Shape Matching Algorithm for Roof Segmentation in Aerial Images

Duccio PIOVANI, Alexander USOLTSEV, Data Science team, Nam.R, Paris

Introduction

Detecting roofs and the segments that compose them is essential for our work at Nam.R: their position as well as many other characteristics is indeed crucial to our activity. The good news is that satellite and aerial images are found today in great abundance, and thanks to the latest developments in Deep Learning and image recognition, spotting roofs is no longer impossible. The bad news is that one need a lot of labels to train high performance Deep Learning models, and those labels are very rare, and how to construct them is not trivial.

In order to build our own roof segment labels dataset, we have exploited a dataset where the 3D coordinates of buildings are contained, including their roof. But of course aerial images are 2D, and therefore in order to compare the two datasets we had to project the roofs coordinates on the 2D plane, while filtering the rest of the building's shape.

Of course this process comes with its problems and its artefacts. Luckily we also have the coordinates of the area for the building's «footprint» which helped us filter out the problematic labels. In order to that we trained an algorithm to recognize shapes that are similar to each other, and by comparing the roofs' shapes with those of the buildings' footprints we were able to get rid of a great numbers of labels that would have prejudicated our model.

3D data roof data to 2D segments

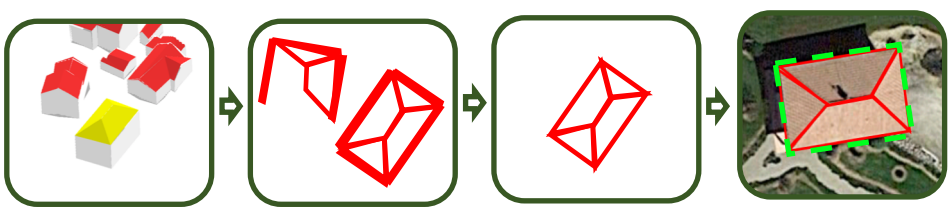
In order to build our own roof segment labels, we have exploited a dataset where the 3D coordinates of buildings are contained, including their roof.
One of the well-know data format to construct 3D models of buildings, cities and landscapes is CityGML format [5].

CityGML is implemented as an XML application schema for the Geography Markup Language which allows to read data with any compatible XML reader. In our case we used Python's standard library The ElementTree XML API to parse our CityGML data source [4].

On the other hand, our aerial images data are in 2D, and therefore in order to use CityGML buildings description we had to project the roofs coordinates from 3D on the 2D plane, while filtering the rest of the building's shape.

Projection conversion comes with its problems and its artefacts. To deal with these artifacts and clean final labels from it, we take advantage of the coordinates of the area for the building's «footprint» we have. We develop Shape Matching algorithm which helped us filter out the problematic labels and match obtained roof segments structure with building's image on the 2D raster.

This process is represented in three major steps below:



Read 3D data → **Project 3D to 2D** → **Filter projection's artifacts** → **Match 2D to building's footprint**

Shape Matching Algorithm

In order to filter the problematic labels, we trained a Shape Matching algorithm to recognize shapes that are similar to each other. Then we applied Shape Matching algorithm to do comparison of the roofs' shapes with buildings' footprints and to get rid of a great numbers of labels that would have prejudicated our model.

We have built an algorithm to give a similarity score to two given polygons, by using information on its geo-localization and its geometrical shape. We have therefore chosen to describe a polygon exploiting a number of geometrical features and mathematical representations

Compactnes $= 4\pi \frac{A_{\text{poly}}}{P}$ **Convexity** $= \frac{A_{\text{poly}}}{A_{\text{convexhull}}}$

where A_{poly} is the area of the polygon, P its perimeter and $A_{\text{convexhull}}$ the area of its convex hull.

Spatial Variance Matrix $= \begin{bmatrix} \sum_k (v_x^k - c_x)^2 & \sum_k (v_x^k - c_x)(v_y^k - c_y) \\ \sum_k (v_x^k - c_x)(v_y^k - c_y) & \sum_k (v_y^k - c_y)^2 \end{bmatrix}$

where (c_x, c_y) are the centroid coordinates and (v_x^k, v_y^k) are the coordinates of the kth vertex. By studying the eigenspace of this matrix we can extract two geometrical features:

- **Orientation Principal Component**: θ_{princ}, tells us the direction of major spatial variation.
- **Eccentricity**: λ_2/λ_1, the ratio between the norm of the principal (λ_1) and second(λ_2) component.

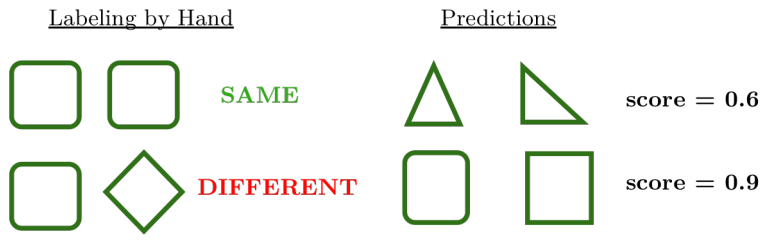
Circular Variance

$$C_v = \sum_k^N (d_k - \bar{d})$$

the variance on the distance of the vertexes to the center (0 for a circle) where $\bar{d}$ is the average distance, N number of vertexes and $d_k = \sqrt{(v_x^k - c_x)^2 + (v_y^k - c_y)^2}$ the distance of the $k-th$ vertex.

This allows us to describe a polygon as a set of numbers.

We have then manually labelled $2*10^4$ polygons by hand, equal or different, and trained a random forest algorithm on the set, to give a similarity score. This consists in the probability of the two polygons to be the same one.

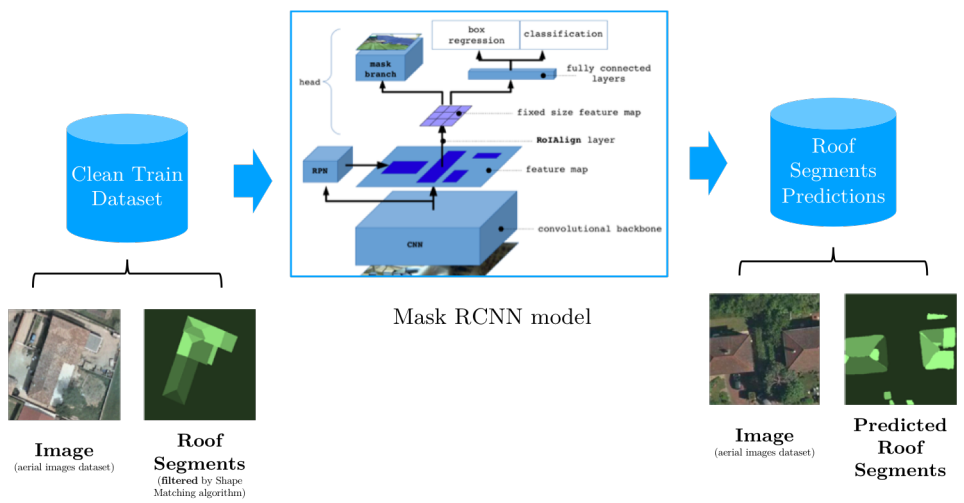


Based on this predictions we are then capable of filtering the shapes for which the projections went for wrong.

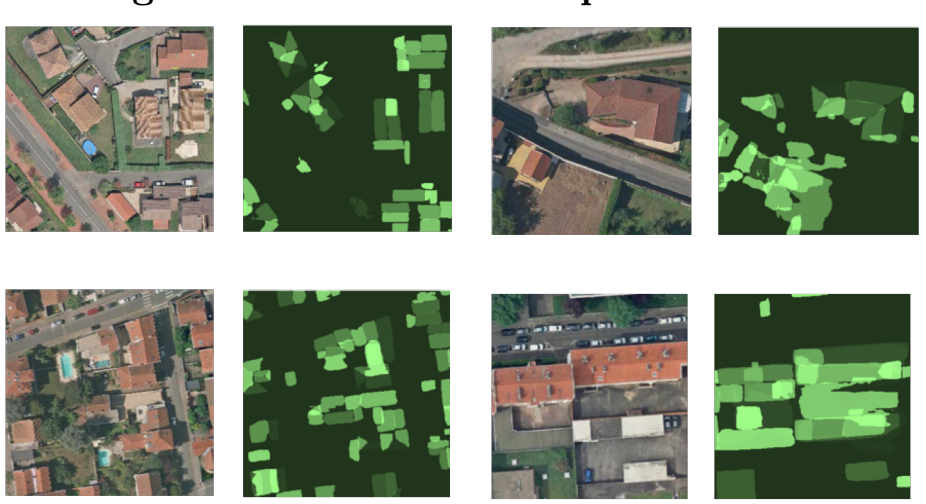
Training Deep Learning model to detect roof segments

As a segmentation model we chose current state-of-the-art Mask RCNN algorithm.
According to the authors of Mask R-CNN, it is a "conceptually simple: underlying object detector has two outputs for each candidate object, a class label and a bounding-box offset; Mask-RCNN add to this a third branch that outputs the object mask. But the additional mask output is distinct from the class and box outputs, requiring extraction of much finer spatial layout of an object. So Mask-RCNN introduce the pixel-to-pixel alignment" [1].

Now when we have labels produced by robust Shape Matching algorithm, we converted polygons into binary masks and prepare a training data consisting of pairs of aerial image and object masks. This training data were fed to the segmentation Deep Learning model which tries to predict correct masks for desired objects. Due to variety of roof's materials, shapes and light conditions that kind of automatic segmentation could be quite challenging. But thanks to big enough volume of labeled data we achieved satisfying performance.



Roof segments detection examples



High quality predicted segments
(low number of false positive segments)

Noisy predicted segments
(high number of false positive segments)

References

[1] Kaiming He and Georgia Gkioxari and Piotr Dollár and Ross B. Girshick, Mask R-CNN, 2017
[2] https://github.com/matterport/Mask_RCNN, Mask R-CNN for Object Detection and Segmentation, 2017
[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: Common objects in context. In ECCV, 2014
[4] https://data.grandlyon.com/imagerie/, Data Grand Lyon, 2015
[5] http://www.opengeospatial.org/standards/citygml , CityGML XML-based format, 2008