Deep learning pipeline for grid classification, segmentation, detection and related problems

Adrien Chan-Hon-Tong

Deep learning pipeline for grid classification, segmentation, detection and related problems

Adrien CHAN-HON-TONG

October 18, 2017

Abstract

We focus on a cluster of computer vision problems containing segmentation, detection and related variants.

We benchmark three deep learning pipelines, each trained following a variante of segmentation, on five problems from the cluster. This benchmark is interesting to anyone looking for a deep learning pipeline to deal with multi tasks and/or who has not a clear idea of metric underlying the desired task.

In our current results, grid segmentation is the most robust training strategy despite being both the simplest and fastest strategies. Such result may interest the computer vision community if confirmed on larger experiments.

1 Introduction

Detection and segmentation are well defined computer vision problems. Now, those problems still come with possible variants. Variant can be the same problem with different metric or slightely different version of the same problem. Typically in detection, there is a widely consensus for the jacard ratio of 0.5 as criterion to accept a possible matching between a prediction and a ground truth box (jacard ratio is intersection over union ratio). But, it is known that changing this ratio may even lead to different ranking of a set of detectors (given the same dataset): one of the lastest example of this statement is [6] where different box proposal algorithms can rank differently following the benchmark rules. Worse, human appreciation of the metric can be problem dependant: a camera control system will need centred detection while a counting system will just need coarse detection. So, there is not way to argue for a variant or an other. Off course, as academics, we need common problem in order to compare each other so we usually take the detection problem *as is*. Now, for anyone who will have to deal with multi tasks and/or who has not a precise idea of the metric underlying the desired task, it can be interesting to evaluate a common pipeline under different problem.

In addition, if this benchmark limitation is not new, it may become critical with the rise of deep learning: due to the end to end nature of

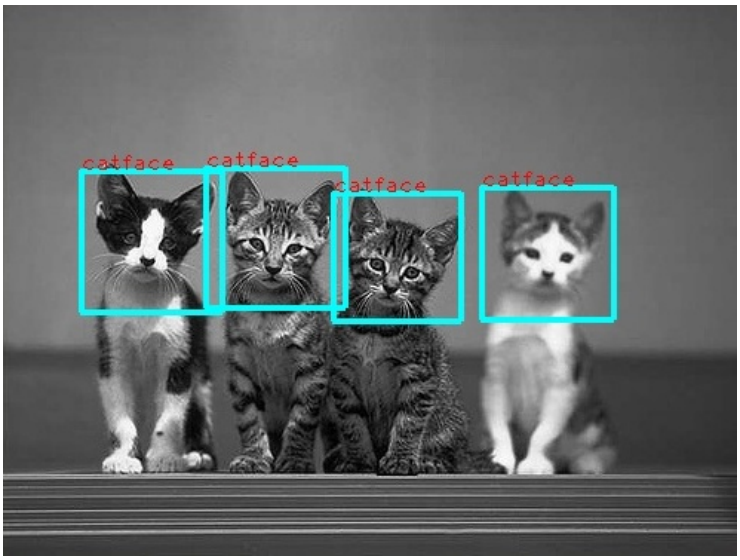

Figure 1: Illustration of object detection: putting bounding boxes on object in the image

deep learning pipeline, we argue that deep learning can be very sensitive to the selected variant. Indeed, when the pipeline is just one network, the loss on the output is the only thing the network know about the problem on which is optimized.

Our contribution is to establish this benchmark of 3 training strategies measured for 5 differents problems on 4 differents public datasets. More precisely, we are especially interested by the variants of detections and segmentations. Then, we compute the *average* performance of deep learning pipeline trained under serveral variants of segmentation and tested under these variants of segmentation and detection.

After a brief review of related work in next section, we describe in section 3 the different variants and links between variants. Then, we describe the results on 4 public datasets before concluding in section 4 and 5.

2 Related works

When the deep learning [8] appears in classification, object detections community does not directly follow the move: even an efficient implementation of [8] could not processed more that 100 small images per second. As the dominant framework for object detection was based on sliding windows processing only 100 windows per second was definitely not enough. At this time, a typical detector was [4]. In [4], hand crafted features are extracted from all the images (train and test). Then a learning problem is created by putting in one side features extracted from ground truth boxes

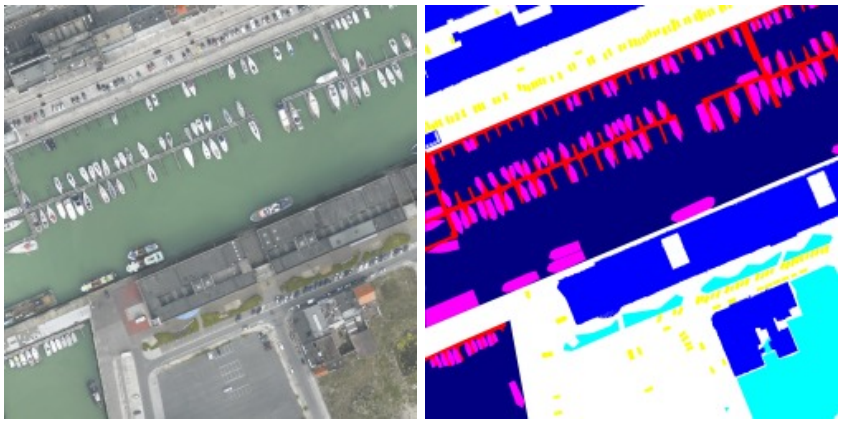

Figure 2: Illustration of semantic segmentation: give a class to each pixel

and on the other side features extracted from random boxes which do not match ground truth. This problem is solved by svm (with an implementation close to [3]). At test time, the image is explored by sliding windows, the svm model is applied on the features of each windows. The set of windows selected by the svm is then pruned by non maximal suppression (svm will behave likely on two very close windows but by keeping the two boxes somes will likely become false alarmes). Finaly, the learning step can be done multiple times to look for hard negative instead of random ones (this is the so called bootstrapping method).

The real apparition of deep learning in detection was with [5] which both highly increases performance of the previous state of the art and on the same time breaks the sliding windows framework by showing that high performance are possible with region proposal framework. In [5], an ad hoc algorithm extract bounding box in the image folding the detection problem into a classification one without exploring all the boxes of the sliding way. Classification in [5] is then done by [8]. Current state of the art of detections [11] are still based on [5] but the adhoc region proposal is now a deep network based on the same first layers than the classifier.

Also, as convolution can take image from arbitrary spatial dimension, first layers of the network are computed on the complete image and not on windows, fastening the first layers computation by the size of the windows compared to the classical way. This last idea of convolution taking arbitrary image gives rise to a lot of semantic segmentation works like [2, 12] (or [9] for remote sensing). Semantic segmentation (see 2) is the goal of producing semantic mask of an images. Or, with others words, the goal of deciding a semantic label for each pixel of an image.

Optimizing deep learning is straightforward in semantic segmentation: output is a highly structured image with predefined size and a number of channels corresponding to the number of type of objects, on which error with the desired manual annotation can be easily measured. Also, on dense semantic segmentation, there are often multiple labels with large

number of instances. Thus, small object pixel are unbalanced toward surface pixels (e.g. car vs building in 2), but there are multiples surface labels forbidding the deep network to produce a mono label map. Recently [1] shows that this semantic segmentation way can be postraited to produce accurate detection.

Now, for semantic segmentation, [7] shows the relevancy of decreasing the weight given to pixels close to a boundary of the ground truth segmentation. At first view, [7] seems to offer the opposite of bootstrapping: bootstrapping consists in looking for hard negative while [7] offer to discard pixel close to a boundary which are typically hard. 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 discutable errors inherent to the problem.

Again, considering boundaries or not may depend of the finality of the algorithm. If the goal is to produce a precise map from an image, having a pixelwise segmentation is required, and in this case, border pixel have to be considered, otherwise the segmentation will be like coarse and unsuitable for making a map. But if the goal is to produce coarse segmentation as a first step toward detection, it is much more important to focus on non border hard negatives (which will create false detections) and not on border hard negatives (which will likely have absolutely no effect).

Currently, our contribution is strongly inspired from [7]: the goal is to benchmark different strategies to learn deep learning pipeline. Each strategie corresponds to a segmentation related problem. The evaluation consists to evaluate the model under 5 *metric*, corresponding to 5 problems related to detection and segmentation.

In the next section, we will formalize these problems/strategies.

3 Segmentation, detection and variants

We define here the set of variants we consider in following experiments.

3.1 Variants

3.1.1 Segmentation

The goal of segmentation is to classify each pixel with a class from a dictionnary. Ground truth is the expected class for each pixel. In our experiment, we have only two classes (foreground background)

There is straitforward strategy to train a deep network in segmentation. Network should output a probability map with the exact same size as the input image (1 probability per class from the dictionnary). Error between prediction and ground truth is given by a cross entropy loss averaged accross all pixels. In a way, this is a regulated version of accuracy. Backpropagation of the error is done classically.

3.1.2 Coarse segmentation

The goal of coarse segmentation is to classify each pixel with a class like segmentation. Ground truth is the expected class for each pixel like segmentation. But, there is a class unknown in the ground truth. Pixels from the unknown class are not taken into account either at train or test stages. Pipelines are not allowed to classify pixel as unknown - unknown is only present in ground truth.

Training is done exactly like for segmentation with a 0 weight for unknown pixel.

3.1.3 Grid segmentation

The goal of grid segmentation is to classify each bloc of pixels with a class. Bloc size is an input of the problem. Ground truth is the expected class for each bloc.

Typically, bloc size is 16. Image size should be multiple of bloc size.

Training is done exactly like for segmentation each that the output of the network is expected to be a probability map with the same size as the number of blocs.

3.1.4 Detection

The goal of detection is to produce a list of area containing objects of interest. Ground truth is a list of objects: typically a list of bounding boxes.

Pipelines should produce a list of boxes. A predicted box is allowed to match a ground truth boxes if intersection over union is higher than 0.5. Then, a maximal matching is computed to know the number of correct matchs, false detections, miss detections.

In our experiment, we do not directly train for detection - we only evaluate list of boxes from probability map.

3.1.5 Pointing

The goal of pointing is to produce a list of area containing objects of interest without taking in account the spatial extention (only the center are expected). Ground truth is a list of points.

Pipelines should produce a list of points. A predicted point is allowed to match a ground truth points if distance between point is lower than a fixed threshold. Then, a maximal matching is computed to know the number of correct matchs, false detections, miss detections.

In our experiment, we do not directly train for points - we only evaluate list of boxes from probability map.

3.2 Ground truth conversion

Given a dataset annotated for segmentation, it is possible to convert it into a grid segmentation dataset.

We describe here the possible conversions and the conversions possible under assumption.

- Pointing can be infer from detection by taken the center of each box.
- In segmentation, borders are always more ambiguious. So if unknown is allowed, it is naturally to form a coarse segmentation dataset from a segmentation dataset by copying the segmentation after marking unknown all the border between two classes.
- Grid segmentation can be infer from pointing by taking all bloc containing at least one center.
- Pointing can be infered from coarse segmentation by taking the center of connected componant. The underlying idea is that even if two objects are close, in the coarse mask the instance will be separated. This assumption is verify in our experiments.
- Coarse segmentation can be infer from detection by setting the center of each box as foreground, the box as unknown and the outside box as background.

Infering detection from segmentation is not always possible if multiple instance of objects overlap. Under the assumption that objects are sparse and thus not overlapping, we can infer detection ground truth from segmentation ground truth by computing the connected components of each foreground area.

3.3 Prediction conversion

Ground truth convertion are expected to be perfect (because otherwise this is not ground truth anymore).

This is why there is not possible conversion from detection to segmentation but only from detection to coarse segmentation and/or from pointing to grid segmentation and/or from coarse segmentation to pointing.

Now, we are talking about simple post processing needed to convert with more or less faireness prediction from a modality to prediction from other modality. Again, here there will be mistake but this is **needed** to evaluate a mask from grid segmentation in detection setting...

In the following experiments, we choose these simple post processings:

- From grid segmentation, we infer other segmentation map by simple resizing.
- From segmentation, we infer detection by taking the connected component **after** a set of morphological operation (we choose erosion + dilatation).

We use the same operation that the one on the ground truth for other convertion.

Now, we are able to train a pipeline for either segmentation, coarse segmentation and grid segmentation). And from the output of each of these pipeline, we have fixed post processing allowing to produce prediction for all five considered problems (detection, pointing and the three segmentations.

So, we can benchmark each training strategy (three strategies) versus five evaluation settings. This is done on the next section.

4 Experiments

4.1 Datasets

We use 4 datasets for our benchmark: the data fusion contest 2015 [9] (DFC2015), the isprs postdam dataset [13] (POSTDAM), the VEDAI dataset [10], and we also rely 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). [9] provides a ground truth for pixelwise semantic segmentation in 7 classes including vegetation, building, road, car, boat. Here, we downsample image to 5120x5120 (thus resolution is around 10cm). We also binarize the ground truth between car and all other classes. As target are far from each other segmentation ground truth can be converted into detection ground truth as detailled in previous section. We split train and test according to [9].

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 both downsample each 6000x6000 image to 4096x4096 and make the ground truth binary.

VEDAI dataset is a detection dataset. Ground truth is composed of bounding boxes of vehicules. The dataset is composed of around 100 1024x1024 ortho images at 12.5 cm of resolution. We do not use the label associated to the bouding boxes (car, truck, farming vehicule, plane all are considered as vehicule).

Finally, SACLAY is a private dataset formed with IGN ortho images, annotated for car pointing (we just have 1 pixel per car - see previous section). The dataset containts around 20 5000x5000 pixel image at 20cm of resolution (we crop the 4608x4608 corresponding image).

4.2 Results

All results are presented in tables 1, 2, 3 and 4.

For all measurements, we compute the gscore (times 100). The gscore is the product of precision and recall. Precision is the number of *positif predicted as positif* over *predicted positif*. Recall is the number of *positif predicted as positif* over *positif*.

This is important to remark that this metric is not the natural one for segmentation: the natural one should be accuracy (*correctly classified pixel* over *pixel*). Indeed, in accuracy, each training strategies is still better than the two others on the corresponding problem (which is quite expected even if not trivial). But, inversely, there is no simple way to train in order to maximize the gscore. Using the gscore instead of accuracy makes sense to remove negative bias: when there are much more negatif than positif, predict that everything is negative lead to high accuracy.

[1]We want to thank Adle Koeniguer for providing the pointing ground truth of Saclay dataset

[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

evaluation	pixelwise	coarse	grid
detection	2	?	0
pointing	26	?	20
segmentation	29	?	19
coarse segmentation	31	?	15
grid segmentation	57	?	10

Table 1: Results on DFC2015 dataset.
Results are gscore (times 100) - pixelwise corresponds to the pipeline trained on segmentation - coarse to the same pipeline trained for coarse segmentation and grid for this pipeline trained for grid segmentation. See previous section for detail about all different problem.
Results on coarse setting will be available soon.

evaluation	pixelwise	coarse	grid
detection	8	0	5
pointing	30	49	44
segmentation	29	36	45
coarse segmentation	41	56	51
grid segmentation	79	25	28

Table 2: Results on POSTDAM dataset.
Results are gscore (times 100) - pixelwise corresponds to the pipeline trained on segmentation - coarse to the same pipeline trained for coarse segmentation and grid for this pipeline trained for grid segmentation. See previous section for detail about all different problem.

evaluation	coarse	grid
detection	?	?
pointing	?	?
coarse segmentation	?	?
grid segmentation	?	?

Table 3: Results on VEDAI dataset.
Results are gscore (times 100) - pixelwise corresponds to the pipeline trained on segmentation - coarse to the same pipeline trained for coarse segmentation and grid for this pipeline trained for grid segmentation. See previous section for detail about all different problem.
No segmentation is provided for VEDAI so we can not evaluate neither test segmentation.

evaluation	coarse	grid
pointing	0	34
coarse segmentation	8	12
grid segmentation	16	47

Table 4: Results on SACLAY dataset.
Results are gscore (times 100) - pixelwise corresponds to the pipeline trained on segmentation - coarse to the same pipeline trained for coarse segmentation and grid for this pipeline trained for grid segmentation. See previous section for detail about all different problem.
No segmentation and dectection is provided for VEDAI so we can not evaluate neither test segmentation and detection.

In our opinion, the most important observed trend over these experiments is that grid segmentation outperforms on average both segmentation and coarse segmentation. Grid segmentation is sometimes even better than segmentation (resp. coarse segmentation) on segmentation (resp. coarse segmentation) with gscore metric (this is not the case in accuracy). In contrary, grid segmentation based detection (resp. pointing is much more efficient than segmentation based detection (resp. pointing) and coarse segmentation based pointing.

Off course, some complementary experiments could have been done to strengthen the evaluation:

- we evaluate only one network VGG16 cut at conv5_3 for grid segmentation, an segnet like for coarse segmentation or segmentation (which is not the state of the art anymore)
- we evaluate only simple post processing (see previous section: resizing for transforming grid segmentation to other segmentation - morpho + connected component for transforming from mask to box)
- we could evaluate multi objective networks or other variants
- it should always be better to add additionnal datasets

However, these experiments have nevertheless been done on two public datasets (+ one private). Results on VEDAI will be available soon. Partial results on VEDAI confirm the trend. So, we are confident to state that the trend we observe is that grid segmentation is better than segmentation and coarse segmentation for handeling multi tasks or when the exact objective is not well defined.

5 Conclusion

We focus on five close computer vision problems: grid segmentation, coarse segmentation, segmentation, objects detection and object pointing.

We train deep learning network (strongly based on VGG16) for grid segmentation, coarse segmentation and segmentation. Output of the network is directly comparable to ground truth for these three problems.

Then, we evaluate these three networks for all the five problems using simple post processing to transform the native outputs into each expected outputs. This benchmark is done on three public datasets (plus one private - result are not complete on the third public dataset).

The observed trend is that grid segmentation outperforms both coarse segmentation training and segmentation training on *average* over the five problems. This is especially interesting as grid segmentation training is both the fastest and the simplest. Also, annotation can be done much more easily for grid segmentation.

This result could be very interesting for the computer vision community (if confirmed with more complex deep learning pipelines and eventually on larger datasets).

Acknoledgment

We want to thank Adle Koeniguer for providing the pointing ground truth of Saclay dataset.

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] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[4] 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.

[5] 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.

[6] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. How good are detection proposals, really? In *Proceedings of the British Machine Vision Conference*, 2014.

[7] 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.

[8] 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.

[9] 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.

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

[11] 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.

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

[13] 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.