Pointing only for remote sensing images

Adrien Chan-Hon-Tong

Pointing only for remote sensing images

Adrien CHAN-HON-TONG

December 14, 2017

Abstract

There are a lot of works aiming to reduce the need of human annotations: self supervised training, weakly supervised training, interactive verification instead of annotation.

Here, we show on several public dataset that, in remote sensing context, pointing objects i.e. giving the centre of each object can be sufficient for learning a segment before pointing pipeline. Resulting pipeline also reaches honourable score on segmentation.

1 Introduction

1.1 Segment before detect

Since the publication of [11], deep learning is more and more becoming a common tool. High quality deep learning engines like PYTORCH and TENSORFLOW offer simple and efficient way to run a network on GPU and/or on the cloud. In this context, given a specific computer vision problem, designing an end to end deep networks allows to simplify the code and to get an automatic GPU acceleration.

Thus, end to end networks are useful even with slightly lower accuracy/quality that the state of the art for the given problem. As an example, [4] learn end to end network for low level computer vision operator accelerating these low level operation thank to automatic GPU implementation.

Now, some problems are more likely to accept an end to end network. For example, object detection does not trivially accept one. Object detection is one of the oldest computer vision problem, and consists in producing a list of bounding boxes of all class instances from an image (see figure 1). So, the output of a detector is highly unstructured and inhomogeneous: unstructured because expected to be a list whose size is unknown, inhomogeneous because numerical representation of bounding boxes lead either to mix coordinate and scale or either to accept numerical pair of coordinates with large variances. Indeed, state of the art deep networks designed for object detections introduce a lot of non standard operations: pre computation of candidates [8], ROI pooling trick [16] or anchor trick [15].

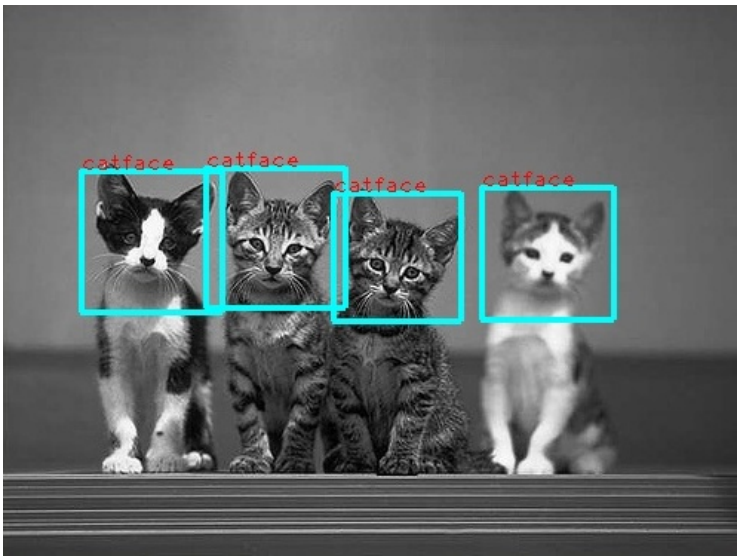

Figure 1: Illustration of object detection: putting bounding boxes on object in the image

Inversely, semantic segmentation is as suitable as classification for end to end deep networks. Semantic segmentation consists in predicting a class for each pixel (or pixel cells of an image). So, output of semantic segmentation pipeline is a known size map with one channel per class (corresponding value can be seen as likelihood of belonging to each class). This is highly structured and homogeneous, allowing very straightforward end to end deep network like [2, 18]

More interesting, in case of remote sensing images, an accurate segmentation mask can be converted into a detection output by considering the connected components: in a remote sensing images, there is only few perspective effects, so an object will rarely occulted an other - this way, all instances will be in different connected components. This one scale property is also the case for lot of medical images including cytology, but in medical images biological objects can merge into complex biological tissues. Inversely, in remote sensing image objects do not overlap each other (e.g. cars do not overlap even on a parking).

Thus, in remote sensing images, segment to detect is a rising paradigm whose an example can be found in [1]. We can also stress that, even on natural image, connexion between detection and segmentation leads to state of the art methods [9].

1.2 Ground truth issue

Now, producing a semantic segmentation ground truth is so time consuming that large semantic segmentation datasets may only appear for market

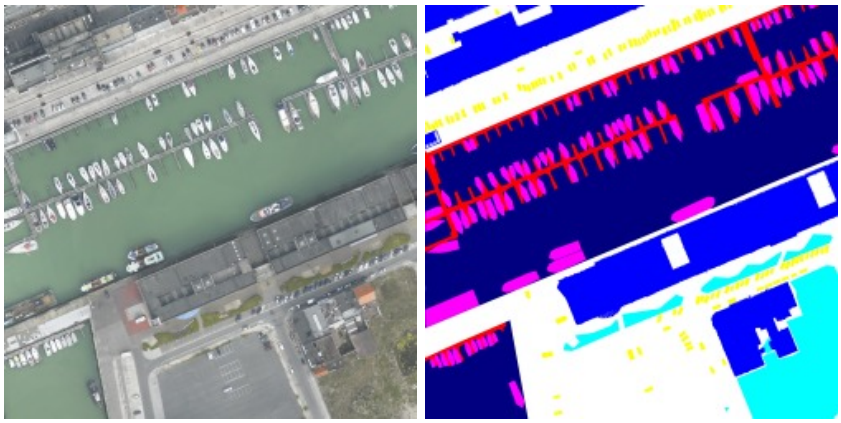

Figure 2: Illustration of semantic segmentation: give a class to each pixel

friend applications (e.g. autonomous driving with CITYSCAPE [5]). In it much more time consuming than just pointing objects i.e. giving the object centres (see figure 3).

In fact, even detection ground truth is time consuming: validating a box is twice more fast than drawing it in [20]. More, [20] does not take into account that drawing boxes leads to much more complex human computer interface than clicking point. Base on private little experiments with disconnected softwares, we currently think that drawing boxes is close to 20 times longer than just pointing objects.

With this order of magnitude in mind, we argue that it is relevant to aim to learn deep segmentation pipeline from pointing ground truth (centres of objects) instead of detection ground truth (boxes of objects) or, off course, segmentation ground truth (masks of objects).

Alternatively, there are lot of works aiming to reduce the need of human annotations: 0 shot learning [3], self supervised training [6] or interactive verification instead of annotation [20, 13]. However, current non supervised trainings have today lower accuracy than supervised one. Also, interactive verification needs a very large software infrastructure to become possible and [13] currently does not really demonstrate that it decreases the human time needed to annotate. It just shows that it decreases the complexity of what is asked to human.

Inversely, our contribution is to offer a (quite simple) trick to learn deep segmentation pipeline with pointing dataset only, in remote sensing context.

This way, we build on the top of the [1] but removing the need for an expensive semantic segmentation ground truth.

Some tricks are presented in the next section. Then, in section 3, we present datasets and results, before conclusion of section 4.

Figure 3: Illustration of pointing dataset: ground truth is the set of object centres

2 Learning from centres

Baseline: Natural segmentation training for convolutional neural networks consists to compare the output mask to the ground truth mask, like if each pixel was a classification data, averaging the loss over all the pixels [2, 18].

Typically, in classification, input of the network is a tensor x with size $B \times 3 \times W \times H$ (batch size, 3 for RGB and $W \times H$ for the spatial sizes of the images), and the output is a tensor p with size $B \times C$ (C being the number of classes). The corresponding ground truth is a vector of size B containing values from $\{0, ..., C-1\}$. Then, the corresponding loss is typically: $loss = \sum_{b=1}^{B} -p[y[b]] + \log(\sum_{c=1}^{C} \exp(p[c]))$. From this loss, derivatives (corresponding to network weights) are computed and weights are updated typically according to stochastic gradient descent paradigm.

In segmentation, the output tensor p has a typical size of $B \times C \times W \times H$ (one likelihood per class and per pixel) but ground truth also has a size $B \times W \times H$ (with values from $\{0, ..., C-1\}$).

So, the loss can be simply computed by seeing both those tensors like classification tensors with batch size being $B \times W \times H$ instead of B: $loss = \sum_{b=1,w=1,h=1}^{B,W,H} -p[y[b]][h][w] + \log(\sum_{c=1}^{C} \exp(p[c][h][w]))$.

Now, such natural way is completely not able to handle centre mask i.e. mask with one foreground pixel per centres. Indeed, such mask is then well to much unbalanced.

At least a bootstrapping procedure for selecting hard negative should be considered (see [7] as an example of old school classifier with sliding windows fashion requiring bootstrapping). However, such procedure is quite iterative and complex. Instead, we offer here 3 tricks to make those mask not so difficult to train with.

This natural way to learn a segmentation deep pipeline will be called *multi classes* or *binary* or *raw training* in the following depending on the ground truth used to train (original mask for *multi classes*, binary mask for *binary* and centre mask for *raw*).

Squared: The first trick is simply to fill a square of predefined size centred on each object centre. Then, training is done as usual.

The main problem with this trick is that pipeline will have to learn from noisy ground truth: background pixel close to a car will be tagged as foreground. But, balance will be globally the same than in the binary mask while needing only object centres.

Inner: The second trick is highly inspired from [10]. [10] shows the relevancy of removing pixels close to a boundary of the ground truth segmentation. Removing border pixel is at first glance the opposite of bootstrapping. However, pixel close to a boundary are more *ambigious* than *hard* (typically, pixel close to boundary are likely to be wrongly annotated (see figure 2 because manual annotation is coarse due to the difficulty to realize it)). In other word, hard example are clear errors from the current detector while boundaries examples are disputable errors inherent to the problem. Indeed, [10] offer to remove pixel from boundaries during training as it increases performance during testing even if boundaries are kept at this stage. This result is not trivial as breaking the symmetry between training and testing (remove border from train but not test) is usually considered as a bad practice. Authors from [10] argues that by removing ambiguous pixel, segmentation pipeline is able to learn better and thus removing large errors localized far from border by not focusing on small border errors.

This strategy will be called *inner training* in the following. The training is done by removing from the $B \times W \times H$ batch the pixel which are not considered: $loss = \sum_{b,w,h \in allowed} -p[y[b]][h][w] + \log(\sum_{c=1}^{C} \exp(p[c][h][w]))$. In practice, this is done by adding a dumb class with class weight equal 0.

Keep the pooling: The idea of making the learning easier to increase performances from [10] leads to an other trick: keeping a pooled segmentation. This trick consists in decomposing the image in cell which correspond to the pixel which will exist in the networks after all the pooling layer. Then, a cell is considered as foreground as soon as it contain an object centre.

Currently, since the apparition of VGG [21], large amount of works have been published to try to restore spatial dimension (dilated convolution, unpooling with index [2], ...). This spatial restoration is natural for pure segmentation. Now, in pointing context, restoring spatial resolution is quite are as we only know centre from each object.

This is why we offer this quite simple trick which will be shown to be quite efficient. Indeed, we argue that it provides 2 advantages: it reduces the unbalance by dividing background instance by the cell size, and then,

it allows to have a simpler training very close to the network structure.

This last trick will be called *pooled training* and consists formally to use the following loss:

$$loss = \sum_{b=1,w=1,h=1}^{B,W',H'} -p[y'[b]][h][w] + \log(\sum_{c=1}^{C} \exp(p[c][h][w]))$$

with W', H' being the size of the image after pooling (typically $H' = \frac{H}{16}$ and y' is the pooled ground truth.

3 Experiments

3.1 Datasets

We use four datasets in our benchmark: the data fusion contest 2015 [12] (DFC2015), the ISPRS POSTDAM dataset [19] (POSTDAM), the VEDAI dataset [14], and we also rely on SACLAY[1] which is a private dataset (but planned to be released).

DFC2015 dataset [2] provides 6 10000x10000 ortho images at very high resolution (5cm). [12] provides a ground truth for pixelwise semantic segmentation in 7 classes including vegetation, building, road, car, boat. Here, we downscale image to 5120x5120 (thus resolution is around 10cm). We split train and test according to [12].

POSTDAM dataset [3] is very similar to DFC2015. It provides 38 ortho images at high resolution (between 5 and 10 cm). As for DFC2015, we also downscale each 6000x6000 image to 4096x4096.

VEDAI dataset is a detection dataset. Ground truth is composed of bounding boxes of vehicles. The dataset is composed of around 100 1024x1024 orthoimages at 12.5 cm of resolution. We do not use the label associated to the bounding boxes.

Finally, SACLAY is a private dataset formed with IGN ortho images, annotated for car pointing. The dataset contains around 20 5000x5000 pixel image at 20cm of resolution (we crop the 4608x4608 corresponding image).

3.2 Dataset conversion

All datasets are converted into vehicle pointing datasets (ground truth is a set of centres of vehicle).

For VEDAI, centres are either extracted from ground truth boxes. All boxes are considered as vehicles (car, truck, farming vehicle, plane: all are considered as vehicle).

For POSTDAM, ISPRS, we extracted connected component of vehicles and compute the centres. Thus, we lost the multi class aspect: we only keep background (all except vehicle vs foreground vehicle).

This label loss is important as it decreases performance. We compare multi classes segmentation (as in [1]) vs binary segmentation in table 1. Currently, we will show in the later that training with pointing may lead

[1] We thank Adle Koeniguer for providing the ground truth for SACLAY

[2] www.grss-ieee.org/.../data-fusion/2015-ieee-grss-data-fusion-contest/

[3] http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html

dataset	*multi classes*	*binary*
DFC2015	-	41
POSTDAM	75	49

Table 1: Comparision of multi classes segmentation versus binary segmentation.

Performance are in Gscore in %.

to the same performance than training with binary segmentation, but both are lower than training with multi class segmentation. The question about why having multi classes helps the classifier is out of the scope of this paper, but, such kind of results are not new (see [17]).

For the unknown and/or mono scale trick, we use either 16 or 32 pixels as border. Thus, for the unknown trick, the centres are tagged as foreground, the 16 or 32 pixels around are tagged as unknown and all other are tagged as background. For the one scale trick, the 16 or 32 pixels around are tagged as foreground and the other as background.

At test time, each pipelines produces a segmentation mask which is post processed to produce a set of points. Post processing is classical erosion dilatation with kernel 7, 9 or 11 and extraction of the centres of connected components.

Predicted points and ground truth points are matched like boxes in detection (criterion for matching is a euclidean distance less than 25 pixels). This leads to correct matches, false alarms and miss detections, and then to precision recall. Then, we form the Gscore (product of precision and recall).

3.3 Result

All pipelines are trained for segmentation in a UNET fashion [18] except for the pooled training where we only use VGG [21].

The main result of this experiment is all training except *pooled training* do not manage to deal correctly with pointing ground truth. Indeed, the gap between performance between *pooled training* and other training is large. Table 2 presents Gscore of all trainings.

An other results is that when matching *pooled training* mask against the original mask, we reach only slightly lower pixelwise performance than the original *raw training*. More precisely, we obtain much more coarse mask from spatial point of view, but, we catch more foreground connected component and have less false foreground connected component. Results are in table 3.

4 Conclusion

We offer a simple trick to learn segmentation pipelines from pointing datasets (ground truth is just the list of object centres). This trick is to forward the pointing mask go through the pooling layer of the network

dataset	*raw*	*square*	*inner*	*pooled*
DFC2015	0	20	16	41
POSTDAM	0	19	14	49
VEDAI	0	9	10	33
SACLAY	0	11	12	34

Table 2: Gscore of training strategies under pointing metric for fcn architecture.

Results are Gscore in %. Datasets are converted into pointing dataset: we keep the centres of each bounding boxes or connected components. *raw* consist to learn directly the centres vs background masks. *square* consist to learn a square centred on the centres vs background masks. *inner* consist to learn the centres vs background masks but removing the border area. *pooled* consist to learn grid quantified mask.

dataset	*binary*	*pooled*
DFC2015	29	29
POSTDAM	45	41

Table 3: Evaluation of segmentation mask learnt from segmentation or centres.

Performance are in Gscore in % (here we count the number of pixels not the number of objects).

in order to reduce unbalance and to help the network to learn something relevant. The pipeline trained with our trick outperforms pipeline trained with other tricks on pointing task.

More our pipeline reaches nearly equivalent performance of semantic segmentation than the original pipeline (the one train on the original binary mask) while trained with centres instead of mask.

These results offer to use only pointing instead of boxes or masks when running out of time to make a dataset ground truth.

Acknoledgment

We thank Adle Koeniguer for providing the ground truth of SACLAY.

References

[1] Nicolas Audebert, Bertrand Le Saux, and Sbastien Lefvre. Segment-before-detect: Vehicle detection and classification through semantic segmentation of aerial images. *Remote Sensing*, 9(4):368, Apr 2017.

[2] Vijay Badrinarayanan, Ankur Handa, and Roberto Cipolla. Segnet a deep convolutional encoder decoder architecture for robust semantic pixelwise labelling. In *arXiv preprint*, 2015.

[3] Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. Improving semantic embedding consistency by metric learning for zero-shot classiffication. In *European Conference on Computer Vision*, 2016.

[4] Qifeng Chen, Jia Xu, and Vladlen Koltun. Fast image processing with fully-convolutional networks. In *IEEE International Conference on Computer Vision*, 2017.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[6] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[9] Kaiming He, Georgia Gkioxari, Piotr Dollr, and Ross B. Girshick. mask r-cnn. In *IEEE International Conference on Computer Vision*, 2017.

[10] Mingyuan Jiu, Christian Wolf, Graham Taylor, and Atilla Baskurt. Human body part estimation from depth images via spatially-constrained deep learning. *Pattern Recognition Letters*, 50:122–129, 2014.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] Adrien Lagrange, Bertrand Le Saux, Anne Beaupere, Alexandre Boulch, Adrien Chan-Hon-Tong, Stéphane Herbin, Hicham Randrianarivo, and Marin Ferecatu. Benchmarking classification of earth-observation data: from learning explicit features to convolutional networks. In *International Geoscience and Remote Sensing Symposium*, 2015.

[13] Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. We don't need no bounding-boxes: Training object class detectors using only human verification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[14] Sébastien Razakarivony and Frédéric Jurie. Vehicle Detection in Aerial Imagery : A small target detection benchmark. *Journal of Visual Communication and Image Representation, Elsevier*, 2015.

[15] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. corr abs/1612.08242 (2016), 2016.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015.

[17] Marko Ristin, Juergen Gall, Matthieu Guillaumin, and Luc Van Gool. From categories to subcategories: large-scale image classification with partial class label refinement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 231–239, 2015.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. Springer International Publishing, 2015.

[19] F. Rottensteiner, G. Sohn, J. Jung, M. Gerke, C. Baillard, S. Benitez, and U. Breitkopf. The isprs benchmark on urban object classification and 3d building reconstruction. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2012.

[20] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Computer Vision and Pattern Recognition*, 2015.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *CoRR*, 2014.