Training An Embedded Object Detector For Industrial Settings Without Real Images

Julia Cohen, Carlos F Crispim-Junior, Jean-Marc Chiappa, Laure Tougne

TRAINING AN EMBEDDED OBJECT DETECTOR FOR INDUSTRIAL SETTINGS WITHOUT REAL IMAGES

Julia Cohen[⋆†] *Carlos Crispim-Junior*[⋆] *Jean-Marc Chiappa*[†] *Laure Tougne*[⋆]

[⋆] Univ Lyon, Lyon 2, LIRIS, F-69676 Lyon, France
[†] DEMS, Saint Bonnet de Mûre, France

ABSTRACT

In an industrial environment, object detection is a challenging task due to the absence of real images and real-time requirements for the object detector, usually embedded in a mobile device. Using 3D models, it is however possible to create a synthetic dataset to train a neural network, although the performance on real images is limited by the domain gap. In this paper, we study the performance of a Convolutional Neural Network (CNN) designed to detect objects in real-time: Single-Shot Detector (SSD) with a MobileNet backbone. We train SSD with synthetic images only, and apply extensive data augmentation to reduce the domain gap between synthetic and real images. On the T-LESS dataset, SSD performs better than Mask R-CNN trained on the same synthetic images, with MobileNet-V2 and MobileNet-V3 Large as backbone. Our results also show the huge improvement enabled by an adequate augmentation strategy.

***Index Terms*—** Object detection, Synthetic dataset, Mobile applications

1. INTRODUCTION

Industry is one of the many fields now relying on computer vision for the automation of different tasks. From maintenance to robotic manipulation, a precise detection of diverse objects is required. In the meantime, deep artificial neural networks have been developed, encouraged by the improved computation capabilities of computers and an increasing availability of massive image datasets [1]. Different levels of analysis have been defined, from classification to semantic segmentation of images. In between, the object detection task consists in the localization of boxes surrounding each object, and identifying the category of the main object in each box. The existence of challenges aiming at resolving these tasks on large-scale datasets greatly benefits the research in computer vision. Among the most successful methods, Convolutional Neural Networks (CNNs) have obtained the best results.

Since industrial applications need to recognize specific objects absent from the usual datasets, it becomes necessary to generate datasets for these objects, a resource-consuming task. Since CAD models exist as a part of the products design process, a promising line of work is to take advantage of the existing 3D data to generate automatically annotated datasets of entirely synthetic images [2]. Moreover, many applications require the CNN to be applied to a stream of images in real-time, on a mobile device such as an embedded computer or a smartphone, which have limited memory and computing capabilities.

Fig. 1. Synthetic training images (top row) and real test images (bottom row) from dataset T-LESS [3].

In this work, we compare the performance of the SSD detector with different MobileNet architectures as feature extractors, in the case of synthetic-to-real domain adaptation. Within such context, we highlight the importance of data augmentation to train a model on synthetic images that will perform detection on real images. We provide object detection results on the public dataset T-LESS [3], that contains challenging texture-less industrial objects.

2. RELATED WORK

In this section, we describe the main approaches for object detection, MobileNets and learning on synthetic images.

Object detection. CNNs enabled a huge performance improvement on the task of object detection. First models with few layers have been outperformed with bigger and deeper architectures, typically composed of two subnetworks: a region proposal stage and a detection stage [4, 5]. This increase in number of layers and parameters is not a suitable prop-

This work is supported by grant CIFRE n.2018/0872 from ANRT.

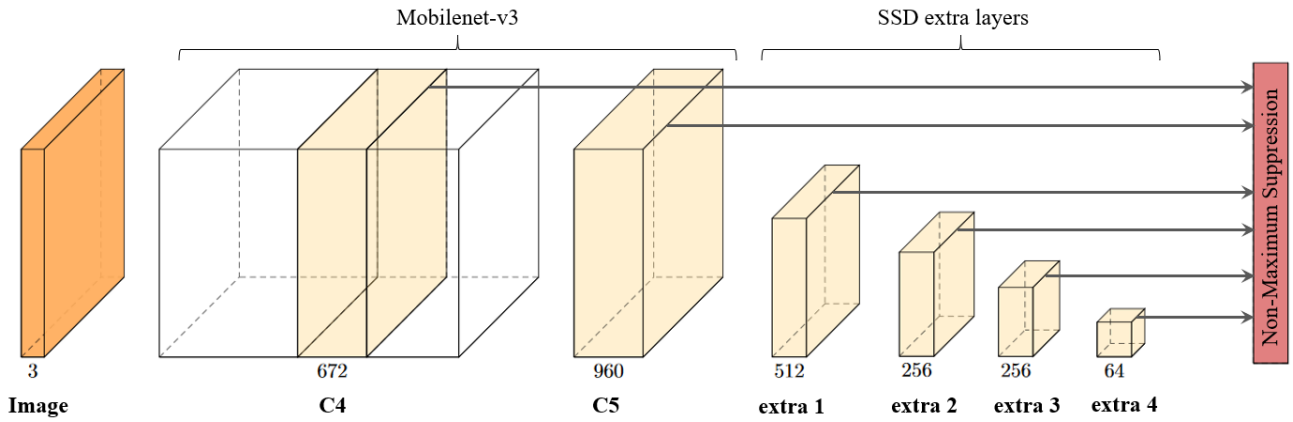

Fig. 2. Model architecture: SSD with MobileNet-V3 backbone.

erty for real-time and embedded applications, which led to the development of small one-stage object detectors for fast inference [6, 7].

MobileNets. Feature extraction is a crucial step for any CNN, and many architectures have been proposed, such as the MobileNet series. MobileNet-V1 [8] introduced the notion of depth-wise separable convolution, as well as the width and resolution multipliers as new hyper-parameters. MobileNet-V2 [9] replaced the simple convolutional layers by inverted residual blocks: a bottleneck with an expansion layer, and residual connections between the bottlenecks of the successive blocks. This memory-efficient residual block enforces the information to be contained in a low-dimensional space. MnasNet [10] added a squeeze-and-excite module into the bottleneck block of MobileNet-V2. Finally, MobileNet-V3 [11] was built as an optimized assembly of the previous blocks, with more efficient non-linearities. While it is better and faster than MobileNet-V2 on a classification task, its accuracy is the same for a faster inference on a detection task. Its adoption by the community has been limited, and recent works still use MobileNet-V2.

Synthetic training data. To train CNNs without any real image, CAD models are used to create synthetic images. The naive approach of pasting a 2D rendered object onto a real background enables to recognize some objects in real images [12, 13, 2], but is limited by the unrealistic aspect of the objects boundaries. Using rendering engines, one can create complete scenes and automatically record annotated images from virtual environments [14, 15]. The limitations of these engines, such as the difficulty to render diverse objects and generate annotations, motivated researchers to develop photorealistic renderers specifically for machine learning, such as BlenderProc [16]. Finally, the advances of Augmented Reality techniques also enables to blend smoothly real and synthetic elements [17]. Domain randomization techniques can be used to generate synthetic images with diverse appearances, such that the real images appear to the network as one more variation of the same domain [18, 19].

This work aims to evaluate if a single-stage object detector with MobileNet feature extractor can effectively learn from photorealistic images only, without seeing a single real image of the objects.

3. PROPOSED APPROACH

3.1. Network architecture

Among the lightweight CNN architectures for object detection, we selected SSD [6] since its size mostly depends on the feature extractor used. The original feature extractor is commonly replaced by ResNet-50 [20] for increased performance or MobileNet-V2 [9] for real-time inference. We compare the performance obtained using MobileNet-V2 and MobileNet-V3. MobileNet-V3 exists in two settings, Large and Small, which differ in the number of layers and maximum size of the feature maps. We evaluate all three architectures, denoted as V2-SSD, V3-SSD and V3small-SSD.

SSD is composed of a set of extra layers and header layers that predict objects locations and categories (Figure 2). The extra layers are added on top of the backbone, while the header layers take as inputs the feature maps from different layers for multi-scale prediction: two sets of predictions are drawn directly from the backbone, and one for each of the extra lay-

Table 1. Object detection results.

Model	mAP
Mask R-CNN	32.8
V3small-SSD (*aug1*)	18.6
V3-SSD (*aug1*)	36.3
V2-SSD (*aug1*)	38.3
V3small-SSD (*aug2*)	23.5
V3-SSD (*aug2*)	46.1
V2-SSD (*aug2*)	**47.7**

Table 2. Results: inference time for a single image (224x224 pixels), allocated memory on GPU and number of parameters.

Model	Inference (ms)	Memory (M)	Parameters (M)
Mask R-CNN	463	181.4	44.0
V3-SSD	35	20.1	4.9
V2-SSD	28	14.5	3.5
V3small-SSD	33	10.6	2.6

ers. A Non-Maximum Suppression (NMS) step is applied to remove duplicate detections. The complete model is built following [9] and [11]: the first SSD header is placed on top of the expansion layer of the bottleneck block with stride 16 (C4), and the extra layers as well as the second header layer are branched on top of the layer with stride 32 (C5). Figure 2 presents the network architecture for V3-SSD. The depths of the feature maps used by SSD are indicated below.

3.2. Data Augmentation

To reduce the domain gap, we rely on data augmentation over photorealistic rendered images. For this purpose, we apply a series of color and geometric augmentations using the Albumentations library [21], each with a random probability of being applied. The augmentations are: brightness, contrast, saturation and hue alterations; color shift; Gaussian, median or motion blur; Gaussian, multiplicative and ISO noise; vertical flip. After these augmentations, the image is randomly cropped to 400x400 pixels, and finally resized to 224x224.

4. EXPERIMENTS AND RESULTS

4.1. Dataset

We evaluate our approach on T-LESS [3], a dataset for 6D pose estimation, containing CAD models and RGB-D images of 30 industry-relevant, texture-less objects. In the context of the 2020 edition of the BOP challenge [23], a synthetic dataset of 50000 photorealistic images was released, generated using the physics-based rendering tool BlenderProc [16] (Figure 1, top row). We used these synthetic images as the only training data, removed 1000 of them for validation, and used the 1000 test images captured using the Primesense CARMINE 1.09 sensor as test dataset (Figure 1, bottom row). We focus on RGB object detection and leave the other modalities provided for further work.

4.2. Evaluation metrics and comparison

We compute the mean Average Precision (mAP) as defined in the Pascal VOC challenge [24] with the framework proposed by Padilla *et al.* [25]. We compare our results against the ones of Mask R-CNN [26] trained on the same synthetic images. Mask R-CNN usually performs better than MobileNets when training and testing on same domain images [27]. However, the size and lower inference time of the model limit its use in embedded applications. We used the model provided by the winners of the 2020 BOP Challenge as the first step of their method to estimate objects 6D pose [22] and apply our evaluation method to the predictions. To identify the influence of data augmentation, we also trained our models with the same augmentation pipeline as Mask R-CNN, which has less variety in the transforms applied. We refer to this data augmentation as *aug1*, and our data augmentation described in Section 3.2 as *aug2*. Note that the training and test images have a size of 540x720 pixels for Mask R-CNN and 224x224 for SSD. For fair comparison, the inference times are always measured for 224x224 pixels.

4.3. Training parameters

We used an existing PyTorch implementation of MobileNet-V3 [1] in order to take advantage of the provided model weights trained on ImageNet [1] for the Large and Small settings. Although the MobileNet-V3 architecture is based on the first version of the article before its publication at ICCV, the only difference is the size of the expansion layer on the 14th bottleneck block (672 instead of 960). For MobileNet-V2, we use the architecture and weights pre-trained on ImageNet provided by the torchvision library [28]. We train the networks until convergence using SGD with a learning rate of 0.05, momentum of 0.9, weight decay of 0.000012 and batch size of 32. The hyper-parameters were determined experimentally with a preliminary random search with V3-SSD and applied to all architectures.

4.4. Quantitative results

Experiments show that, with same data augmentation *aug1*, V2-SSD and V3-SSD both outperform Mask R-CNN (Table 1). With a more complete set of augmentations (*aug2*), performance improves from 36.33% to 46.1% for V3-SSD, and from 36.3% to 47.7% for V2-SSD. Surprisingly, the MobileNet-V2 backbone performs better than MobileNet-V3 with both augmentation settings, while the performance on the validation set was lower, suggesting a better ability to generalize to the real domain. A reason may be the lower number of parameters (Table 2), which prevents the model from learning less useful features (i.e., features representative

[1] https://github.com/d-li14/mobilenetv3.pytorch

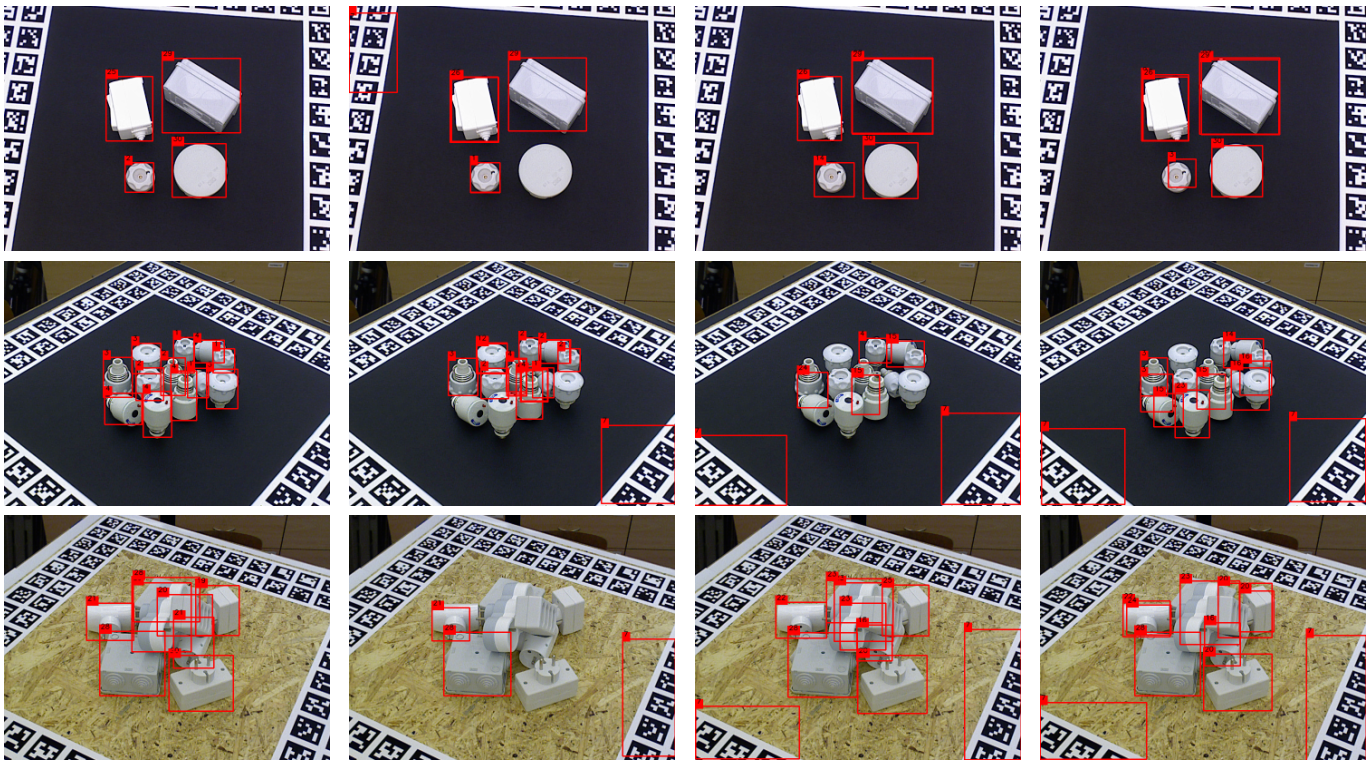

Fig. 3. Detection results. From left to right: ground-truth, Mask R-CNN [22], V3-SSD (ours), V2-SSD (ours).

of the synthetic nature of the training images). However, V3small-SSD obtains a lower performance in both augmentation settings.

To extract features relevant to the real domain, previous works fixed the weights of the backbone pre-trained on ImageNet, obtaining either an improved [12, 2] or degraded [19] performance. In our experiments, the network did not converge when updating only the SSD weights during training. We assume that the features learned on ImageNet do not transfer to T-LESS because the color and texture are not discriminative. The realism of the synthetic training set is then a strong requirement to reduce the reality gap without any real image.

Regarding the inference time, SSD is about 10 times faster than Mask R-CNN (Table 2). It is worth noting that about half the time taken by the SSD models corresponds to the NMS, which was not optimized. The model size in memory is a measurement of the memory allocated when loading the model on the GPU. V2-SSD is both smaller and faster than V3-SSD, and performs better, which justifies its use in embedded applications. With our implementation, V3-SSD and V3small-SSD seem to have the same inference time, even though the latter contains half the number of parameters.

4.5. Qualitative results

Figure 3 shows the detected bounding boxes for Mask R-CNN, V3-SSD and V2-SSD, as well as the ground truth boxes. A recurring mistake of all methods is to identify the markers around the scene as an object (usually with category 7, a rectangular block of 3 sockets). While Mask R-CNN misses objects, both V3-SSD and V2-SSD duplicate detections with different labels. This error may come from the way we applied NMS: In order to allow the detection of objects when one occludes the other, we remove only the bounding boxes of the same categories when they intersect more than a given threshold. Applying the standard NMS (regardless of the category) would remove the duplicates of different categories, although possibly keeping the wrong ones.

5. CONCLUSION

We evaluated the suitability of single-stage object detectors trained only on synthetic images for embedded detection applications. We show that such models outperform a larger model as Mask R-CNN on texture-less industry-related objects, especially with the curated data augmentation method. SSD with MobileNet-V2 as feature extractor achieves the best performance, with faster inference and lower memory requirements than the more recent MobileNet-V3. Further work should study the relevance of the proposed augmentation techniques on Mask R-CNN, as well as transferring the synthetic images to the real domain before training the object detector.

6. REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *CVPR*, 2009, pp. 248–255.

[2] J. Cohen, C. Crispim-Junior, C. Grange-Faivre, and L. Tougne, "CAD-based learning for egocentric object detection in industrial context," in *VISAPP*, 2020, vol. 5, pp. 644–651.

[3] T. Hodan, P. Haluza, S. Obdrzalek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An rgb-d dataset for 6d pose estimation of texture-less objects," in *WACV*, 2017, pp. 880–888.

[4] R. Girshick, "Fast R-CNN," in *ICCV*, 2015, pp. 1440–1448.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *TPAMI*, vol. 39, no. 6, pp. 1137–1149, 2017.

[6] W. Liu *et al.*, "SSD: Single shot multibox detector," in *ECCV*, 2016, pp. 21–37.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016, pp. 779–788.

[8] A. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018, pp. 4510–4520.

[10] M. Tan *et al.*, "MnasNet: Platform-aware neural architecture search for mobile," in *CVPR*, 2019, pp. 2820–2828.

[11] A. Howard *et al.*, "Searching for MobileNetV3," in *ICCV*, 2019, pp. 1314–1324.

[12] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige, "On pre-trained image features and synthetic images for deep learning," in *ECCV Workshops*, 2018, pp. 682–697.

[13] J. Josifovski, M. Kerzel, C. Pregizer, L. Posniak, and S. Wermter, "Object detection and pose estimation based on convolutional neural networks trained with synthetic data," in *IROS*, 2018, pp. 6269–6276.

[14] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *CVPR*, 2016, pp. 4340–4349.

[15] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *CVPR*, 2016, pp. 3234–3243.

[16] M. Denninger *et al.*, "BlenderProc," *arXiv preprint arXiv:1911.01911*, 2019.

[17] H. A. Alhaija, S. K. Mustikovela, L. M. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets computer vision: Efficient data generation for urban driving scenes," *IJCV*, vol. 126, no. 9, pp. 961–972, 2018.

[18] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IROS*, 2017, pp. 23–30.

[19] J. Tremblay *et al.*, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *CVPR Workkshops*, 2018, pp. 969–977.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[21] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *Information-an International Interdisciplinary Journal*, vol. 11, no. 2, pp. 125, 2020.

[22] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "CosyPose: Consistent multi-view multi-object 6d pose estimation," in *ECCV*, 2020, pp. 574–591.

[23] T. Hodan *et al.*, "BOP: Benchmark for 6d object pose estimation," in *ECCV*, 2018, pp. 19–35.

[24] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[25] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms.," in *IWSSIP*, 2020, pp. 237–242.

[26] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *TPAMI*, vol. 42, no. 2, pp. 386–397, 2020.

[27] J. Huang *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *CVPR*, 2017, pp. 3296–3297.

[28] S. Marcel and Y. Rodriguez, "Torchvision the machine-vision package of torch," in *ACM Multimedia*, 2010, pp. 1485–1488.