Ensemble Methods for Object Detection

Ángela Casado-García and **Jónathan Heras**[1]

Abstract. Object detection is one of the most important topics of computer vision since it has many applications in several fields. Object detection models can be improved thanks to ensemble techniques; however, the process of ensembling object detectors poses several challenges. In this paper, we present an ensemble algorithm that can be applied with any object detection model independently of the underlying algorithm. In addition, our ensemble method has been employed to define a test-time augmentation procedure for object detection models. Our ensemble algorithm and test-time augmentation procedure can be used to apply data and model distillation for object detection, two semi-supervised learning techniques that reduce the number of necessary annotated images to train a model. We have tested our methods with several datasets and algorithms, obtaining up to a 10% improvement from the base models. All the methods are implemented in an open-source library.

1 INTRODUCTION

Object detection is a fundamental task in computer vision since it is a key step in many real-world applications such as security [2], satellite imagery [12] or healthcare [35]. Over the last few years, a lot of progress has been made in this field thanks to the use of deep convolutional neural networks [51], and deep detectors have achieved impressive results in large detection datasets such as Pascal VOC [13] and MS COCO [28]. In addition, as in many other machine learning tasks, the accuracy and robustness of object detectors can be greatly improved thanks to the application of ensemble methods [50]; for instance, the mmAP in the COCO dataset was improved from 50.6 to 52.5 in [33], or the mAP in the Pascal VOC dataset increased by 3.2% in [37]. In fact, the leading methods on datasets like Pascal VOC or MS COCO are based on the usage of ensembles [18, 20, 33].

However, the process of ensembling object detectors poses several challenges. First of all, some ensemble approaches for object detection depend on the nature of the detection models — for example, the procedure to ensemble models explained in [18] can only be applied to models based on the FasterRCNN algorithm — therefore, these methods cannot be generalised and lack the diversity provided by the ensemble of different algorithms. Related to the previous point, those ensemble methods require the modification of the underlying algorithms employed to construct the models, and this might be challenging for many users. In order to deal with this problem, there are ensemble methods that work with the output of the models [44, 48]; but, again, they are focused on concrete models, and only work if the models are constructed using the same framework. Finally, it does not exist an open-source library that provides general ensemble methods for object detection, and this hinders their use.

[1] Department of Mathematics and Computer Science, Universidad de La Rioja, Spain, email: {angela.casado,jonathan.heras}@unirioja.es

In this paper, we tackle the aforementioned challenges by designing a generic method that serves to ensemble the output produced by detection algorithms; that is, *bounding boxes* which indicate the position and category of the objects contained in an image. The method can be employed with any detection model independently of its underlying algorithm and the framework employed to construct it. In particular, the contributions of this work are the following:

- We present a general method for ensembling object detectors independently of the underlying algorithm; and, in addition, we devise several voting strategies to carry out the ensembling process.
- As a by-product of our ensemble method, we define a test-time augmentation procedure that can be applied to boost the accuracy of object detection models. Moreover, we explain how to reduce the burden of annotating images in the context of object detection using two semi-supervised learning techniques, based on ensemble methods, known as model and data distillation.
- We conduct a comprehensive study of the impact of our ensemble method and the devised voting strategies, and show the benefits of this method as well as the advantages of using test-time augmentation and distillation methods.
- We implement our methods in the EnsembleObjectDetection library, available at `https://github.com/ancasag/ensembleObjectDetection`. This open-source library can be extended to work with any object detection model independently of the algorithm and framework employed to construct it.

The rest of the paper is organised as follows. In the next section, we provide the necessary background to understand the rest of the paper. Subsequently, our approach to ensemble object detection algorithms, and the extension of such an approach for test-time augmentation, and data and model distillation are presented in Section 3. In addition, and also in Section 3, we present the main highlights of the library that implements our methods. After that, an analysis of the impact of our methods on different datasets is provided in Section 4. The paper ends with a section of conclusions and further work.

2 BACKGROUND

In this section, we briefly provide the necessary background and notation needed to understand the rest of the paper. We start by formally presenting the task of object detection.

2.1 Object detection

Object detection is the task of determining the position and category of multiple objects in an image. Formally, an object detection model can be seen as a function that given an image I returns a list of detections $D = [d_1, \ldots, d_N]$ where each d_i is given by a triple $[b_i, c_i, s_i]$

that consists of a bounding box, b_i, the corresponding category, c_i, and the corresponding confidence score, s_i.

Currently, the most successful object detection models are based on deep learning algorithms, and they can be split into two groups: one-stage and two-stage detectors. The former divide the image into regions that are passed into a convolutional neural network to obtain the list of detections — these algorithms include techniques such as SSD [30] or YOLO [38]. The two-stage object detectors employ region proposal methods, based on features of the image, to obtain interesting regions, that are later classified to obtain the predictions — among these algorithms, we can find the R-CNN family of object detectors [39] or Feature Pyramid Network (FPN) [29]. Independently of the underlying algorithm, the accuracy of these detection models can be improved thanks to the application of ensemble methods.

2.2 Ensemble learning

Ensemble methods combine the predictions produced by multiple models to obtain a final output [50]. These methods have been successfully employed for improving accuracy in several machine learning tasks, and object detection is not an exception. We can distinguish two kinds of ensembling techniques for object detection: those that are based on the nature of the algorithms employed to construct the detection models, and those that work with the output of the models.

In the case of ensemble methods based on the nature of the algorithms, different strategies have been mainly applied to two-stage detectors. Some works have been focused on ensembling features from different sources before feeding them to the region proposal algorithm [25, 33], others apply an ensemble in the classification stage [17, 7], and others employ ensembles in both stages of the algorithm [18, 20, 24]. In the case of ensemble methods based on the output of the models, the common approach consists in using a primary model which predictions are adjusted with a secondary model. This procedure has been applied in [48] by combining Fast-RCNN and Faster-RCNN models, in [37] by combining Fast-RCNN and YOLO models, and in [44] by using RetinaNet and Mask R-CNN models. Another approach to combine the output of detection models is the application of techniques to eliminate redundant bounding boxes like Non-Maximum Suppression [19], Soft-NMS [4], NMW [52], fusion [49] or WBF [45]. However, these techniques do not take into account the classes of the detected objects, or the number of models that detected a particular object; and, therefore, if they are blindly applied, they tend to produce lots of false positives.

In our work, we propose a general method for ensembling the output of detection models using different voting strategies, see Section 3.1. The method is independent of the underlying algorithms and frameworks, and allows us to easily combine a variety of multiple detection models. In addition, our method opens the door to apply techniques based on ensembles such as test-time augmentation.

2.3 Test-time augmentation

Data augmentation [42, 41] is a technique widely employed to train deep learning models that consists in generating new training samples from the original training dataset by applying transformations that do not alter the class of the data. There is a variant of data augmentation for the test dataset known as *test-time augmentation* [43]. This technique creates random modifications of the test images, performs predictions on them, and, finally, returns an ensemble of those predictions.

Due to the cost of collecting data in the context of object detection, data augmentation strategies such as random scaling [37] or cropping [30] are widely employed [54]. On the contrary, and due to the lack of a general method to combine predictions of object detectors, test-time augmentation has been mainly applied in the context of image classification [43]. As far as we are aware, test-time augmentation has only been applied for object detectors in [49], and only using colour transformations. This limitation is due to the fact that some transformations, like flips or rotations, change the position of the objects in the image and this issue must be taken into account when combining the predictions. The method presented in Section 3.2 deals with this problem and allows us to apply test-time augmentation with any object detection model.

3 METHODS

In this section, we explain our ensemble algorithm for combining the output of object detection models. Such an algorithm can be particularised with different strategies that are also explained in this section. Moreover, we explain how our algorithm can be applied for test-time augmentation, and data and model distillation. This section ends by highlighting the main features of the library where we have implemented our methods.

3.1 Ensemble of object detectors

We start by explaining the procedure that we have designed to combine object detections obtained from several sources. The input of our ensemble algorithm is a list $LD = [D_1, \ldots, D_m]$ where each D_i, with $i \in \{1 \ldots m\}$, is a list of detections for a given image I as explained in Section 2.1. Usually, each D_i comes from the predictions of a detection model; but, as we will see in Section 3.2, this is not always the case. In general, each list D_i is a list of detections produced using a particular method M_i for a given image.

Given the list LD, our ensemble algorithm consists of four steps. First of all, the list LD is flattened in a list $F = [d_1, \ldots, d_k]$, since the provenance of each detection d_i is not relevant for the ensembling algorithm. Subsequently, the elements d_i of F are grouped together based on the overlapping of their bounding boxes and their classes. In order to determinate the overlap of bounding boxes, the IoU metric [40] is employed. Considering two bounding boxes b_1 and b_2, the IoU formula for finding the overlapped region between them is given by

$$IoU(b_1, b_2) = \frac{area(b_1 \cap b_2)}{area(b_1 \cup b_2)}$$

This measure is employed to group the elements of F producing as a result a list $G = [D_1^G, \ldots, D_m^G]$ where each D_i^G is a list of detections such that for all $\bar{d}(= [\bar{b}, \bar{c}, \bar{s}]), \hat{d}(= [\hat{b}, \hat{c}, \hat{s}]) \in D_i^G$, $IoU(\bar{b}, \hat{b}) > 0.5$, and $\bar{c} = \hat{c}$. At this point, each $D_i^G \in G$ is focused on a particular region of the image, and the size of D_i^G will determine whether our algorithm considers whether such a region actually contains an object. Namely, this decision can be taken using three different voting strategies:

- Affirmative. In this strategy, all the lists D_i^G are kept. This means that whenever one of the methods that produce the initial predictions says that a region contains an object, such a detection is considered as valid.
- Consensus. In this case, only the lists D_i^G which length is greater than $m/2$ (where m is the size of the initial list LD) are kept.

This means that the majority of the initial methods must agree to consider that a region contains an object. The consensus strategy is analogous to the majority voting strategy commonly applied in ensemble methods for images classification [3].

- Unanimous. In the last strategy, only the lists D_i^G which length is equal to m are kept. This means that all the methods must agree to consider that a region contains an object.

After applying one of the aforementioned strategies, we end up with a list $G' \subseteq G$. Since each list $D_k^{G'} \in G'$ might contain several detections for the same region, the last step of our algorithm is the application of the non-maximum suppression (NMs) algorithm to each $D_k^{G'}$. The final result is a list $D = [d_1, \dots, d_n]$ with the ensemble detections. Our ensemble algorithm is summarised graphically in Figure 1.

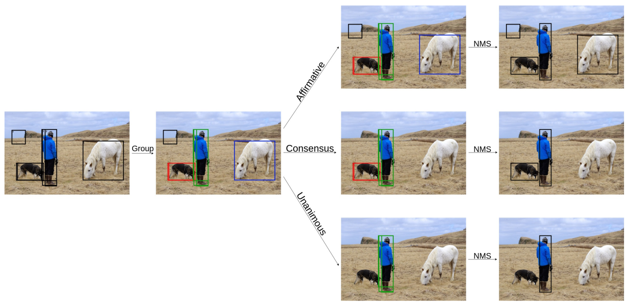

Figure 1. Example of the workflow of our ensemble algorithm. Three methods have been applied to detect the objects in the original image: the first method has detected the person and the horse; the second, the person and the dog; and, the third, the person, the dog, and an undefined region. The first step of our ensemble method groups the overlapping regions. Subsequently, a voting strategy is applied to discard some of those groups. The final predictions are obtained using the NMs algorithm.

From a theoretical point of view, the affirmative strategy reduces the number of objects that are not detected (false negatives) — some objects that are not detected with a concrete approach might be detected by the others — but increases the number of incorrect detections (false positives) — this is due to the fact that the false positives obtained with each approach are accumulated. The unanimous strategy has the opposite effect, it reduces the number of false positives but increases the number of false negatives — since all the approaches that generated the initial detections must agree to detect an object. In general, the consensus strategy provides a better trade-off, and, therefore, at first glance, the affirmative and unanimous strategies might seem too lose and too restrictive to be useful. However, as we will show in Section 4, they can produce better results than the consensus approach depending on the performance of the detection models (for instance, if the detection models produce few false positives, but lots of false negatives, the affirmative strategy might be more useful than the other two strategies).

As we explained at the beginning of this section, the most natural way of producing the input of our ensemble algorithm is by using the predictions that are outputted by several object detection models. In addition, there are other ways, for instance, combining the detections of a model for multiple transformations of an image, this is known as test-time augmentation.

3.2 Test-time augmentation for object detectors

Test-time augmentation (TTA) in the context of image classification is as simple as applying multiple transformations to an image (for example, flips, rotations, colour transformations, and so on), making predictions for each of them using a particular model, and finally returning the ensemble of those predictions [43]. On the contrary, in the context of object detection, TTA is not as straightforward due to the fact that there are some transformations, like flips or crops, that alter the position of the objects. This explain why the works that apply TTA for object detection only apply colour operations [49] — since those transformations do not alter the position of the objects in the image. This limitation of the TTA method is faced in this section taking as basis the ensemble algorithm presented previously.

First of all, we define the notion of *detection transformation*. Given an image I and a list of detections for I, D, a detection transformation is an operation that returns a transformed image I^t and a list of detections D^t such that the size of D^t is the same of D, and all the objects detected by D in I are detected by D^t in I^t.

Example 3.1. Given an image I of size (W_I, H_I) where W_I and H_I are respectively the width and height of I, and a list of detections $D = [d_1, \dots, d_n]$ such that for each $d_i = [b_i, c_i, s_i]$ and b_i is given by (x_i, y_i, w_i, h_i) where (x_i, y_i) is the position of the top-left corner of b_i, and w_i and h_i are respectively the width and height of b_i; then, the horizontal flip detection transformation applies an horizontal flip to the image I, and returns it together with the list $D^t = [d_1^t, \dots, d_n^t]$ where $d_i^t = [(W_I - x_i, y_i, w_i, h_i), c_i, s_i]$. Another example is the equalisation transformation that applies the histogram equalisation to the image I and returns it together with $D^{t'} = D$. These examples are depicted in Figure 2.

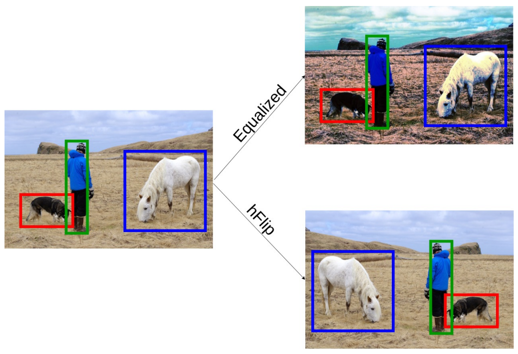

Figure 2. Example of horizontal flip detection transformation and equalisation transformation

Now, we can define the following procedure to apply TTA for object detection. Given an image I, an object detection model M, and a list of image transformations $T_1, \dots, T_n$, we proceed as follows. First of all, we apply each image transformation T_i to I, obtaining as a result new images $I_1, \dots, I_n$. Subsequently, we detect the objects in each I_i using the model M, and produce the lists of detections $D_1, \dots, D_n$. For each, (I_i, D_i), we apply a detection transformation that returns a list of detections D_i^t in the correct position for the original image I. Finally, we ensemble the predictions using the procedure presented in the previous section using one of the three voting strategies. An example of this procedure is detailled in Figure 3.

The ensemble of models and TTA can be employed to improve the accuracy of object detectors, see Section 4. In addition, they are the basis of two semi-supervised learning techniques that tackle one of

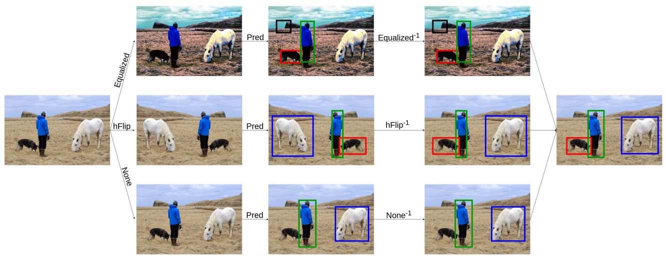

Figure 3. Example of the workflow of TTA for object detectors. First, we apply three transformations to the original image: a histogram equalisation, a horizontal flip, and a none transformation (that does not modify the image). Subsequently, we detect the objects in the new images, and apply the corresponding detection transformation to locate the objects in the correct position for the original image. Finally, the detections are ensembled using the consensus strategy.

the main problems faced when training object detection models: the annotation of images.

3.3 Data and model distillation

Deep learning methods are data demanding, and acquiring and annotating the necessary amount of images for constructing object detection models is a tedious and time-consuming process that might require specialised knowledge [23]. This has lead to the development of semi-supervised learning techniques [53], a suite of methods that use unlabelled data to improve the performance of models trained with small dataset of annotated images. Self-training [53] is a particular case of semi-supervised learning where the model predictions are employed as ground truth to train a new model. However, training a model on its own predictions does not usually provide any benefit; and this has lead to the development of techniques like *data distillation* and *model distillation*.

Data distillation [34] applies a trained model on manually labelled data to multiple transformations of unlabelled data, ensembles the multiple predictions, and, finally, retrains the model on the union of manually and automatically labelled data. Similarly, model distillation [5] obtains multiple predictions of unlabelled data using several models, ensembles the result, and retrains the models with the combination of manually and automatically annotated data. Both techniques can also be combined as shown in [21].

Even if data distillation was applied to object detection in [34], these method have not been widely employed in the context of object detection due to the lack of a library for ensembling predictions of detection models. This problem is overcome thanks to the techniques and the library developed in our work, and, as we show in Section 4, different ensembling schemes to the one proposed in [34] might have a better impact on the distillation methods.

3.4 Design of the library

The techniques presented throughout this section have been implemented as an open-source library called EnsembleObjectDetection. This library has been implemented in Python and relies on several third-party libraries like Numpy [32], OpenCV [31], or CLoDSA [8] (the last one provides the functionality to implement the image and detection transformations of the TTA procedure).

As we have previously mentioned, the EnsembleObjectDetection library has been designed to be applicable to models constructed with any framework and underlying algorithm. To this aim, we have defined an abstract class called `IPredictor` with a `predict` method that takes as input a folder of images, and produces as a result XML files in the Pascal VOC format containing the predictions for each image of the input folder. Then, for each detection framework or library that we want to include in the EnsembleObjectDetection library, we have to provide a class that extends the `IPredictor` class and implements the `predict` method. Using the output produced by the `predict` method, the user can apply ensembling and TTA with any detection model.

Currently, the EnsembleObjectDetection library supports models trained using the Darknet [36] and MxNet [10] frameworks, and several Keras libraries [1, 27]. The procedure to extend the library with models from other libraries is explained in the project webpage.

4 RESULTS

In this section, we conduct a thorough study of the ensemble methods presented in the previous section by using three different datasets.

4.1 Pascal VOC

In the first case study, we use the Pascal VOC dataset [14], a popular project designed to create and evaluate algorithms for image classification, object detection and segmentation. This dataset consists of natural images containing objects of 20 categories; and, the metric to evaluate the performance of detection models in this dataset is the mean average precision (mAP) over each category [40].

For our experiments with the Pascal VOC dataset, we have employed 5 models pre-trained for this dataset using the MxNet library [10]; namely, a Faster R-CNN model, two YOLO models (one using the darknet backbone and another one using the mobilenet backbone) and two SSD models (one using the Resnet backbone and the other using the mobilenet backbone). The performance of these models on the Pascal VOC test set is given in the first five rows of Table 1. As can be seen in such a table, the best model is the YOLO model using the darknet backbone with a mAP of 69.78%. Such a mAP can be greatly improved using model ensembling and TTA.

For model ensembling, we conduct an ablation study by considering the ensemble of the five models, the ensemble of the three models with the best mAP (that are Faster R-CNN, YOLO with the darknet backbone, and SSD with the Resnet backdone), and the three models with the worst mAP (that are YOLO with the mobilenet backbone and the two SSD models). The results for such an ablation study are provided in the last 9 rows of Table 1. In those results, we can notice that all the ensembles conducted with the affirmative strategy obtain better results than the individual models — the best result is obtained by ensembling the three best models (mAP of 77.50%, almost an 8% better than the best individual model). On the contrary, the unanimous strategy produces worse results than the individual models; and the consensus strategy only achieves a better mAP when the three best models are combined. These results are due to the fact that the individual models produce few false positives, and some objects that are detected by one of the models are missed by the others. Therefore, the affirmative strategy helps to greatly reduce the number of false negatives but without considerably increasing the number of false positives; on the contrary, the unanimous strategy is too restrictive and increases the number of false negatives. Something similar happens with the consensus strategy. If we focus on the results for each particular category, we can notice an improvement of up to a 10% with respect to the results obtained by the best individual model.

Datasets	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster R-CNN	0.69	0.70	0.77	0.71	0.64	0.50	0.78	0.78	0.80	0.45	0.74	0.58	0.79	0.80	0.79	0.70	0.42	0.75	0.67	0.77	0.67
SSD mobilenet	0.62	0.59	0.70	0.61	0.51	0.33	0.68	0.71	0.78	0.43	0.57	0.61	0.69	0.79	0.71	0.61	0.36	0.59	0.60	0.77	0.67
SSD resnet	0.64	0.62	0.80	0.70	0.57	0.42	0.78	0.79	0.88	0.50	0.73	0.63	0.78	0.80	0.80	0.70	0.39	0.69	0.65	0.79	0.69
YOLO darknet	0.69	**0.80**	0.72	0.70	0.57	0.60	0.80	0.80	0.81	0.43	0.75	0.63	0.78	0.81	0.71	0.70	0.39	0.71	0.65	0.79	0.70
YOLO mobilenet	0.59	0.62	0.71	0.52	0.49	0.43	0.70	0.71	0.70	0.36	0.66	0.47	0.68	0.71	0.62	0.61	0.24	0.60	0.55	0.70	0.61
All affirmative	0.77	0.79	0.80	0.79	**0.72**	0.64	**0.87**	0.86	0.88	**0.55**	0.83	0.68	**0.87**	0.87	0.80	0.78	0.51	0.79	**0.75**	0.78	0.76
All consensus	0.68	0.71	0.72	0.71	0.60	0.44	0.79	0.80	0.80	0.45	0.76	0.63	0.80	0.81	0.71	0.70	0.39	0.71	0.65	0.78	0.62
All unanimous	0.51	0.54	0.63	0.45	0.35	0.27	0.62	0.63	0.63	0.32	0.53	0.42	0.63	0.63	0.63	0.53	0.17	0.54	0.51	0.62	0.54
Three best affirmative	**0.77**	0.79	**0.80**	**0.79**	0.71	**0.64**	0.86	**0.87**	**0.89**	0.55	**0.83**	**0.69**	**0.87**	**0.88**	**0.80**	**0.78**	**0.51**	**0.79**	0.75	**0.84**	**0.76**
Three best consensus	0.71	0.71	0.80	0.71	0.60	0.52	0.80	0.80	0.81	0.51	0.76	0.64	0.80	0.81	0.80	0.70	0.47	0.71	0.71	0.78	0.70
Three best unanimous	0.61	0.63	0.72	0.63	0.52	0.35	0.71	0.72	0.81	0.39	0.61	0.57	0.71	0.72	0.72	0.62	0.33	0.62	0.58	0.71	0.62
Three worst affirmative	0.73	0.77	0.79	0.77	0.66	0.56	0.78	0.79	0.80	0.52	0.80	0.62	0.79	0.80	0.80	0.77	0.48	0.76	0.73	0.79	0.74
Three worst consensus	0.66	0.71	0.71	0.62	0.60	0.43	0.70	0.71	0.80	0.44	0.68	0.56	0.71	0.80	0.72	0.70	0.39	0.69	0.64	0.79	0.69
Three worst unanimous	0.52	0.54	0.63	0.54	0.44	0.27	0.62	0.63	0.63	0.32	0.53	0.49	0.63	0.63	0.63	0.53	0.24	0.53	0.50	0.62	0.53

Table 1. Results for the Pascal VOC dataset applying our model ensemble algorithm. The first five rows provide the result for the base models. The next three rows correspond with the ensemble for the base models. Rows 9 to 11 contain the results of applying the ensemble techniques to the three best base models (SSD resnet, YOLO darknet and Faster R-CNN); and the last three rows contain the results of ensembling the worst three best models (YOLO mobilenet, SSD mobilenet and Faster R-CNN). The best results are in bold face.

In addition, we have applied TTA to all the base models to improve their accuracy. Namely, we have applied three kinds of data augmentations: colour transformations (applying gamma and histogram normalitation, and keeping the original detection), position transformations (applying a horizontal flip, a rotation of 10º, and keeping the original prediction) and the combination of both. Moreover, for each augmentation scheme, we have applied the three voting strategies, see Table 2. As in the case of model ensembling, all the models are improved (the improvement ranges from 0.08% to 5.54%) thanks to TTA when using the affirmative strategy; but the most beneficial augmentation scheme varies from model to model. For instance, the SSD model with the resnet backbone is improved with the three augmentation schemes; but, the Faster R-CNN model only improves with the colour scheme. Regarding the other voting strategies, the unanimous strategy always obtains worst results than the original models; and the consensus strategy only gets better results in some cases. The explanation for these results is the same provided previously for model ensembling. It is also worth noting that adding more augmentation techniques does not always improve the ensembling results.

	No	TTA Colour			TTA Position			TTA All		
	TTA	Aff.	Cons.	Una.	Aff.	Cons.	Una.	Aff.	Cons.	Una.
Faster R-CNN	0.69	0.69	0.69	0.08	0.53	0.53	0.22	0.63	0.61	0.21
SSD mobilenet	0.62	0.63	0.63	0.09	0.58	0.58	0.52	0.61	0.58	0.47
SSD resnet	0.64	0.70	0.70	0.08	0.65	0.65	0.60	0.68	0.63	0.09
YOLO darknet	0.69	**0.71**	**0.71**	0.09	0.68	0.68	0.63	0.70	0.68	0.57
YOLO mobilenet	0.59	0.61	0.61	0.10	0.57	0.57	0.50	0.61	0.58	0.44

Table 2. Results for the Pascal VOC dataset applying TTA. In the first column, we provide the result for the models without applying TTA. The rest of the table is divided into three blocks of three columns (one per each voting strategy): the first block provides the results with colour transformations, the second contains the results for position transformations, and, the last block presents the results combining all the transformations. The best results are in bold face.

As a conclusion for this study, we can say that model ensembling is more beneficial than TTA; and, this is due to the fact that the former introduces a higher variability (thanks to the heterogeneity of models) in the predictions than the latter.

4.2 Stomata detection

In the second example, we apply TTA and data distillation to two proprietary datasets of stomata images. Stomata (singular "stoma") are pores on a plant leaf that allow the exchange of gases, mainly CO_2 and water vapor, between the atmosphere and the plant. Stomata respond to changes in the environment and regulate the photosynthesis of plants, and thus their productivity [22, 6].

In order to analyse stomata of plant leaves, plant biologists take microscopic images of leaves, and manually measure the stomata density in those images. This is a tedious, error-prone, time-consuming and subjective task due to the large number of stomata in each image but, it can be automatised by means of detection algorithms. In particular, we have constructed a stomata detection model using the YOLO algorithm implemented in the Darknet framework [9]. The YOLO model was trained by using 4,050 stomata images, and it was evaluated on a test set of 450 images using the F1-score and the mAP, see the first row of Table 3. As can be seen from that table, the number of false positives (FP) is considerably higher than the number of false negatives (FN), and this will have an impact in the voting strategy to apply. In this section, we show how such a model can be improved thanks to TTA.

Datasets	F1-score	TP	FP	FN	mAp
Original	0.90	16600	2341	1239	0.84
Affirmative	0.88	**17003**	3600	**836**	0.80
Consensus	0.92	16509	1551	1330	0.84
Unanimous	0.80	12272	**589**	5567	0.61
Colour	**0.93**	16502	1324	1337	**0.85**
Flips	0.92	16480	1448	1359	0.85
Rotations	0.91	16463	1999	1376	0.82
Flips & colour	0.92	16572	1529	1267	0.84
Flips & rotations	0.92	16580	1659	1259	0.84
Rotations & colour	0.92	16556	1633	1283	0.84

Table 3. Results of our Yolo model for stomata detection. In the first row, we provide the results for the original model. In the next three rows, we have applied TTA with 9 transformations using the three voting strategies; and, in the next six rows, we have applied the TTA method for different kinds of transformations and using the consensus strategy. The best results are in bold face.

First of all, we have applied TTA by using 9 transformations: three colour transformations (histogram normalisation, gamma correction and Gaussian blurring), three flips (vertical, horizontal and both) and three rotations (90º, 180º and 270º). Moreover, we have applied the three voting schemes — the results for these experiments are given in rows 2 to 4 in Table 3. As can be seen in this table, the only strategy that improves the results is the consensus approach, that improves a 2% the F1-score. Note that each strategy has the expected effect, the affirmative scheme increases the number of FP and decreases the number of FN; on the contrary, the unanimous strategy has the opposite effect. Then, the consensus strategy provides the best trade-off by considerably reducing the number of FP but only slightly increasing the number of FN.

In addition to the above results, we have also inspected the impact of each kind of transformation for this dataset. In particular, we have applied TTA using colour transformations, rotation transformations, flip transformations, and their combinations — see the last 6 rows of Table 3. In this case, we only included the consensus strategy in Table 3 since the other strategies produced the same effect previously explained. As can be seen from those results, the same improvement obtained using all the transformation can be achieved by using only some of them — this considerably reduces the tome needed to apply TTA. In fact, we achieved better results by applying TTA using only colour transformations. This indicates that it is necessary to study different combinations of transformations to find the one that produces the best results; and this shows the benefits of having a library like the one presented in this work.

Finally, we have also studied the benefits of data distillation in the context of stomata detection. It is worth noting that each stomata image contains approximately 45 stomata, and, hence, annotating those images is a time-consuming task. Therefore, the application of semi-supervised learning techniques, like data distillation, can reduce the burden of annotating those images. In our data distillation experiments, see Table 4, we have employed a dataset of stomata images from a different variety than the original dataset employed for the results of Table 3. Such a dataset contains 450 annotated images for training, 150 annotated images for testing, and 1,620 unlabelled images. Using the 450 annotated images, we constructed a YOLO model that achieved a F1-score of 0.85 and a mAP of 0.8 when using a confidence threshold of 0.25, and a F1-score of 0.83 and a mAP of 0.79 when using a confidence threshold of 0.5. Using such a model, we have applied data distillation using three schemes: applying colour transformations (gamma correction, histogram, normalisation and Gaussian blurring), applying flips (vertical, horizontal and both), and combining colour and flip transformations. Moreover, we have used the three voting schemes, see Table 4. In this case, the best strategy consists in applying all the transformations together with the unanimous voting scheme. This improves a 3% the F1-score value, and an 8% the mAP. Note that using the unanimous strategy, we can increase the threshold to consider a detection as correct since using such a strategy the new model is trained with images where the detections have been agreed by the predictions of the 9 transformations.

Based on the results obtained for this example, we have shown the benefits of trying different alternatives for TTA; and, in addition, that techniques like data distillation can produce accurate models starting from small datasets of images.

4.3 Table detection

In the last case study, we analyse the effects of model ensembling and model distillation for table detection — an important problem since it is a key step to extract the semantics from tabular data [11]. To this aim, we have employed the ICDAR2013 dataset [16], and the Word part of the TableBank dataset [26]. Both datasets have been designed to test table detection algorithms; however, the ICDAR2013 is too small to directly apply deep learning algorithms (it only contains 238 images), and the TableBank dataset is a big enough dataset (160k images) but it was semi-automatically annotated and contains several incorrect annotations. Therefore, these two datasets provide the perfect scenario for applying model ensembling and model distillation. In particular, we have trained different models for the ICDAR2013 dataset, and have improved them by using our ensemble algorithm, and by applying model distillation using the TableBank dataset.

	Confidence 0.25		Confidence 0.5	
Datasets	F1-score	mAP	F1-score	mAP
Original	0.85	0.80	0.83	0.79
All Aff.	0.84	0.82	0.80	0.82
All Cons.	0.86	0.85	0.87	0.85
All Una.	0.86	0.88	**0.88**	**0.88**
Colour Aff.	0.86	0.86	0.83	0.86
Colour Cons.	0.82	0.78	0.77	0.78
Colour Una.	0.74	0.77	0.62	0.77
Flips Aff.	0.83	0.80	0.76	0.78
Flips Cons.	0.04	0.61	0	0.55
Flips Una.	0.01	0.30	0	0.30

Table 4. Results of data distillation for the stomata dataset. This table is divided into two blocks: in the first block (columns 2 and 3), we use a confidence threshold of 0.25; and, in the second block (the last two columns), we use a confidence threshold of 0.5. In the first row of the table, we provide the result for the original model. In the next three rows, we present the results of applying data distillation with colour and flip transformations. In rows 5 to 7, we include the results of applying data distillation only using colour transformations, and the last three rows present the results of applying data distillation using flip transformations. The best results are in bold face.

	F1@0.6	F1@0.7	F1@0.8	F1@0.9	WAvgF1
Mask R-CNN	0.58	0.52	0.39	0.19	0.39
SSD	0.85	0.79	0.62	0.26	0.59
YOLO	0.83	0.8	0.69	0.33	0.63
Affirmative	0.86	0.81	0.69	0.37	0.65
Consensus	**0.88**	**0.84**	**0.75**	**0.42**	**0.69**
Unanimous	0.4	0.36	0.28	0.08	0.26
Mask R-CNN Aff.	0.58	0.54	0.39	0.11	0.37
Mask R-CNN Cons.	0.68	0.63	0.49	0.13	0.45
Mask R-CNN Una.	0.57	0.48	0.29	0.06	0.32
SSD Aff.	0.77	0.71	0.58	0.24	0.54
SSD Cons.	0.87	0.8	0.67	0.32	0.63
SSD Una.	0.82	0.74	0.56	0.26	0.56
YOLO Aff.	0.75	0.71	0.57	0.18	0.52
YOLO Cons.	0.88	0.8	0.67	0.32	0.63
YOLO Una.	0.77	0.69	0.52	0.16	0.50

Table 5. Results for the ICDAR2013 dataset applying model ensembling and model distillation. The first three rows are the results for the base models. The next three rows include the results of applying our ensemble method with the three different voting strategies; and, the next three blocks provide the results of applying model distillation to the three base algorithms. The best results are in bold face.

In our experiments, we have split the ICDAR2013 dataset into a training set of 178 images and a testing set of 60 images. Using the training set, we have constructed three models using the YOLO algorithm, implemented in Darknet, the SSD algorithm, implemented in MxNet, and the Mask RCNN algorithm, implemented in the Keras library [15] — note that we have employed different libraries and algorithms, but our ensemble library can deal with all of them. These three models have been evaluated (see the three first rows of Table 5) in the testing set using the W1Avg F1-score [46], a metric that computes the weighted sum of the F1-score using different IOU thresholds ranging from 0.6 to 0.9 — the F1-score at 0.6 is employed to measure the number of tables that are detected, even if the detection bounding boxes are not perfectly adjusted; and, on the contrary, the F1-score at 0.9 measures the tables that are detected with a bounding box perfectly adjusted to them. As can be seen in Table 5, the best model is obtained using the YOLO algorithm (WAvgF1-score of 0.63).

The three models can be improved thanks to model ensembling, and model distillation. First of all, we have ensembled the three models using the three voting strategies (see rows 3 to 5 of Table 5), and we have obtained an improvement of 2% using the affirmative strategy, and a 6% using the consensus approach; as we have seen previously, the unanimous approach is too restrictive and obtains worst results.

Moreover, we have applied model distillation using the TableBank dataset; applying the three voting strategies, and retraining the three models, see the last 9 rows of Table 5. Using this approach we have improved the SSD model a 4%, the Mask R-CNN model a 6%; but, the YOLO model does not improve at all. However, if we inspect the F1-score value at 0.6, the improvement is more evident, SSD improves a 2%, Mask R-CNN a 10%, and YOLO a 5%. This is due to the fact that the ensemble of models usually produces bounding boxes that are not perfectly adjusted to the objects; the issue of improving those adjustments remain as further work.

As a conclusion of this example, we can again notice the benefits of applying our ensemble algorithm, and the improvements that can be achieved applying model distillation when a large dataset of images is available, even if it is not annotated.

5 CONCLUSIONS AND FURTHER WORK

In this work, we have presented an ensemble algorithm that works with the bounding boxes produced by object detection models, and, hence, it is independent of the underlying algorithm employed to construct those models. Our ensemble algorithm can be particularised with three voting strategies (affirmative, consensus, and unanimous) that have a different effect depending on the performance of the base models. Namely, the affirmative strategy works better when the detections of the base models are mostly correct (that is, there are few false positives) but several objects are left undetected (that is, there are lots of false negatives); the unanimous strategy obtains better results in the opposite case; and, the consensus strategy provides a better trade-off when there is not a significant difference between false negatives and false positives in the base models.

In addition, the ensemble method presented here has been employed to define a test-time augmentation procedure for object detection that improves the accuracy of object detection models. Moreover, the ensemble of models and test-time augmentation are the basis for data and model distillation, two semi-supervised learning techniques that can considerably reduce the number of images that must be manually annotated to train an object detection model; but that, up to now, have not been broadly adopted in the context of object detection due to the lack of a clear ensemble method.

As a by-product of this work, we have developed the open-source EnsembleObjectDetection library that implements all the methods presented here. This library provides support for models constructed with several algorithms and deep learning frameworks, and can be easily extended with others. Our methods and library have been tested with several datasets, and we have improved some models up to a 10%.

Several tasks remain as further work. First of all, we want to test our techniques with other datasets like COCO [28]. Moreover, we would like to test whether techniques like Soft-NMS [4], NMW [52], fusion [49] or WBF [45] produce better results than the NMS algorithm currently employed, or, at least, more fine-grained detections. Another interesting aspect to study is whether our ensembling procedure can be employed as a defense against adversarial attacks [47]. Finally, both in test-time augmentation and data distillation, it remains the question of deciding the image transformations that produce better results. Currently, this decision is taken by the users that employ our method, but it would be interesting to study automatic methods like the ones presented in [54].

ACKNOWLEDGEMENTS

This work was partially supported by Ministerio de Economía y Competitividad [MTM2017-88804-P], Ministerio de Ciencia, Innovación y Universidades [RTC-2017-6640-7], Agencia de Desarrollo Económico de La Rioja [2017-I-IDD-00018], and the computing facilities of Extremadura Research Centre for Advanced Technologies (CETA-CIEMAT), funded by the European Regional Development Fund (ERDF). CETA-CIEMAT belongs to CIEMAT and the Government of Spain. We also thank Álvaro San-Sáez for providing us with the stomata datasets.

REFERENCES

[1] W. Abdulla. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. `https://github.com/matterport/Mask_RCNN`, 2017.

[2] S. Akcay et al., 'Using Deep Convolutional Neural Network Architectures for Object Classification and Detection Within X-Ray Baggage Security Imagery', *IEEE Transactions on Information Fosics and Security*, **13**(9), 2203–2215, (2018).

[3] D. Ballabio, R. Todeschini, and V. Consonni, 'Recent Advances in High-Level Fusion Methods to Classify Multiple Analytical Chemical Data', *Data Handling in Science and Technology*, **31**, 129–155, (2019).

[4] N. Bodla et al., 'Soft-NMS: improving object detection with one line of code', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'17, pp. 5561–5569, (2017).

[5] C. Bucila, R. Caruana, and A. Niculescu-Mizil, 'Model compression: making big, slow models practical', in *Proceedings of the 12th International Conference on Knowledge Discovery and Data Mining*, KDD'06, pp. 535–541, (2006).

[6] B. R. Buttery, C. S. Tan, R. I. Buzzell, J. D. Gaynor, and D. C. MacTavish, 'Stomatal numbers of soybean and response to water stress', *Plant and Soil*, **149**(2), 283–288, (1993).

[7] Z. Cai and N. Vasconcelos, 'Cascade R-CNN: Delving Into High Quality Object Detection', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'18, pp. 6154–6162, (2018).

[8] A. Casado-García et al., 'CLoDSA: A Tool for Image Augmentation in Classification, Localization, Detection and Semantic Segmentation Tasks', *BMC in Bioinformatics*, **20**(323), (2019).

[9] A. Casado-García, J. Heras, and A. Sanz-Saez, 'Towards the automatic analysis of stomata images', in *Proceedings of the International Conference on Computer Aided Systems Theory (EUROCAST'19)*, (2019).

[10] T. Chen et al., 'MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems', *CoRR*, **abs/1512.01274**, (2015).

[11] B. Coüasnon and A. Lemaitre, *Handbook of Document Image Processing and Recognition*, chapter Recognition of Tables and Forms, 647–677, Springer International Publishing, 2014.

[12] A. V. Etten, 'You Only Look Twice: Rapid Multi-Scale Object Detection In Satellite Imagery', *CoRR*, **abs/1805.09512**, (2018).

[13] M. Everingham et al., 'The Pascal Visual Object Classes (VOC) Challenge', *International Journal of Computer Vision*, **88**(2), 303–338, (2010).

[14] M. Everingham et al., 'The Pascal Visual Object Classes Challenge: A Retrospective', *International Journal of Computer Vision*, **111**(1), 98–136, (2015).

[15] F. Chollet and others. Keras. `https://github.com/fchollet/keras`, 2015.

[16] M. C. Gobel, T. Hassan, E. Oro, and G. Orsi, 'ICDAR2013 Table Competition', in *12th ICDAR Robust Reading Competition*, ICDAR'13, pp. 1449–1453. IEEE, (2013).

[17] J. Guo and S. Gould, 'Deep CNN Ensemble with Data Augmentation for Object Detection', *CoRR*, **abs/1506.07224**, (2015).

[18] K. He et al., 'Deep Residual Learning for Image Recognition', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'16, pp. 770–778, (2016).

[19] J. Hosang, R. Benenson, and B. Schiele, 'Learning non-maximum suppression', *CoRR*, **abs/1705.02950**, (2017).

[20] J. Huang et al., 'Speed/accuracy trade-offs for modern convolutional object detectors', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'17, pp. 3296–3305, (2017).

[21] T. Huang, J. A. Noble, and A. I. L. Namburete, 'Omni-Supervised Learning: Scaling Up to Large Unlabelled Medical Datasets', in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, MICCAI'17, pp. 572–580, (2018).

[22] J. Hughes et al., 'Reducing Stomatal Density in Barley Improves Drought Tolerance without Impacting on Yield', *Plant Physiology*, **174**(2), 776–787, (2017).

[23] J. Irvin et al., 'Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison', in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, volume 33 of *AAAI'19*, pp. 590–597, (2019).

[24] J. Lee, S. Lee, and S-I. Yang, 'An Ensemble Method of CNN Models for Object Detection ', in *Proceedings of the IEEE Conference on-Information and Communication Technology Convergence (ICTC'18)*, ICTC'18, pp. 898–901, (2018).

[25] J. Li, J. Qian, and Y. Zheng, 'Ensemble R-FCN for Object Detectio', in *Proceedings of the International Conference on Computer Science and its Applications*, volume 474 of *CSA'17*, pp. 400–406, (2017).

[26] M. Li et al., 'TableBank: Table Benchmark for Image-based Table Detection and Recognition', *CoRR*, **abs/1903.01949**, (2019).

[27] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Keras retinanet. `https://github.com/fizyr/keras-retinanet`, 2017.

[28] T-Y. Lin et al., 'Microsoft COCO: Common Objects in Context', in *Proceedings of the European Conference on Computer Vision*, volume 8693 of *ECCV'14*, pp. 740–755, (2014).

[29] T-Y. Lin et al., 'Feature Pyramid Networks for Object Detection ', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'17, pp. 936–944, (2017).

[30] W. Liu et al., 'Ssd: Single shot multibox detector', in *Proceedings of the European Conference on Computer Vision*, volume 9905 of *ECCV 2016*, pp. 21–37, (2016).

[31] J. Minichino and J. Howse, *Learning OpenCV 3 Computer Vision with Python*, Packt Publishing, 2015.

[32] T. E. Oliphant, *A guide to NumPy*, volume 1, Trelgol Publishing USA, 2006.

[33] C. Peng et al., 'MegDet: A Large Mini-Batch Object Detector ', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'18, pp. 6181–6189, (2018).

[34] I. Radosavovic et al., 'Data Distillation: Towards Omni-Supervised Learning', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'18, pp. 4119–4128, (2018).

[35] S. Ramachandran et al., 'Using YOLO based deep learning network for real time detection and localization of lung nodules from low dose CT scans', in *Proceedings of Medical Imaging 2018: Computer-Aided Diagnosis*, p. 53, (2018).

[36] J. Redmon. Darknet: Open Source Neural Networks in C. `http://pjreddie.com/darknet/`, 2013–2016.

[37] J. Redmon et al., 'You Only Look Once: Unified, Real-Time Object Detection ', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'16, pp. 779–788, (2016).

[38] J. Redmon and A. Farhadi, 'Yolov3: An incremental improvement', *CoRR*, **abs/1804.02767**, (2018).

[39] S. Ren, K. He, R. Girshick, and J. Sun, 'Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks', *Advances in Neural Information Processing Systems*, **28**, 91–99, (2015).

[40] A. Rosebrock, 'Intersection over union (iou) for object detection', in *PyImageSearch*, (2018). `https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/`.

[41] P. Simard et al., 'Tangent prop – a formalism for specifying selected invariances in an adaptive network', in *Proceedings of the International Conference on Neural Information Processing Systems*, volume 4 of *NIPS'91*, pp. 895–903, (1992).

[42] P. Simard, D. Steinkraus, and J. C. Platt, 'Best practices for convolutional neural networks applied to visual document analysis', in *Proceedings of the International Conference on Document Analysis and Recognition*, volume 2 of *ICDAR'03*, pp. 958–964, (2003).

[43] K. Simonyan and A. Zisserman, 'Very deep convolutional networks for large-scale image recognition', in *Proceedings of the 3rd International Conference on Learning Representations*, ICLR'15, (2015). `http://arxiv.org/abs/1409.1556`.

[44] I. Sirazitdinov et al., 'Deep neural network ensemble for pneumonia localization from a large-scale chest x-ray database', *Computers and Electrical Engineering*, **78**, 388–399, (2019).

[45] R. Solovyev and W. Wang, 'Weighted Boxes Fusion: ensembling boxes for object detection models', *CoRR*, **abs/1910.13302**, (2019).

[46] C. Y. Suen et al. ICDAR2019 Table Competition. `http://icdar2019.org/`, 2019.

[47] F. Tramèr et al., 'Ensemble Adversarial Training: Attacks and Defenses', in *Proceedings of the 6th International Conference on Learning Representations*, ICLR'18, pp. 1–20, (2018).

[48] N. Vo et al., 'Ensemble of Deep Object Detectors for Page Object Detection', in *Proceedings of the International Conference on Ubiquitous Information Management and Communication*, IMCOM'18, pp. 1–5, (2018).

[49] P. Wei, J. E. Ball, and D. T. Anderson, 'Fusion of an Ensemble of Augmented Image Detectors for Robust Object Detection', *Sensors*, **18**(3), 1–21, (2018).

[50] *Ensemble Machine Learning: Methods and Applications*, eds., C. Zhang and Y. Ma, Springer, 2012.

[51] Z-Q. Zhao et al., 'Object Detection With Deep Learning: A Review', *IEEE Transactions on Neural Networks and Learning Systems*, 1–21, (2019).

[52] H. Zhou et al., 'CAD: Scale Invariant Framework for Real-Time Object Detection', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'17, pp. 760–768, (2017).

[53] *Introduction to Semi-Supervised Learning*, eds., X. Zhu and A. B. Goldberg, Morgan & Claypool Publishers, 2009.

[54] B. Zoph et al., 'Learning Data Augmentation Strategies for Object Detection', *CoRR*, **abs/1906.11172**, (2019).