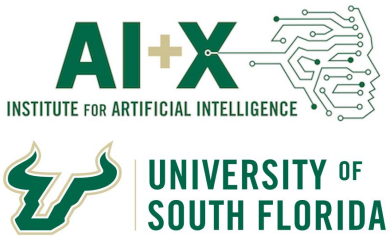
Measuring Airport Activity from Sentinel-2 Imagery to Support Decision-Making during COVID-19 Pandemic

Rodrigo Minetto, Maurício Pamplona Segundo, Sudeep Sarkar

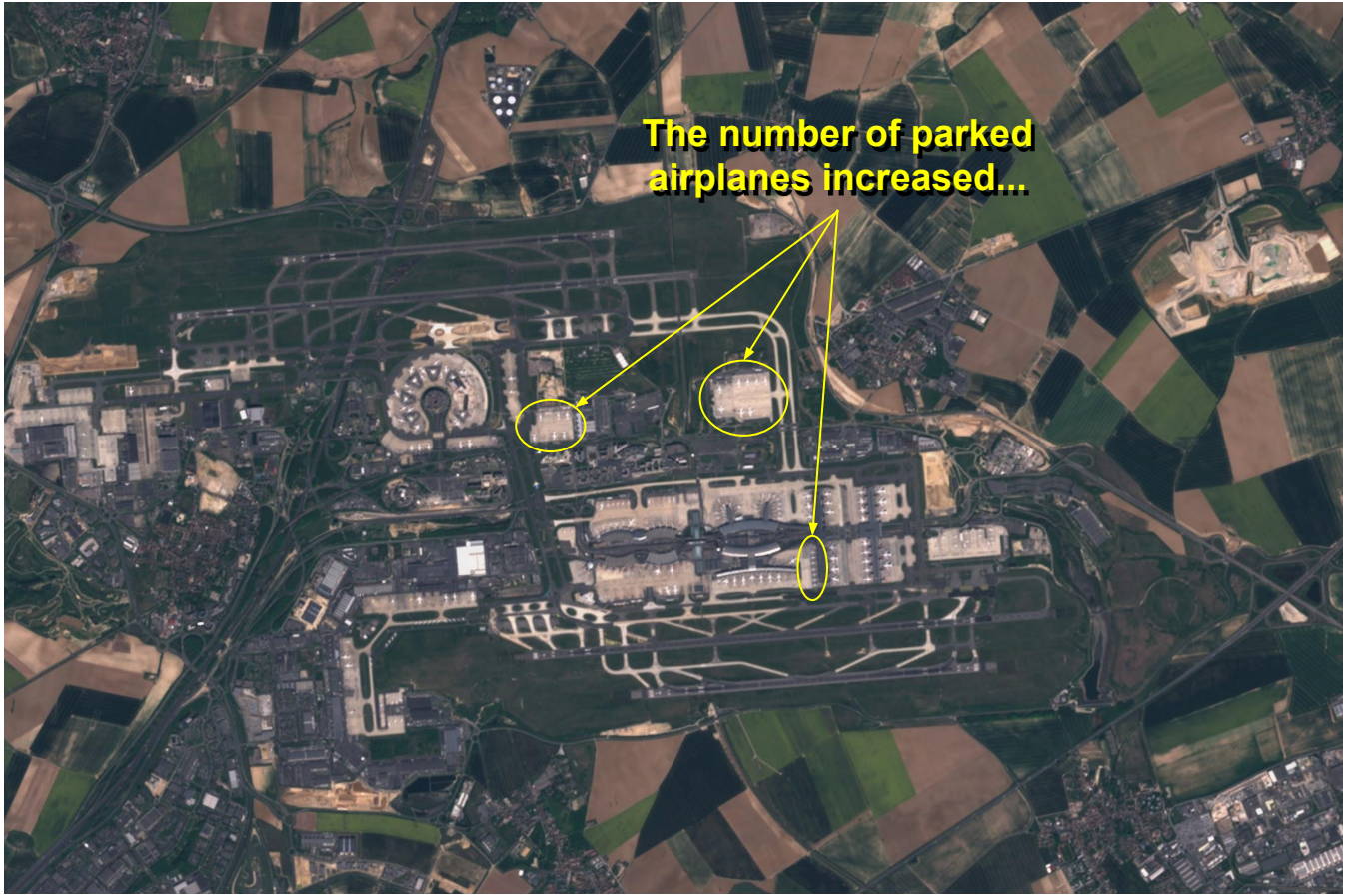


Goal

- Count the number of airplanes in an airport over time
  - COVID-19 considerably reduced the number of commercial flights
  - Indicator of economic and human activity
- Analyse the temporal signal and identify anomalies
  - Unexpected changes can be reported to the authorities
- Define standard behavior
  - Indicator of economic recovery

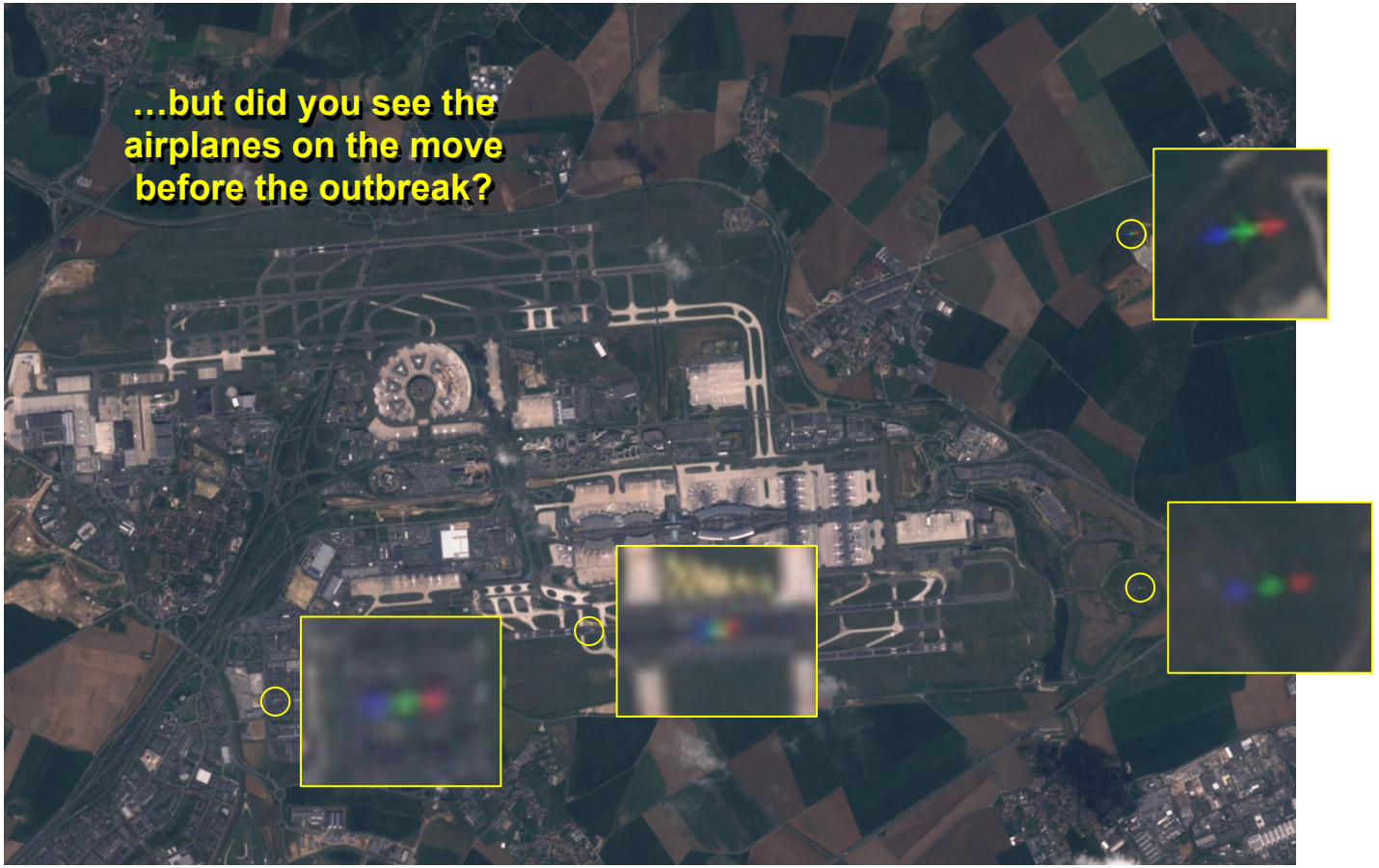
*Contains modified Sentinel-2 data processed by Euro Data Cube

Charles de Gaulle Airport before the COVID-19 outbreak (11 Apr 2019)



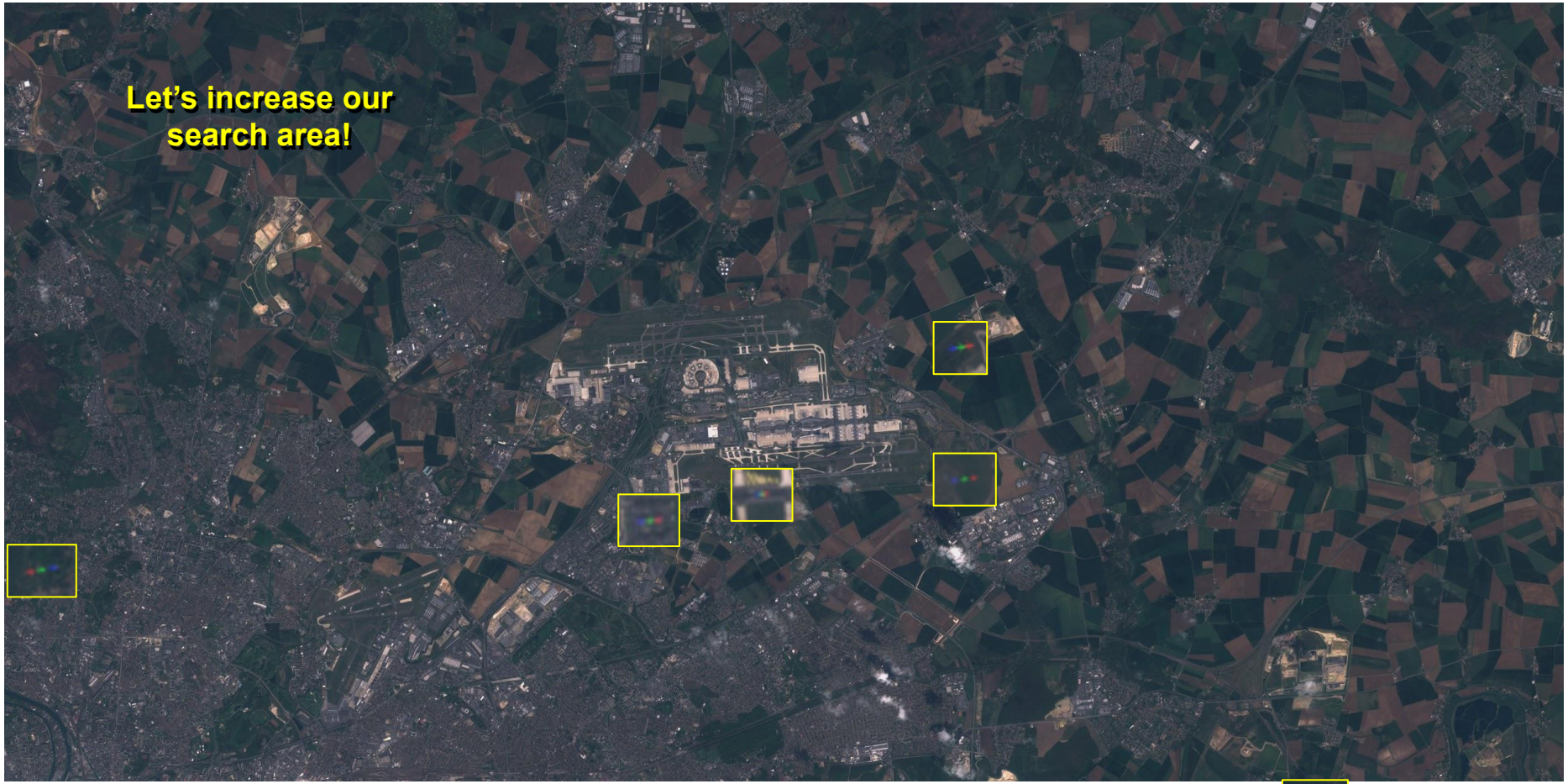
*Contains modified Sentinel-2 data processed by Euro Data Cube

Charles de Gaulle Airport after the COVID-19 outbreak (15 Apr 2020)



*Contains modified Sentinel-2 data processed by Euro Data Cube

Charles de Gaulle Airport before the COVID-19 outbreak (11 Apr 2019)



*Contains modified Sentinel-2 data processed by Euro Data Cube

Before (11 Apr 2019)
(6 flying airplanes)

*Contains modified Sentinel-2 data processed by Euro Data Cube

After (15 Apr 2020)
(0 flying airplanes)

Goal

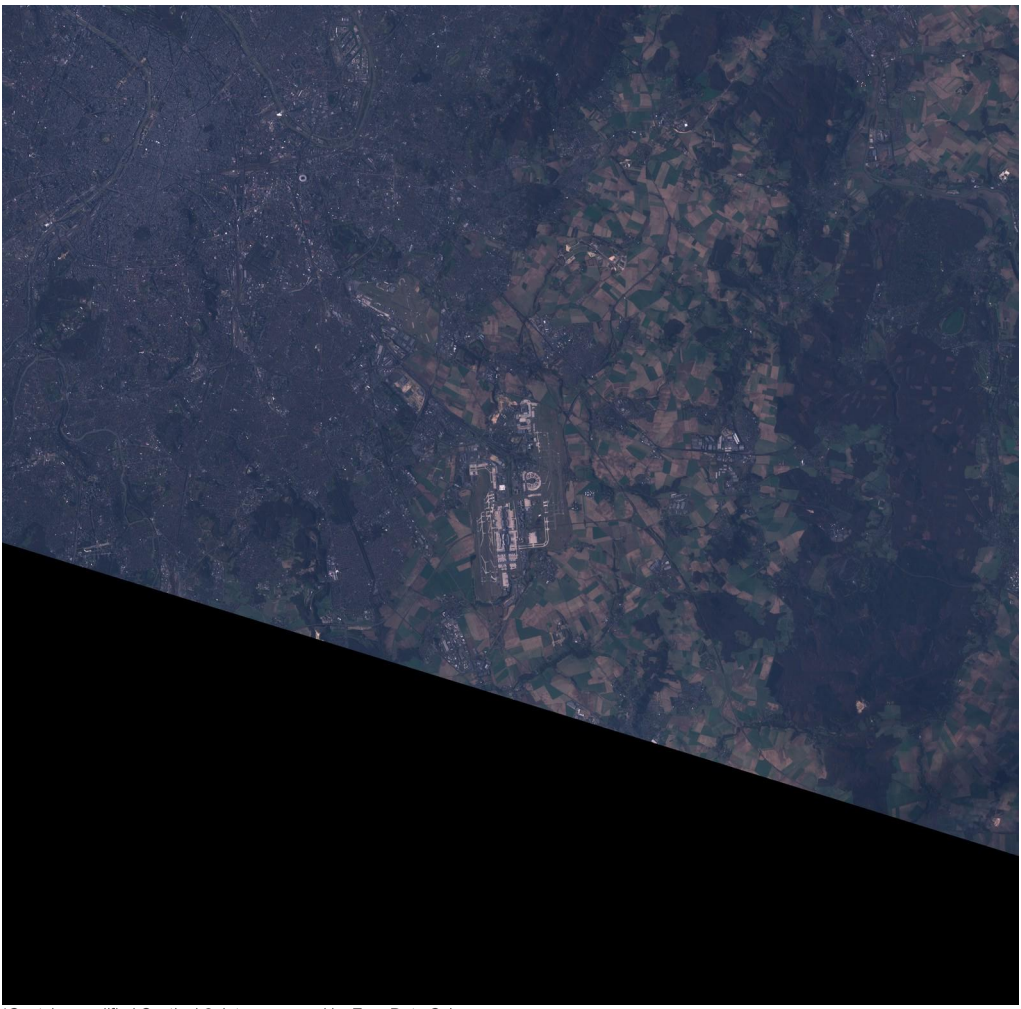
- Count the number of parked/flying airplanes in an airport over time
  - COVID-19 considerably reduced the number of commercial flights
  - Indicator of economic and human activity
  - Number of parked airplanes is expected to grow
  - Number of flying airplanes is expected to drop
- Analyse the temporal signal and identify anomalies
  - Unexpected changes can be reported to the authorities
- Define standard behavior
  - Indicator of economic recovery

Satellite image acquisition

```
$ python3 download.py
```

```
(saves downloaded images in the folder ‘./images/’ with name YYYY-MM-DD-hh-mm-ss.png)
```

- Use sentinel-hub and eo-learn
- Download Sentinel-2 images of an extended area around the Charles de Gaulle Airport from 01/01/2018 to 04/28/2020 (~2.3 years)
- Automatically discard images covered by clouds
- Can be easily adapted to other locations or periods of time



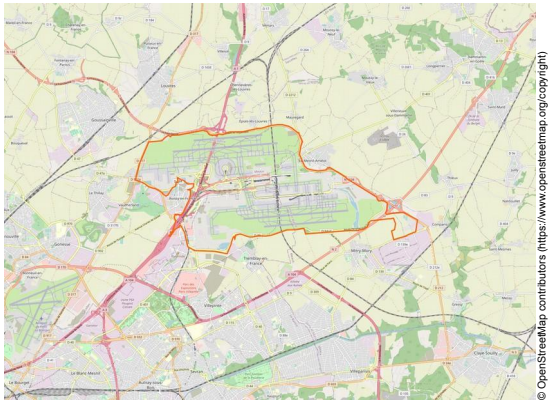
*Contains modified Sentinel-2 data processed by Euro Data Cube

Region of interest (ROI) for parked airplanes

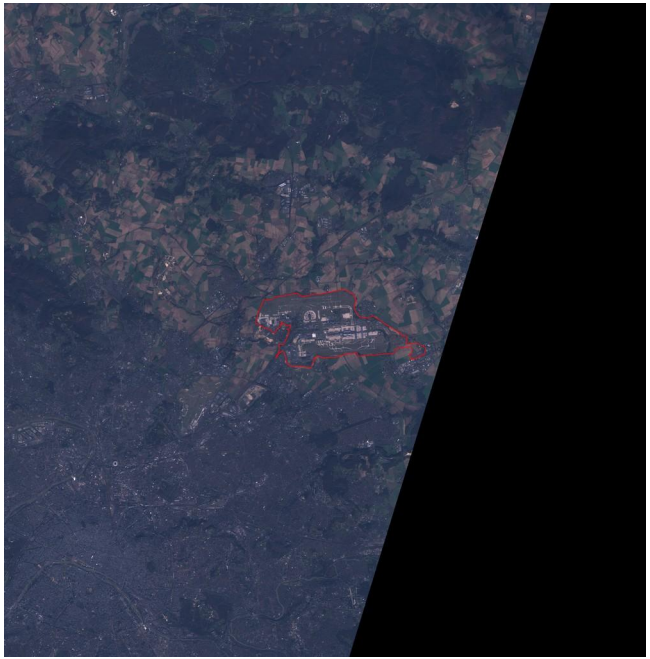
```
$ python3 roi.py
```

```
(saves ROI images in the folder ‘./images/’ with name YYYY-MM-DD-hh-mm-ss-roi.png)
```

- Retrieve airport boundaries from OpenStreetMap
- Use the smallest enclosing rectangle to crop downloaded images



Downloaded image



*Contains modified Sentinel-2 data processed by Euro Data Cube

ROI

*Contains modified Sentinel-2 data processed by Euro Data Cube

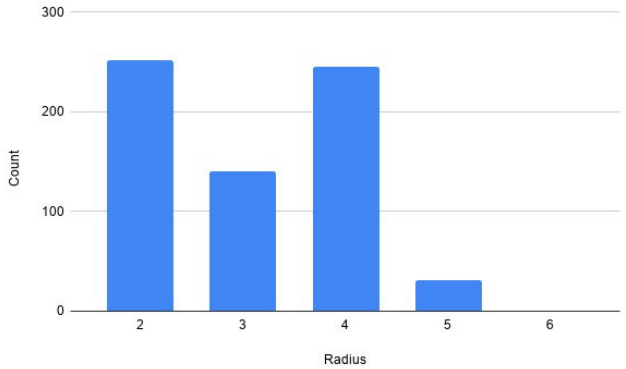
Training a detector for parked airplanes

- Two class problem: background and parked airplane
- Annotated five ROI images from 01/01/2018 to 04/30/2018
  - Skipped snow covered images
  - Minimum enclosing circle for each airplane
  - Total of 668 parked airplanes annotated

*Contains modified Sentinel-2 data processed by Euro Data Cube

Architecture

- Single-scale Fully Convolutional Network
  - Train for patches, run for large images
- Fixed detection window size of 17x17 pixels
  - Maximum airplane size (diameter of 12 pixels) plus an extra margin



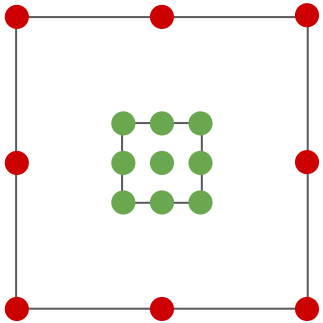
Architecture

Layer	# filters	Kernel size	Stride	Tensor size
Input				17x17x3
Conv/Relu/BN	64	3x3	1x1	15x15x64
MaxPool		3x3	1x1	13x13x64
Conv/Relu/BN	128	3x3	1x1	11x11x128
MaxPool		3x3	1x1	9x9x128
Conv/Relu/BN	256	3x3	1x1	7x7x256
MaxPool		3x3	1x1	5x5x256
Conv/Sigmoid	1	5x5	1x1	1x1x1

*** 20% dropout after every max-pooling

Training strategy

- First stage of training
  - Crop 17x17 samples following the pattern below
  - Step size of 1 pixel for positive samples and annotation radius for negative ones



  - Add random crops to the negative set (max 1:2 size ratio for positive:negative sets)
  - Train for 3000 iterations (bs: 256, Adam, lr: 0.0001)
    - Random flips (horizontal and vertical) and 90-degree rotations as data augmentation

Training strategy

- Following stages of training
  - Update negative set
    - Use current model weights for inference
    - Binarize output image with a 0.5 threshold
    - Find blobs and compute center (avg of pixel coordinates) and size (number of pixels)
    - Apply NMS using size of blobs as score and distance between their centers as overlap (at least 4 pixels apart to be considered different detections)
    - Use ground truth to find false positives (at least 8 pixels apart from all annotations)
    - Replace up to half of the negative samples with newly discovered false positives
  - Train for 3000 iterations (bs: 256, Adam, lr: 0.0001)
    - Random flips (horizontal and vertical) and 90-degree rotations as data augmentation

Training result

```
$ python3 train_parked.py
```

(saves model as ‘./models/parked.pytorch’)

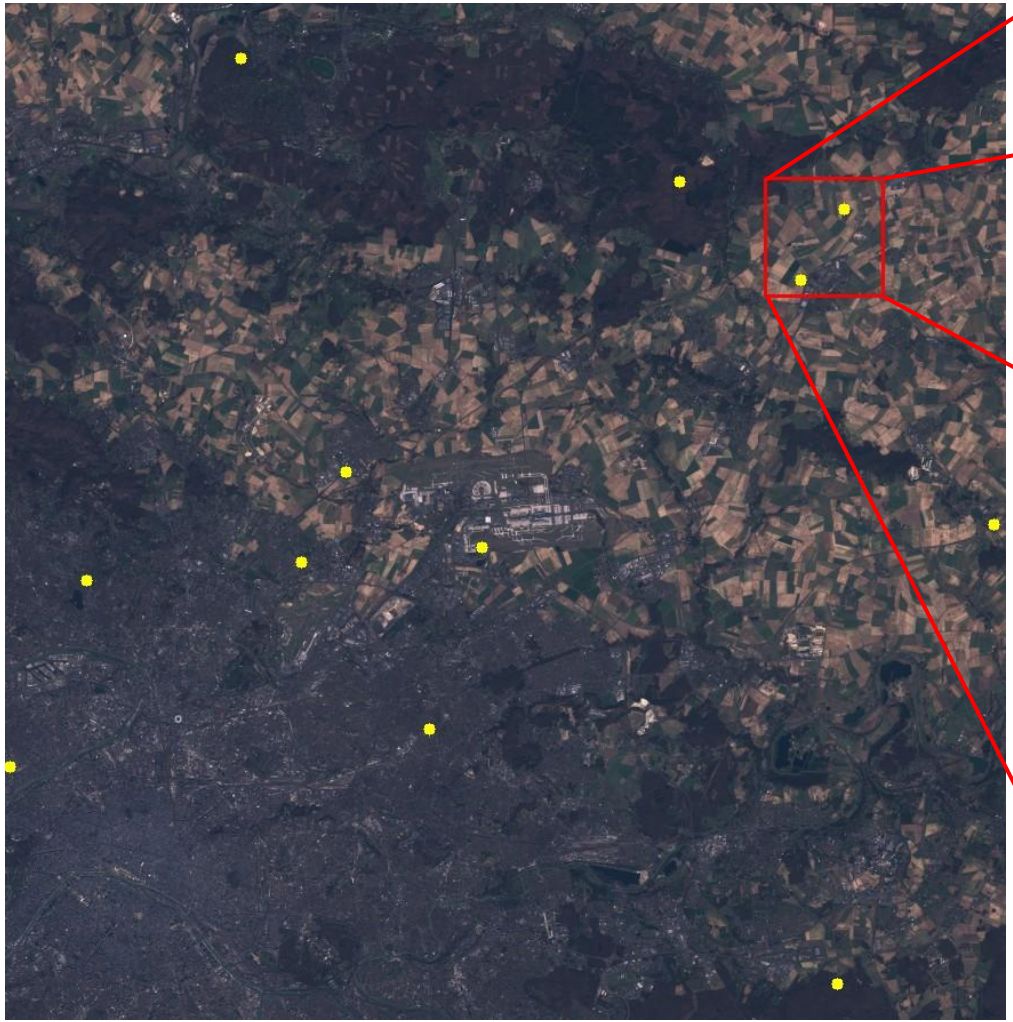
Stage	Detection rate	False detection rate
#1	91.01%	91.07%
#2	89.22%	57.79%
#3	90.27%	17.96%
#4	90.87%	17.53%
#5	89.97%	14.27%
#6	**91.17%**	**9.51%**

*Contains modified Sentinel-2 data processed by Euro Data Cube

Training a detector for **flying** airplanes

- Two class problem: background and flying airplane
- Annotated **18 full** images from 01/01/2018 to **06/30/2018**
  - Skipped snow covered images
  - Minimum enclosing circle for each airplane
  - Total of **190** flying airplanes annotated

***** differences between parked and flying are highlighted in orange *****

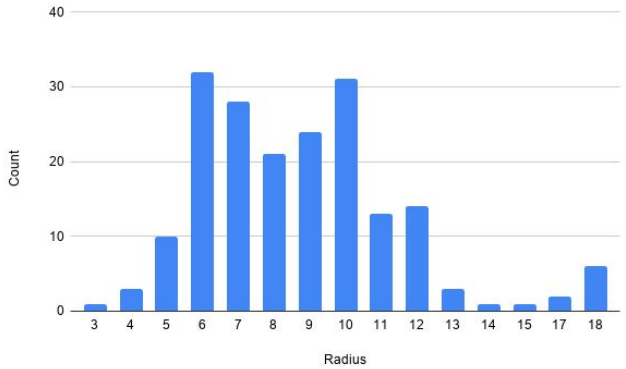


*Contains modified Sentinel-2 data processed by Euro Data Cube

*Contains modified Sentinel-2 data processed by Euro Data Cube

Architecture

- Single-scale Fully Convolutional Network
  - Train for patches, run for large images
- Fixed detection window size of **51x51** pixels
  - Maximum airplane size (diameter of **36** pixels) plus an extra margin



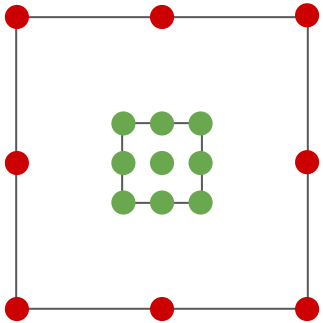
Architecture

	Layer	# filters	Kernel size	Stride	Tensor size
	Input				51x51x3
	Conv/Relu/BN	16	5x5	1x1	47x47x16
	MaxPool		5x5	1x1	43x43x16
	Conv/Relu/BN	32	5x5	1x1	39x39x32
	MaxPool		5x5	1x1	35x35x32
(3x)	Conv/Relu/BN	64	5x5	1x1	(31x31|23x23|15x15)x64
	MaxPool		5x5	1x1	(27x27|19x19|11x11)x64
	Conv/Sigmoid	1	11x11	1x1	1x1x1

*** 20% dropout after every max-pooling

Training strategy

- First stage of training
  - Crop **51x51** samples following the pattern below
  - Step size of **3** pixel for positive samples and annotation **25** for negative ones



  - Add random crops to the negative set (max 1:2 size ratio for positive:negative sets)
  - Train for **2000** iterations (bs: 256, Adam, lr: 0.0001)
    - Random flips (horizontal and vertical) and 90-degree rotations as data augmentation

Training strategy

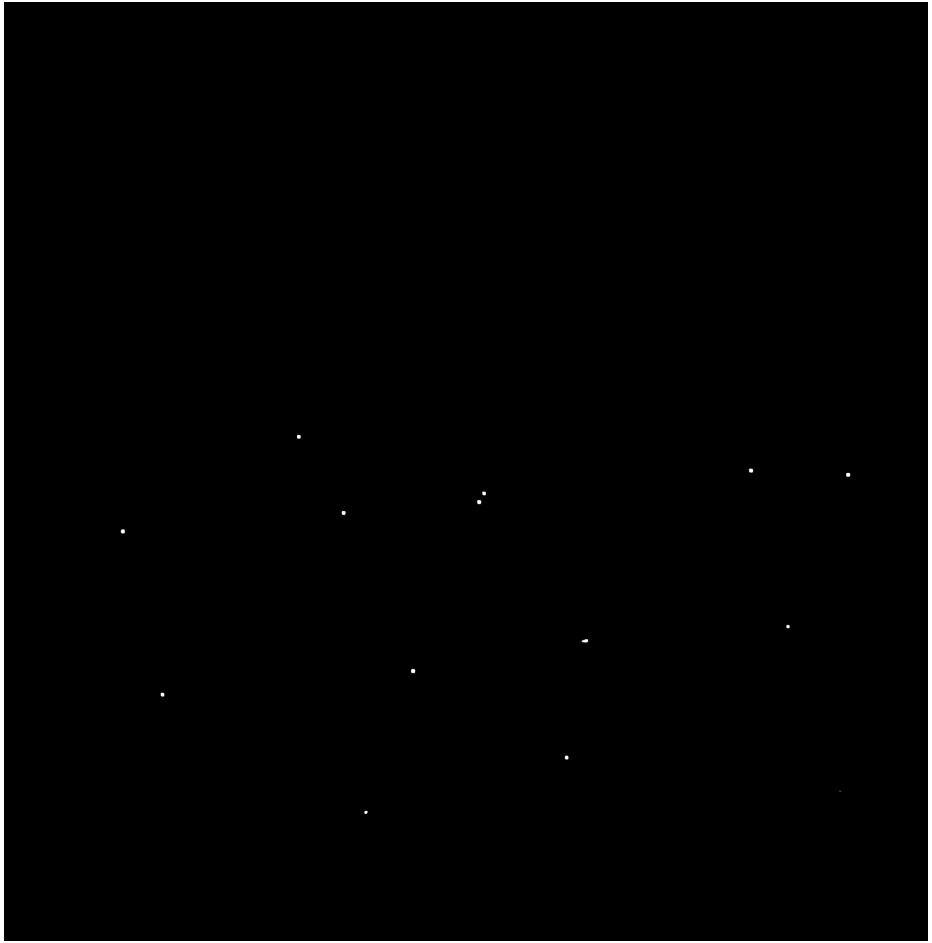
- Following stages of training
  - Update negative set
    - Use current model weights for inference
    - Binarize output image with a 0.5 threshold
    - Find blobs and compute center (avg of pixel coordinates) and size (number of pixels)
    - Apply NMS using size of blobs as score and distance between their centers as overlap (at least **25** pixels apart to be considered different detections)
    - Use ground truth to find false positives (at least **25** pixels apart from all annotations)
    - Replace up to half of the negative samples with newly discovered false positives
  - Train for **2000** iterations (bs: 256, Adam, lr: 0.0001)
    - Random flips (horizontal and vertical) and 90-degree rotations as data augmentation

Training result

```
$ python3 train_flying.py
```

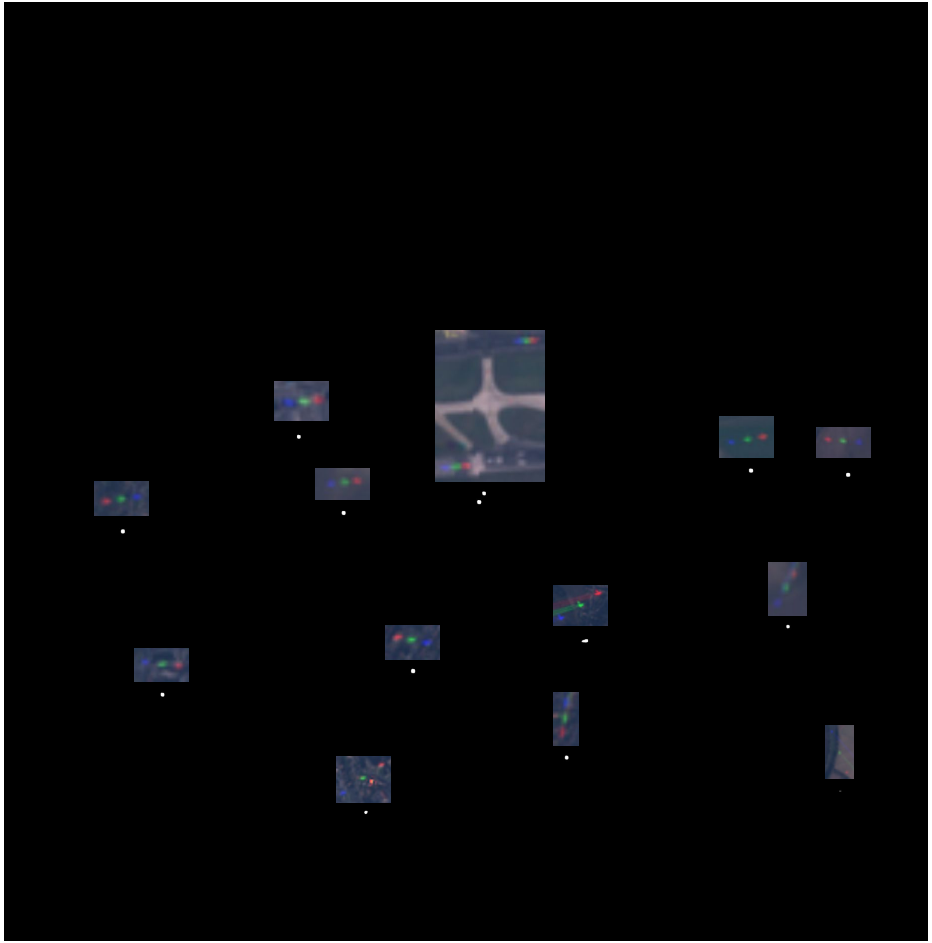
(saves model as ‘./models/flying.pytorch’)

Stage	Detection rate	False detection rate
#1	99.47%	92.81%
#2	99.47%	53.45%
#3	99.47%	2.07%
#4	99.47%	3.08%
#5	**99.47%**	**1.05%**



*Contains modified Sentinel-2 data processed by Euro Data Cube

*Contains modified Sentinel-2 data processed by Euro Data Cube



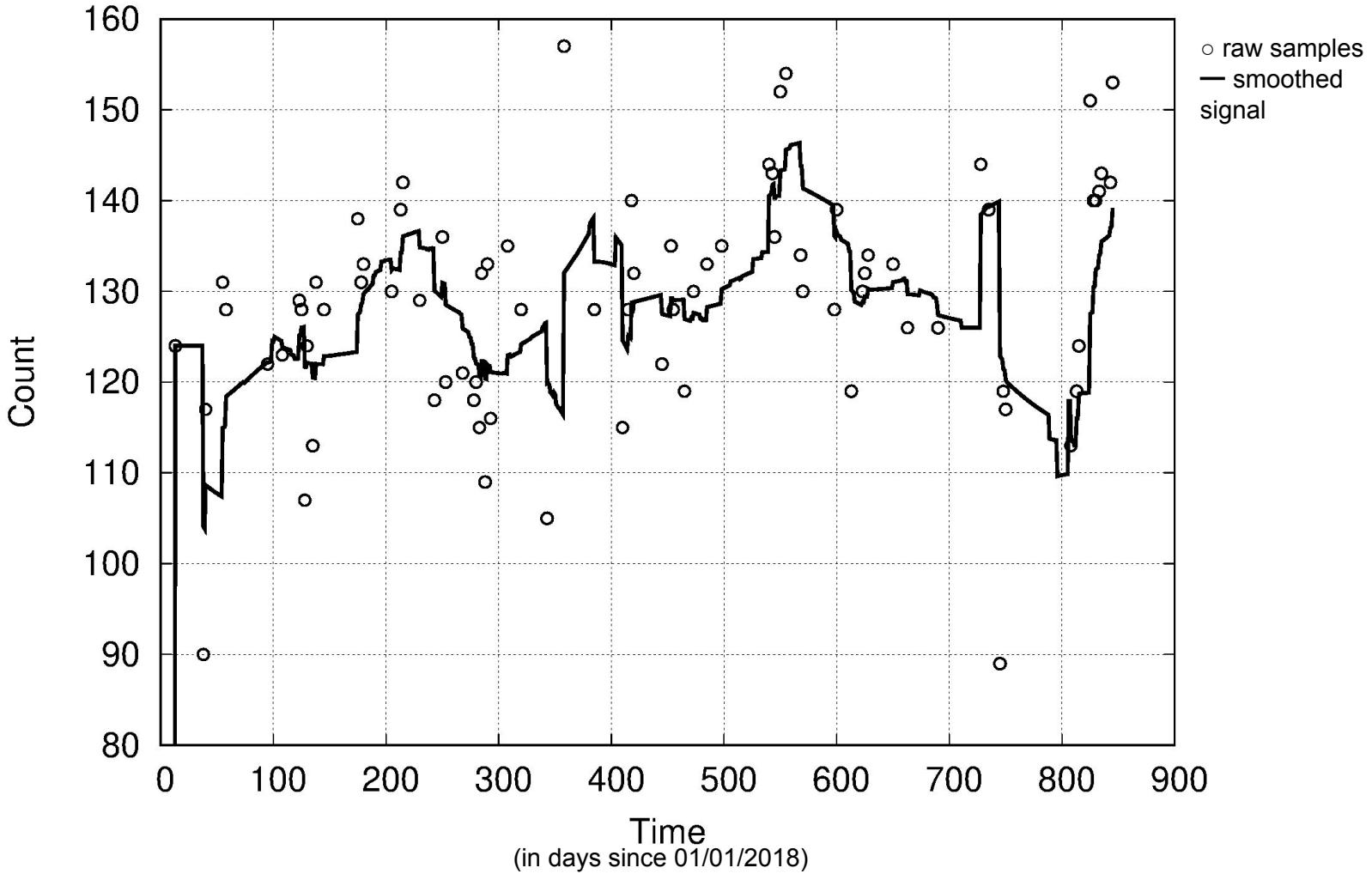
*Contains modified Sentinel-2 data processed by Euro Data Cube

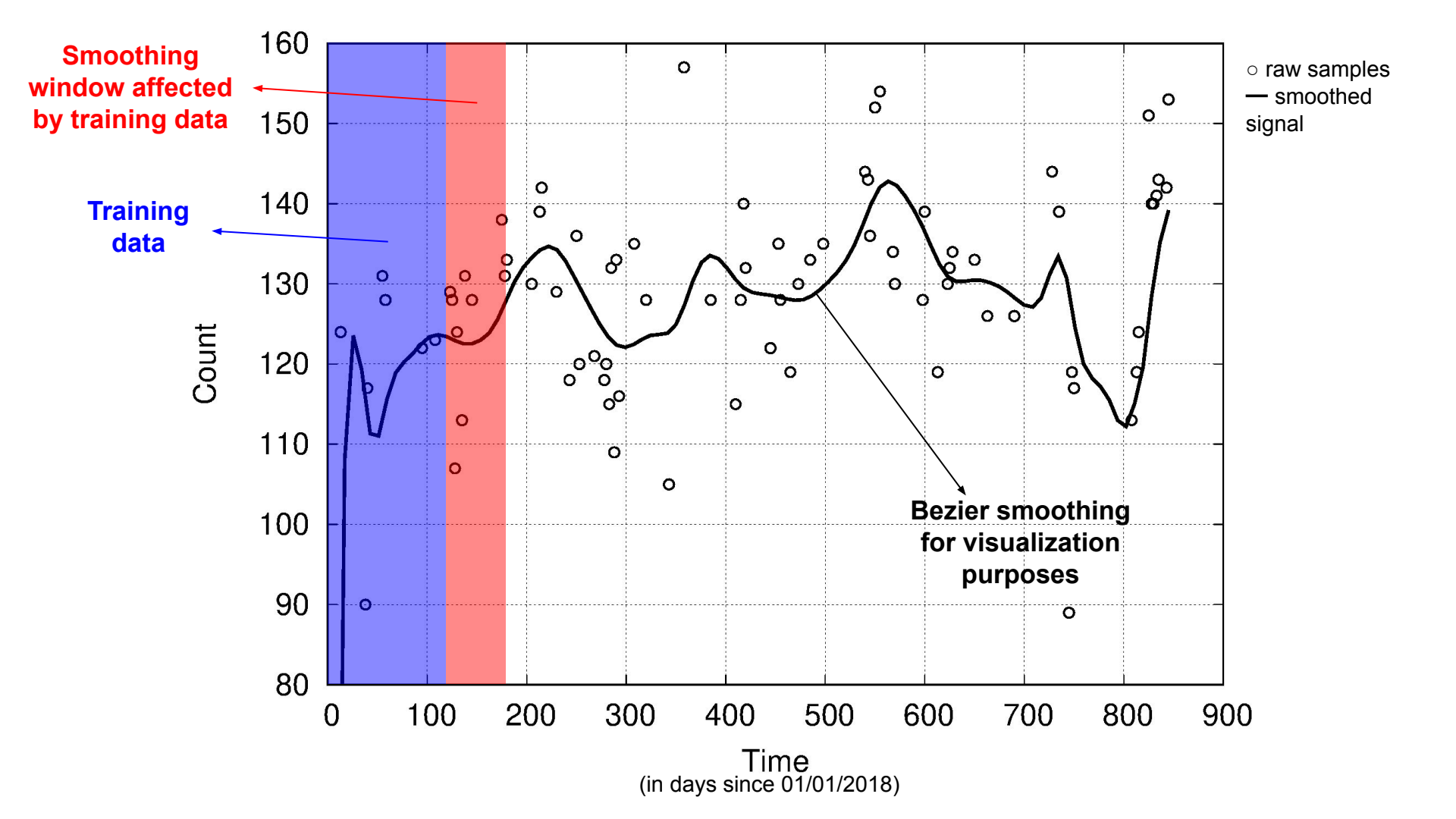
Temporal analysis of parked airplanes

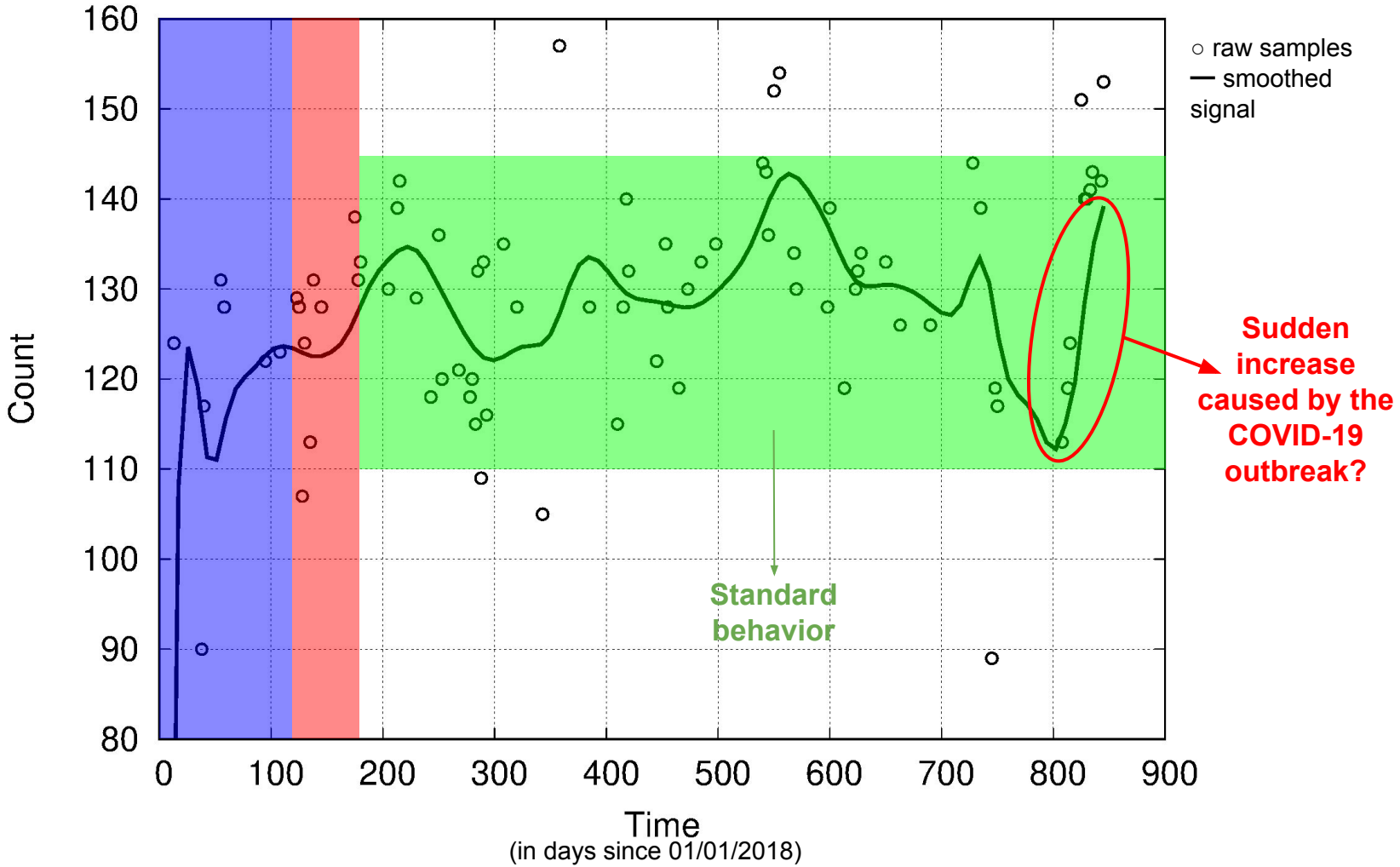
```
$ python3 test_parked.py
```

```
(saves temporal sequence as ‘./parked.csv’)
```

- Count airplanes in satellite images since 01/01/2018
- Temporal smoothing using a 60-days sliding window
  - Window ends in the present day (only sees the past)
  - Gaussian weighting (σ = 30 days)


160
150
140
130
120
110
100
90
80
Count
0
100
200
300
400
500
600
700
800
900
Time
(in days since 01/01/2018)
○ raw samples
— smoothed signal


Smoothing window affected by training data
Training data
Count
160
150
140
130
120
110
100
90
80
0
100
200
300
400
500
600
700
800
900
Time
(in days since 01/01/2018)
Bezier smoothing for visualization purposes
○ raw samples
— smoothed signal


160
150
140
130
120
110
100
90
80
Count
0
100
200
300
400
500
600
700
800
900
Time
(in days since 01/01/2018)
○ raw samples
— smoothed signal
Standard behavior
Sudden increase caused by the COVID-19 outbreak?

Temporal analysis of parked airplanes

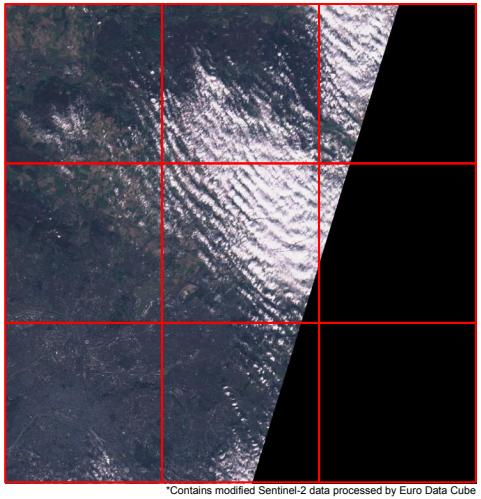
- Counting is not very accurate
  - Parked airplanes are too small
  - White regions (snow, small clouds, etc) affect the result
- In terms of quantity, the peak caused by the COVID-19 outbreak is not outside the standard behavior zone
- In terms of oscillation, a sudden increase with large magnitude can be seen as an unexpected behavior

Temporal analysis of flying airplanes

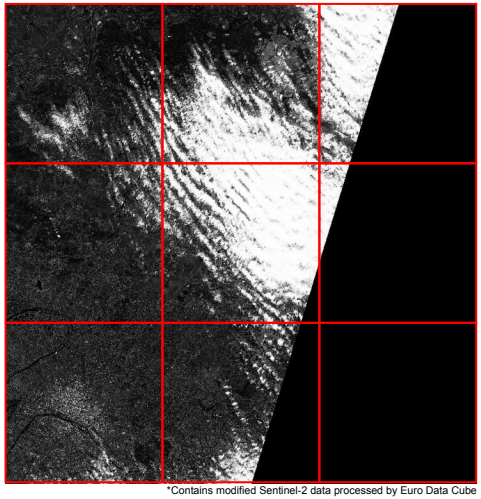
```
$ python3 test_flying.py
```

```
(saves temporal sequence as './flying.csv')
```

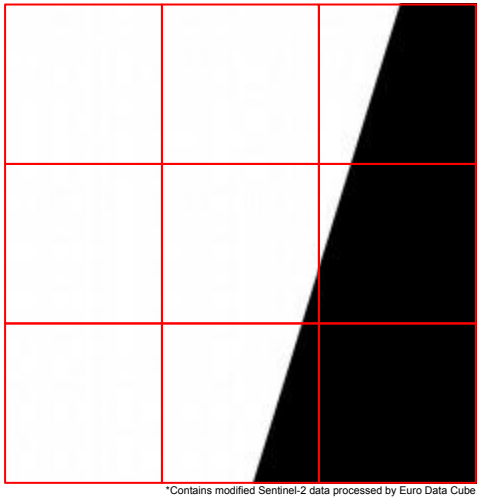
- Count airplanes in satellite images since 01/01/2018
- Temporal smoothing using a 60-days sliding window
  - Window ends in the present day (only sees the past)
  - Gaussian weighting (σ = 30 days)
- Divide image in nine blocks
  - Discard blocks covered by clouds, but maintain as much data as possible
  - Run temporal analysis for each block separately
  - Output signal is the sum of all smoothed block signals



Input image

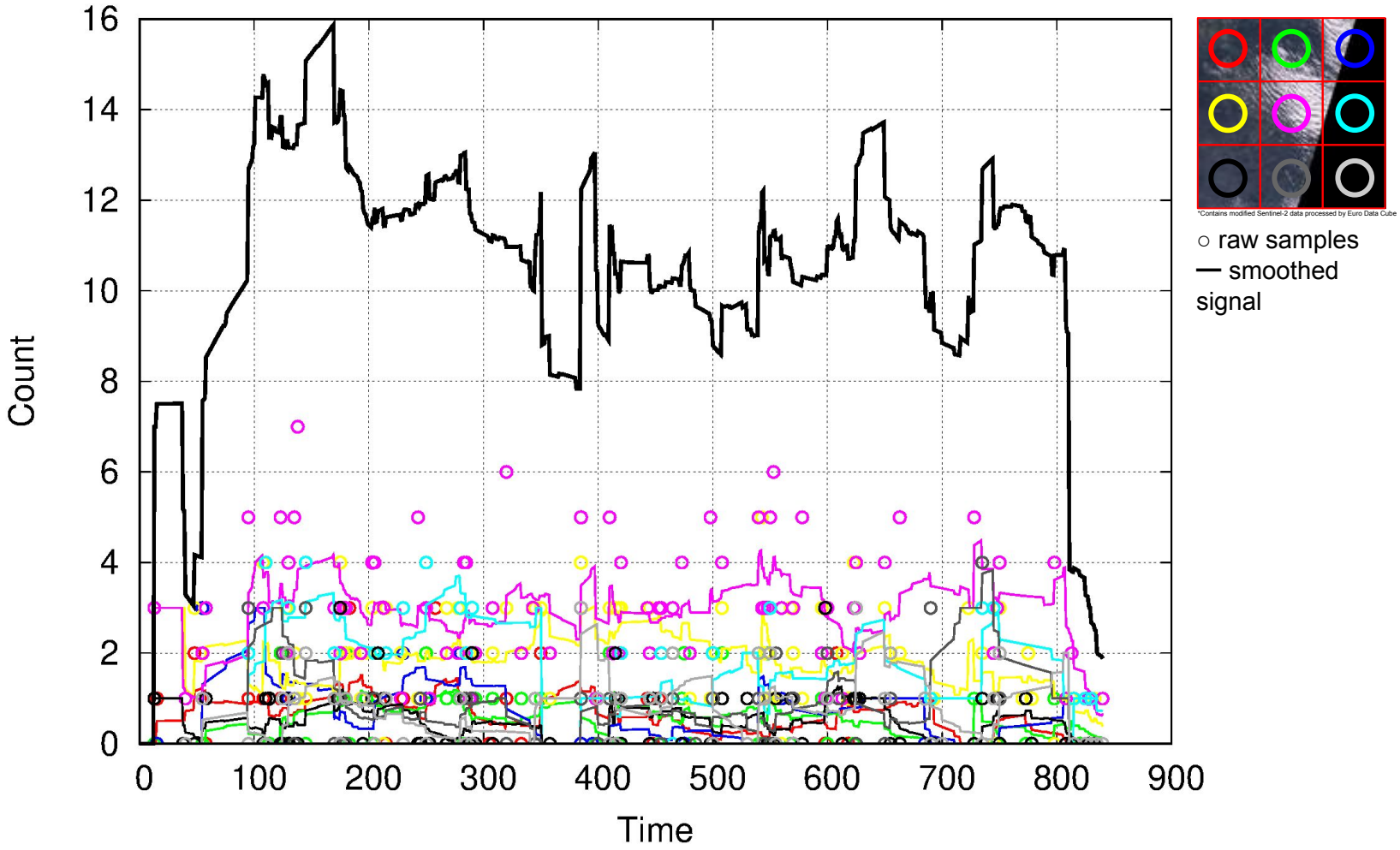


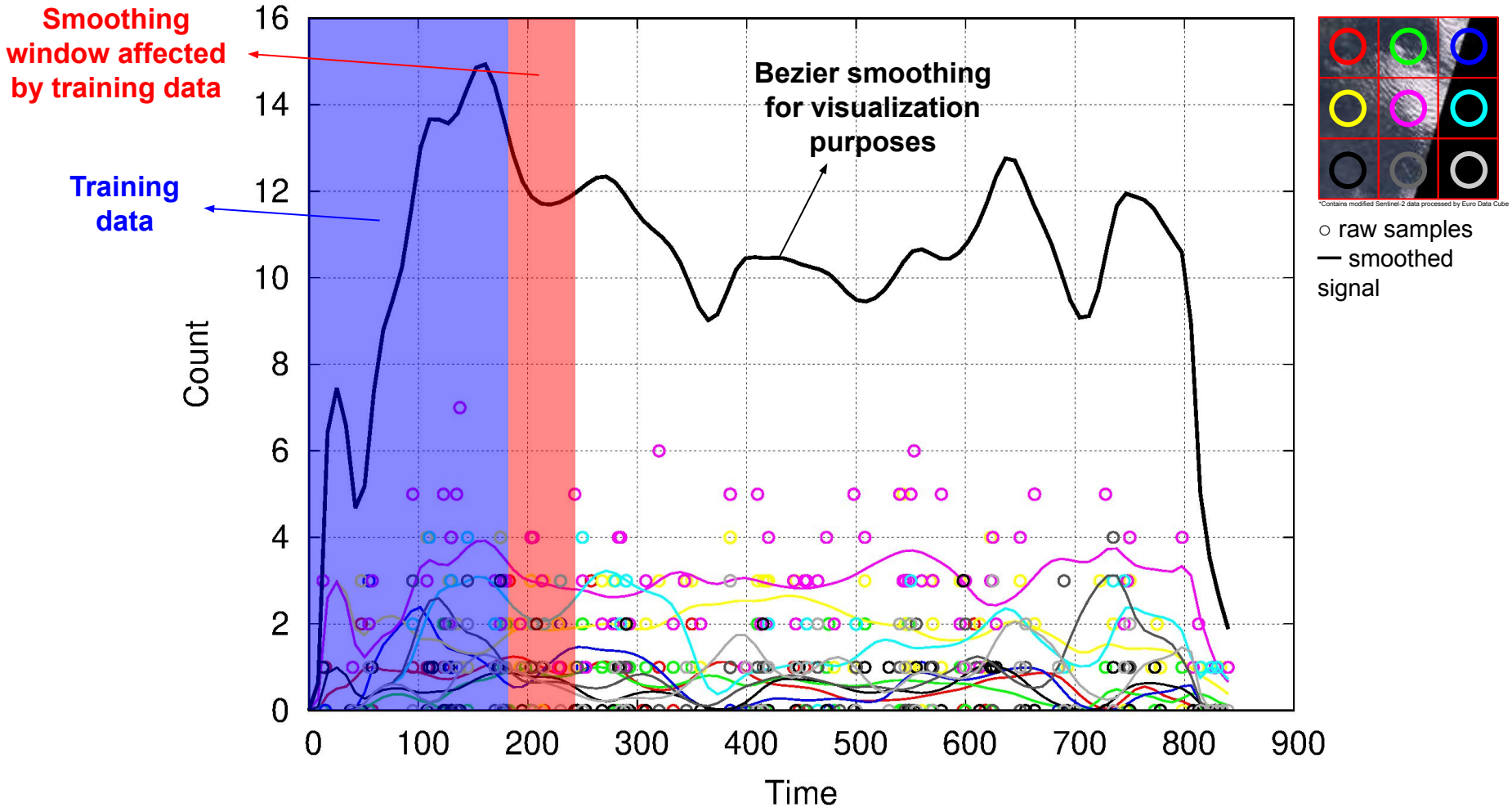
Cloud mask

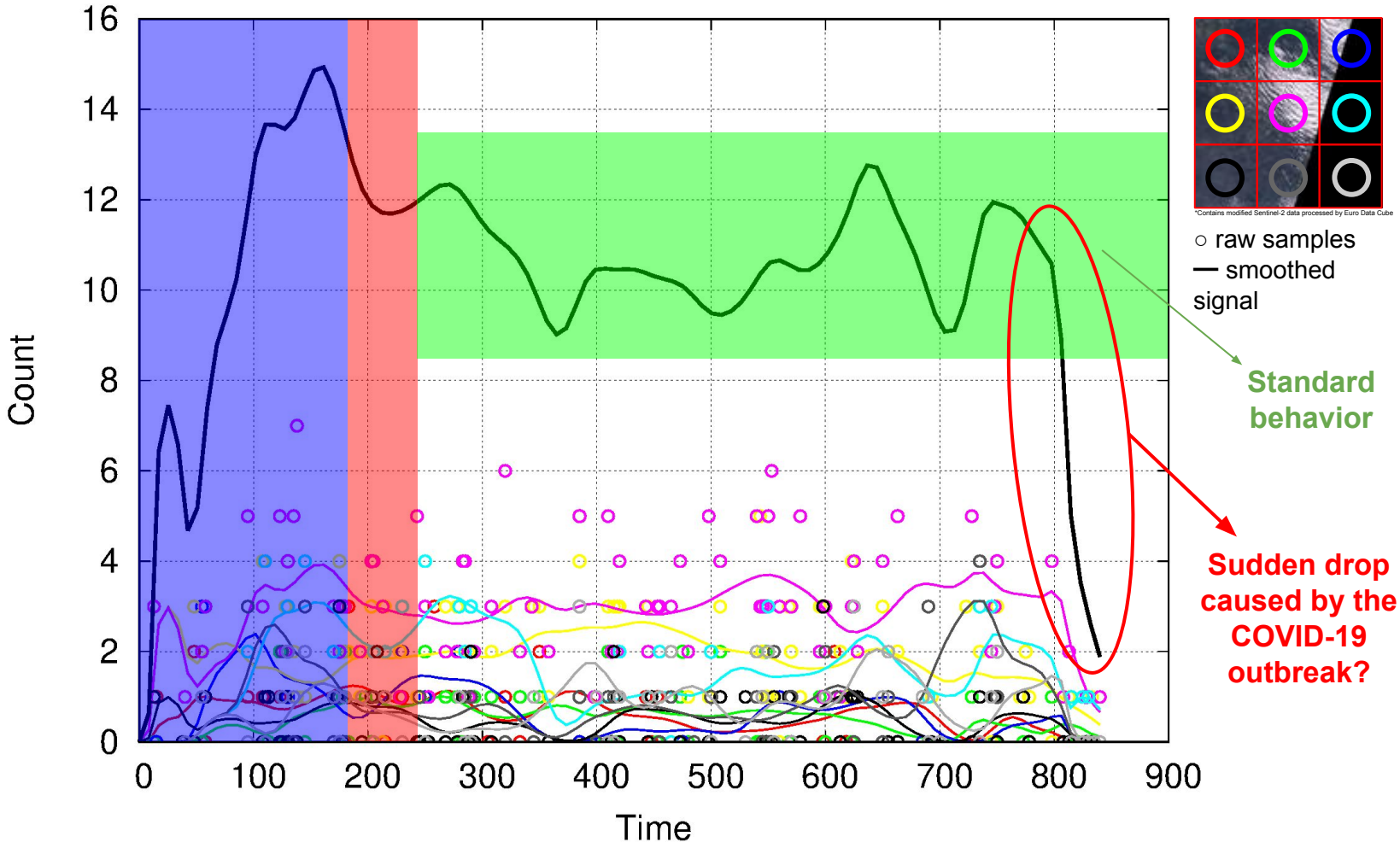


Valid pixel mask

Handling data absence and occlusions:
- Divide image in nine blocks;
- Count flying airplanes in each of them separately;
- Discard blocks with more than 30% pixels classified as cloud;
- Discard blocks with less than 90% of valid pixels;
- Ignore detections with blob size smaller than 10 pixels.


16
14
12
10
8
6
4
2
0
Count
0
100
200
300
400
500
600
700
800
900
Time
*Contains modified Sentinel-2 data processed by Euro Data Cube
○ raw samples
— smoothed signal


Smoothing window affected by training data
Training data
Bezier smoothing for visualization purposes
Count
Time
0
2
4
6
8
10
12
14
16
0
100
200
300
400
500
600
700
800
900
*Contains modified Sentinel-2 data processed by Euro Data Cube
○ raw samples
— smoothed signal


16
14
12
10
8
6
4
2
0
Count
0
100
200
300
400
500
600
700
800
900
Time
*Contains modified Sentinel-2 data processed by Euro Data Cube
○ raw samples
— smoothed signal
Standard behavior
Sudden drop caused by the COVID-19 outbreak?

Temporal analysis of flying airplanes

- Counting is very accurate
  - Flying airplanes occupy a larger area than parked airplanes
  - Pattern is very discriminative
- The drop caused by the COVID-19 outbreak is not a normal behavior in terms of both quantity and oscillation
- Can indicate whether airport activity is back to normal or not

Automatic detection of behavioral changes

- Anomaly detection
  - Predict future measurements and check the error
  - Long-Short-Time-Memory (LSTM) networks
    - 100 hidden units
- 80% of the temporal sequence for training
  - 20% dropout to avoid overfitting
- 20% of the temporal sequence to test
- An anomaly occurs when the LSTM prediction deviates 20% from the real measurements

Automatic detection of behavioral changes

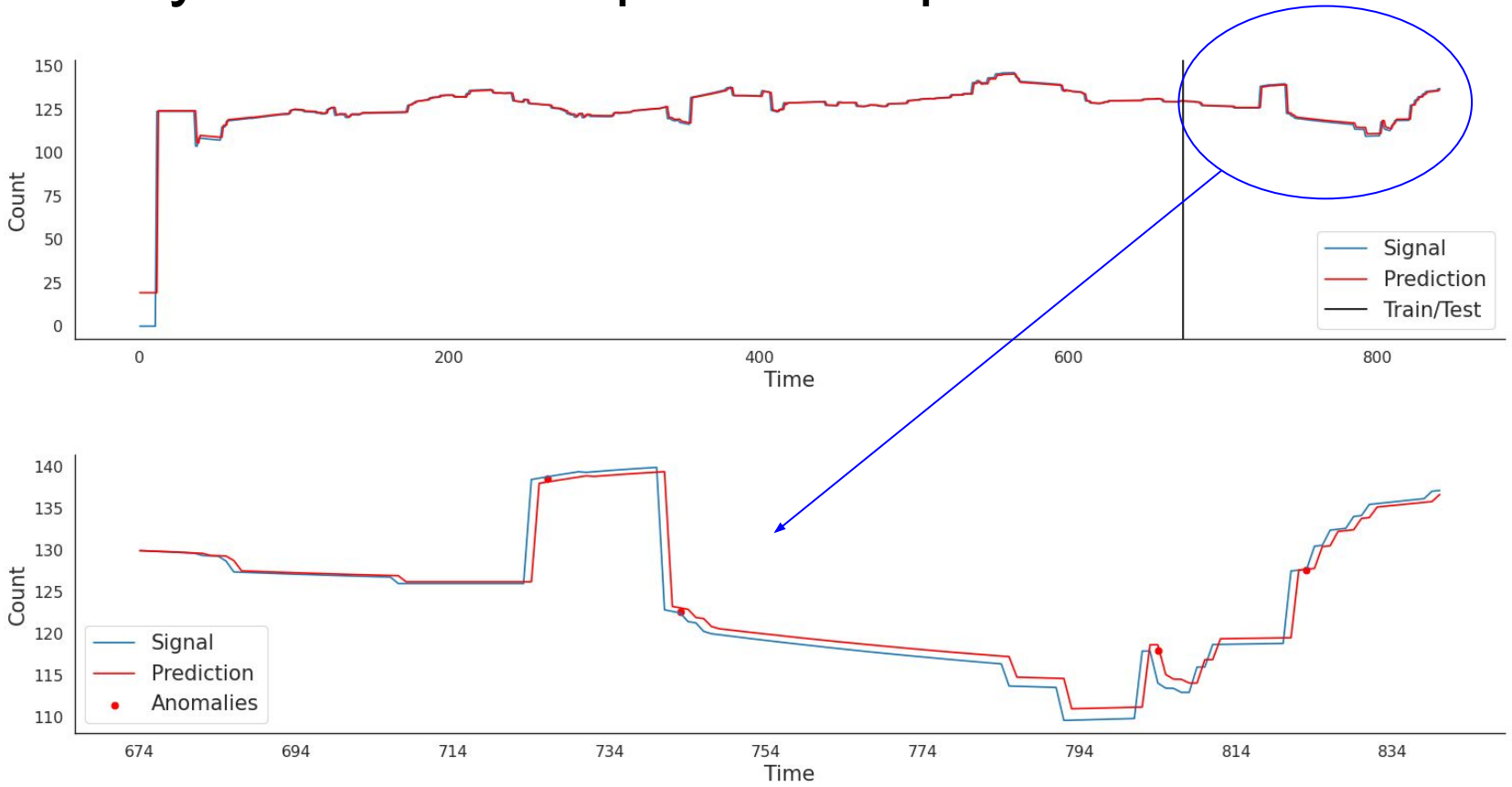
```
$ python3 detect_anomalies.py parked.csv
```

or

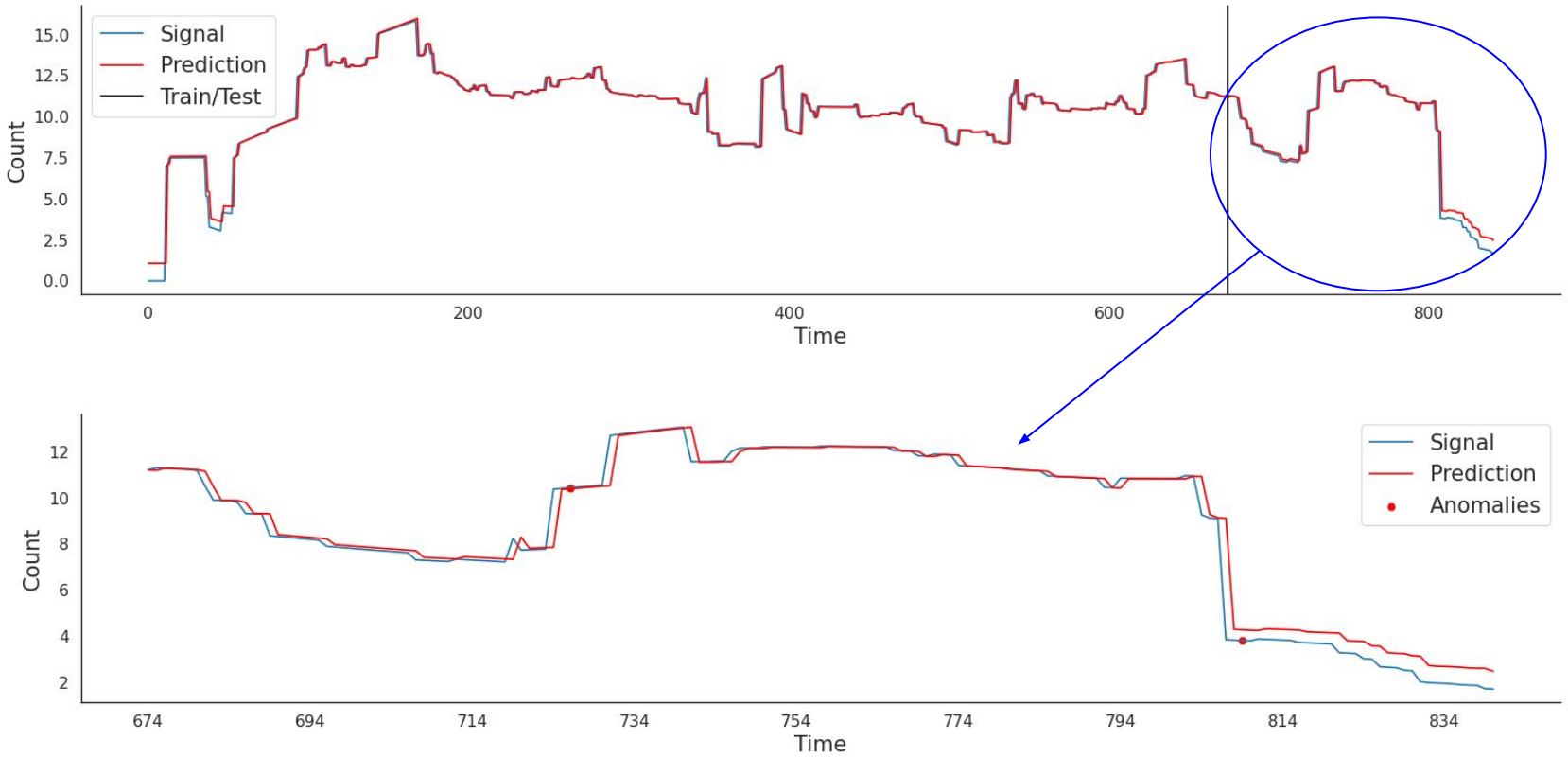
```
$ python3 detect_anomalies.py flying.csv
```

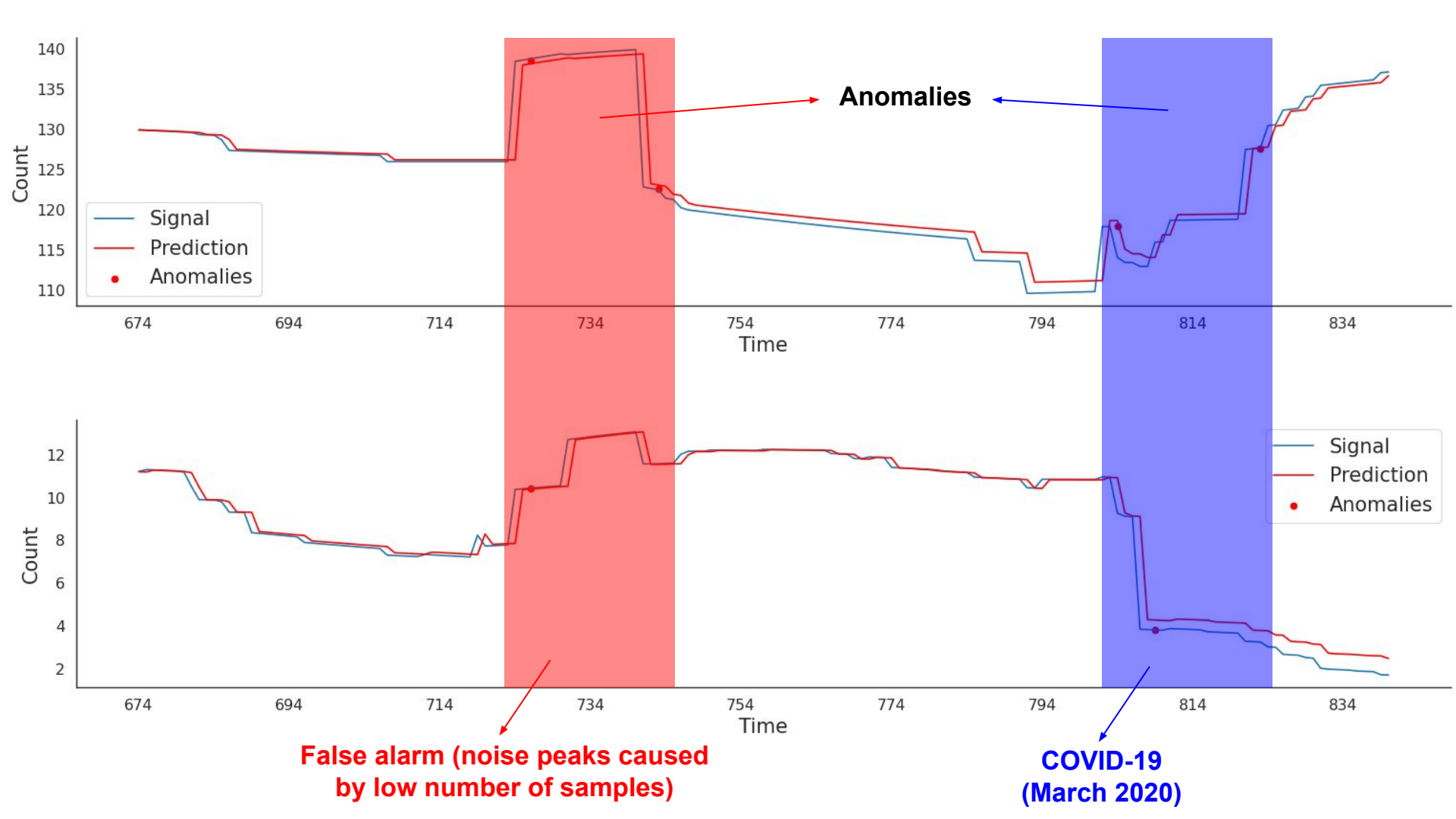
(saves results as ‘result.png’ and ‘result_zoom.png’)

Anomaly detection for parked airplanes



Anomaly detection for flying airplanes




Anomalies
Count
Time
Signal
Prediction
Anomalies
140
135
130
125
120
115
110
674
694
714
734
754
774
794
814
834
12
10
8
6
4
2
False alarm (noise peaks caused by low number of samples)
COVID-19 (March 2020)

Final remarks

- We show how to automatically detect parked and flying airplanes over time
- We correlate changes in the number of airplanes with the COVID-19 outbreak
- We define standard behavior zones that can be used as a recovery indicator
- We show how to automatically identify anomalies in temporal sequences, which can be used to keep track of several places of interest simultaneously