Benchmarking losses for deep learning laxness detection

Adrien Chan-Hon-Tong

Benchmarking losses for deep learning laxness detection

Adrien CHAN-HON-TONG

July 31, 2017

Abstract

In object detection, classical rule for accepting a match between a ground truth area and a detection is a 0.5 jacard ratio. But does user care about this ? Especially, can this rule be well handled by deep network training ? And if no, may user accept some relaxation of the problem if it can help the training ?

In this paper, we benchmark several strategies to perform object detection with end to end deep network when metric is relaxed. Our preliminary results on different public datasets show that under this relaxations, some strategies are very facilitative to train the network outperforming the same network learned with classical strategies.

1 Introduction

Interactions between deep learning users and deep learning designers usually go through one classical academic problem. However, users are not aware of the difficulties introduced by this problem which may be harder than really needed, and inversely, deep learning designers are not aware of the exact needs of the users and are not well able to explain what could be relaxed and how it could be facilitative. Off course, we acknoledge that, as academics, we need common problem in order to compare each other. But, when it came to adapt deep network to some real life problems, it may be relevant to relax the classical problem.

In this article, we apply this statement to object detection in image (see figure 1), which is one of oldest computer vision problem. For academic purpose, object detection is a well defined problem. Each testing image is annotated by a set of bounding boxes (eventually labelled) which form the ground truth. Detector should take one image as input and product a set of boxes (eventually labelled and eventually with confidence level). A detection is allowed to match an annotation if intersection over union of the two boxes is greater than 0.5 (and if label are the same when present). A maximization of the number of matchs is computed. Then, each non matched annotation is a miss detection and each non matched prediction is a false alarm. Classical measure of the fairness of the detector is given by combining the precision (number of correct detections over

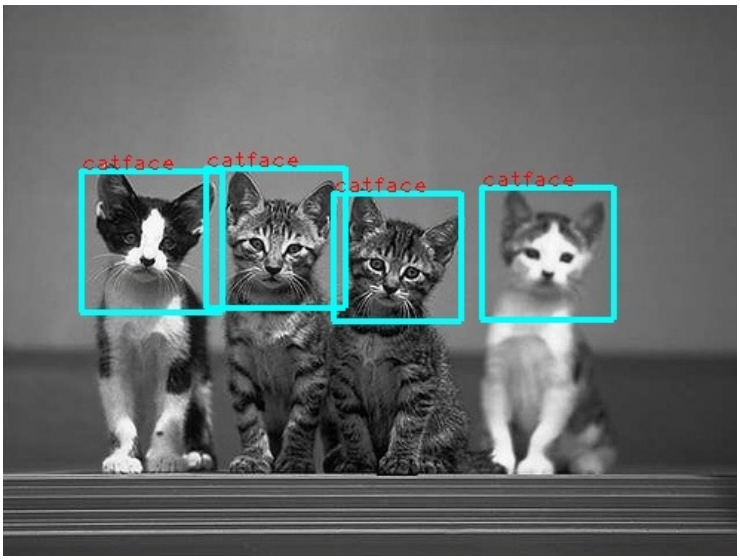


Figure 1: Illustration of object detection: putting bounding boxes on object in the image

number of detections) and the recall (number a correct detections over number of annotations), for example with the so called F score or G score. Alternatively, detection are sorted according to the confidence level, and, a elementary score is computed for each rank. All this elemantary score are merged into a global score (for example by considering that elemantary scores form a curve and by computing the area under the curve).

If, for example, the detector is applied to count cars on a parking this definition of the problem is completely relevant typically because 1 to 1 matching is needed to produce an accurate estimation of the number of cars. But, there are a lot of application where this strict definition is not relevant. For example, if detections should be checked by an human anyway, the only matter is to produce coarse area of interest recovering the objects (because both fine spatial extention estimation and 1 to 1 matching are useless: typically multiple detections of a common object may not be an issue).

These considerations are not new, and, at first side, one just need to do little ajustement on a detector to adapt it to a new metric.

This is probably true for hand crafted detector pipeline where only small blocks rely on training because changing the metric will have low influence on hand crafted block and/or because depending of the metric some block will be changed but without affecting the behaviour of previous/following blocks. But, on end to end deep learning network, metric is the only given prior on the problem, thus changing the metric (i.e changing the network loss) may have a critical influence on the fairness of the detector.

The contribution of this paper is indeed to show that losses do have a critical influence on the performance at least for end to end deep learning detectors.

More precisely, we focus on object dectection but under a relaxed metric (classical metric is replaced by a more user oriented ones). We show that similar network using same training data can behave highly differently on testing data depending on the way we model this detection problem. Our preliminary results show that modeling the problem as a simple grid classification is very facilitative for network training probably because it match the pooling structure of the network.

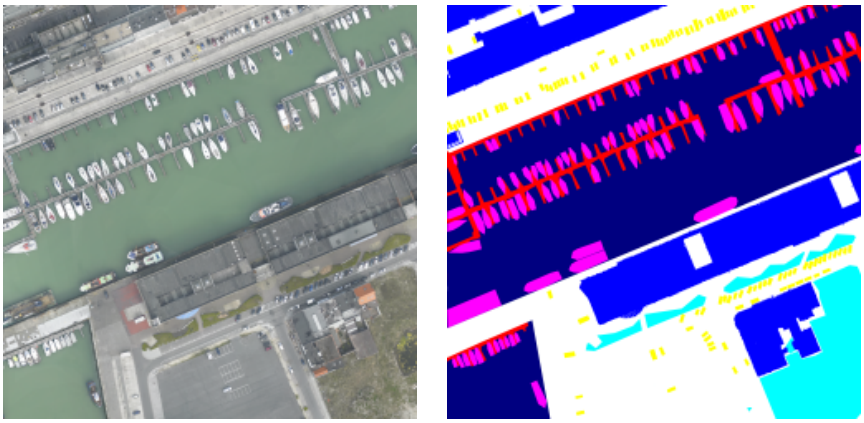
In the next section, we will present related work, then we will present the models we selected for the benchmarking. Section 4 will present different datasets and results before the section 5 conclusion.

2 Related works

When the deep learning [7] appear in classification, object detections community does not directly follow the move: even an efficient implementation of [7] 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 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 [7]. 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 [8] for remote sensing). Semantic segmentation (see 2) is the goal of producing semantic mask of



(a) a remote sensing images (b) a semantic mask

Each color in the semantic mask is expected to correspond to a kind of object in the image. For example, boats in the image are in pink in the semantic mask while cars are yellow.

Figure 2: Example of semantic segmentation of a remote sensing image

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, [6] shows the relevancy of decreasing the weight given to pixels close to a boundary of the ground truth segmentation. At first view, [6] seems to offer the opposite of bootstrapping: bootstrapping consists in looking for hard negative while [6] 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, in [6], testing performances are measured with strict metric but still increase whereas training is done like if metric was borderless. So, in this case, the increase is only due to a relaxation that help to learn and not just because the metric is less strict. This is a critical point: off course, performances automatically increase when relaxing the metric but some relaxation of the metric may allow a completely different way to handle the problem which can provide a performance gap and not just a small automatic increase. Idealy this performance gap could be visible even when testing with the strict metric. Now, in our case, as the relaxation may even permit to change the output format of the pipeline, such evaluation will not always be possible.

So, after this fast review of related works, there are at least three strategies to handle detection: classifier based, segmentation bases and borderless segmentation based. In the next section, we will formalize the our soft metric and it interaction with all different strategies for detection and describe experiment protocols.

3 Different models for laxness detections

3.1 Laxness detections

In this paper we focus on producing an *output* that will be postraited by a person to finally lead to detections. Thus, the goal is to produce something that help to detect most targets while minimizing the time to be post processed. The clear use case of such detector is to help to process very large remote sensing images (e.g. car detections on an area of 20km x 20km acquired per a pleiade satellite corresponds to detection of 10x10 object into a 40000x40000 images).

Given some user recommandation, we decide to evaluate the performance of the detector with a very different metric that the classical one. The detector output is expected to be a set of points. The user is expected to look around each of these points/alarms. Thus, missing targets are annotation far from any of these points while false alarms are points far for any annotation. However, the number of missing targets has less sense than in classical detection because several close targets correspond to only one missing alarm. Also, the ratio of the true alarm divided by the total alarms could be completely biased (because by adding alarm around an easy target, one can make this ratio arbitrarity close to 1). Also two alarms close to a common hard negative should count as one false alarm, not 2. Thus, performance is not directly computed between alarms and annotations but between alarms and *optimal* alarm output. Given the ground truth (which can be either center of object or mask or bounding boxes), one can easily approximate the optimal alarm allocations. So, in our experiment, soft recall will correspond to the number of alarms from the optimal allocations that have to be added to cover all targets divided by the number of alarms in this optimal allocation. And, soft precision will correspond to the number of pixels covered by non missing alarms

from the optimal allocation divided by the number of pixels covered by all alarms (this way having two redondant alarms will not significatively decrease precision), all divided by the area of alarms (precision is thus almost invariant to spatial resolution).

We will describe formally the selected metric. Let assume that the user is able to visualize a dxd image easily (in other words, visualizing a dxd image centred around an alarm is fast and natural). A pixel q will be covered by an alarm p if $\left\|p_k^* - p_j^*\right\| \leq d$. Let $p_1^*, ..., p_{N^*}^*$ be the target positions (these position can be infered from or bounding boxes or masks if ground truth is more then just center annotations - the number of targets may be lower if ground truth is a mask in which several targets are too close to have their own connected component in pixel but seeing our metric this should not be an issues). The optimal allocation of alarms can be approximated by the greedy algorithm leading to $p_1, ..., p_N$:

1. if there are no targets left return
2. compute $i = \arg\max_j \left|\left\{k, \left\|p_k^* - p_j^*\right\| \leq d\right\}\right|$
3. add p_i^* to the alarm list and remove covered targets.
4. go to 1

See annexe for discussion about this greedy algorithm. Now, let $q_1, ..., q_M$ be a set of predicted alarms. We compute L the alarms required from $p_1, ..., p_N$ to complete $q_1, ..., q_M$ in order to recover all targets with the following algorithm:

1. remove all targets covered by $q_1, ..., q_M$
2. if there are no targets left return
3. compute i the first index such that $\exists j, \left\|p_i - p_j^*\right\| \leq d$
4. add p_i to the missing alarms and remove targets covered by p_i
5. go to 2

In our experiment, we define the soft recall as $\frac{N-|L|}{N}$. Now, let $C^+ = \{q, \exists i \in \{1, ..., N\} - L, \|p_i - q\| \leq d\}$ (the set of pixels covered by the non missing alarm of the *optimal* allocation) and $C = \{q, \exists i, \|q_i - q\| \leq d\}$ (the set of pixels covered by the produced alarms). We define the soft precision as $\frac{|C^+|}{|C|d^2}$.

In our opinion, these two definitions soft recall and soft precision keep the spirit of recall and precision while adding a kind of clustering of both targets and predictions: 2 close targets correspond to 1 missing alarm as they will be covered by 1 alarm from the optimal allocation - 2 close false alarms will count almost like 1 false alarm as they will cover the same area. Also, using the optimal allocation allows to bypass bias issue: for example dividing by M instead of N in soft recall has no sense as one can add multiple alarms around an easy targets.

Given this new rules to evaluate the output, several strategies are possible to deal with this laxness detection problem.

3.2 Classical approach

In R-CNN, a first algorithm is applied to produce candidate. The precision of this first algorithm is expected to be very low and the main requirement for such algorithm is to be training free. Candidates are pruned by non maximal suppression.

A training time, candidates are matched with the ground truth and a classifier (a deep network) is learnt on the positive/negative (positive are candidate which match the ground truth - negative are candidate which not match).

At testing time, the classifier is applied on all candidate and accepted candidates form the detections.

3.3 Segmentation

In segment before detect, we convert the bounding boxes annotations into a binary mask (background for pixel outside all boxes and foreground for pixel inside one or more boxes).

At training time, we learn a fully convolutionnal deep network in segmentation fashion (we want each pixel be well classified).

At test time, we produce detection by putting a bounding box around each connected component when using strict detection rules.

3.4 Coarse segmentation

The third model to handle detection is like segment before detect but instead of producing a binary mask by labelling background all pixels outside all boxes and foreground all pixels inside one or more boxes, we produce a trinary mask: all pixels outside all boxes are background, all centers of all boxes are foreground and all other pixels are border.

When training border pixels are discarded from the optimization, thus the network only has to predict correctly the center of object and not the ground truth bounding boxes of the object.

3.5 Pooled detection

The last model to handle detection is the simpler and can only handle laxness detection: we explore the image by spaced sliding windows (with step corresponding for example to the average size of objects), each window containing an object should be classified as positive and each window containing no object should be classified as negative.

This can be seen as a segmentation pooled by a factor corresponding to the selected sliding step. This can also be seen as a R-CNN with R outputing a tilding of the image.

At testing time, we just output the center of the windows accepted as positive.

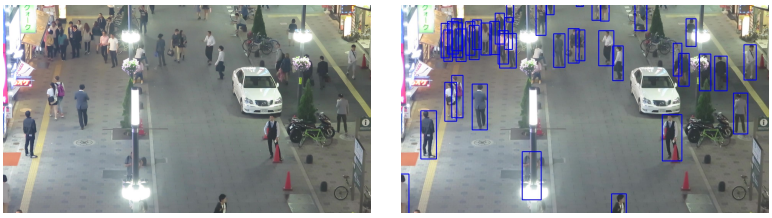


Figure 3: Illustration of the MOT 16 dataset.

4 Experiments

4.1 Datasets

We use 3 datasets to evaluate the influence of these different strategies for detection: MOT 2016 (MOT16) [9], VIRAT aerial dataset [10] (not video surveillance data - we use only the aerial videos). As no annotation has been released, we annotate it ourselves using our tool (annotations are planned to be released). We also rely on SACLAY which is a private dataset (but planned to be released).

MOT16 (https://motchallenge.net) is a multi objects tracking dataset: detection are provided, the goal is to keep temporal id on the detection. Here, we use this dataset only to learn to produce a semantic mask corresponding to the detections (we only keep person detection). The training part of MOT16 is composed of 6 HD videos taken from a pedestrian or car or surveillance camera (see figure 3).

The VIRAT aerial dataset is a set of videos which are low resolution and contain camera motions, highly textured background and very small object. As no public annotation has been released for this dataset, we annotated a subset of the frames in a person detection setting. In order to provide a diversity of situations, we chose to annotate about thirty sub videos of 400 frames containing at least one person distributed over the dataset (but discarded infrared images). Figure 4 shows examples of images from this sub dataset.

The SACLAY dataset is a set of aerial images at 20cm of resolution. Annotation is a set of point localized on cars.

4.2 Results

Currently, we have only preliminary results computed on a subset of VIRAT in table.

However, from this preliminary results, we have noticed that coarse segmentation perform poorly while pooled segmentation outperforms the others. As a matter of fact, coarse segmentation increase the unbalance (only 1 pixel positive per object all other are either discarded or background) which is not the case for pooled (1 cell positive per object when their are distant but the number of negative cell is globally the number of pixel divided by the size of the cell).

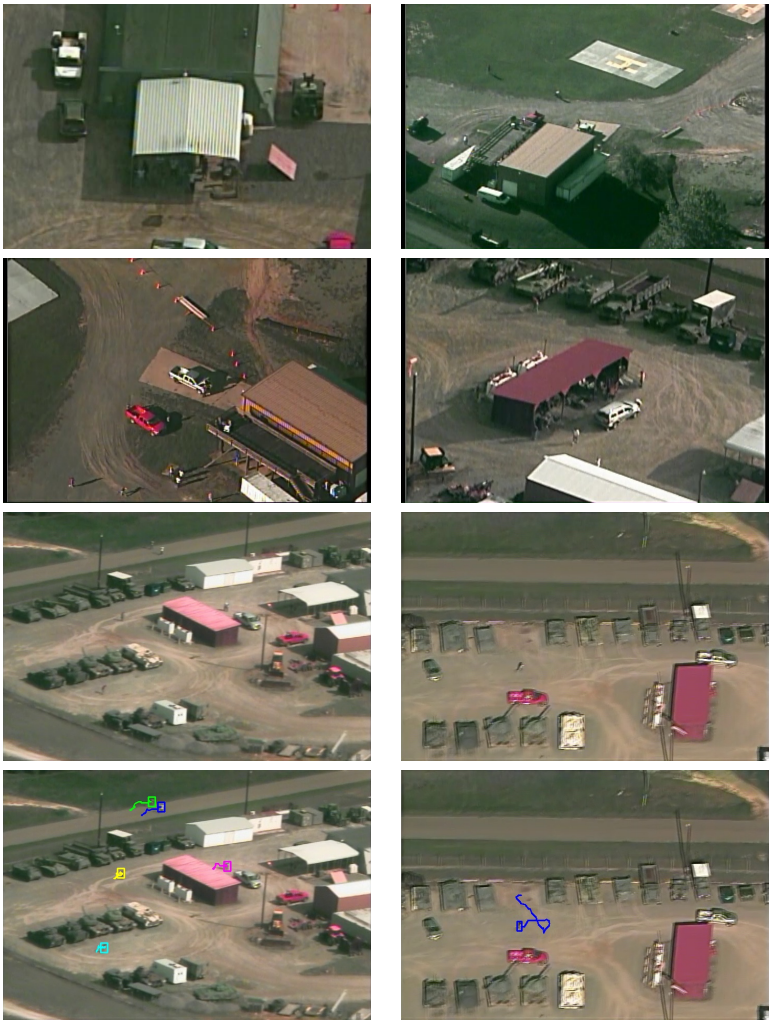


Figure 4: Illustration of the subset of VIRAT (plus annotation).

5 Partial conclusion

Even if our benchmarking is currently done on a small part of public datasets, the current trends seems clear: losses used to learning deep learning detectors have a dramatic influence over the results.

This invite to strengthen our current benchmark as the persistence of the current features will be an interesting knowledge for the community.

Acknoledgment

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

Annex

Optimal allocation

Given a number $d > 0$ and a set of points $p_1^*, ..., p_{N^*}^*$, computing the minimal set of points $p_1, ..., p_N$ such that $\forall i, \exists j, \|p_i^* - p_j\|_\infty \leq d$ is close to classical clustering and optimization problem (typically k-means). If we add the constraint that $p_1, ..., p_N$ should be a subset of $p_1^*, ..., p_{N^*}^*$, then it is a sub problem of vertex cover problem. Because, we can considered the graph (V, E) with V be $\{1, ..., N\}$ and $\forall i, j \in V, ij \in E \Leftrightarrow \left\|p_i^* - p_j^*\right\|_\infty \leq d$.

This two statements do not provide any proof that optimal allocation is NP-complet (because for example the set of vertex cover instance corresponding to an optimal allocation instance may be restricted to a subclass for which there are P algorithm). However, from computer vision point of view, having to solve such combinatorial problem (which may be NP-complet even if not proved here) is a clear issue. This is why we choose to rely on a simple greedy approximation (see 3.1).

This is not hard to imagine case in which this greedy approximation will be far from the optimal one. However, we evaluate both this greedy approximation and the exact two solutions (depending if alarm should be on a input point or not) on a part of VIRAT dataset (see 4.1). Exact solutions were computed almost by brute force: we modelize the two problems as MILP (with big M trick when alarms are not requested to be input points) and use MILP solver (lpsolve). Thus, computation has been very intensive even for a small number of points. Yet, the tree solutions were completely equivalent. This result argues that our greedy approximation is sufficient for our use case.

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] 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.

[7] 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.

[8] 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.

[9] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *CoRR*, abs/1603.00831, 2016.

[10] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *conference on Computer Vision and Pattern Recognition*, 2011.

[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.