Capstone Proposal

Roberto Jiménez Sánchez
February 25th, 2020

Domain background

The project is framed within the computer vision field, in the object detection domain. Specifically, this project deals with object detection from satellite images. There is a growing number of satellites orbiting around Earth and in some cases, their mission is to take pictures of Earth's surface. The applications for algorithms of object detection for satellites are many; the example in particular that inspired this project comes from Ayush et al. (2020), that use satellite images to characterize poverty in a country by region. In another line of work, Biermann et al. (2020) use satellite images to detect plastic debris in the ocean.

Problem statement

To the best of the author's knowledge, object detection in itself does not seek to solve a particular problem. But many problems can be solved by an algorithm capable to perform object detection on satellite data. Going back to the example of poverty, as mentioned by Ayush et al. (2020), a common problem when dealing with poverty is that the data is very scarce. When it is available for a region, it might not be updated periodically. Thus, it is difficult to assess the performance of the policies that have been put in place to mitigate poverty. Ayush et al. (2020) propose using satellite Earth images to fill this gap.

Dataset

The dataset that will be used is the one provided by xView challenge. This dataset is composed of satellite images containing different types of objects at different regions in Earth and covers 1400km^2 of surface. 847 images are available to the public, with 0.3m resolution, containing over 1 million objects. The objects are distributed over 8 parent classes which are divided further into 60 child classes. As an example, a parent class is "Passenger Vehicle" and the associated child classes are "Bus" and "Small car". The topology of parent classes is provided here below:

- Fixed-Wing Aircraft
- Passenger Vehicle
- Truck
- Railway Vehicle
- Maritime Vessel
- Engineering Vehicle
- Building
- None

The “None” class contains classes with no common parent, as for example “Shipping Container” or “Vehicle Lot”, For further detail of the child classes, the reader is referred to the xView challenge website or to Lam *et al.* (2018).

The dataset images are roughly 3300pixels^2 in size. The objects vary in size from 3 meters to greater than 3000 meters (so 10 to 10000 pixels respectively). The dataset is split into train, validation and test sets. Only the train and validation sets are available to the public, and the validation set has no labels or bounding boxes provided to the public. However, the challenge provides the possibility to evaluate a model over the validation set on the challenge website, by submitting a docker container. An image example (cropped and resized to smaller size) is provided below.

Figure 1 Sample image from the dataset (the image has been cropped and reduced in size)

This dataset allows training a model to detect objects from satellite images, which can be used as a first step on the process to produce poverty maps, as detailed in the next section “Solution Statement”.

Solution statement

To obtain poverty maps, Ayush *et al.* 2020 follow this process: first, they train a model that can perform object identification from satellite images. From the objects identified by the satellite images, a classwise object count can be obtained for any satellite image. These classwise object counts are added by regions, and this regional information with the objects is used to train a new model which has to predict the poverty index of that region. In order to predict this index, the model is fed with new data obtained from household surveys. In this capstone project, only the first step of the process will be completed, the rest are considered out of the scope: train a model that can perform object identification from satellite images.

Benchmark model

xView challenge provides a baseline model based on TensorFlow, which details can be found in Lam *et al.* (2018). The model uses the Single Shot Multibox Detector meta-architecture (SSD). The model is evaluated recurring to the whole set of images. To train the model, the images are fed in smaller sizes; three approaches are used: (i) dividing the images into images of 300px^2 (Vanilla dataset); (ii) using a multi-resolution dataset, which creates image chips of different sizes (300px^2, 400px^2 and 500px^2, Multires dataset) and (iii) an augmented dataset, which uses the multi-resolution dataset and adds image augmentation (shifting, rotation, noise and blurring) (Augmented dataset).

The results are evaluated using 30% of the data by recurring to the mAP metric (which is detailed in the evaluation section below):

	Vanilla dataset	Multires dataset	Augmented dataset
mAP	0.1456	0.2590	0.1549

Another benchmark model has been found in Medium, by Fong, 2018. In this case, the model used is YOLOv3, and the mAP score is 0.085. For this second benchmark only the train images were used to train and evaluate the model, as will be the case for this capstone project, and it might be more appropriate for comparison.

Evaluation

The model will be evaluated with the metrics provided at the xView challenge, which is the mAP (mean Average Precision). The mAP is the mean over classes of the AP for each class (Henderson and Ferrari, 2016). The AP of a class is given by the area under the precision/recall curve for the detections. An example of such a curve and how it is constructed is provided in Figure 2.

The steps provided following are obtained from Henderson and Ferrari, 2016, and completed with the Medium article "mAP (mean Average Precision) for Object Detection" by Jonathan Hui. To understand the curve, it is first necessary to define the overlap between two boundaries, which in this project will be defined by the IoU, the Intersection over Union. The IoU of two boundaries is simply defined as:

$$IoU = (area\ of\ intersection)/(area\ of\ union)$$

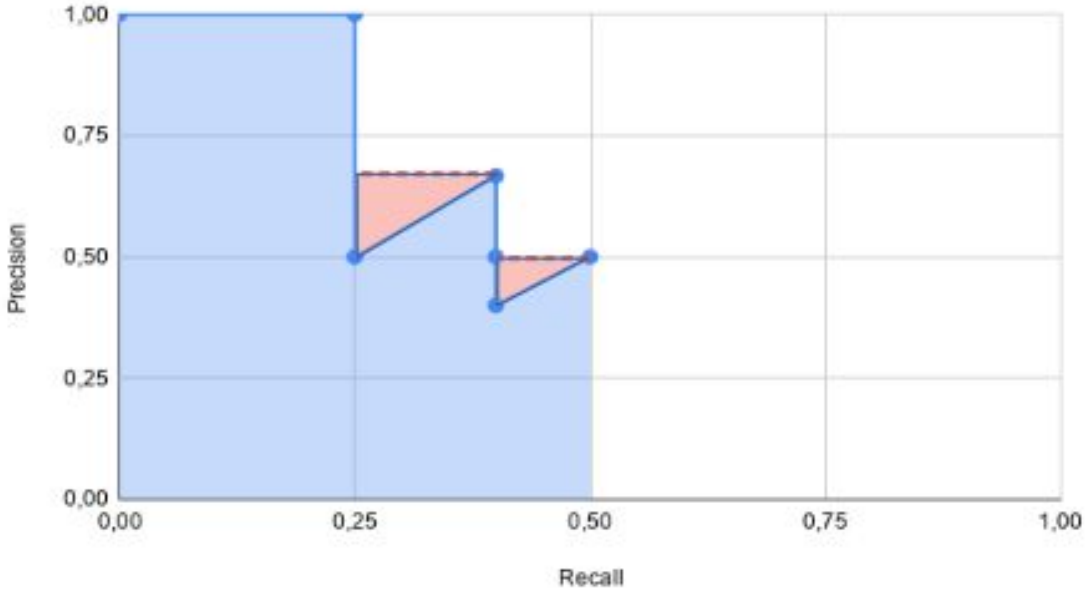

Figure 2 Example of precision/recall curve

To construct the PR curve the following steps need to be followed (we are working with a single class at this stage):

1. Map each detection provided by the algorithm to its most overlapping ground truth instance, as long as the overlap is larger than 50% IoU.
2. The highest scored detection mapped to a ground truth instance is considered as a true positive. All the other detections are false positives. Multiple detections of the same object are considered to be false detections. The ground truth objects that have not been detected are considered as false negatives.
3. The predictions are ranked from higher to lower confidence scores.
4. Recall and precision are computed progressively advancing through this rank of predictions, by recurring to their common definitions, except for the fact that only a subset of data will be used at each stage:

$$Recall = TP/(TP + FN) \qquad Precision = TP/(TP + FP)$$

Below, the example that yields the precision/recall curve in Figure 2 is provided. In this example we assume that there are 6 ground truth instances of a given class. The algorithm has made 6 predictions, of which 3 have an IoU > 50% to ground instances. As a consequence, 3 predictions have no associated ground truth instances and FN=3 for all the steps.

Rank	Correct?	Cumulated TP	Cumulated FP	**Recall**	**Precision**
1	TP	1	0	**0,25**	**1**
2	FP	1	1	**0,25**	**0,5**

3	TP	2	1	**0,4**	**0,67**
4	FP	2	2	**0,4**	**0,5**
5	FP	2	3	**0,4**	**0,4**
6	TP	3	3	**0,5**	**0,5**

Before calculating the area under recall precision curve, the jigsaw effect is smoothed, taking the highest precision value to the right for any given recall value. This is represented by the red triangles in Figure 2. The AP value is obtained then as the blue area plus the red area depicted in Figure 2.

Project Design

To begin, the data will have to be analyzed to assess if all provided bounding boxes for the different objects make sense. It is also important to understand at this stage the different object sizes that can be found in the images, looking at differences between classes, and also differences for the same class. We will look both at this distribution of sizes for both the parent and child classes. This information will drive the way the images will be cropped to construct a dataset to train the algorithm and the choice of the algorithm. The number of images for each of the classes will be analyzed as well.

Then, from the raw images a dataset will have to be created, with the images cropped into several chips of a given smaller size than can be fed into the model for training and to perform inference. This new dataset will have to be split into at least train and test data (and eventually evaluation data), trying to keep a sufficient number of samples of each class for each set of data (and roughly the same proportion of classes between sets).

The framework chosen to develop the model is Pytorch. The idea is to start from a model that is already used for object detection, as for example YOLO or the Single Shot Detector (SSD). There is a new version of YOLO (YOLOv5) and it could be interesting comparing the performance of YOLOv5 to that obtained by the benchmark model with YOLO v3. The model for this particular problem will have to be built around one of those two. In order to choose one or the other, or even a different model, it will be done based on literature on the subject, on reported performances on the literature for this particular dataset or for similar satellite datasets, and also on complexity and resources that will be required to train the model.

Whatever it is the chosen model, it will have to be adapted so it can read the images from the dataset, and also provide predictions for the number of classes in the dataset. This will be done by creating a customised neural network that will be attached to the pretrained model. In principle, the idea is to train only this customised neural network.

One of the possible blocking points on this project will be the resources that will be needed to train the algorithm. To alleviate the process, first some tests will be performed on the author's local computer. Then, the idea is to use Amazon Web Services with Sagemaker, the provided workspace for the dog breed classification problem and/or Google Colab Notebooks to complete the full training process. If still the required training was too computer heavy, one of the solutions that are being considered is training the model to detect only the

parent classes. In that case, the results from Ayush *et al.* (2020) can be used as a benchmark to which compare the model. Finally, the model performance will be measured in the test images that have been reserved for that purpose.

References

Ayush, K.; Uzkent, B.; Burke, M.; Lobell, D.; and Ermon, S. 2020. Generating Interpretable Poverty Maps using Object Detection in Satellite Images. arXiv preprint arXiv:2002.01612

Biermann, L., Clewley, D., Martinez-Vicente, V. *et al.* Finding Plastic Patches in Coastal Waters using Optical Satellite Data. *Sci Rep* **10,** 5364 (2020). https://doi.org/10.1038/s41598-020-62298-z

Fong, R. 2018. The XView Dataset and Baseline Results. Medium article. Consulted 03/03/2021. https://medium.com/picterra/the-xview-dataset-and-baseline-results-5ab4a1d0f47f

Henderson P., Ferrari V. (2017) End-to-End Training of Object Class Detectors for Mean Average Precision. In: Lai SH., Lepetit V., Nishino K., Sato Y. (eds) Computer Vision – ACCV 2016. ACCV 2016. Lecture Notes in Computer Science, vol 10115. Springer, Cham. https://doi.org/10.1007/978-3-319-54193-8_13

Hui, J. 2018. mAP (mean Average Precision) for Object Detection. Medium article. Consulted 03/03/2021. https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

Lam, D.; Kuzma, R.; McGee, K.; Dooley, S.; Laielli, M.; Klaric, M.; Bulatov, Y.; and McCord, B. 2018. xView: Objects in Context in Overhead Imagery. arXiv preprint arXiv:1802.07856

xView challenge: http://xviewdataset.org/