SCALE-AWARENESS FOR MORE ACCURATE OBJECT DETECTION USING MODIFIED SINGLE SHOT DETECTORS

V. Tsironis[1,*], C. Stentoumis[1], N. Lekkas[1], A. Nikopoulos[1]

[1] up2metric P.C., GR-11521, Michail Mela 21, Athens, Greece -
(tsironisbi, christos, nick.lekkas, alexandros.nikopoulos)@up2metric.com

Commission II, WG II/III

KEY WORDS: Object Detection, Classification, Image Processing, Small Object Detection.

ABSTRACT:

Object detection performance is directly related to the apparent size of the object to be detected, thus most state-of-the-art algorithms dedicate different detection heads for each object size. In this work, we propose an end-to-end pipeline to adapt a single-shot object detector (SSD) to the underlying object size distribution of the target detection domain. Our contributions are the adjustments to the detector architecture and the introduction of a novel batch sampling method. To validate the effect of our method, we chose a task-specific highly specialized object detection and classification dataset of tomato fruits that apart from bounding box information, it also contains class information for three ripening stages of each tomato fruit.
More specifically, the major motivation and contributions are discussed in relation to the recent bibliography. Next, an extensive analysis of our pipeline is presented, where the concept of scale alignment is thoroughly presented along with the novel sampling method. Following the results of a series of experiments, we conclude that our pipeline significantly improves over the "off-the-shelf" base single-shot detector and its detection performance is comparable to more elaborate algorithms, especially if we measure detection performance slightly disregarding box localization. Lastly, we include a stratified ablation study in the closing sections where we measure the impact of each step along our proposed SSD adaptation pipeline.

1. INTRODUCTION

Object Detection (OD) is one of the most challenging problems in computer vision, aiming to determine the location of certain objects on images and videos, as well as to classify them among specific classes. Typically, the localization of an object is described by a bounding box. OD is an enabling technology for a wide range of applications such as autonomous driving, object and people tracking, and in different fields, such as space, security, transportation, and industry. Nowadays, several either two-stage detectors, e.g. region proposal CNN (Girshick et al., 2014), Faster RCNN (Ren et al., 2015), and Mask RCNN (He et al., 2020), or one-stage detectors, e.g. RetinaNet (Lin et al., 2017), Single Shot Detector (Liu et al., 2016) and YOLO (Redmon and Farhadi, 2018), are considered standard practise for common detection tasks.

Most often, OD models are tied to standard datasets used for pretraining, such as the COCO (Lin et al., 2014) or the Open-Images (Kuznetsova et al., 2020) dataset, which affects their response to object sizes different than the ones in the dataset. Existing works, as discussed in the following sub-section, are based on a variety of different approaches that tackle the problem of multi-scale object detection with remarkable efficiency. However, most of these approaches are either new architectures all together, or heavily tied to a specific type of detection model. Hence, our motivation lies in the need for a higher-level universal approach that could potentially benefit a handful of target detection model architectures into significantly improving their respective detection performance. That said, we propose a pipeline to accurately align detection set of scales to the target domain object size distribution using single-shot object detectors. We validate our proposition on "*small object detection*" which is a research sub-field of particular interest due to its inherent "by-definition" difference in object size distribution compared to standard OD datasets. The evaluation is based on the task-specific but intriguing *TomatOD* dataset (Tsironis et al., 2020).

This work introduces two major contributions: firstly, we present a straightforward pipeline to configure an object detection model so that it can perform optimally in the size range of a given target object; secondly, we introduce a novel training strategy which significantly improves the detection performance of a model avoiding the complexity of other similar methods. We provide source code for our scale-aware SSD implementation at `https://github.com/up2metric/scale-aware-SSD/`.

2. RELATED WORK

In this section we present some recent advancements in the sub-field of "*small object detection*" and we further focus on related work that manipulates the scale space of the detection, since its more closely related to our approach. Finally, we briefly discuss the Single-Shot Detector (SSD) MobileNet v2 object detection algorithm upon which we base the demonstrator of our method.

2.1 Small object detection

A plethora of real-world applications is heavily dependent on accurate small object detection such as industrial (Qiu et al., 2019), agriculture (Wang et al., 2020; Zhang et al., 2020), sports (Xu et al., 2018), or security (Abe et al., 2008) applications. Small object detection is of particular interest for earth observation purposes such as the case of object detection in optical airborne (Wang et al., 2018), or satellite (Ren et al., 2018; Zhang et al., 2019) very high resolution data.

* Corresponding author

Standard detection algorithms typically perform poorly in such scenarios, compared to their performance on more common datasets like COCO (Lin et al., 2014) or Open-Images (Kuznetsova et al., 2020). In the literature there are several approaches to mitigate this problem (Tong et al., 2020); some include the design of entirely new detectors, e.g. they employ generative adversarial networks (GANs) to detect small objects in a context-aware manner (Bai et al., 2018). Similarly, (Bell et al., 2016) perform multi-scale ROI pooling with their inside-outside network (ION) resulting in improved performance for small object detection. Most approaches, however, rely on modifying existing models to improve their detection performance on smaller objects as in the work of (Cui et al., 2018) where a set of de-convolution layers are coupled to certain feature maps of an SSD detector. (Cao et al., 2019) present an adaptation of Faster RCNN where a multi-scale feature fusion technique is used to improve small object detection performance. Also notable is the work of (Kisantal et al., 2019) where the authors discuss augmentation techniques that are beneficial for boosting a model's small object detection performance, while (Chen et al., 2021) further discuss some scale-aware augmentation techniques.

When dealing with the "small object detection" problem the model architecture is not the only unknown that needs to be addressed as the training strategy also plays a crucial role in building an accurate small object detector. In bibliography there have been proposed several such strategies like SNIP (Singh and Davis, 2018), an approach where, during training, only selected ground truth boxes and proposals of a specified size range for a particular resolution are used to back-propagate the gradients through the network. Similarly, SNIPER (Singh et al., 2018) only processes context regions around ground truth boxes at the proper scale instead of processing a whole multi-scale pyramid of features. Finally, (Bodla et al., 2017) propose a soft-NMS post-process to integrate detections from a variety of scales ultimately improving detection performance for small objects.

2.2 Scale-awareness in object detection

Context and *scale awareness* are critical qualities of a detection system to properly identify objects in a wide range of potential sizes. Most existing detection systems have some multi-scale structure in place, either via multi scale fusion like the region proposal network (RPN) found in faster region-based CNN (FRCNN), or an FPN-based structure similar to that in Retina models, or even a pyramidal feature hierarchy like the one present in SSD detectors. Furthermore there is a plethora of approaches that are specifically designed to incorporate multi-scale features resulting in *scale-invariant* detectors like (Sambolek and Ivasic-Kos, 2021) or (Zhang et al., 2018).

However, there are some algorithms that are specifically optimized for *scale awareness* such as SAN from (Kim et al., 2018), a scale-aware network that models scale-dependent relationships by mapping convolutional features from different scales onto a scale-invariant subspace. (Liu et al., 2017) use a recurrent rolling mechanism to propagate along the scale dimension given the computation of a single fixed-scale feature map. (Li et al., 2019) implement a parameter sharing technique across multiple parallel detection branches, each one having a different receptive field, thus corresponding to a different scale.

2.3 SSD MobileNet v2 detector

This work builds upon the architecture of the SSD MobileNet v2 detector, which has a typical SSD architecture, i.e. it is a multi-scale feature learning architecture exploiting the paradigm of *"Pyramidal Feature Hierarchy"*. In this particular variant the detector exploits the fully convolutional *encoder* part of the MobileNet v2 image classification network (Sandler et al., 2018). MobileNet v2 is a very deep yet lightweight architecture because it leverages an inverted residual structure to construct the very computational efficient *bottleneck* blocks. We chose SSD MobileNet v2 detector as the base for our methodology for two key reasons: firstly, it exploits a paradigm that allows for a streamlined and straightforward implementation of our pipeline; secondly, it is an ultra lightweight detector capable of performing adequately even on low-powered embedded devices with little compromise on input image resolution. In detail, the direct attachment of the detection heads to certain layers of the encoder, allows for the direct manipulation of the receptive field corresponding to each detection head. Our goal has been to develop a detection system capable of running near real-time on embedded devices, like the Nvidia Jetson Nano, while still retaining the detection performance of more complex detectors, like FRCNN. In this context, the SSD MobileNet v2 object detector is a perfect fit for our needs.

3. METHODOLOGY

The *scale-aware* approach presented in this work constitutes a generic technique to adapt mainstream single-shot detectors so that they can take advantage of the singular object scale distribution of the target training and testing dataset. Here, we present the novel detector adaptation pipeline in a more streamlined fashion, since we focus on the adaptation of SSD MobileNet v2 for the task-specific *TomatOD* dataset (Tsironis et al., 2020); however, our technique can expand to literally any single-shot detector and dataset.

3.1 Target dataset scale distribution

The first step of our method is to identify the overall object scale distribution of the target dataset. To accomplish this we calculate the percentile relative size of each bounding box, i.e. the proportion of the diagonal size of each box over the diagonal size of the image. Figure 1 shows the object scale distribution in the case of *TomatOD*.

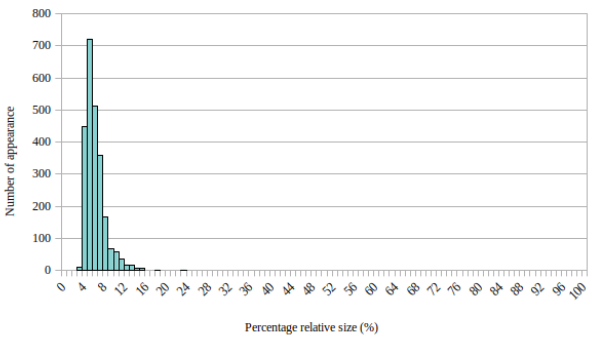


Figure 1. *TomatOD* dataset percentile object size distribution.

One can observe that the target dataset consists of objects with a relative size varying from 5% to 16%, having a few outliers

on either side. Given that we target a 512 × 512 input image size we directly infer that the absolute box size ranges mostly between 25 to 80 pixels. In the following step, we will leverage this information to design a task-specific version of the SSD MobileNet v2 detector.

3.2 Scale alignment for detection heads and anchor selection

Most deep-learning based object detectors use a multi-scale grid of *"default boxes"* or *"anchors"* to define the basic detection scale set, that corresponds to a set of *detection heads* in the model's architecture, and an a priori spatial distribution of *possible object locations* on any input image. Our goal is to determine an *optimal set of anchors* in order to better align the detector's basic detection scale set to the absolute object size distribution of the target dataset, as discussed earlier.

To accomplish this we investigate the *encoder* part of the SSD MobileNet v2 detector and for each block we calculate its respective receptive field (Table 1). In its standard implementation SSD MobileNet v2 uses *block-14* and *output* blocks of the MobileNet v2 encoder to attach the first two detection heads, which correspond to the two *"smallest"* scales of the anchor set. For the rest detection heads a set of extra features layers is used.

Setting	Output size (px)	Receptive field (px)
Input	512 × 512	1
init-conv	256 × 256	3
Block-1	256 × 256	7
Block-2	128 × 128	11
Block-3	128 × 128	19
Block-4	64 × 64	27
Block-5	64 × 64	43
Block-6	64 × 64	59
Block-7	32 × 32	75
Block-8	32 × 32	107
Block-9	32 × 32	139
Block-10	32 × 32	171
Block-11	32 × 32	203
Block-12	32 × 32	235
Block-13	32 × 32	267
Block-14	16 × 16	299
Block-15	16 × 16	363
Block-16	16 × 16	427
Output	16 × 16	491

Table 1. Block-wise receptive fields and block output size for 512 × 512 input image size for the MobileNet v2 encoder.

On the other hand we opted for *block-5* and *block-8* blocks as the attachment point for our first two detection heads, while we maintained the extra features layers approach for the rest of the remaining detection heads (Figure 2). These design choices resulted in a set of {43 / 107 / 141 / 207} pixels for the effective receptive field of the feature maps where the detection heads attach. In contrast, the original implementation had, respectively, a set of {299 / 491 / 557 / 687} pixels. This design choice was driven by the pre-calculated absolute object size distribution for the target dataset, where the dominant size range was found to be between 25 and 80 pixels with some rare instances on both extremes, e.g. a few instances were as small as 15-20 pixels and some other as big as 120-140 pixels. By intuition, we argue that there should be a relation between the anchor size and the effective receptive field of the respective detection head, however in any case the effective receptive field should be larger than the corresponding anchor diagonal size. After several experiments discussed in Section 5, we opted for a set of {25 / 38 / 58 / 87} pixels as the corresponding reference diagonal sizes for the 1:1 aspect ratio anchors for each detection scale that describes optimally the absolute object size distribution of the target dataset.

Overall, the average receptive field (RF) to anchor size ratio was about 2.3 while for the original implementation it was about 10.6 (Table 2). This indicates an increased *alignment* of the model's detection scales (dataset-dependent) to the available image area for detecting an object (architecture-dependent). We argue that such *alignments scores* should preferably lie within an optimal range of [1.5, 2.5], as derived from the experiments in Section 5.

	Original		Custom	
Anchor size	RF	A-score	RF	A-score
25 px	299 px	12.0	43 px	1.7
38 px	491 px	12.9	107 px	2.8
58 px	557 px	9.6	141 px	2.4
87 px	687 px	7.9	207 px	2.4
mean		**10.6**		**2.3**

Table 2. Alignment scores (A-score) for original and custom SSD architecture given a dataset-specific anchor sizes set.

3.3 Training techniques and Dynamic Negative Sampling

Apart from the aforementioned architectural interventions, we introduced a set of training techniques in order to maximize our model's potential. Specifically, we addressed the class imbalance problem by introducing a strategy for determining a set of class weights for the foreground classes. This strategy is discussed in the section 5. Furthermore, we introduced a novel batch-sampling method, namely *"Dynamic Negative Sampling"* (DNS) to improve on the original 3:1 negative-to-positive hard-mining approach of the original SSD implementation. During the sampling procedure, DNS will keep all positive training samples and all misclassified negative samples (negative samples that are classified as positive classes). This procedure will lead to mini-batches of variable lengths in each training step, depending on the number of misclassified negative samples.

DNS was inspired by the realisation that a fixed positive / negative sampling ratio can really hurt the algorithm in cases where the negative samples are very easy to detect and correctly classify. In such cases, the constant number of negative samples can really weaken the positive signals that try to differentiate between different classes, making the training procedure more difficult and time-consuming. The intuition behind the DNS algorithm is that the variable mini-batch lengths can create an *adversarial effect* in the training procedure. As the length of each mini-batch shrinks, positive samples will start to dominate it, generating stronger positive signals in each training step. Those signals will start to classify more positive samples correctly, but they will also force more samples to be misclassified, hence turning them into *negative*, resulting in larger mini-batch sizes. On the contrary, as the length of each mini-batch increases, negative samples will start to dominate it, forcing the training procedure to focus on eliminating them, resulting eventually in smaller mini-batch sizes. Those two contradictory states alternate continuously during the training procedure, leading eventually to a best-of-both-states equilibrium, in which only an essential amount of negative samples is kept in each mini-batch.

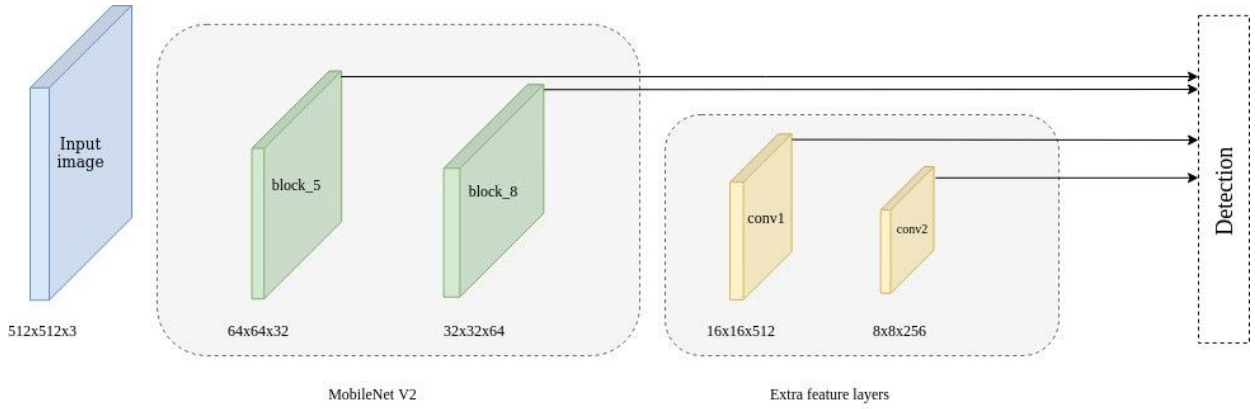


Figure 2. Target dataset scale-aware custom SSD architecture.

4. RESULTS

In this section we discuss the detection performance of our approach is discussed in comparison to several industry standard object detectors. Initially, we present the dataset used to train and evaluate our models along with the complete experimental setup that we used. A quantitative analysis of the results follows where all models are compared using several variants of the, widely used for detection tasks, *mAP* family of metrics. Finally, some qualitative observations are being noted about the difference of performance across all models tested.

4.1 Experimental setup

TomatOD dataset was chosen to evaluate the proposed method for adapting SSD for a handful of reasons. At first, TomatOD has a radically different object size distribution compared to standard datasets, such as COCO or Open-Images, yet it does not qualify as a "small object"-only dataset, such as DoTA. This variance in target object sizes is ideal to demonstrate the impact of our scale-dependent approach. Secondly, the TomatOD dataset is a *real-world* dataset, i.e. it has a constrained amount of labelled images, whose image quality (resolution, sharpness and brightness) is close to what should be expected from low-cost machine vision cameras in real-world conditions. At last, the TomatOD is an overall tough dataset for object classification since its target categories are strongly correlated since they vary among three different ripening stages of the same object (tomato fruit).

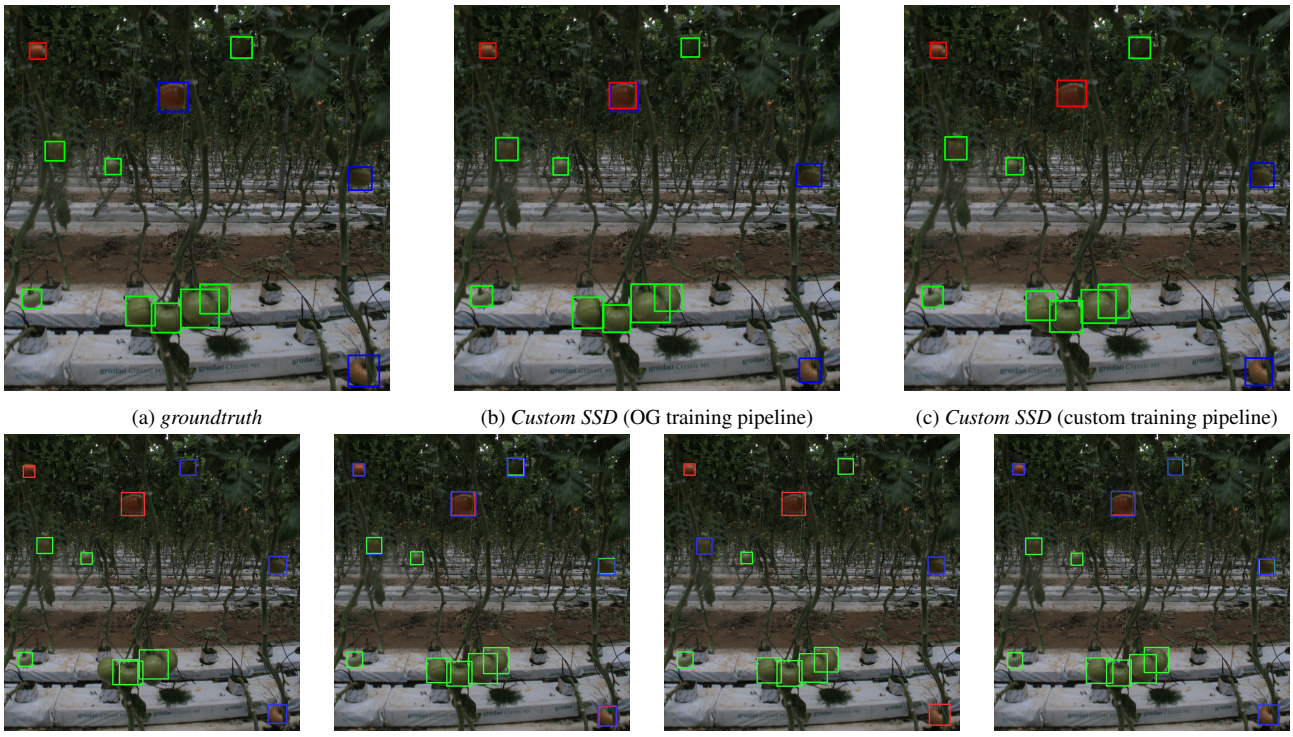
We compared our approach to three other object detection models: the SSD MobileNet v2 from the Tensorflow OD API, the Faster RCNN ResNet50 FPN, the Faster RCNN ResNet101 FPN and the Retina ResNet50 from the Detectron 2 framework, all pretrained at the COCO dataset. The first one, draws the baseline performance since it is the model we based our methodology on, while the other two models represent the state-of-the-art for object detection applications. We selected an input image size of 512×512 for all the detection models, including ours. We opted for a minimal set of transformations / augmentations, apart from the obvious resize to 512×512, which includes random horizontal flip and random sized crop. To properly evaluate our model we used the well established *mAP* and *mAP:0.5* object detection metrics from the COCO dataset. Furthermore, we measured inference time for all models on a system based on an Nvidia RTX 3090 graphics card.

4.2 Quantitative Results

The overall best results for each algorithm are displayed in Table 3. Regarding our custom SSD model we included two distinct variants; one that differs only architecture-wise to the base SSD MobileNet v2 detector and another one that includes all training techniques that we discussed in subsection 3.3. Comparing our model to the one its based on, we immediately observe that both evaluation metrics are significantly improved. In detail, the *mAP* metric improved 12.5% while the *mAP:0.5* metric improved by 20.9%; this outstanding improvement highlights the potential of single-shot detectors to produce adequately good detection results when are biased towards the target size distribution. Specifically, we argue that by assigning the detection heads to *earlier* feature maps, to associate the effective receptive field to the anchor box size, the resulting anchor boxes grid is much denser and thus better aligned *a priori* to the ground-truth bounding boxes.

Taking into account all the proposed training techniques our model performs better than both FRCNN variants on the (crucial) *mAP:0.5* metric, and lacks by 4-6% on the *mAP* metric. Retina R50 is the overall best performing detector outperforming our model by 12.2% and 1.6% on the *mAP* and the *mAP:0.5* metric respectively. Overall, our custom SSD model posts a strong performance on the *mAP:0.5* metric, a metric that measures the ability of a model to successfully detect the existence of an object without enforcing strict bounding box localization criteria. On the other hand the generic *mAP* metric measures both the detection and localization capability of a detector. Given the simplistic and straightforward structure of the SSD approach, when compared to complex architectures like FRCNN and Retina, our model is bound to inferior localization performance due to its SSD architecture.

Regarding computational efficiency, we measured the mean inference speed in *frames per second (FPS)*. As presented in Table 4, our custom SSD model not only matches, but improves on the computational efficiency of the SSD Mobilenet v2. This is an expected result since our model attaches its detection heads earlier to the MobileNet v2 encoder resulting in fewer operations in total. Compared to FRCNN and Retina algorithms our model outperforms them both by a significant margin of about 5-7$\times$. Our model was also tested for computational performance on embedded platforms, yielding about 8-10 FPS mean inference speed on a Nvidia Jetson Nano. On this specific platforms all FRCNN and Retina models performed much worse, resulting in $<$1 FPS inference performance.



(a) *groundtruth* (b) *Custom SSD* (OG training pipeline) (c) *Custom SSD* (custom training pipeline)

(d) *SSD MobileNet v2* (e) *Faster RCNN R50* (f) *Retina R50* (g) *Faster RCNN R101*

Figure 3. Visualized detection results for all tested detection models on a sample test image.

Detector	mAP	mAP:0.5
	%	%
FRCNN R50 FPN	43.8	83.6
FRCNN R101 FPN	41.0	79.9
Retina R50	49.7	88.0
SSD MobileNet v2	22.3	57.8
Custom SSD *(OG train pipeline)*	34.8	78.7
Custom SSD *(custom train pipeline)*	37.3	86.4

Table 3. Overall detection results (after hyperparameter tuning) for the *TomatOD* dataset.

Detector	Input size	mean FPS
FRCNN R50 FPN	512 × 512	51
FRCNN R101 FPN	512 × 512	31
Retina R50	512 × 512	43
SSD MobileNet v2	512 × 512	153
Custom SSD *(OG train pipeline)*	512 × 512	**200**
Custom SSD *(custom train pipeline)*	512 × 512	**200**

Table 4. Inference performance on an Nvidia RTX 3090 based system.

4.3 Qualitative results

In Figure 3 some characteristic cases of the detection performance difference between the models are showcased. In the case of our model *("Custom SSD")* we can argue that, in general, performs adequately well. In further detail, we accomplish detection performance similar to much more complex architectures like FRCNN and Retina. Classification-wise the performance of our detector is equivalent to that of FRCNN and Retina but localization-wise it can't match the, by-design superior, capacity of the other two models. Compared to the baseline SSD MobileNet v2, however, we can easily observe that our approach performs much better in every aspect. The baseline model suffers from inability to correctly classify the object into the right category (in Figure 3 there are two such cases) while, also, there are object instances that failed to be detected all together (in Figure 3 there are several undetected instances on the tomato cluster). The localization performance of the baseline method is only lacking by a slim margin compared to ours for the cases where both detectors have successfully identified an object. That observation may indicate that the overall localization accuracy of the model might be constrained by the common underlying SSD architecture. Finally, we can observe that our best model, i.e. the one trained with our proposed pipeline, excels in avoiding multiple detections of the same object when much more complex detectors fail to do so. We believe that this is indicative of the overall impact of our DNS sampler in conjunction with the proper utilisation of class balancing weights during training for the foreground classes.

5. ABLATION STUDY

In this section we will evaluate the impact of each individual component of our method. The study is presented in a stratified way so that it represents our proposition on how to develop a similar custom detector in virtually any dataset. Step-by-step, we initially search for an optimum set of anchors, next we determine sequentially the best choice of foreground class weights and the best performing batch sampling method for hard positive/negative sample mining. Finally, we perform some performance stability tests over a series of repeated experiments and evaluate the potential impact of pre-training our model on a common "small-object" detection dataset.

5.1 Anchor search

Our first concern is no other than selecting an optimal set of anchor sizes. We refer to the diagonal size of the 1:1 aspect ratio base anchors. It's common that, size-wise, the other anchors (different aspect ratios) derive in a procedural way from those base anchors so we can safely constraint our search only to base 1:1 anchors. In the common case for an SSD architecture there are 4 such base anchors, a design we choose to adopt as well.

To come up with a selection of possible base anchor sets we need some information from the target dataset. As heavily discussed during previous sections we compute the absolute diagonal object size distribution for the target domain. In the case of TomatOD it is a range of 25-80 pixels with some additional outliers. Next, we need to target a specific "alignment" score, most commonly a target value in the range 1.8-2.5. Last, we adopt a systematic multiplicative rule to go from one scale to the next. In our case we tried both a 1.5× and a 1.7× multiplication factor.

Anchor set	a-score	mAP	mAP:0.5
		%	%
20/41/81/164	1.9	31	74
25/38/58/87	2.3	**35**	**79**
25/43/74/124	1.9	**35**	**79**
28/45/72/115	1.9	34	**79**
30/45/60/80	2.2	**35**	75
30/45/60/90	2.1	34	78

Table 5. Anchor search experiments
Four anchor sizes per set, anchor size in pixels.

All experiments were conducted given a basic setup, e.g. the original SSD 3:1 batch sampler, no class weighting and xavier-initialized weights where Image-Net pretrained ones were unavailable. In Table 5 all sets under investigation are presented. Two sets yielded the best results. Both sets have 25 pixels as the smallest anchor diagonal size, however one has a 1.5× and the other a 1.7× multiplication factor. Furthermore, in both cases we have an acceptable "alignment" score, hence both sets are excellent choices for our case. However, we selected {25/38/58/87} pixels for our model because as seen in Figure 1 there are more "small" objects than "bigger" ones in our dataset so we opted for the set that "covers" in a denser way the "small" object spectrum of the size distribution.

5.2 Class imbalance

Apart from being a challenging task-specific detection dataset, *TomatOD* dataset also poses a hard object classification problem for two main reasons: at first, as seen in Figure 4, it is characterized by heavy class imbalance in favour of 1 of the 3 class (unripe); secondly, the classification task is an inter-species classification task, so there is a continuous, and not discrete as usual, transition between classes "unripe"-"semi-ripe" and "semi-ripe"-"ripe".

Figure 4. *TomatOD* dataset object class distribution.

To countermeasure these challenges we experimented with three approaches regarding balancing the classes through class weights. Initially we used a set of weights directly derived from the class distribution of the dataset, i.e. unripe(u):1, semi-ripe(s):4, ripe(r):3.7. Next, we tried a different approach by selecting weights that heavily "promote" the "middle" class (semi-ripe), i.e. u:1, s:5, r:1. Finally, we opted for a hybrid approach that is essentially a combination of the previous two, i.e. unripe(u):1, semi-ripe(s):8, ripe(r):3.7.

Weights set	mAP	mAP:0.5
	%	%
no weights	35	79
u:1, s:4, r:3.7	**36**	**82**
u:1, s:8, r:3.7	33	80
u:1, s:5, r:1	**36**	80

Table 6. Class weights search experiments.
(u: unripe, s: semi-ripe, r: ripe)

For the experiments we used a fixed *25/38/58/87 (px)* set of anchors, the original 3:1 batch sampling process and we used a randomly (xavier) initialized model, except for the encoder part where ImageNet pretrained weights were used. As shown in Table 6 the classical approach of using class weights derived from the target distribution yielded the best results. The overall improvement was about 1% for the *mAP* metric and 3% for the *mAP:0.5* metric. It is also worth noting that virtually all three weight computation methods resulted in superior performance over the no-weights baseline for the critical *mAP:0.5* metric, and only the *hybrid* approach resulted in worse performance for the *mAP* metric.

5.3 Batch sampling method

Apart from the architectural interventions presented in this work, we proposed a novel batch sampling method, namely *Dynamic Batch Sampling (DNS)*. In this subsection we investigate the effect *DNS* has when compared to the standard 3:1 maximal negative-to-positive hard mining batch sampling algorithm for the SSD models. For completeness we also conducted some experiments with a naive 1:1 negative-to-mining hard sampling method. In Table 7 the results of this study are presented. The experiments were conducted given a *25/38/58/87 (px)* set of anchors, unripe(u):1, semi-ripe(s):4, ripe(r):3.7 class weights and xavier-initialized weights where ImageNet pretrained ones were unavailable.

Sampling method	mAP	mAP:0.5
	%	%
basic (3:1)	36	82
DNS	**37**	**86**
1:1	34	79

Table 7. Hard-mining batch sampling method search.

Interpreting the results we can clearly observe that DNS provides a significant boost especially for the *mAP:0.5* metric where the overall improvement is about 4%. Smaller gains about 1% are observed for the *mAP* metric. That model, in particular, is the one that achieved the best metric values for both *mAP* and *mAP:0.5* metric. Regarding 1:1 hard-mining sampler, it performed poorly as expected. That series of experiments, however, indicated another strong advantage of *DNS* versus the 3:1 sampler. DNS converges at a much faster rate to the solution, in our case in about 40-50 epochs of training, while the standard approach required hundreds (150-250) of epochs. That result proves experimentally our argument, discussed in Section 3, about the adversarial nature of the *DNS*.

5.4 Initialization impact and pretraining

One crucial difference of our approach when compared to off-the-shelf detection solutions is that our model *is not pretrained* on a generic detection dataset, such as COCO or OpenImages. In this series of experiments we investigate whether our (in part) randomly initialized non-pretrained model can benefit from pretraining on a well-known dataset for *small* object detection such as the DoTA dataset (Xia et al., 2018). To assess the randomness of the final resulting models, we also repeated each experiment 15 times. All experiments were conducted using {25 / 38 / 58 / 87} set of anchors, (unripe: 1, semi-ripe: 4, ripe: 3.7) class weights and our *DNS* hard-mining batch sampling method. The results are shown in Table 8

	mAP			mAP:0.5		
Pretrained	mean	std	max	mean	std	max
	%	%	%	%	%	%
no	**33.2**	1.9	**37.3**	**81.3**	2.7	**86.4**
yes	32.16	**1.5**	34.3	79.0	**2.2**	82.4

Table 8. Metrics variance over a course of experiments for a DoTA pretrained custom SSD model versus a randomly initialized one.

Reviewing the results, one can observe that the pretrained models have a smaller variance in their performance; this is to be expected since the pretrained model has, in theory, a much better initial state. As a side effect, the pretrained models also converged significantly faster to a stable solution. However, quite unexpectedly, the non-pretrained model has generally ended-up being significantly superior compared to the DoTA pretrained ones. We interpret this result by arguing that a randomly (xavier) initialized model has the potential to *discover* better overall solutions that might exist out of the area of influence of the (fixed) pretrained initial state. In other words, it is a situation similar to the *"exploration vs exploitation" dilemma* that we commonly face in Reinforcement Learning problems.

At last, we should note that we experimented with a few augmentation options, specifically *Random Horizontal Flip* and *Random Crop*, but this caused a reduction in performance by 1-2% on both metrics. Although augmentations are crucial to the generalization of a model for real world applications, their effect was statistically insignificant in this case.

6. CONCLUSION

In this work, we investigated the problem of generating a lightweight purpose-built object detector that can potentially achieve performance directly comparable to state-of-the-art two-stage object detectors without sacrificing, if not improving, the computational efficiency of the base single-shot detector it was built upon. Our work proved that it is possible to specifically adapt a lightweight SSD model with minimal effort and without the need of pretraining to come up with a model that can perform similar to much more complicated algorithms, while remaining deployable in low-power and low-cost platforms. The proposed method is easily generalizable to several other single-shot architectures (e.g. RPN, Retina, and YOLO among others), while our DNS sampling method can be used "as-is" in almost any recent object detection algorithm.

As a future work, an investigation of the effect of *DNS* for a variety of object detectors would be very interesting. Furthermore, the possibility of designing custom detectors without relying on complex ImageNet-pretrained encoder networks, but instead opting for lightweight FCN xavier-initialized, should be explored. In such a scenario, real-time task-specific object detection on high resolution images on virtually any embedded device would become a reality.

ACKNOWLEDGEMENTS

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code T1EDK-04171).

REFERENCES

Abe, D., Segawa, E., Nakayama, O., Shiohara, M., Sasaki, S., Sugano, N., Kanno, H., 2008. Robust small-object detection for outdoor wide-area surveillance. *IEICE transactions on information and systems*, 91(7), 1922–1928.

Bai, Y., Zhang, Y., Ding, M., Ghanem, B., 2018. Sod-mtgan: Small object detection via multi-task generative adversarial network. *Proceedings of the European Conference on Computer Vision (ECCV)*, 206–221.

Bell, S., Zitnick, C. L., Bala, K., Girshick, R., 2016. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2874–2883.

Bodla, N., Singh, B., Chellappa, R., Davis, L. S., 2017. Soft-nms–improving object detection with one line of code. *Proceedings of the IEEE international conference on computer vision*, 5561–5569.

Cao, C., Wang, B., Zhang, W., Zeng, X., Yan, X., Feng, Z., Liu, Y., Wu, Z., 2019. An improved faster R-CNN for small object detection. *IEEE Access*, 7, 106838–106846.

Chen, Y., Li, Y., Kong, T., Qi, L., Chu, R., Li, L., Jia, J., 2021. Scale-aware Automatic Augmentation for Object Detection. *arXiv preprint arXiv:2103.17220*.

Cui, L., Ma, R., Lv, P., Jiang, X., Gao, Z., Zhou, B., Xu, M., 2018. MDSSD: multi-scale deconvolutional single shot detector for small objects. *arXiv preprint arXiv:1805.07009*.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2020. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2), 386-397.

Kim, Y., Kang, B.-N., Kim, D., 2018. San: Learning relationship between convolutional features for multi-scale object detection. *Proceedings of the European Conference on Computer Vision (ECCV)*, 316–331.

Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J., Cho, K., 2019. Augmentation for small object detection. *arXiv preprint arXiv:1902.07296*.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Kolesnikov, A. et al., 2020. The open images dataset v4. *International Journal of Computer Vision*, 1–26.

Li, Y., Chen, Y., Wang, N., Zhang, Z., 2019. Scale-aware trident networks for object detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017. Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision*, 2980–2988.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., 2014. Microsoft coco: Common objects in context. *European Conference on Computer Vision*, Springer, 740–755.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A. C., 2016. Ssd: Single shot multibox detector. *European conference on computer vision*, Springer, 21–37.

Liu, Y., Li, H., Yan, J., Wei, F., Wang, X., Tang, X., 2017. Recurrent scale approximation for object detection in cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 571–579.

Qiu, Z., Wang, S., Zeng, Z., Yu, D., 2019. Automatic visual defects inspection of wind turbine blades via YOLO-based small object detection approach. *Journal of Electronic Imaging*, 28(4), 043023.

Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*.

Ren, Y., Zhu, C., Xiao, S., 2018. Small object detection in optical remote sensing images via modified faster R-CNN. *Applied Sciences*, 8(5), 813.

Sambolek, S., Ivasic-Kos, M., 2021. Automatic Person Detection in Search and Rescue Operations Using Deep CNN Detectors. *IEEE Access*, 9, 37905–37922.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520.

Singh, B., Davis, L. S., 2018. An analysis of scale invariance in object detection snip. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3578–3587.

Singh, B., Najibi, M., Davis, L. S., 2018. Sniper: Efficient multi-scale training. *arXiv preprint arXiv:1805.09300*.

Tong, K., Wu, Y., Zhou, F., 2020. Recent advances in small object detection based on deep learning: A review. *Image and Vision Computing*, 97, 103910.

Tsironis, V., Bourou, S., Stentoumis, C., 2020. TomatOD: Evaluation of Object Detection Algorithms on a New Real-World Tomato Dataset. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43, 1077–1084.

Wang, C., Bai, X., Wang, S., Zhou, J., Ren, P., 2018. Multiscale visual attention networks for object detection in VHR remote sensing images. *IEEE Geoscience and Remote Sensing Letters*, 16(2), 310–314.

Wang, Q.-J., Zhang, S.-Y., Dong, S.-F., Zhang, G.-C., Yang, J., Li, R., Wang, H.-Q., 2020. Pest24: A large-scale very small object data set of agricultural pests for multi-target detection. *Computers and Electronics in Agriculture*, 175, 105585.

Xia, G.-S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., Zhang, L., 2018. Dota: A large-scale dataset for object detection in aerial images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3974–3983.

Xu, J., Kanokphan, L., Tasaka, K., 2018. Fast and accurate object detection using image cropping/resizing in multi-view 4k sports videos. *Proceedings of the 1st International Workshop on Multimedia Content Analysis in Sports*, 97–103.

Zhang, H., Wang, K., Tian, Y., Gou, C., Wang, F.-Y., 2018. MFR-CNN: Incorporating multi-scale features and global information for traffic object detection. *IEEE Transactions on Vehicular Technology*, 67(9), 8019–8030.

Zhang, Q., Liu, Y., Gong, C., Chen, Y., Yu, H., 2020. Applications of deep learning for dense scenes analysis in agriculture: A review. *Sensors*, 20(5), 1520.

Zhang, W., Jiao, L., Liu, X., Liu, J., 2019. Multi-scale feature fusion network for object detection in vhr optical remote sensing images. *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 330–333.