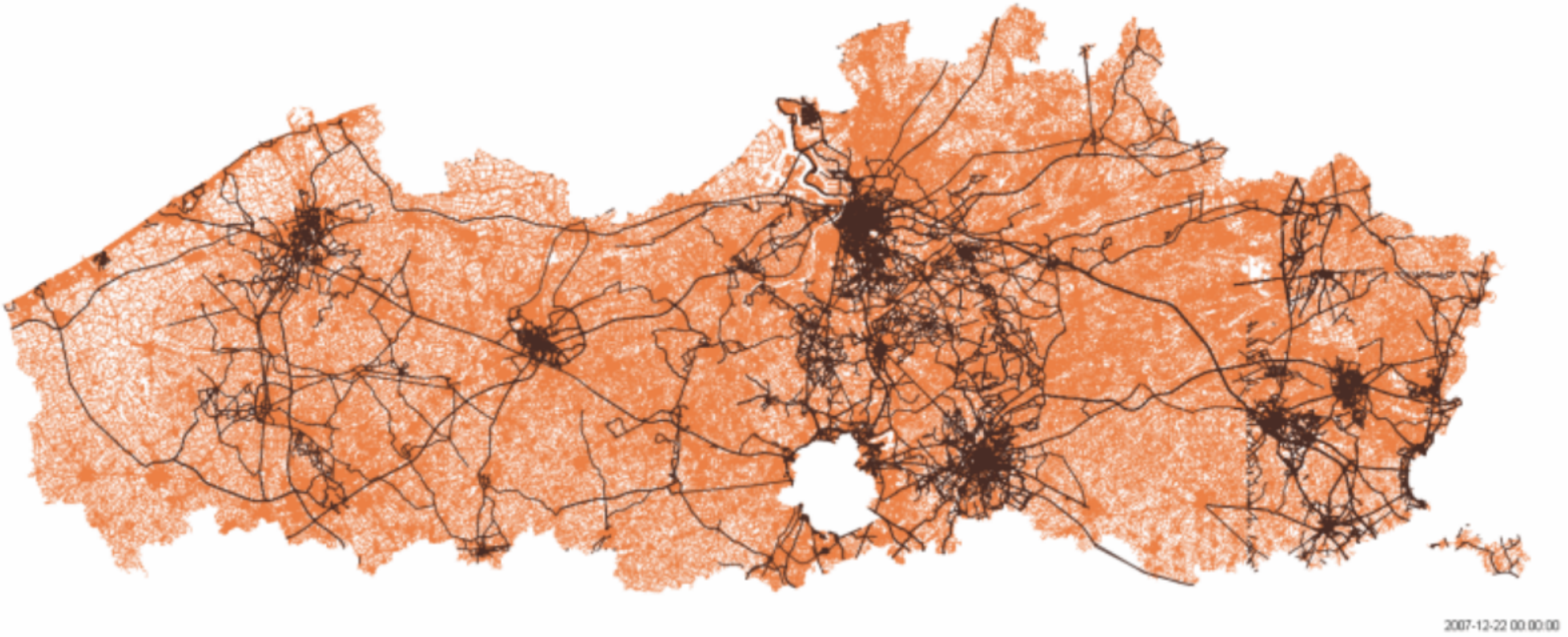
How Deep Learning, could help to improve GeoSpatial data quality ? an OSM use case

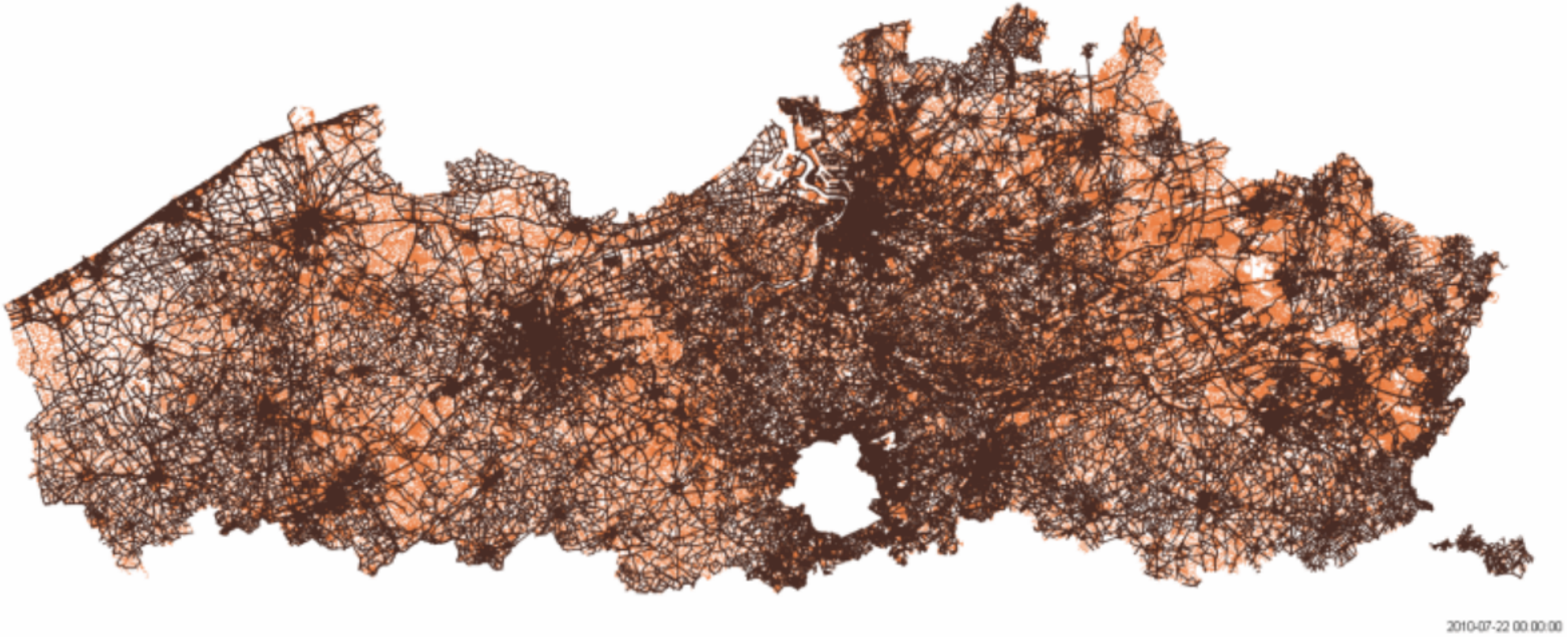
@o_courtin

FOSDEM 2018



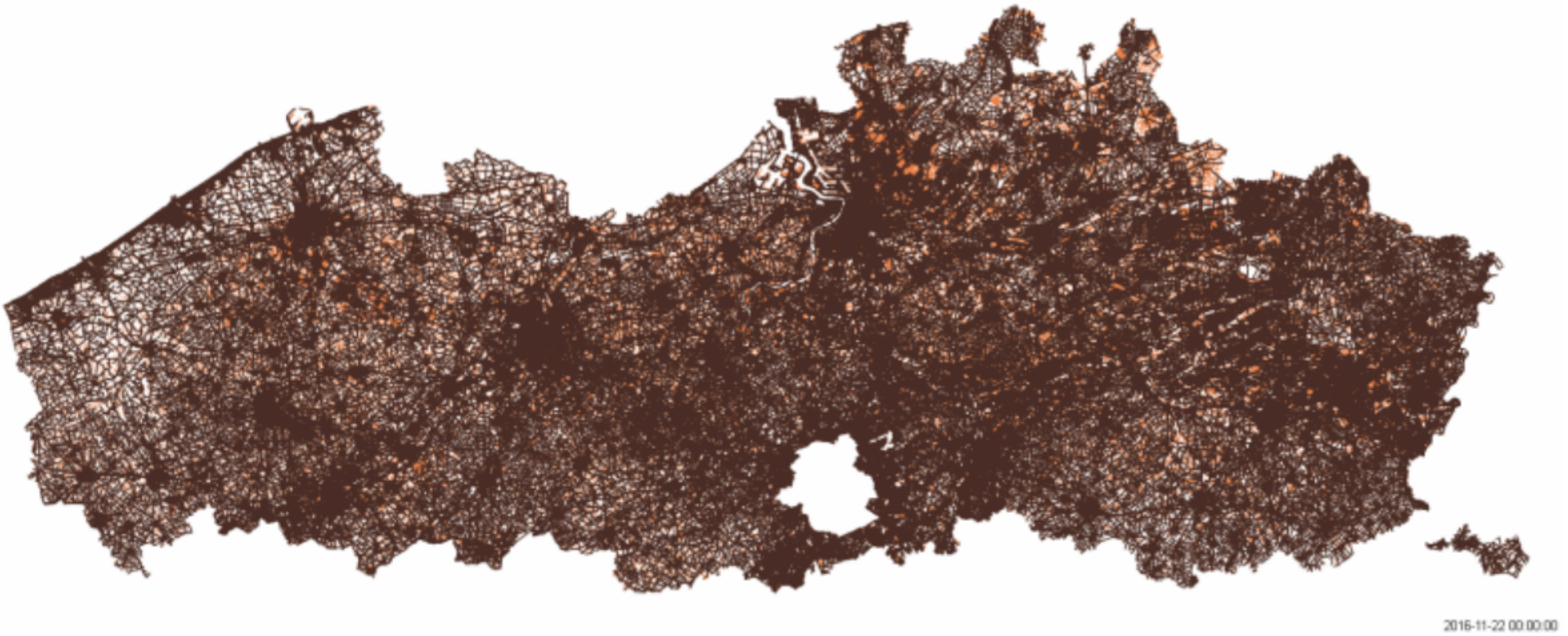
2007-12

http://www.osm.be/assets/images/road-completion2.gif



2010-07

http://www.osm.be/assets/images/road-completion2.gif



2016-11

http://www.osm.be/assets/images/road-completion2.gif

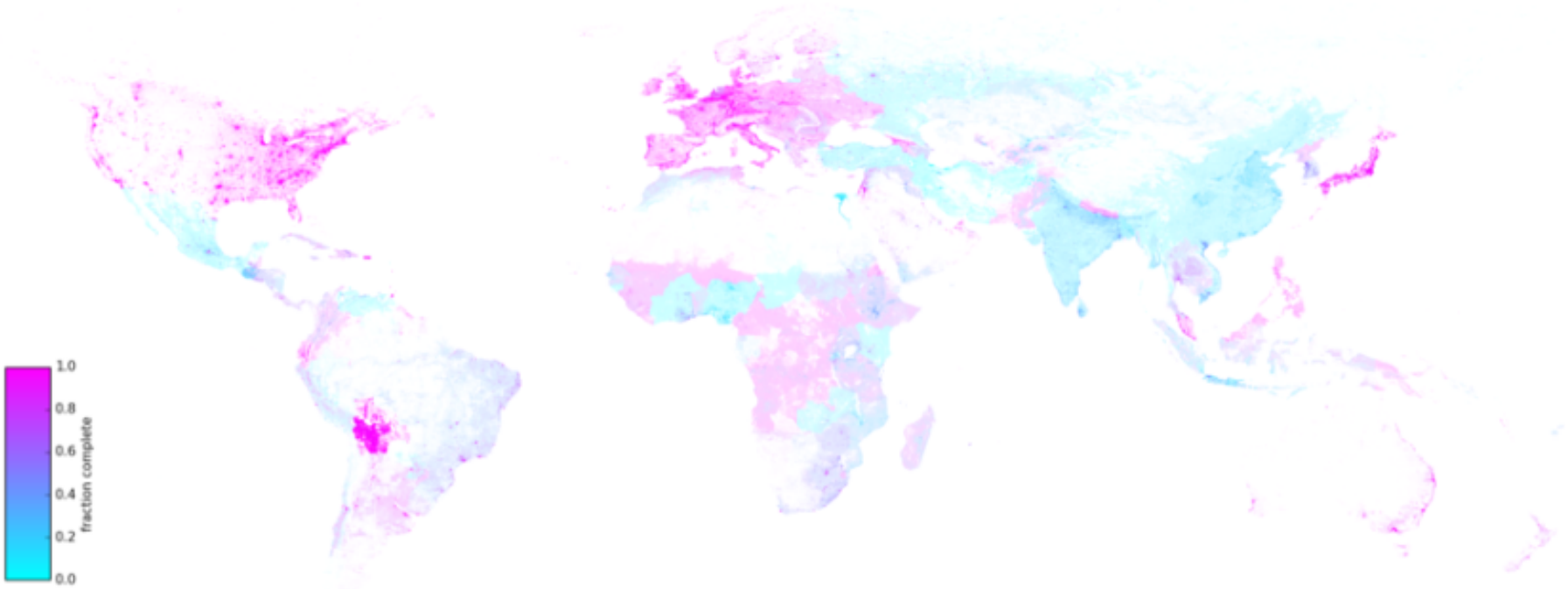


Fig 6. Completeness of the OSM dataset, by grid cell, January 2016. The fraction complete is estimated by the multilevel model. The color intensity represents the number of estimated street edges, thus highlighting parts of the world with a denser street network. The full-resolution image is available online.

http://wellbeing.ihsp.mcgill.ca/publications/Barrington-Leigh-Millard-Ball-PLOS2017.pdf

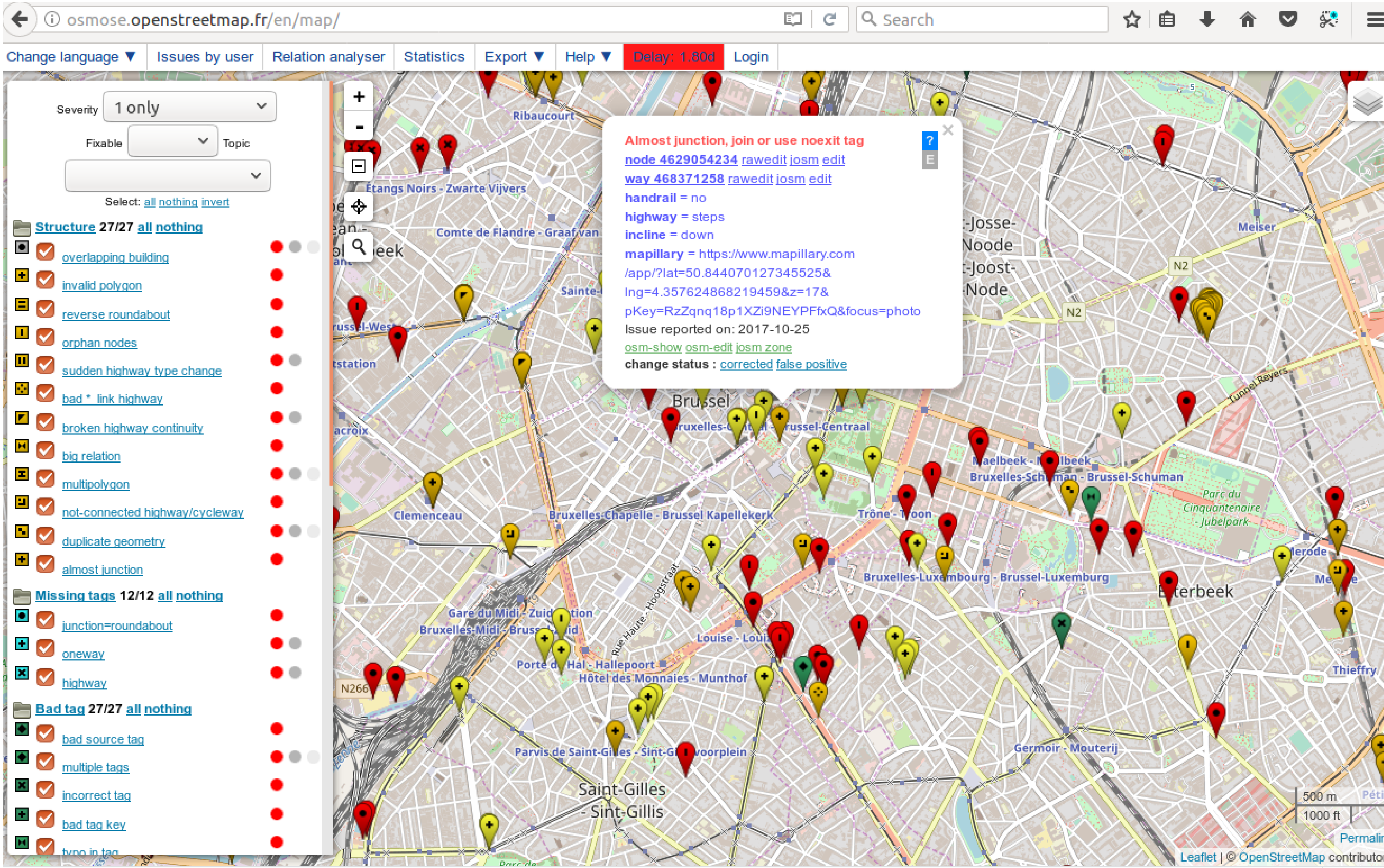
Error detection tools

Error Detecting Tools check the OSM data for potential data errors, inaccuracy or sparsely mapped places. Users should check if these structures are really wrong (false positives usually occur and there are not really mapping rules which are set in stone) and correct the data for a continuously rising data quality.

Comparison of some of the following tools

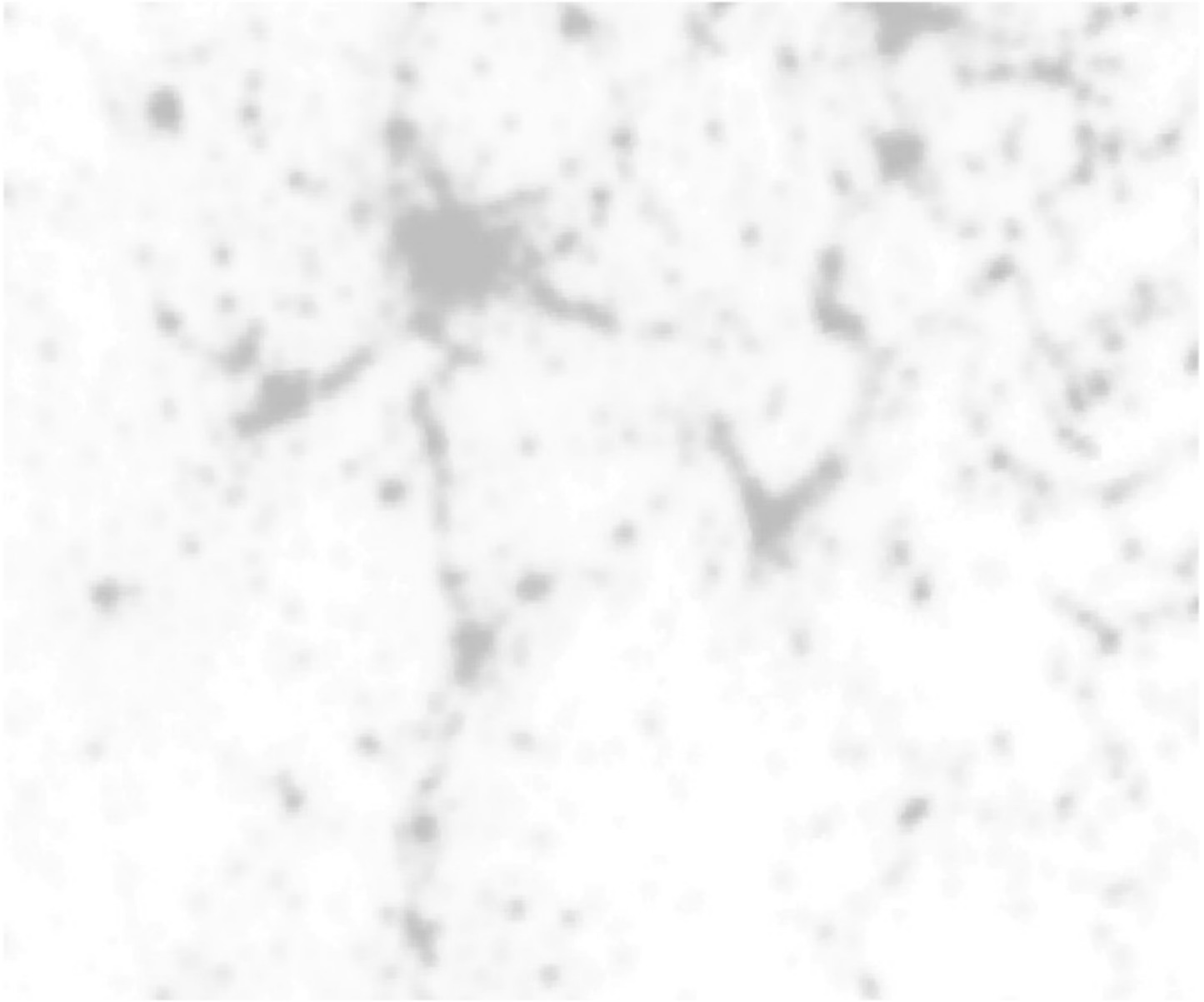
Tool	Coverage	Error types	Display mode	Fix suggestion	Downloadable	API	Correction guide
Keep Right	World	Many (50+)	Marker map	no	yes	yes	German only
Osmose	World	Many (200+)	Marker map	yes	yes	yes	yes
JOSM/Validator	Local	Many	List	yes	N/A	N/A	For some problems
OSM Inspector	World/Partial	Many	Rendered map	no	yes	N/A	no
Maproulette	World/Partial	Many (10+)	One feature at time	no	yes	yes	no

http://wiki.openstreetmap.org/wiki/Quality_assurance


osmose.openstreetmap.fr/en/map/
Search
Change language ▼
Issues by user
Relation analyser
Statistics
Export ▼
Help ▼
Delay: 1.80d
Login
Severity
1 only
Fixable
Topic
Select: all nothing invert
Structure 27/27 all nothing
overlapping building
invalid polygon
reverse roundabout
orphan nodes
sudden highway type change
bad *_link highway
broken highway continuity
big relation
multipolygon
not-connected highway/cycleway
duplicate geometry
almost junction
Missing tags 12/12 all nothing
junction=roundabout
oneway
highway
Bad tag 27/27 all nothing
bad source tag
multiple tags
incorrect tag
bad tag key
typo in tag
Almost junction, join or use noexit tag
node 4629054234 rawedit josm edit
way 468371258 rawedit josm edit
handrail = no
highway = steps
incline = down
mapillary = https://www.mapillary.com/app/?lat=50.844070127345525&lng=4.357624868219459&z=17&pKey=RzZqnq18p1XZi9NEYPFfxQ&focus=photo
Issue reported on: 2017-10-25
osm-show osm-edit josm zone
change status : corrected false positive
Ribaucourt
Etangs Noirs - Zwarte Vijvers
Comte de Flandre - Graaf van
Clemenceau
Bruxelles-Chapelle - Brussel Kapellekerk
Gare du Midi - Zuidstation
Porte de Hal - Hallepoort
Hôtel des Monnaies - Munthof
Louise - Louiza
Parvis de Saint-Gilles - Sint-Gillisvoorplein
Saint-Gilles - Sint-Gillis
Trône - Troon
Bruxelles-Luxembourg - Brussel-Luxemburg
Maelbeek - Maalbeek
Bruxelles-Schuman - Brussel-Schuman
Parc du Cinquantenaire - Jubelpark
Etterbeek
Germoir - Mouterij
Thieffry
Meiser
Tunnel Reyers
N2
N266
Brussel
500 m
1000 ft
Permalink
Leaflet | © OpenStreetMap contributors

What about using an other dataset, to hilight (in)consistencies ?

Light pollution map



Open Data from : http://geodata.grid.unep.ch - 2003 Raster

SELECT corr (pop_density, light)::numeric(4,4) FROM own.commune;

0.6533

- - OSM 08/2014
SELECT corr (road_density, light)::numeric(4,4) FROM own.commune;

0.7573

- - OSM 08/2016
SELECT corr (road_density, light)::numeric(4,4) FROM own.commune;

0.7782

IN CS, IT CAN BE HARD TO EXPLAIN THE DIFFERENCE BETWEEN THE EASY AND THE VIRTUALLY IMPOSSIBLE.

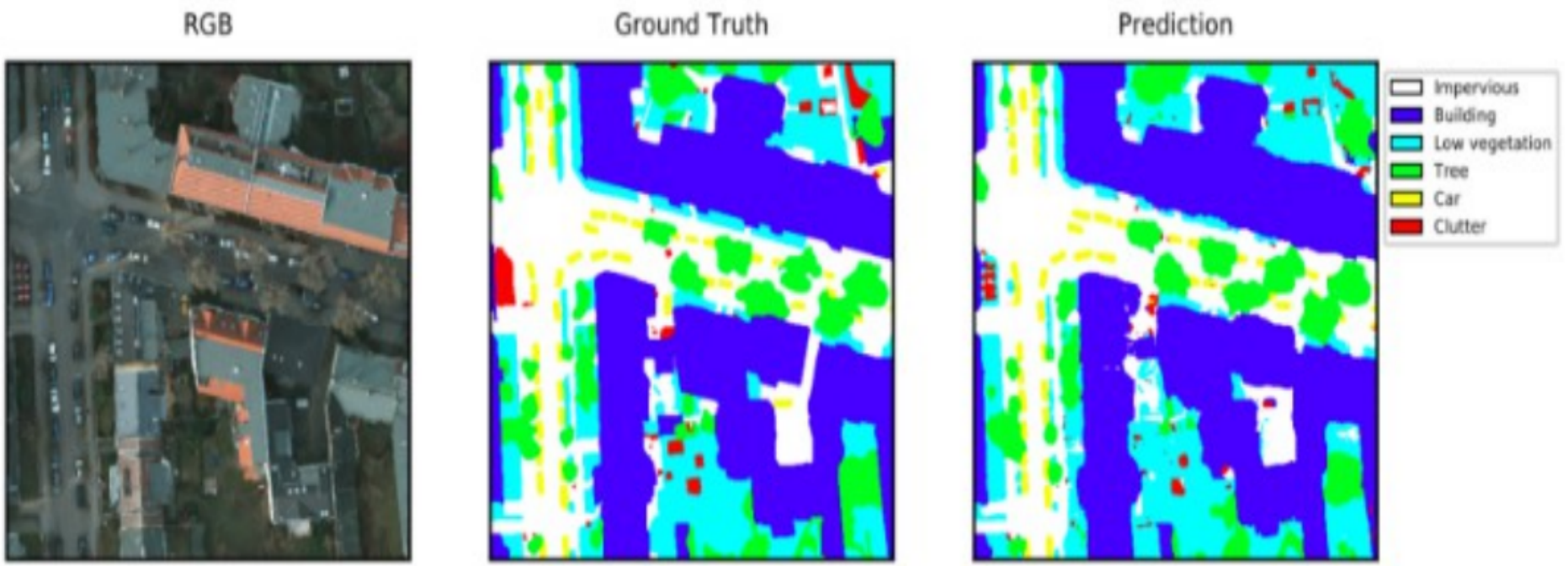
azavea

SOFTWARE DEVELOPMENT

Deep Learning for Semantic Segmentation of Aerial Imagery

By Rob Emanuele on May 30th, 2017

https://www.azavea.com/blog/2017/05/30/deep-learning-on-aerial-imagery/



Pre-trained ResNet50 with ImageNet on IR-R-G

	Overall	Impervious	Building	Low Vegetation	Tree	Car	Clutter
Validation	85.8	89.1	91.8	82.0	83.3	93.7	63.2
Test	89.2	91.4	96.1	86.1	86.6	93.3	46.8

```
# The number of output labels
nb_labels = 6

# The dimensions of the input images
nb_rows = 256
nb_cols = 256

# A ResNet model with weights from training on ImageNet. This will
# be adapted via graph surgery into an FCN.
base_model = ResNet50(
    include_top=False, weights='imagenet', input_tensor=input_tensor)

# Get final 32x32, 16x16, and 8x8 layers in the original
# ResNet by that layers's name.
x32 = base_model.get_layer('final_32').output
x16 = base_model.get_layer('final_16').output
x8 = base_model.get_layer('final_x8').output

# Compress each skip connection so it has nb_labels channels.
c32 = Convolution2D(nb_labels, (1, 1))(x32)
c16 = Convolution2D(nb_labels, (1, 1))(x16)
c8 = Convolution2D(nb_labels, (1, 1))(x8)

```

```
23.
24. # Resize each compressed skip connection using bilinear interpolation.
25. # This operation isn't built into Keras, so we use a LambdaLayer
26. # which allows calling a Tensorflow operation.
27. def resize_bilinear(images):
28.     return tf.image.resize_bilinear(images, [nb_rows, nb_cols])
29.
30. r32 = Lambda(resize_bilinear)(c32)
31. r16 = Lambda(resize_bilinear)(c16)
32. r8 = Lambda(resize_bilinear)(c8)
33.
34. # Merge the three layers together using summation.
35. m = Add()([r32, r16, r8])
36.
37. # Add softmax layer to get probabilities as output. We need to reshape
38. # and then un-reshape because Keras expects input to softmax to
39. # be 2D.
40. x = Reshape((nb_rows * nb_cols, nb_labels))(m)
41. x = Activation('softmax')(x)
42. x = Reshape((nb_rows, nb_cols, nb_labels))(x)
43.
44. fcn_model = Model(input=input_tensor, output=x)
```

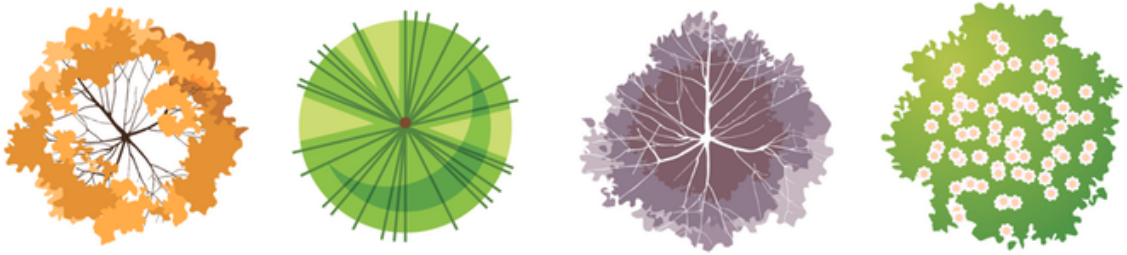
Dstl Satellite Imagery Feature Detection

Can you train an eye in the sky?

$100,000 · 419 teams · 8 months ago

Overview

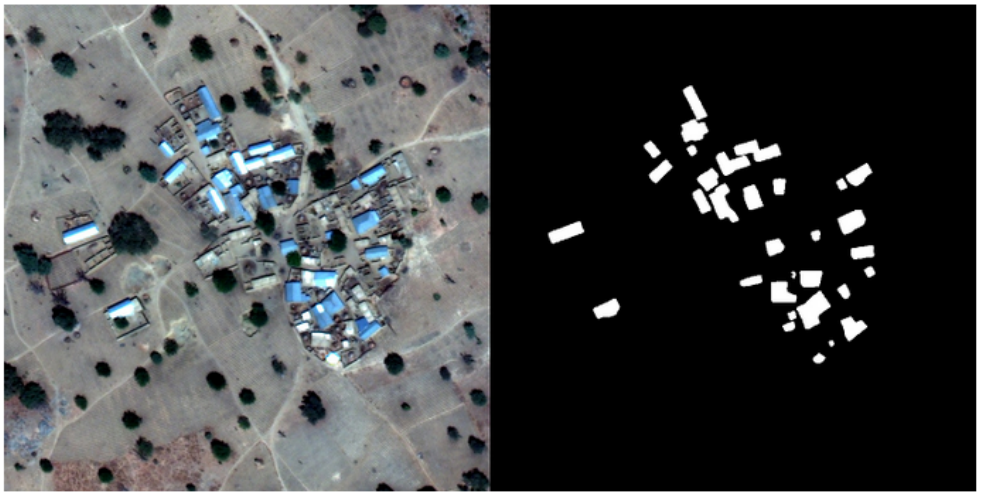
The proliferation of satellite imagery has given us a radically improved understanding of our planet. It has enabled us to better achieve everything from mobilizing resources during disasters to monitoring effects of global warming. What is often taken for granted is that advancements such as these have relied on labeling features of significance like building footprints and roadways fully by hand or through imperfect semi-automated methods.



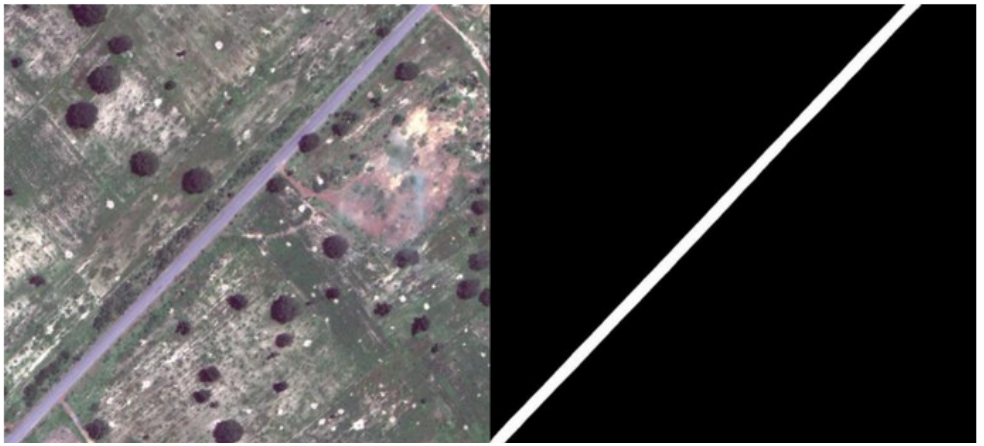
As these large, complex datasets continue to increase exponentially in number, the Defence Science and Technology Laboratory (Dstl) is seeking novel solutions to alleviate the burden on their image analysts. In this competition, Kagglers are challenged to accurately classify features in overhead imagery. Automating feature labeling will not only help Dstl make smart decisions more quickly around the defense and security of the UK, but also bring innovation to computer vision methodologies applied to satellite imagery.

Final results

Below we present a small sample of the final results from our models:

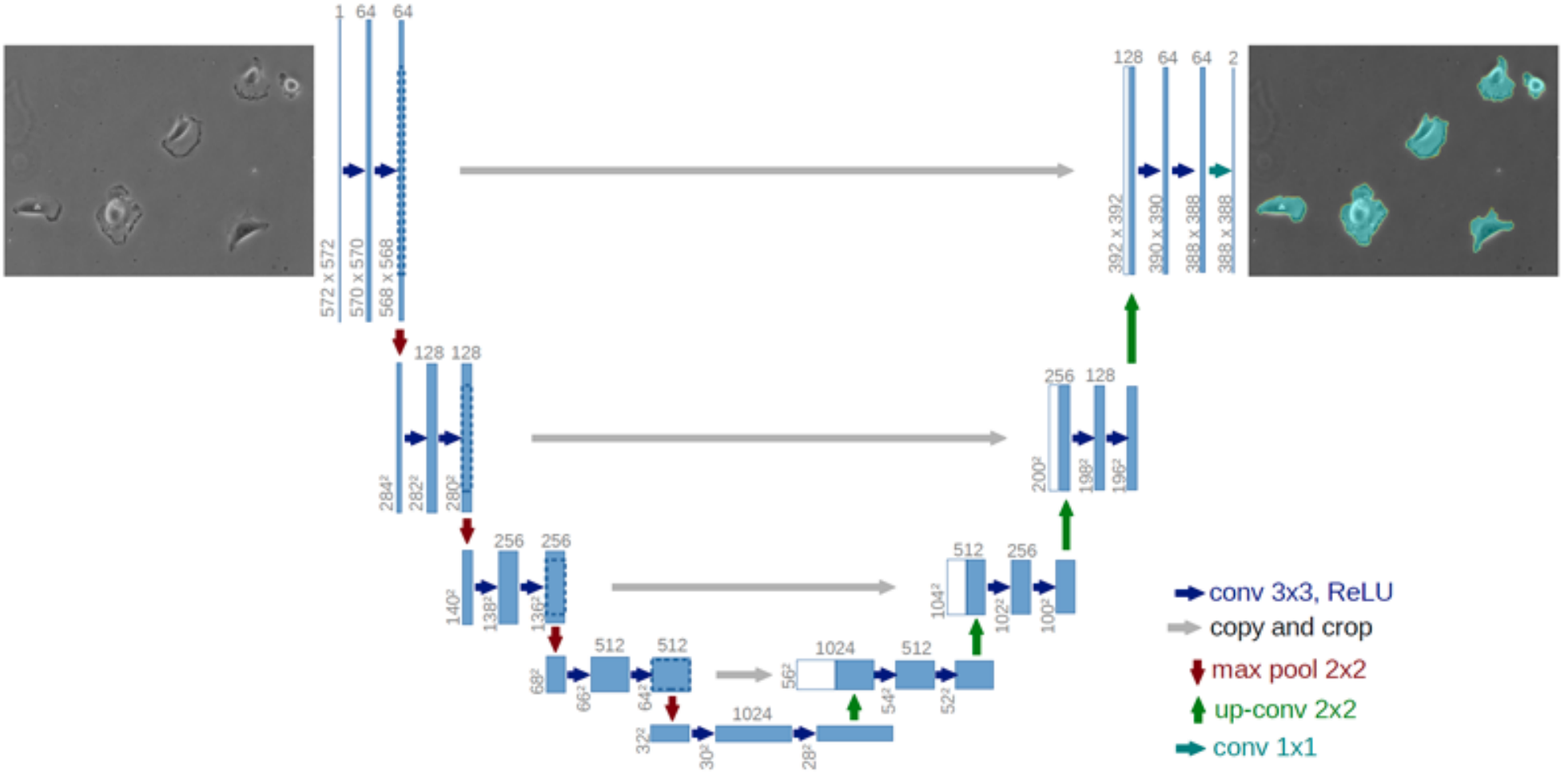


Buildings



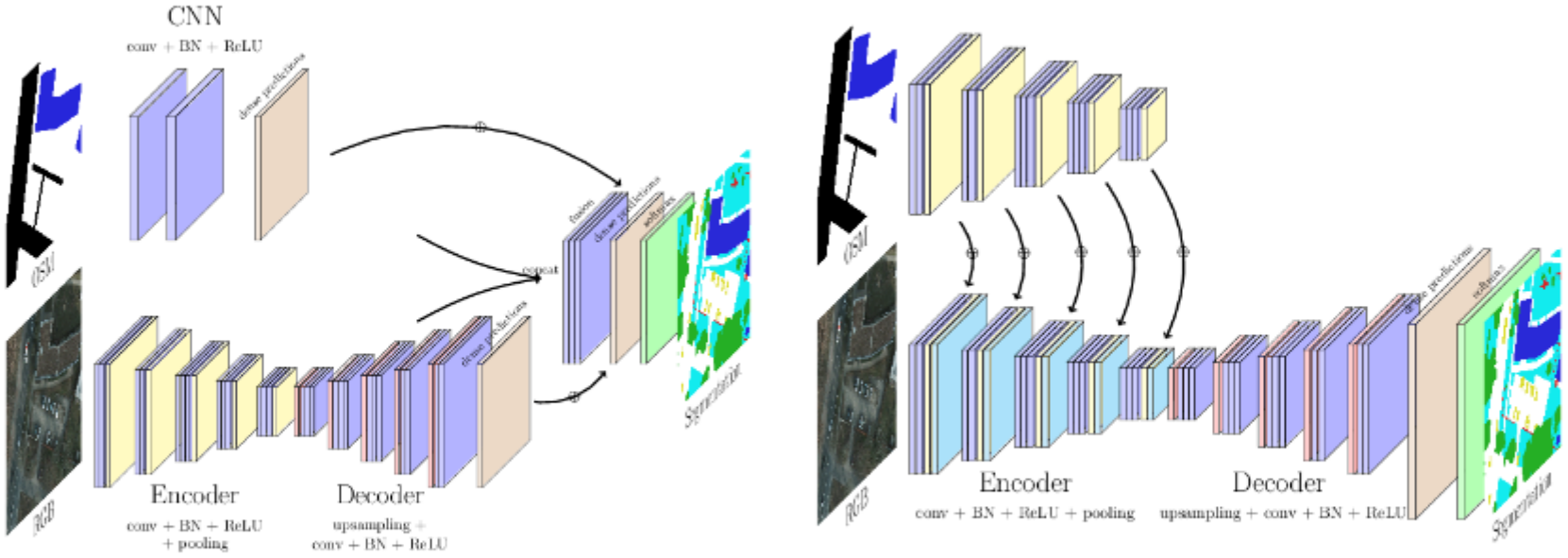
Type	Wavebands	Pixel resolution	#channels	Size
grayscale	Panchromatic	0.31 m	1	3348 x 3392
3-band	RGB	0.31 m	3	3348 x 3392
16-band	Multispectral	1.24 m	8	837 x 848
	Short-wave infrared	7.5 m	8	134 x 136

https://blog.deepsense.ai/deep-learning-for-satellite-imagery-via-image-segmentation/



U-Net: Convolutional Networks for Biomedical Image Segmentation

https://arxiv.org/abs/1505.04597



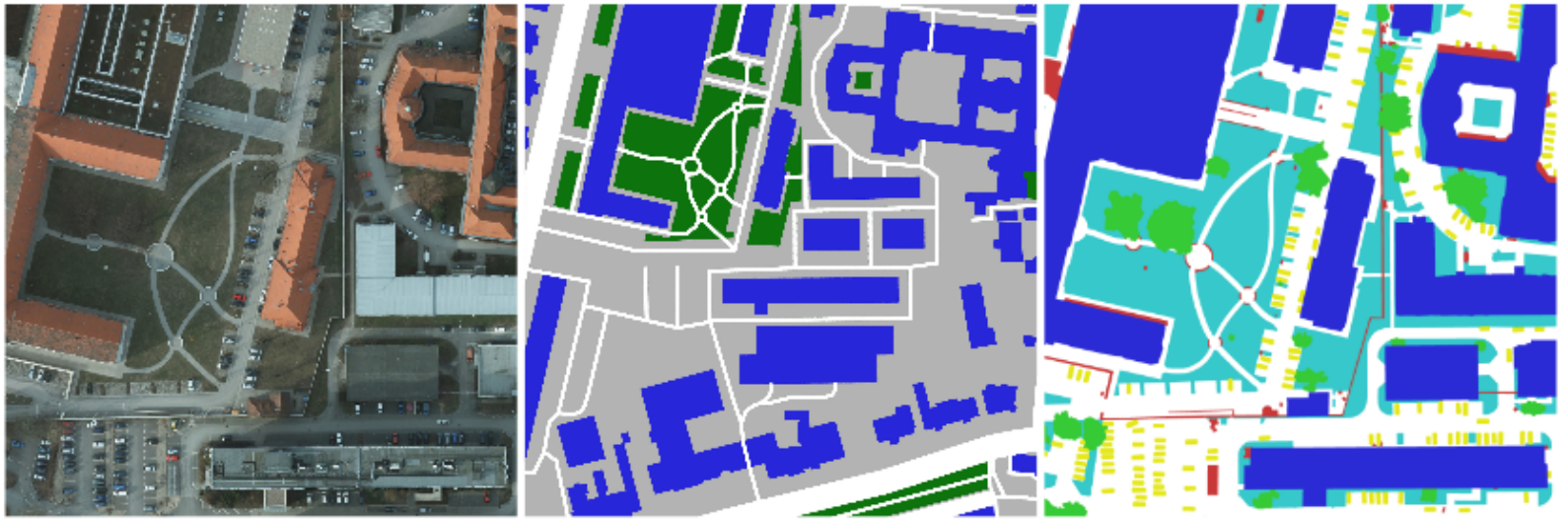
(a) Optical and OSM data fusion using residual correction [11].

(b) FuseNet [13] architecture applied to optical and OSM data.

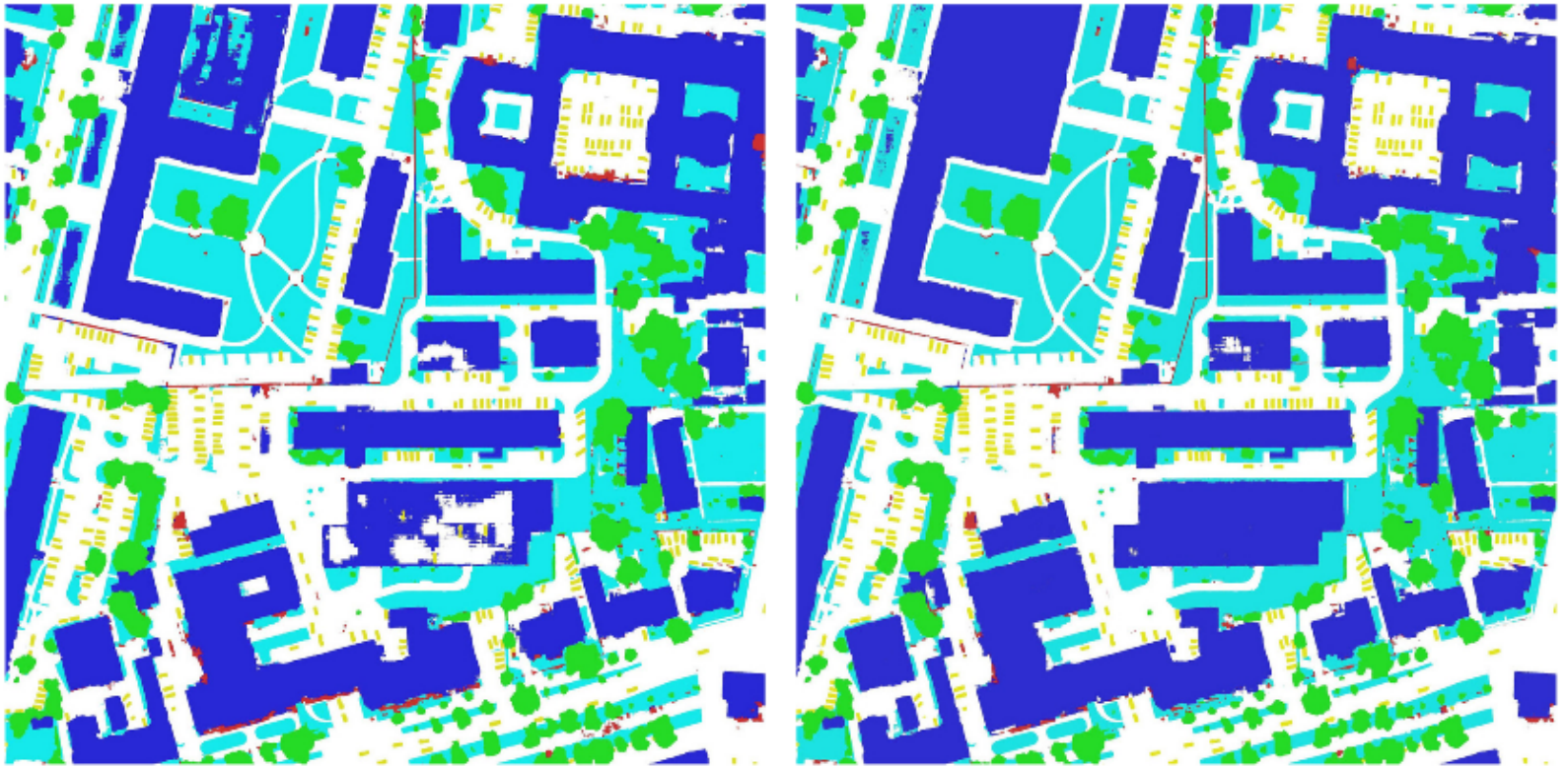
Figure 1: Deep learning architectures for joint processing of optical and OpenStreetMap data.

Nicolas Audebert, Bertrand Le Saux, Sébastien Lefèvre. Joint Learning from Earth Observation and OpenStreetMap Data to Get Faster Better Semantic Maps. EARTHVISION 2017 IEEE/ISPRS CVPR Workshop. Large Scale Computer Vision for Remote Sensing Imagery, Jul 2017, Honolulu, United States. 2017.

https://hal.archives-ouvertes.fr/hal-01523573



(a) RGB input (b) OSM input (c) Target ground truth



(d) SegNet (RGB) (e) FuseNet (OSM+RGB)

Figure 4: Excerpt from the classification results on Potsdam

Ben Hamner @benhamner · Nov 12

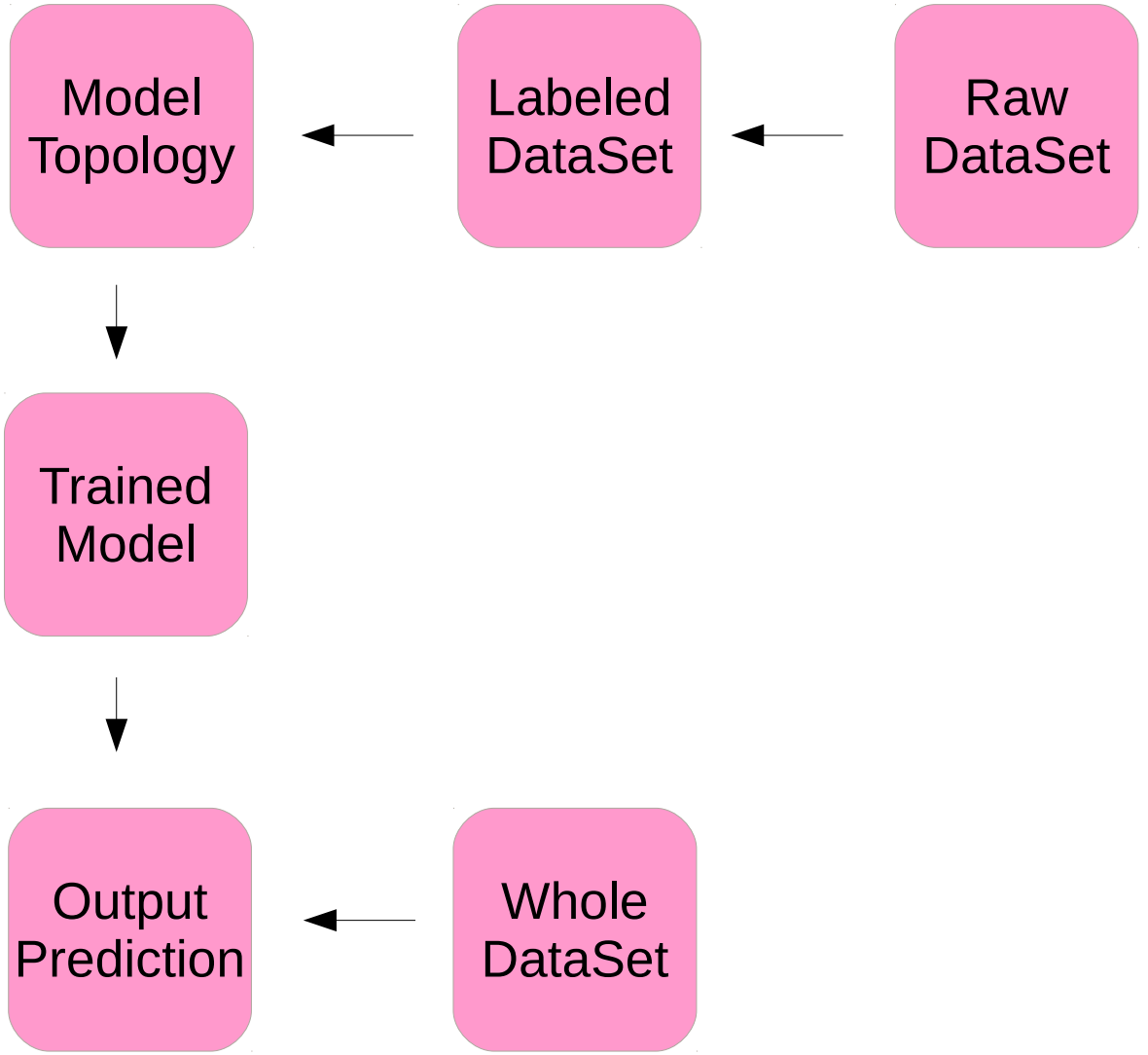
Easy parts of applying machine learning:

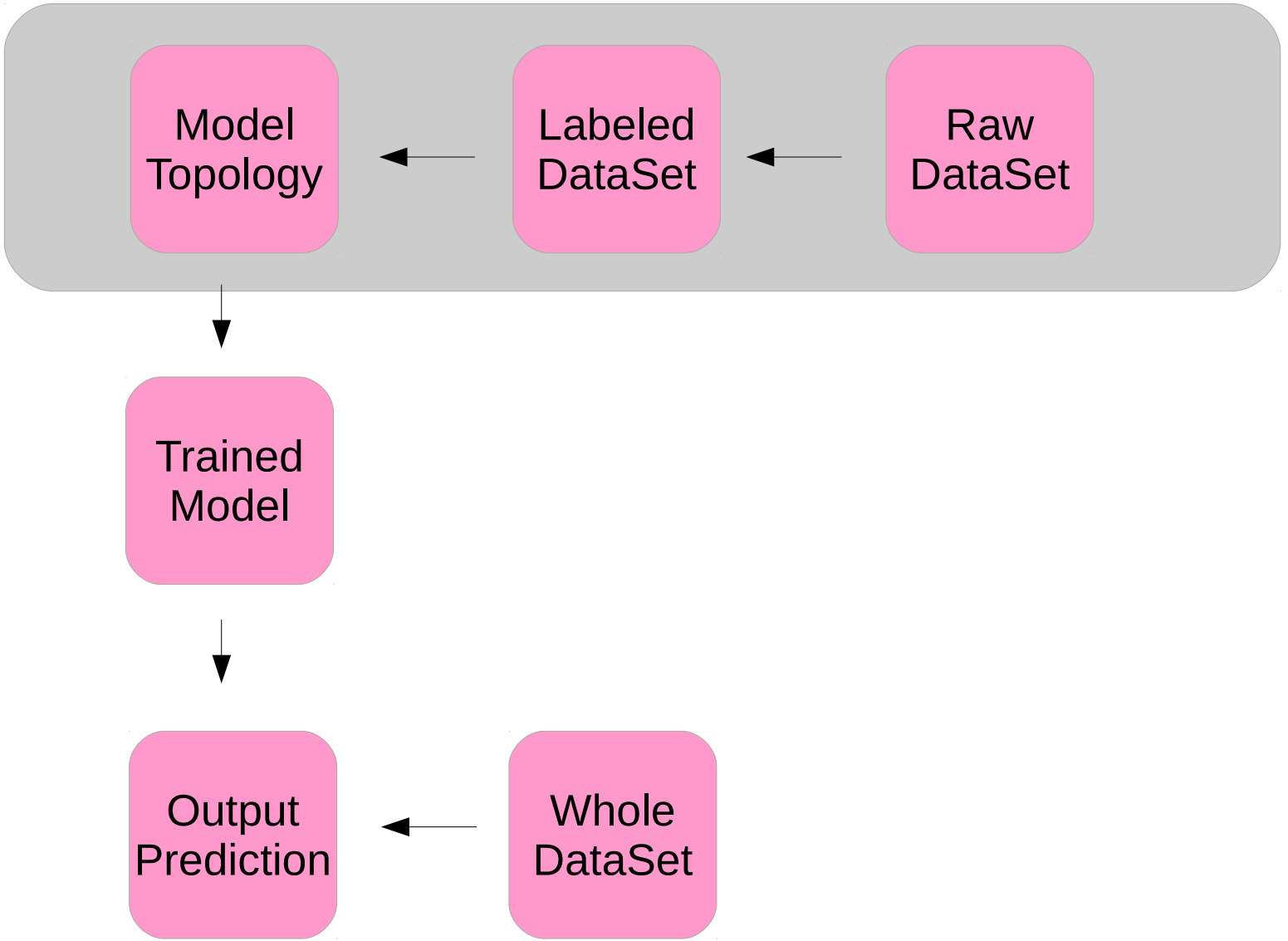
.fit()
.predict()

Hard parts:

.clean()
.transform()
.get_data()
.frame_problem()
.debug()
.handle_nonstationarities()
.handle_missing_inputs()

35 792 2.0K


Model Topology
Labeled DataSet
Raw DataSet
Trained Model
Output Prediction
Whole DataSet


Model Topology
Labeled DataSet
Raw DataSet
Trained Model
Output Prediction
Whole DataSet

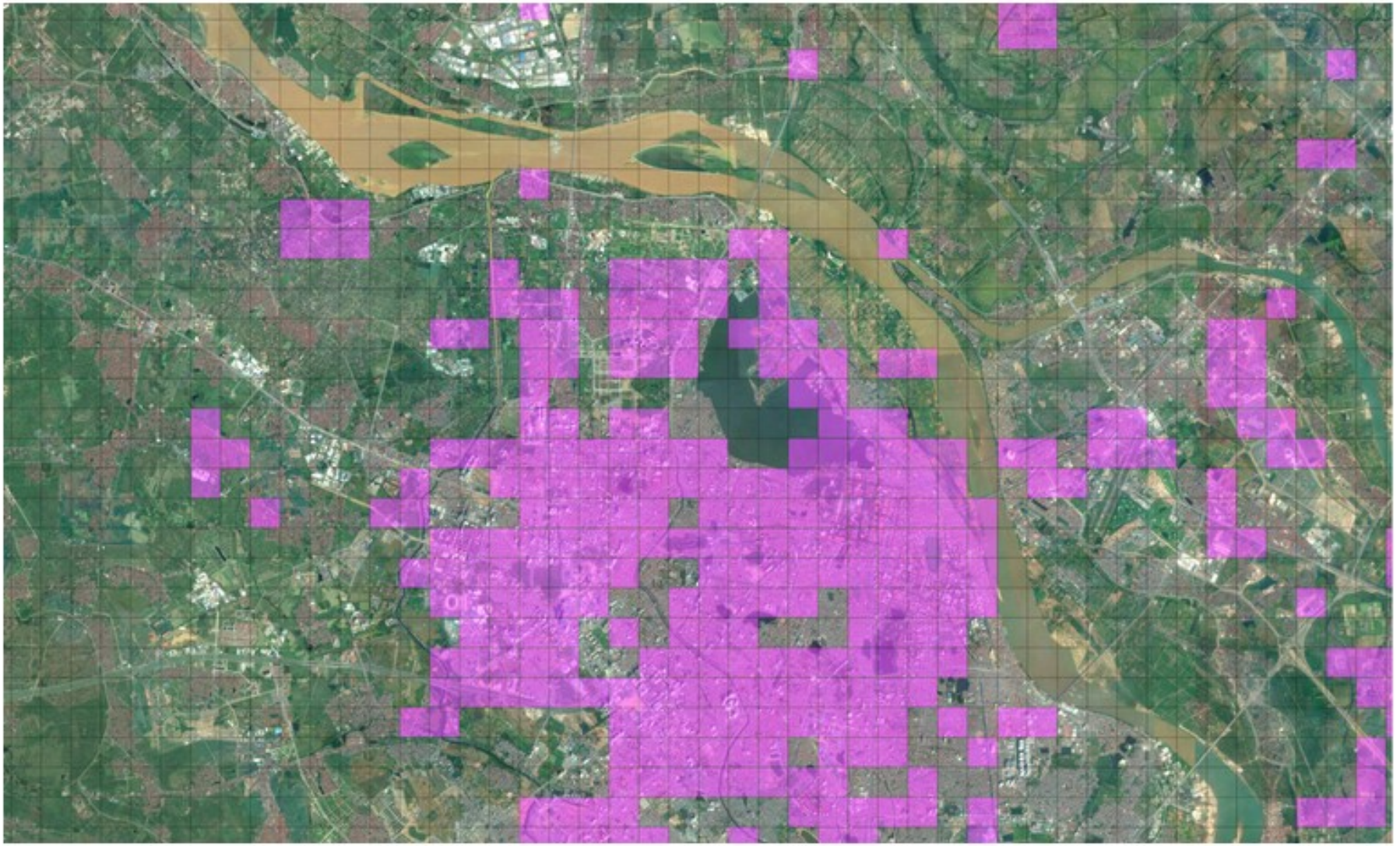
developmentSEED

BY DREW BOLLINGER ON JAN 11, 2018

Quickly plug satellite imagery into your favorite machine learning framework

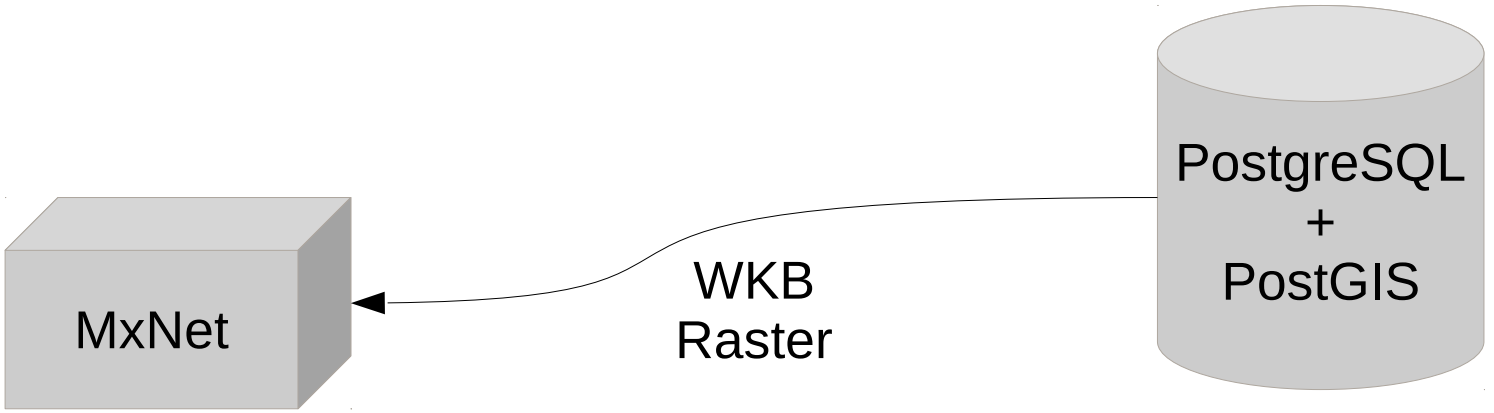
Creating labelled image chips doesn't have to be hard

https://developmentseed.org/blog/2018/01/11/label-maker/



Purple building tile labels overlaid over Mapbox Satellite Imagery.

https://developmentseed.org/blog/2018/01/19/sagemaker-label-maker-case/

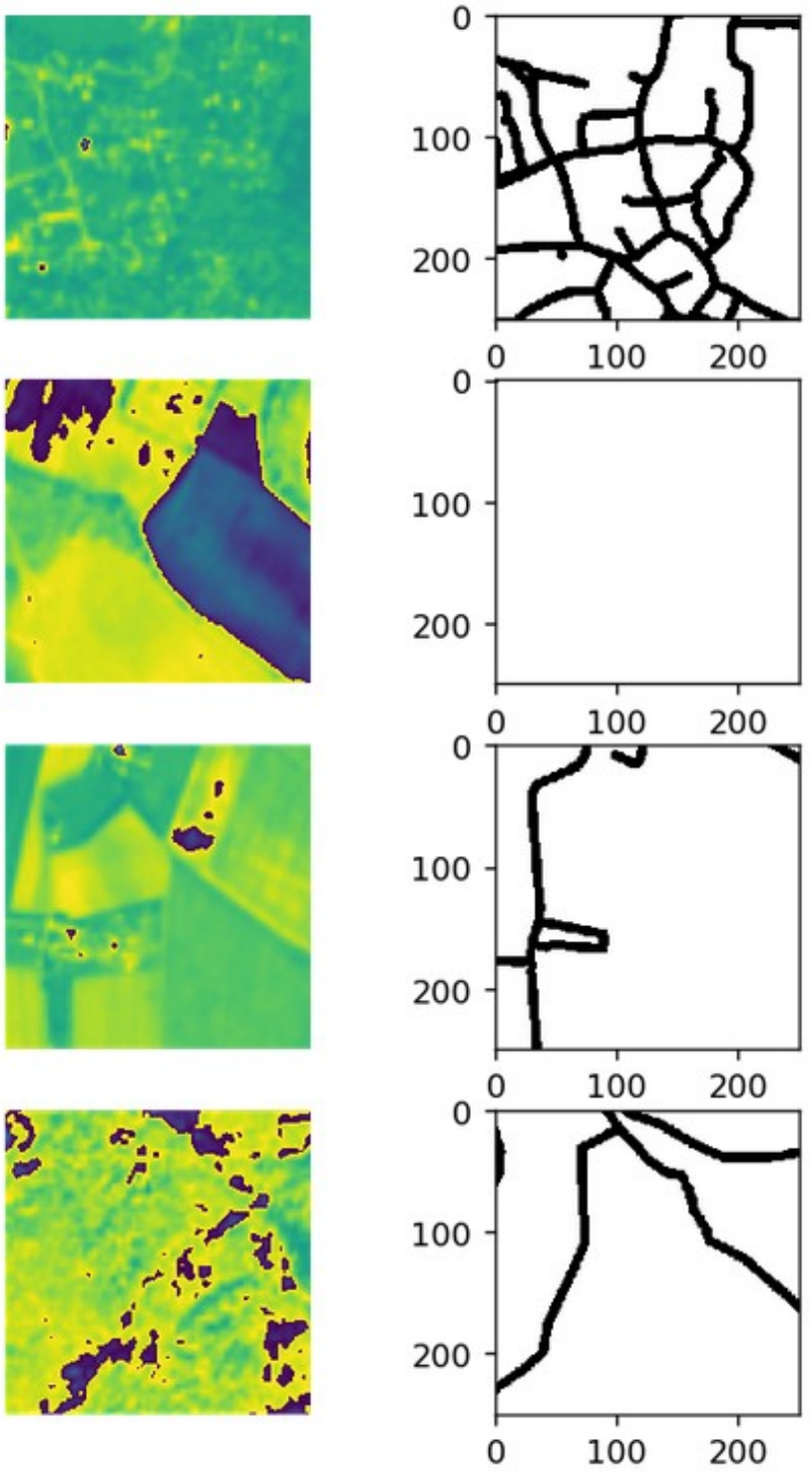

PostgreSQL
+
PostGIS
WKB
Raster
MxNet

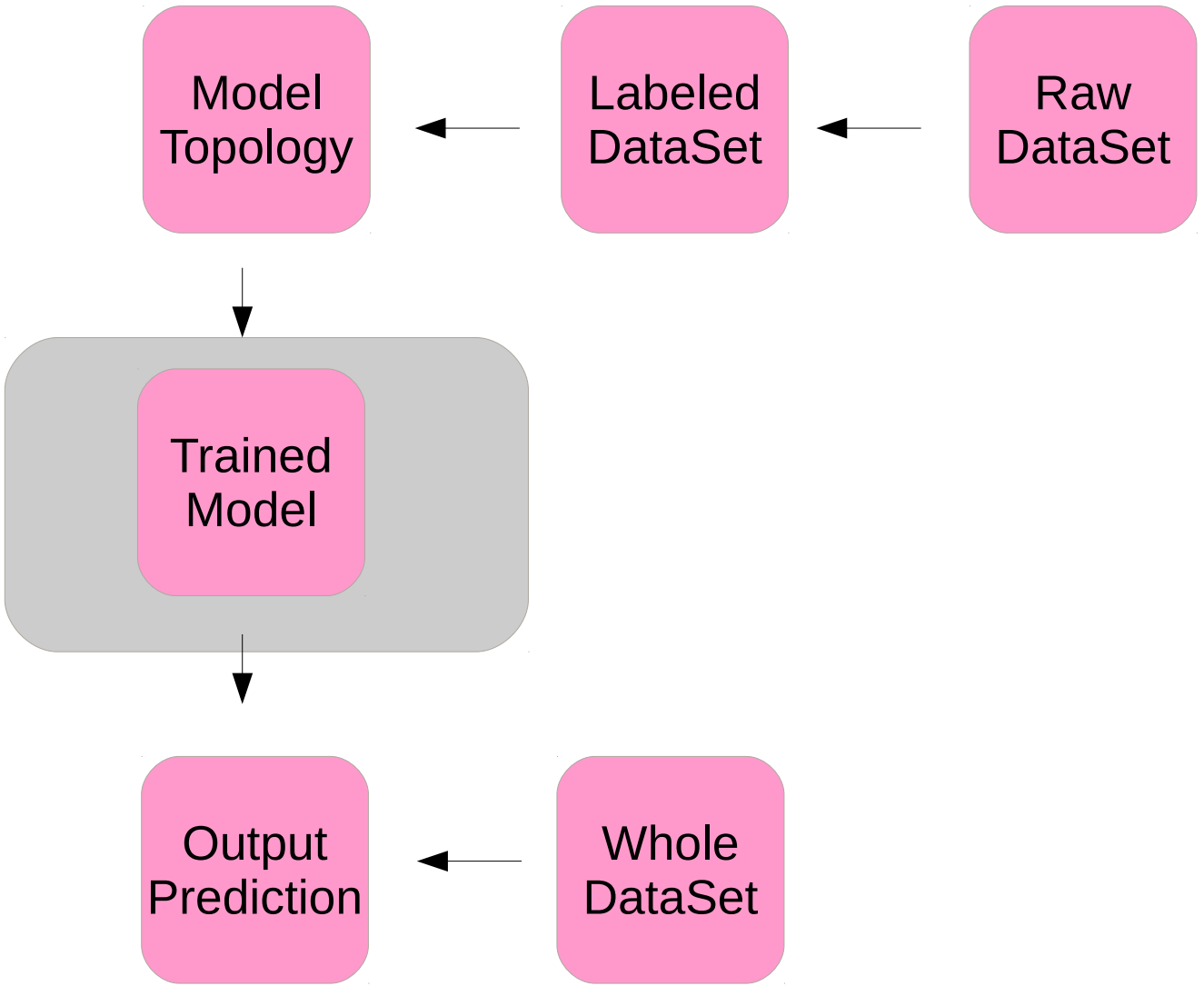
```
class SimpleIter(mx.io.DataIter):
    def __init__(self, data_names, data_shapes, data_gen,
                 label_names, label_shapes, label_gen, num_batches=10):
        self._provide_data = zip(data_names, data_shapes)
        self._provide_label = zip(label_names, label_shapes)
        self.num_batches = num_batches
        self.data_gen = data_gen
        self.label_gen = label_gen
        self.cur_batch = 0

    def __iter__(self):
        return self

    def reset(self):
        self.cur_batch = 0

    def __next__(self):
        return self.next()

    @property
    def provide_data(self):
        return self._provide_data

    @property
    def provide_label(self):
        return self._provide_label

    def next(self):
        if self.cur_batch < self.num_batches:
            self.cur_batch += 1
            data = [mx.nd.array(g(d[1])) for d,g in zip(self._provide_data, self.data_gen)]
            label = [mx.nd.array(g(d[1])) for d,g in zip(self._provide_label, self.label_gen)]
            return mx.io.DataBatch(data, label)
        else:
            raise StopIteration
```

https://mxnet.incubator.apache.org/tutorials/basic/data.html

```
WITH
origins AS (SELECT ('{{855878,6534055},{878721,6533022},{873294,6541341},{870027,6524893}}'::float[]) AS ul ),
tiles AS (
     SELECT row_number() OVER() as tid,
            ST_SetSRID(
                ST_MakeEnvelope(ul[i][1], ul[i][2], ul[i][1] + 1250, ul[i][2] + 1250)
                , 2154
            ) AS geom
     FROM origins, generate_subscripts((SELECT ul FROM origins), 1) AS i
  ),
tile_rast AS
(
    SELECT  tiles.tid,
            ST_AddBand(
                ST_SetSRID(
                    ST_MakeEmptyRaster(
                        250, 250,
                        ST_Xmin(tiles.geom)::float8,
                        ST_Ymax(tiles.geom)::float8,
                    2.5),
                2154),
            '8BUI') AS rast
    FROM tiles
),
images AS
(
    SELECT  tile_rast.tid,
            tile_rast.rast AS tile_rast,
            ST_MapAlgebra(
                ST_AddBand(tile_rast.rast, '8BUI'::text), 1,
                ST_Resample(ST_Grayscale(ST_Union(image.rast)), tile_rast.rast, 'bilinear'), 1,
                '[rast2] ', NULL, 'FIRST', '[rast2]'
            ) AS rast

    FROM tile_rast, LATERAL
    (
        SELECT rast
        FROM sat.s2
        WHERE ST_Intersects(s2.rast, tile_rast.rast)

    ) AS image

    GROUP BY tile_rast.rast, tile_rast.tid
  ),
labels AS (
    SELECT  tile_rast.tid,
            ST_MapAlgebra(
                 tile_rast.rast,
                 ST_AsRaster(label.geom, tile_rast.rast, '8BUI'),
                 '([rast2])::integer', NULL, 'FIRST', '([rast2])::integer'
             ) AS rast

    FROM tile_rast, LATERAL
    (
       SELECT ST_ClipByBox2D(ST_Buffer(ST_Union(osm.way), 10),
                             ST_Envelope(tile_rast.rast)) geom
       FROM planet_osm_line osm
       WHERE osm.highway IS NOT NULL AND (osm.route = 'road' OR osm.route IS NULL)
       AND ST_Intersects(osm.way, tile_rast.rast)

       GROUP BY tile_rast.rast, tile_rast.tid

    ) AS label
)
SELECT Box3D(images.rast) AS bbox,
       ST_AsBinary(images.rast) AS data,
       CASE WHEN labels.rast IS NULL
           THEN ST_AsBinary(images.tile_rast)
           ELSE ST_AsBinary(labels.rast)
       END AS label
FROM  labels RIGHT JOIN images ON images.tid = labels.tid
```

```
batch_size = 2
max_iter = 2

geo_iter = GeoIter(
    'postgresql://o:xxx@127.0.0.1:5433/osm_qa',
    (850000,6524040,890960,6565000), 2154, (256, 256), (10, 2.5),

    """
         SELECT ST_ClipByBox2D(ST_Buffer(ST_Union(osm.way), 6),
                               ST_Envelope(tile_rast.rast)) geom
         FROM planet_osm_line osm
         WHERE osm.highway IS NOT NULL AND (osm.route = 'road' OR osm.route IS
NULL)
         AND ST_Intersects(osm.way, tile_rast.rast)
    """,

    """
        SELECT ST_Grayscale(ST_Union(s2.rast)) AS rast
        FROM sat.s2
        WHERE ST_Intersects(s2.rast, tile_rast.rast)
    """,

    batch_size, max_iter)
```


0
100
200
0
100
200
0
100
200
0
100
200
0
100
200
0
100
200
0
100
200
0
100
200


Model Topology
Labeled DataSet
Raw DataSet
Trained Model
Output Prediction
Whole DataSet

MxNet multi GPU (easy) handling

And if (really) needed, multi machines training

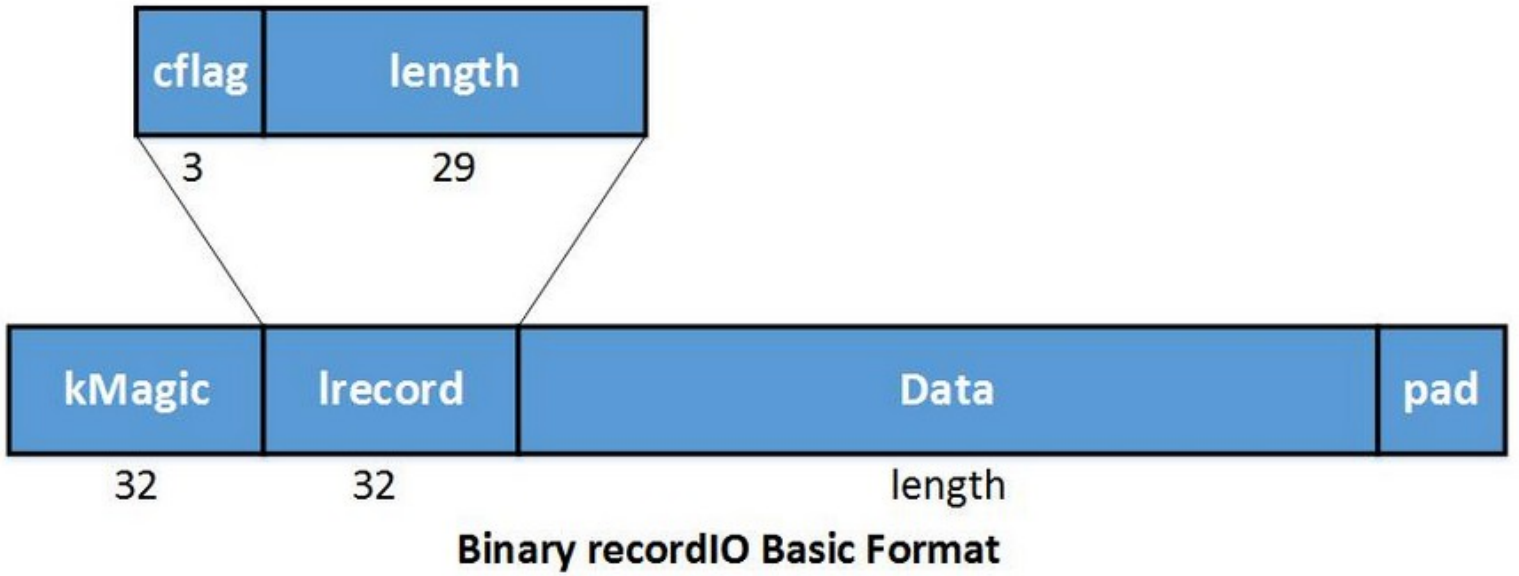
https://mxnet.incubator.apache.org/how_to/multi_devices.html

MxNet RecordIO fast data loader

Data Format

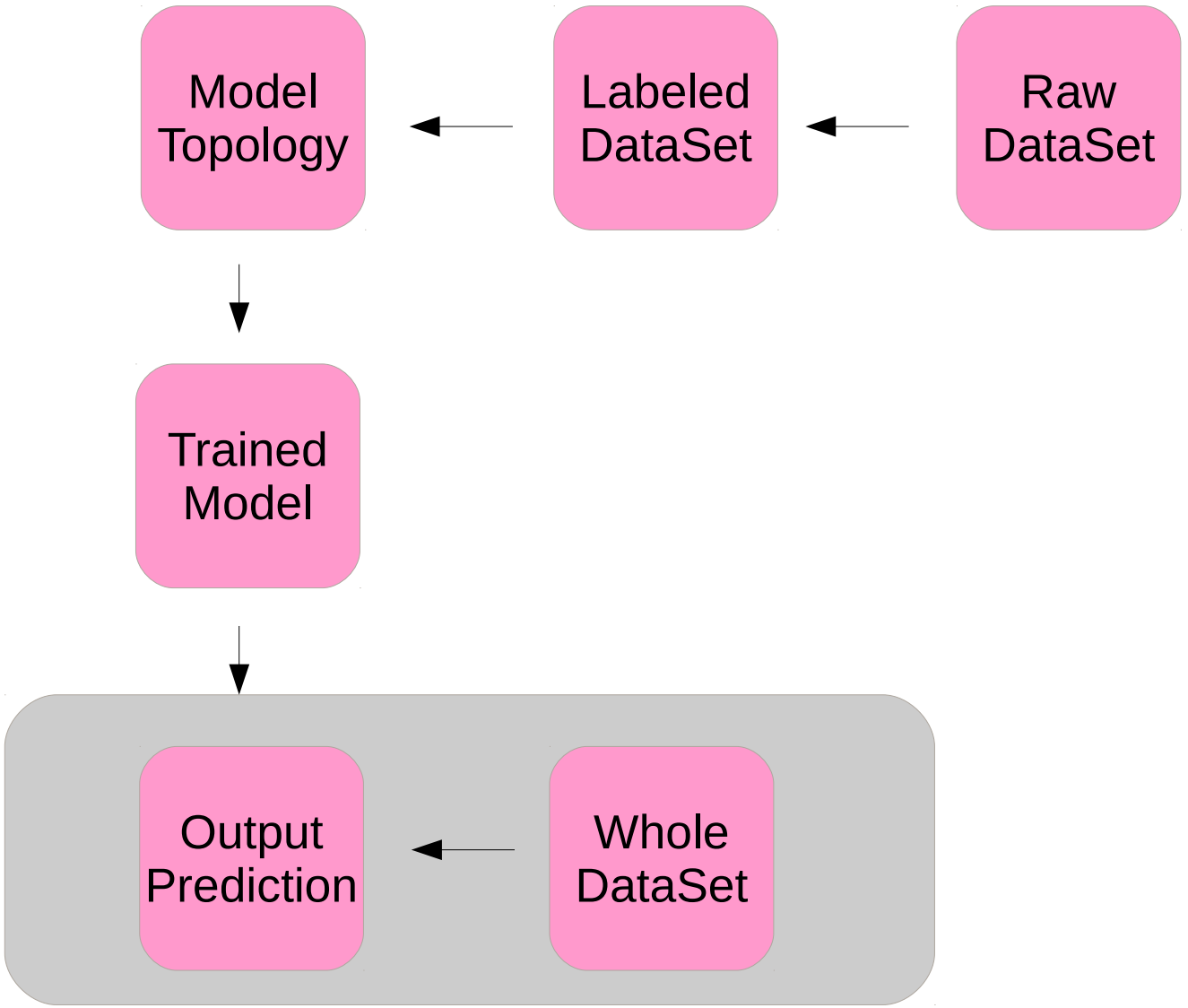
Since the training of deep neural network often involves large amounts of data, the format we choose should be both efficient and convenient. To achieve our goals, we need to pack binary data into a splittable format. In MXNet, we rely on the binary recordIO format implemented in dmlc-core.

Binary Record



Binary recordIO Basic Format

https://mxnet.incubator.apache.org/architecture/note_data_loading.html


Model Topology
Labeled DataSet
Raw DataSet
Trained Model
Output Prediction
Whole DataSet

Could we MapReduce a map ?

Structuration by OpenDataSet

#1 – DIY stage

#2 – Good Training DataSet publicly available

#3 – Efficient PreTrained model publicly available

#4 – Out of the box app

Labelled Datasets

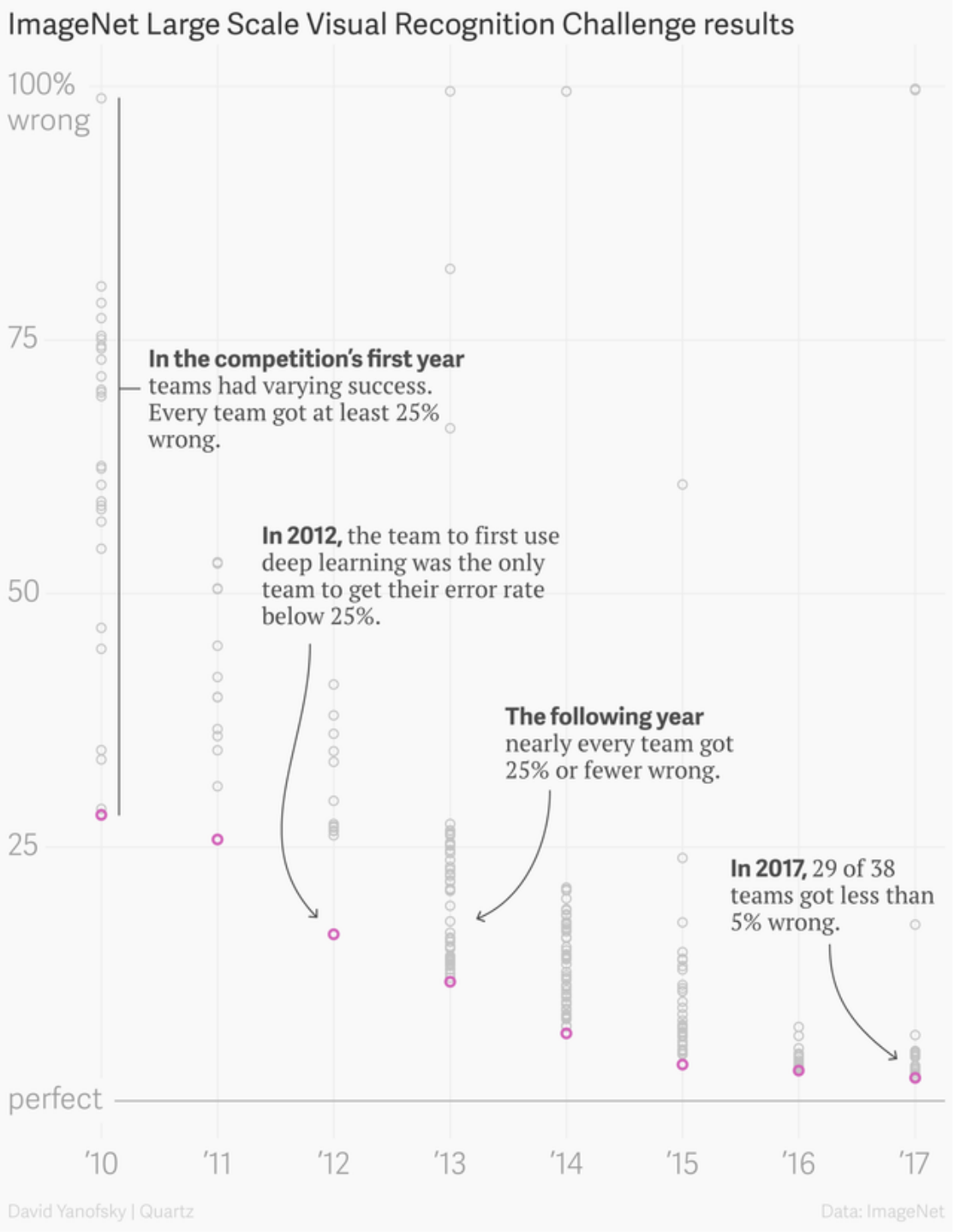
Volodymyr PhD: https://www.cs.toronto.edu/%7Evmnih/data/

SpaceNet: https://aws.amazon.com/public-datasets/spacenet/

ISPRS: http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html
http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html

EuroSAT: https://arxiv.org/pdf/1709.00029.pdf

DeepSAT: http://csc.lsu.edu/%7Esaikat/deepsat/


ImageNet Large Scale Visual Recognition Challenge results
100% wrong
75
50
25
perfect
'10
'11
'12
'13
'14
'15
'16
'17
In the competition's first year teams had varying success. Every team got at least 25% wrong.
In 2012, the team to first use deep learning was the only team to get their error rate below 25%.
The following year nearly every team got 25% or fewer wrong.
In 2017, 29 of 38 teams got less than 5% wrong.
David Yanofsky | Quartz
Data: ImageNet

Next Steps

Lower resolution imagery ability (as Sentinel-2 or PlanetLab)

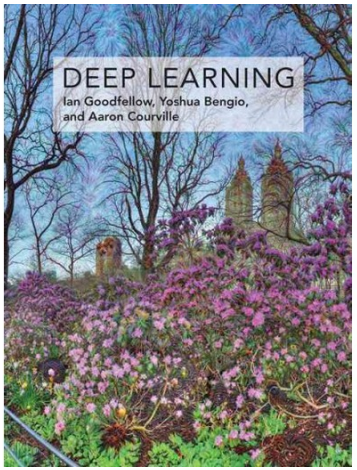
RL

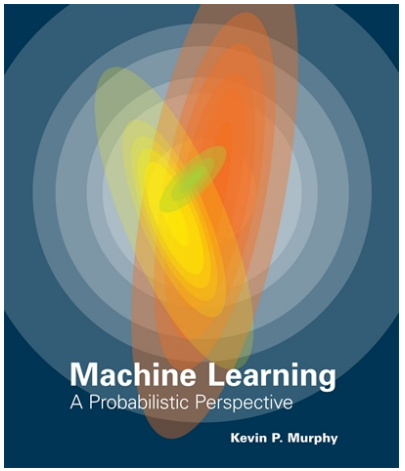
Human Learning

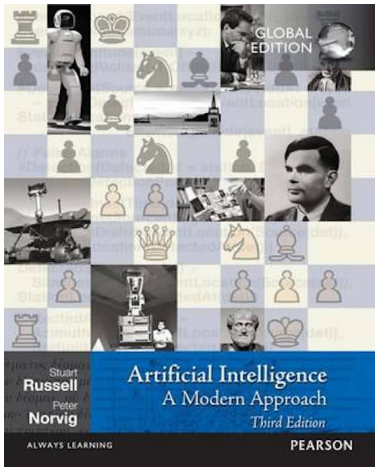
https://www.college-de-france.fr/site/yann-lecun/course-2015-2016.htm

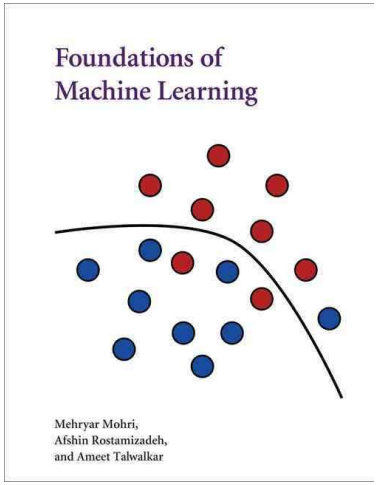
http://cs231n.stanford.edu/syllabus.html

https://raw.githubusercontent.com/mrgloom/Semantic-Segmentation-Evaluation/master/README.md

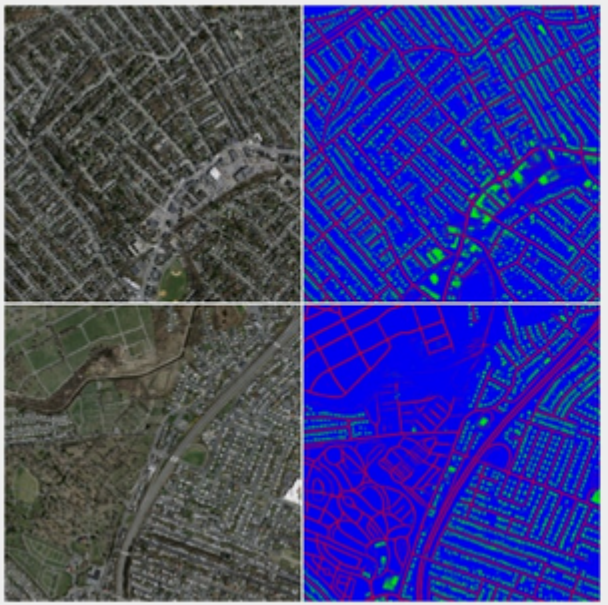








http://crowdsourcing.topcoder.com/spacenet



Mapping Challenge

By Humanity & Inclusion

Starting_soon

8 FOLLOW

https://www.crowdai.org/challenges/mapping-challenge

Conclusions

@data_pink

www.datapink.com