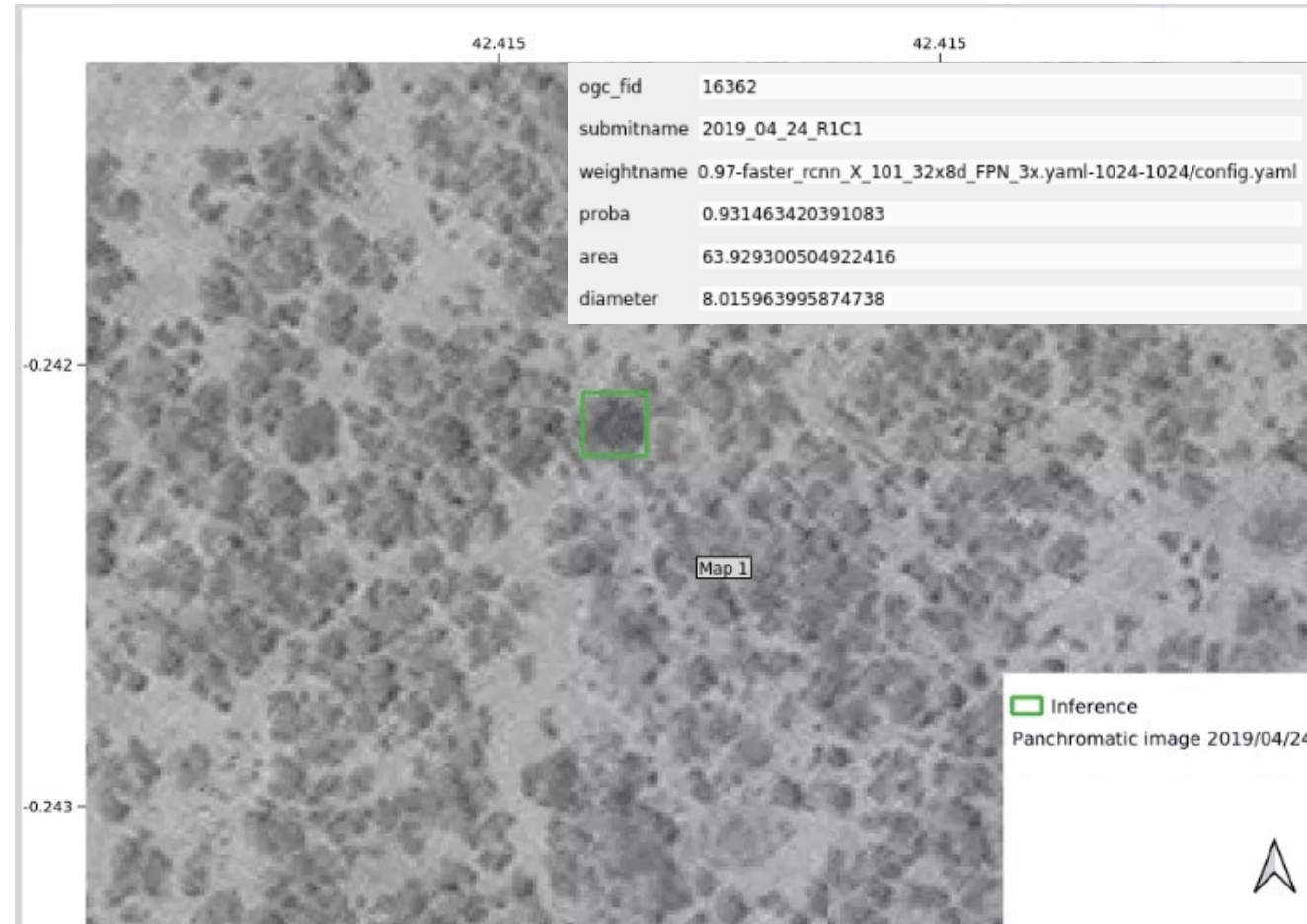
Final results – F1 score / TP rate

Image set	TP	FP	FN	AP50	Precis	Recall	F1-score	Backbone/ConvNet	Learning rate	Momentum	# Regions
Panchromatic	289	52	18	84.3	0.85	0.94	0.89	faster_rcnn_X_101	0.001	0.96	1024
Panchromatic	281	26	26	81.9	0.91	0.91	0.91	faster_rcnn_X_101	0.0007	0.96	2048
RGB	147	42	37	70.1	0.78	0.79	0.78	faster_rcnn_X_101	0.001	0.96	1024
RGB	145	27	39	68.7	0.84	0.79	0.81	faster_rcnn_X_101	0.0007	0.96	2048



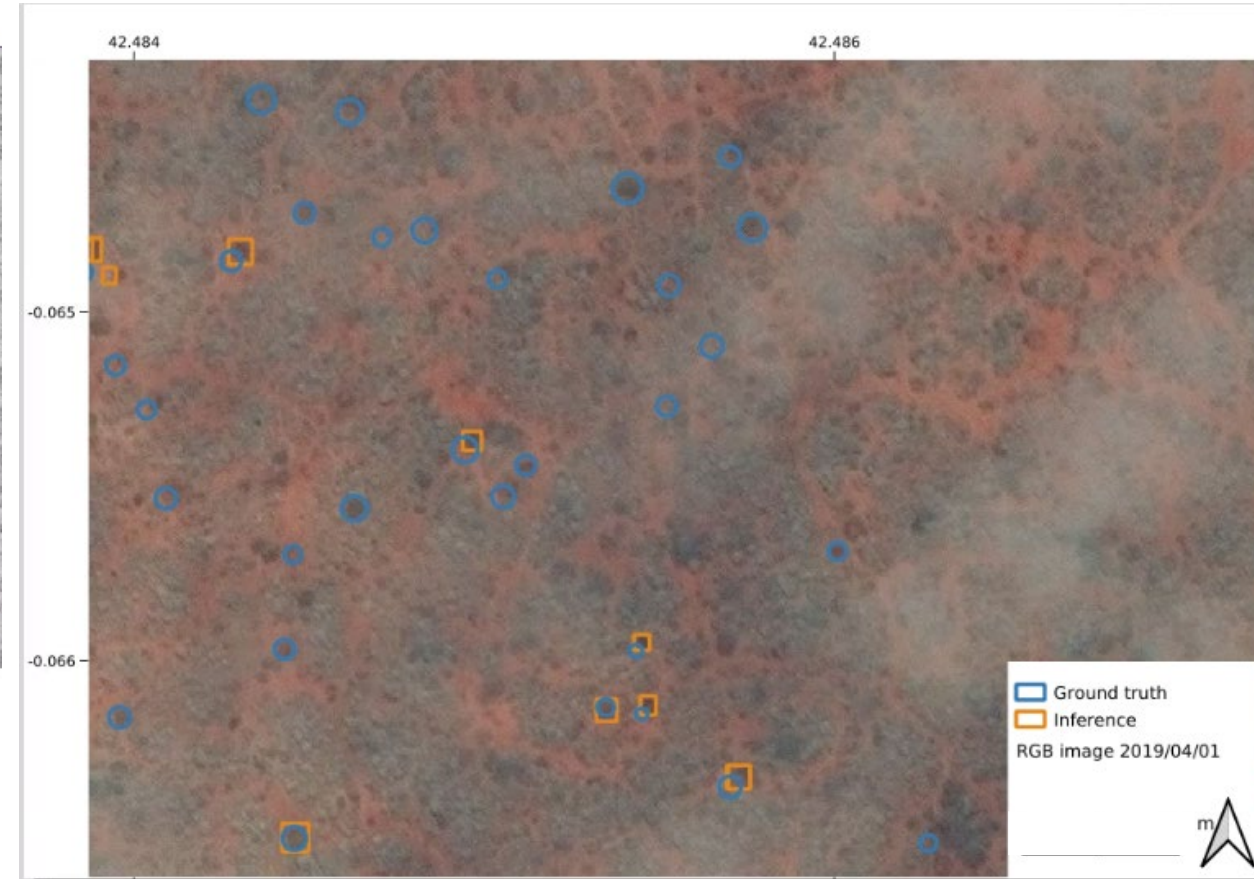
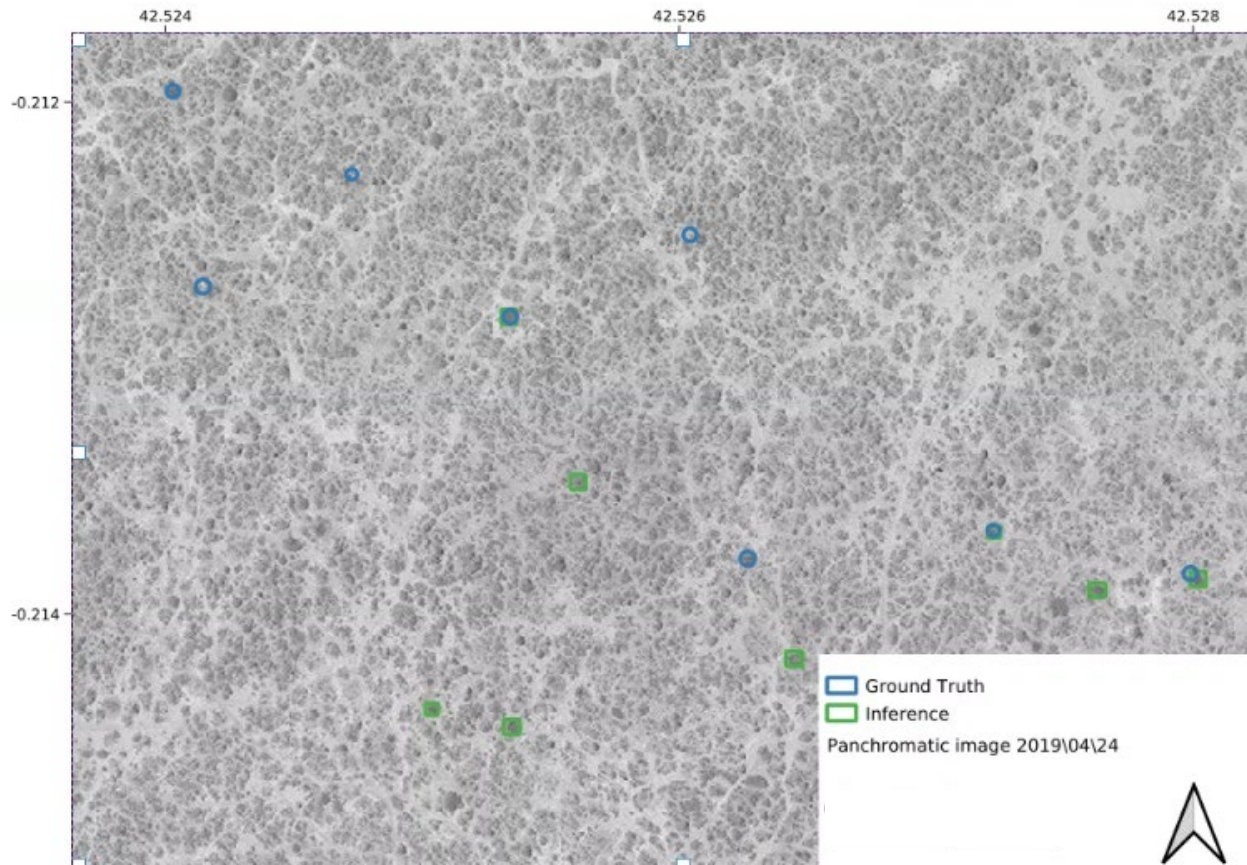
Inference on the whole data

- Inference with the 4 models mentioned
- Creation of a PostgreSQL DB
- Union of the results for each type of image
- Postprocess overlapping results
- Each geometry has:
 - Area
 - Diameter
 - Tile used for the inference
 - Model configuration
 - Confidence of the model



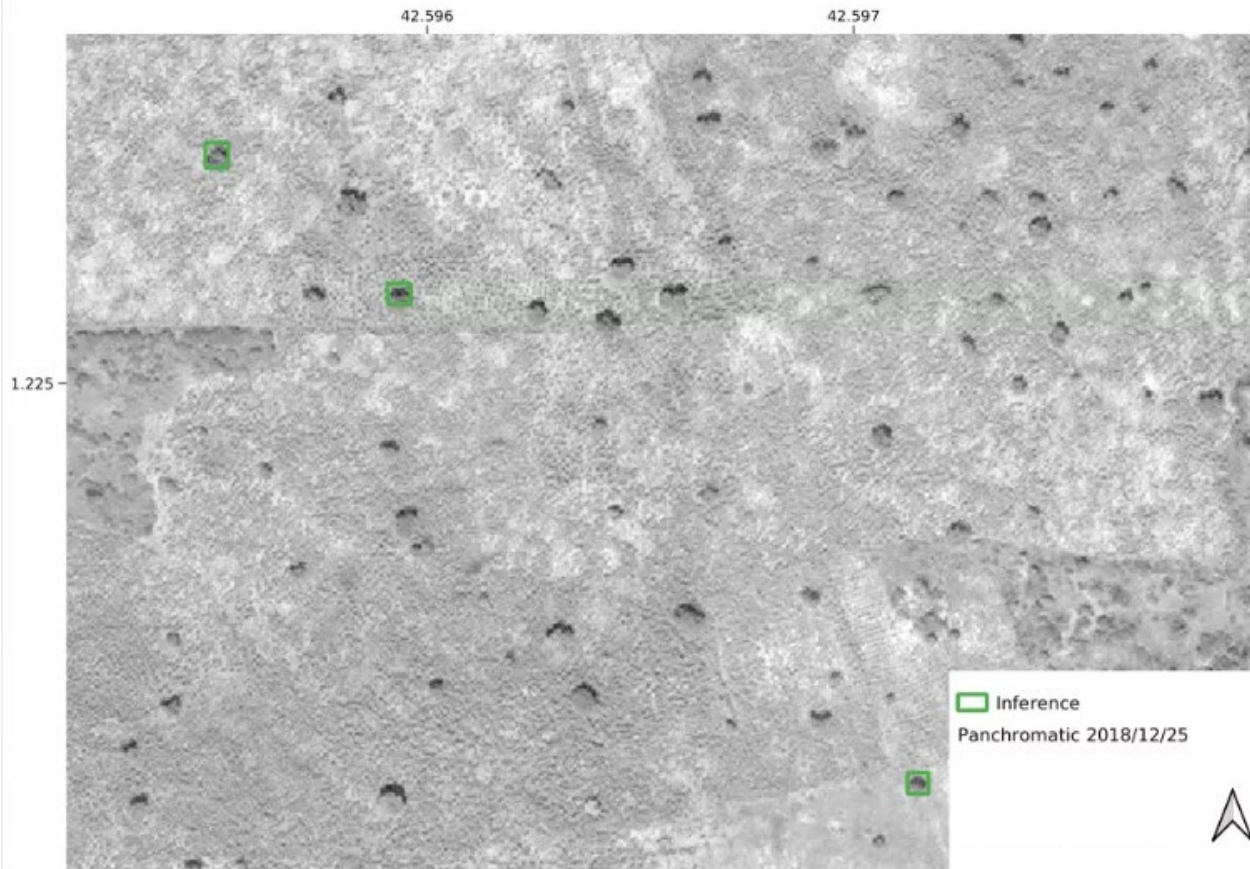
Results explanation

We fail on overpopulated tiles, especially in RGB



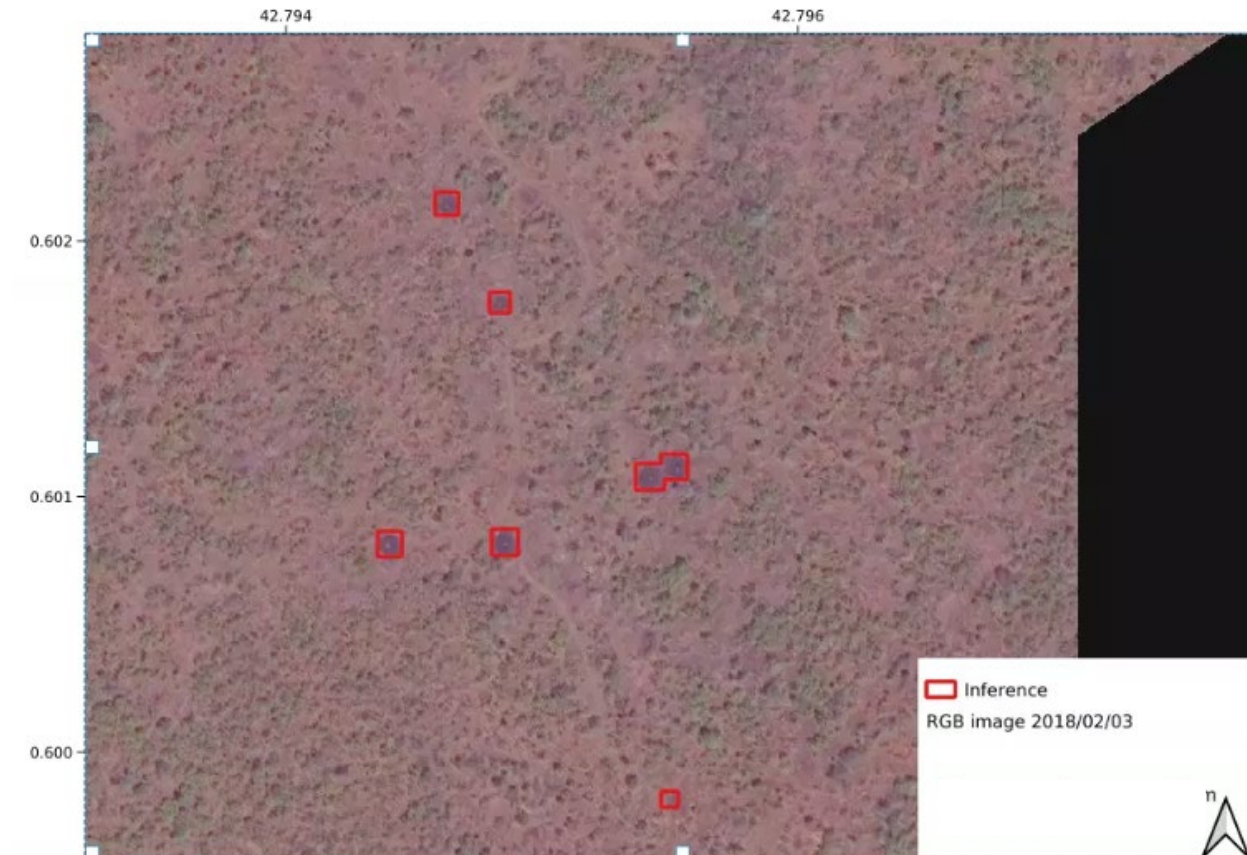
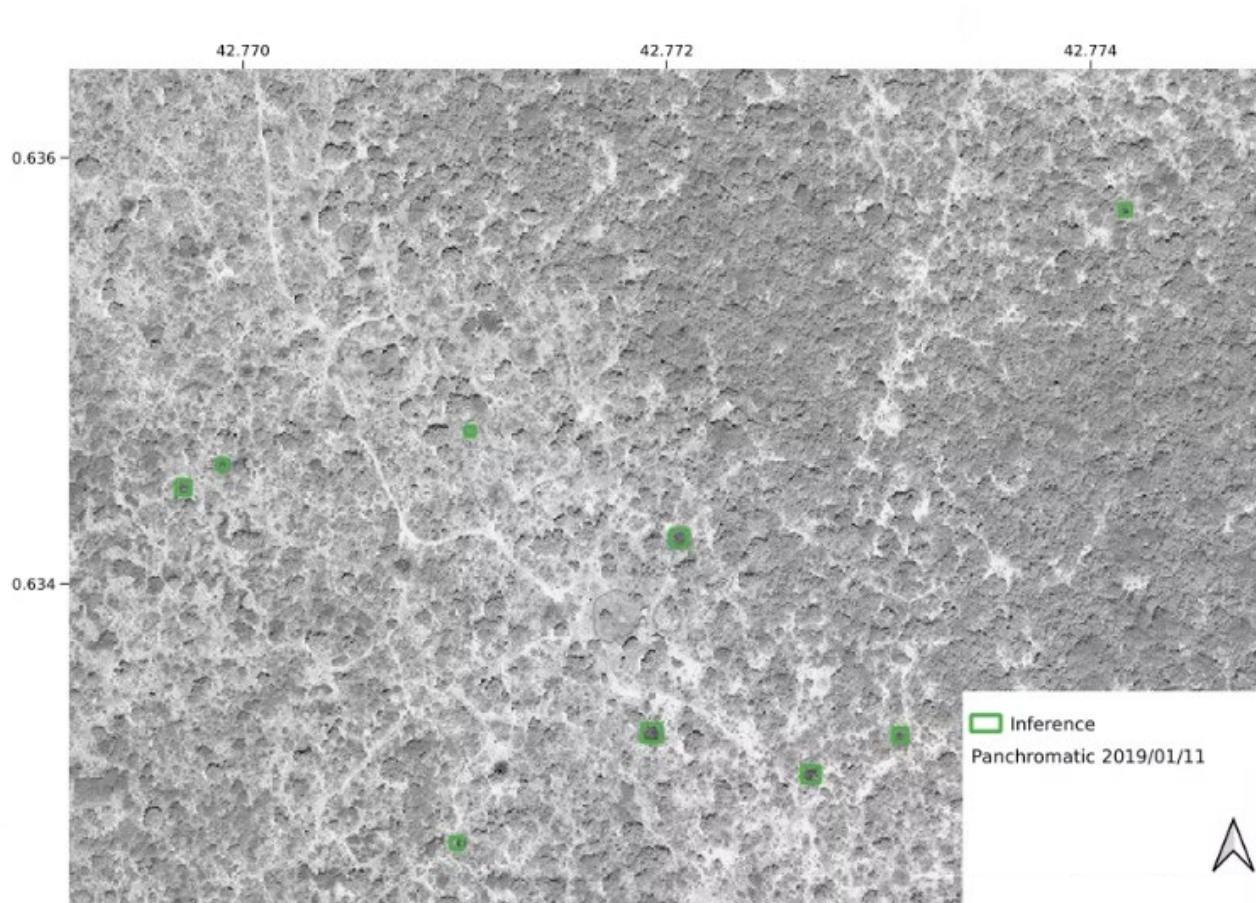
Results explanation

We have some miss classifications with trees



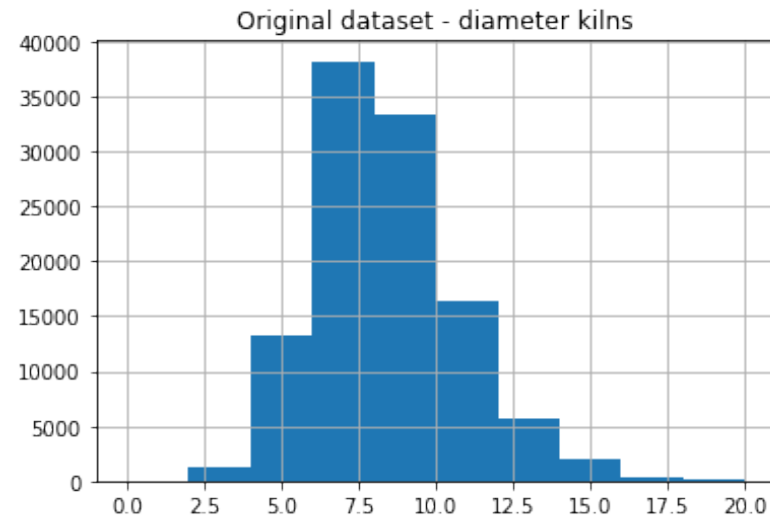
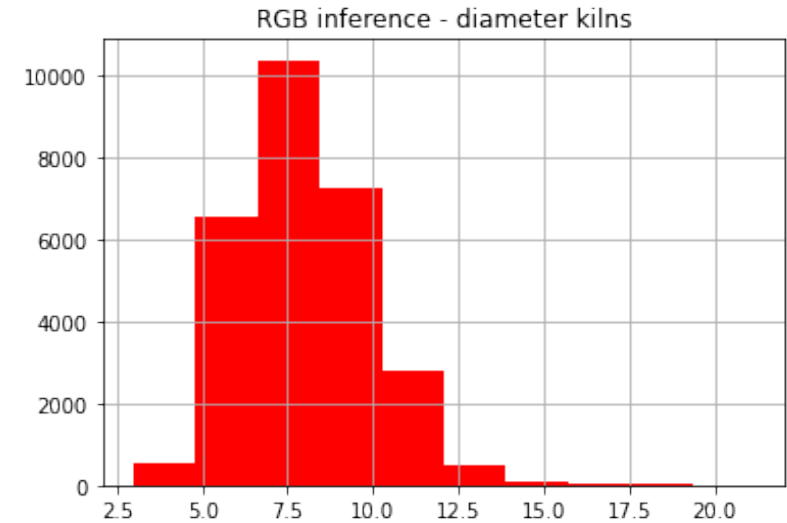
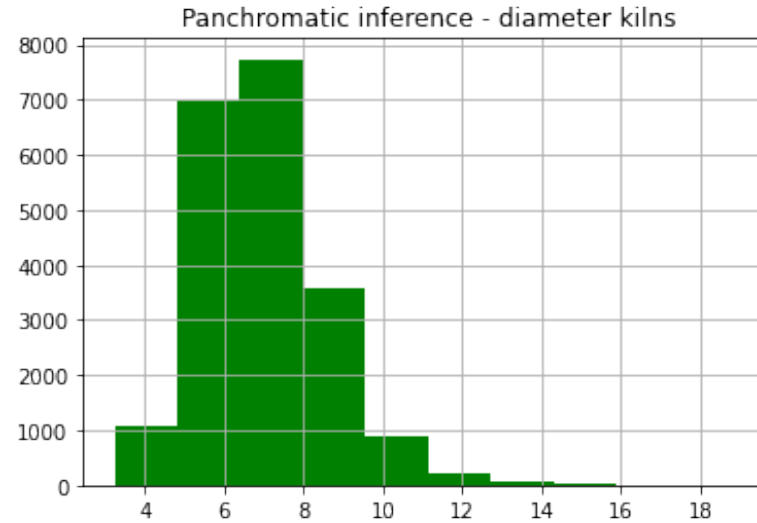
Results explanation

We detect kilns in areas that were not originally delineated



Results explanation

Generalize in size and different sensors



Conclusion and perspectives

- AI/CV approach can be used to automatically identify kilns related to illegal charcoal production on VHR images
- This two-stages training can overcome omission/misalignments problems on the original data
- For the final models we achieve F1-Scores:
 - Panchromatic: 0.91
 - RGB: 0.81
- We create an easy operational pipeline to reproduce this work
- Code will be available soon as several Jupyter-notebooks
- This code can be re-used for other object location and classification problems



Install detectron2 and other packages

```
import torch, torchvision
torch.__version__

!pip install funcy
!git clone https://github.com/akaraziewicz/cocospilt.git cocospilt

# You may need to restart your runtime prior to this, to let your installation take effect
# Some basic setup
# Setup detectron2 logger
import detectron2
from detectron2.utils.logger import setup_logger
setup_logger()

# Import some common libraries
import matplotlib.pyplot as plt
import numpy as np
import cv2

# Import some common detectron2 utilities
from detectron2.engine import DefaultPredictor
from detectron2.config import get_cfg
from detectron2.utils.visualizer import Visualizer
from detectron2.data import MetadataCatalog, DatasetCatalog
plt.rcParams['figure.figsize'] = [12, 8]
plt.rcParams['figure.dpi'] = 100 # 200, e.g. is really fine, but slower

RGB = True

if RGB == False:
    path_ann = '/eos/jedopp/data/projects/REFOCUS/data/swalm_cvat/pancro_first_iter/'
    path_imgs = '/scratch/pancro_first'
    img_src = '/eos/jedopp/data/projects/REFOCUS/data/swalm_v2/inputs/pancro/img_with_ann/'
else:
    path_ann = '/eos/jedopp/data/projects/REFOCUS/data/swalm_cvat/rgb_first_iter/'
    path_imgs = '/scratch/rgb_first'
    img_src = '/eos/jedopp/data/projects/REFOCUS/data/swalm_v2/inputs/RGB/img_with_ann/'
```

Get data and visualize

move the data to scratch before running train

```
mkdir /scratch/rgb_first
mkdir /eos/jedopp/data/projects/REFOCUS/data/swalm_v2/outputs

# -*- coding: utf-8 -*-
import json
import shutil
import os

with open(path_ann+'instances_default.json') as json_file:
    data = json.load(json_file)

data['categories'] = [{'id': 0, 'name': 'kiln', 'supercategory': ''}]
images = data['images'].copy()
for i in data['images']:
    print(i['file_name'])
    i['file_name'] = os.path.basename(i['file_name'])
    src = '{}{}'.format(img_src, i['file_name'])
    i['file_name'] = os.path.basename(src)
```

Thank you

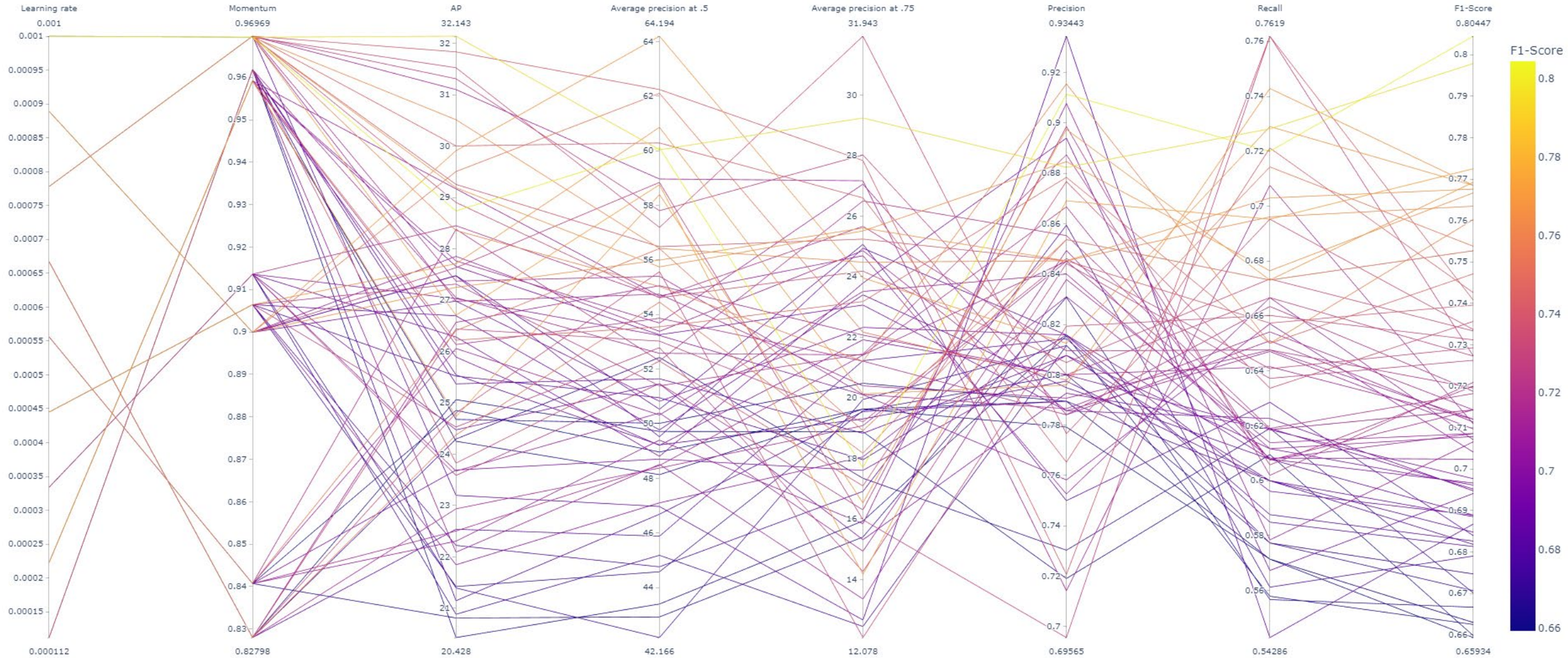
Get in touch!

Laura.Martinez-Sanchez@ec.europa.eu

This presentation has been prepared for internal purposes. The information and views expressed in it do not necessarily reflect an official position of the European Commission or of the European Union.

Except otherwise noted, © European Union (year). All Rights Reserved

Hyperparameter tuning



Hyperparameter tuning

