Localizing Objects with Self-Supervised Transformers and no Labels — Supplementary Material —

Oriane Siméoni[1] Gilles Puy[1] Huy V. Vo[1,2] Simon Roburin[1,3] Spyros Gidaris[1]
Andrei Bursuc[1] Patrick Pérez[1] Renaud Marlet[1,3] Jean Ponce[2,4]

[1]Valeo.ai, Paris, France
[2]Inria and DIENS (ENS-PSL, CNRS, Inria), Paris, France
[3]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
[4]Center for Data Science, New York University, New York, USA

A Ablation Study

A.1 Which transformer features to choose?

As explained in subsection 3.2 of the main paper, we chose to use the keys $\mathbf{k}_p$ of the last attention layer as patch features $\mathbf{f}_p$ in LOST. As we will see here, this choice provides the best localization performance among other alternatives. Specifically, in the first section of Table 5, we report the performance of LOST when using as patch features $\mathbf{f}_p$ either the keys $\mathbf{k}_p$, the queries $\mathbf{q}_p$, or the values $\mathbf{v}_p$ of the attention layer. We see that, when using the queries $\mathbf{q}_p$ or the values $\mathbf{v}_p$, LOST's performance deteriorates by at least 11 CorLoc points compared to using the keys $\mathbf{k}_p$.

Another way to measure the similarity between two patches in a transformer architecture is to use the scalar product between the queries and the keys. We thus test substituting

$$\tilde{a}_{pq} = \begin{cases} 1 & \text{if } \mathbf{q}_p^\top \mathbf{k}_q + \mathbf{k}_p^\top \mathbf{q}_q \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

for a_{pq} in Eq. (1) in the main paper, when selecting the first, initial seed. Note that this choice of $\tilde{a}_{pq}$ ensures the symmetry of the adjacency matrix. We test this new choice of similarity matrix when using the queries, keys or values in the seed expansion step, i.e., in $\mathcal{S}$, and in the box extraction steps, i.e., in $\mathbf{m}$ as defined in Eq. (3) in the main paper.

Finally, we also test another alternative by changing the definition of $\mathcal{S}$ to $\tilde{\mathcal{S}} = \{q \mid q \in \mathcal{D}_k \text{ and } \mathbf{q}_q^\top \mathbf{k}_{p^*} + \mathbf{k}_q^\top \mathbf{q}_{p^*} \geq 0\}$ and changing the definition of m_q in Eq. (3) to

$$\tilde{m}_q = \begin{cases} 1 & \text{if } \sum_{s \in \mathcal{S}} \left(\mathbf{k}_q^\top \mathbf{q}_s + \mathbf{q}_q^\top \mathbf{k}_s \right) \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Results in Table 5 show that all these alternatives using queries and keys yield results that are not as good as when using the keys as patch features.

Seed selection	Expansion & Box extrac.	k	CorLoc
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{q}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{q}_p$ in $\mathcal{S}$ and m_q	100	30.8
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{v}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{v}_p$ in $\mathcal{S}$ and m_q	100	50.5
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{k}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	100	61.9
$\tilde{a}_{pq}$ defined in 4	$\mathbf{f}_p = \mathbf{q}_p$ in $\mathcal{S}$ and m_q	100	30.8
$\tilde{a}_{pq}$ defined in 4	$\mathbf{f}_p = \mathbf{v}_p$ in $\mathcal{S}$ and m_q	100	29.9
$\tilde{a}_{pq}$ defined in 4	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	100	30.7
$\tilde{a}_{pq}$ defined in 4	using $\tilde{\mathcal{S}}$ and $\tilde{m}_q$	100	30.8
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{k}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	1	38.3
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{k}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	50	58.8
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{k}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	150	61.8
a_{pq} with $\mathbf{f}_{p,q} = \mathbf{k}_{p,q}$ in Eq. (1)	$\mathbf{f}_p = \mathbf{k}_p$ in $\mathcal{S}$ and m_q	200	61.2

Table 5: **Ablation study.** CorLoc performance on VOC2007 for different choices of transformer features in the seed selection, expansion and box extraction steps, as well as influence on the results of the parameter k (maximum number of patches with the lowest degree, in $\mathcal{D}_k$, for seed expansion).

A.2 Importance of the seed expansion step

We analyse here the importance of the seed expansion step that is controlled by k. The seed expansion step allows us to enlarge the region of interest so as to include all the parts of an object and not only the part localized from the first, initial seed.

Table 5 presents the impact of the parameter k, which corresponds to the maximum number of patches that can be used to construct the mask $\mathbf{m}$. We notice that, without seed expansion (i.e., $k = 1$), there is a drastic drop in the localization performance. The performance then improves when increasing k to 100-150 with a slight decrease at 200.

Visualizations of results with $k = 1$ and $k = 100$ are presented in the Figure 3 of the main paper and Figure 4 here. We see that the boxes in yellow obtained with $k = 1$ are small and localized on probably what is the most discriminative part of the objects. Increasing k permits us to increase the size of the box and localize the object better. We also present in Figure 5 cases of failures where the seed expansion step is either insufficient to localize the whole object or yields a box containing multiple objects.

A.3 Analysis of DINO-seg

In this section, we investigate alternative setups of the baseline DINO-seg which is based on the work of Caron *et al*. [13]. They are presented in Table 6.

First, instead of using the best attention head over the entire dataset (as we did in the main paper), we evaluate the localization accuracy of DINO-seg for each one of the 6 available heads. We find out that one head in particular, namely head 4, captures objects well, whilst results with other heads are much lower. Due to its superior performance, in the main paper we report DINO-seg using head 4.

We also explore dynamically selecting one box per image among boxes corresponding to the different heads using some heuristics. We report the two variants that gave the best results. In the first variant, we consider selecting the box corresponding to the head with the biggest connected component ('DINO-seg BCC'). However, it yields worse results than with head 4. We also try selecting, over the 6 boxes of the different heads, the box that has the highest average IoU overlap with the remaining 5 boxes ('DINO-seg HAIoU'). It improves over

Method	VOC07_trainval	VOC12_trainval	COCO20k
DINO-seg [head 0]	25.9	24.6	30.1
DINO-seg [head 1]	36.2	35.9	35.8
DINO-seg [head 2]	32.1	33.2	31.6
DINO-seg [head 3]	21.6	20.0	26.3
DINO-seg [head 4]	45.8	46.2	**42.1**
DINO-seg [head 5]	35.5	42.1	26.5
DINO-seg BCC	38.8	45.2	28.8
DINO-seg HAIoU	**46.1**	**47.6**	40.8
LOST (ours)	**61.9**	**64.0**	**50.7**

Table 6: **DINO-seg ablation study.** We compare here CorLoc results on datasets VOC07_trainval, VOC12_trainval and COCO20k when applying the DINO-seg method to create a box from the different heads of the attention layer. Also, DINO-seg BCC selects the box/head that produces the biggest connected component, and DINO-seg HAIoU selects the box/head that has the highest average IoU with the other 5 boxes. We additionally report results with our method LOST for comparison.

Number K of clusters	20	25	30	40
Mean AP (%)	29.9	29.4	34.0	32.2

Table 7: **Impact of number of clusters in object detection.** Results, using the mean AP@0.5 (%) across all the classes, on VOC07 test. All models are trained using LOST's pseudo-boxes (i.e., LOST + OD) on the VOC07 and VOC12 trainval sets. The number of classes in VOC is 20.

DINO-seg [head 4] by 1 point on both VOC07 and VOC12. However, as shown in Table 6, it still performs significantly worse than LOST in this single-object discovery task.

A.4 Impact of the number of clusters on class-aware detection training

For the unsupervised class-aware detection experiments of the main paper, we assume that we know the exact number of object classes present in the used dataset, i.e., 20 in the VOC dataset, and use the same number of K-means clusters. Here we only assume that we have a rough estimate of the number of classes and study the impact of the requested number of clusters on the performances of the unsupervised detector.

To that end, in Table 7, we provide the mean AP across all the 20 VOC classes when using 20, 25, 30 and 40 clusters. For the case when we use more clusters than the 20 classes of the VOC dataset, Hungarian matching, which is used for reporting the AP results, will map to the VOC classes only the 20 most fitted available clusters. Thus, when reporting the per-class AP results, we ignore the detections in these unmatched clusters (since they have not been mapped to any ground-truth class).

In Table 7, we observe that our unsupervised detector achieves good results for all the numbers of clusters. Interestingly, for 30 and 40 clusters there is a noticeable performance improvement. Similar findings have been observed on prior clustering work [2, 36, 61].

A.5 Impact of the non-determinism of the K-means clustering

We investigate the impact of the randomness in the K-means clustering on the results of the object detector. To that end, we repeat 4 times, using different random seeds, the unsupervised

Method	VOC07_noh	VOC12_noh
OSD [67]	40.7	-
DDT+ [72]	43.4	46.3
rOSD [68]	49.3	51.2
LOD [69]	48.0	50.5
LOST	54.9	57.5

Table 8: CorLoc results on the VOC07_noh and VOC12_noh datasets.

class-aware object detection experiment with LOST + OD† (using the model trained on the union of VOC07 and VOC12 trainval sets, cf. Table 3 in subsection 4.3 of the main paper). We obtain a standard deviation of 0.8 for the AP@0.5 %, which shows that the method is fairly insensitive to the randomness of the clustering method.

B More quantitative results and comparisons

B.1 Results on more datasets used in previous work

For completeness, we present in Table 8 results on the datasets used in previous object discovery works [67, 68, 69, 72]. In particular, we evaluate our method on the datasets VOC07_noh and VOC12_noh datasets (also named VOC_all in literature). They are subsets of the trainval set of the well-known PASCAL VOC 2007 and PASCAL VOC 2012 datasets containing 3550 and 7838 images respectively. These subsets exclude all images containing only objects annotated as "hard" or "truncated" and all boxes annotated as "hard" or "truncated".

B.2 Multi-object discovery results

We compare in Table 9 the object discovery performance of different methods in the setting where multiple regions are returned per image. This setting has been explored in [68] and [69].

Following [69], instead of considering the object recall (detection rate) for a given number of predicted regions per image, as in [68], we consider as a metric a form of Average Precision adapted to the task, that we name here "odAP". It is the average of the AP of predicted objects for each number of predicted regions, from one to the maximum number of ground-truth objects in an image in the dataset. This odAP metric thus does not depend on the number of detections per image and remains related to AP, which is a standard metric for object detection. [69] actually uses two variants of this metric: odAP50, where a prediction is correct if its intersection-over-union (IoU) with one of the ground-truth boxes is at least 0.5, and odAP@[50-95], the average odAP value at 10 equally-spaced values of the IoU threshold between 0.5 and 0.95.

As LOST only returns one region per image, we only consider here LOST + CAD, which is the output of a class-agnostic detector (CAD) trained with LOST boxes, and we compare it to other existing approaches. It can be seen that LOST + CAD outperforms significantly all the previous methods, including the class-agnostic detector trained with LOD [69] boxes (LOD [69] + CAD).

Method	odAP50			odAP@[50-95]		
	VOC07_trainval	VOC12_trainval	COCO20k_trainval	VOC07_trainval	VOC12_trainval	COCO20k_trainval
Kim *et al.* [38]	9.5	11.8	3.93	2.49	3.11	0.96
DDT+ [72]	8.7	11.1	2.41	3.0	4.1	0.73
rOSD [68]	13.1	15.4	5.18	4.29	5.27	1.62
LOD [69]	13.9	16.1	6.63	4.47	5.34	1.98
LOD [69] + CAD	15.8	20.9	7.26	5.03	7.07	2.28
LOST + CAD	**19.8**	**24.9**	**7.93**	**6.71**	**8.85**	**2.51**

Table 9: Multi-object discovery performance in odAP (Average Precision for object discovery) of our method and the baselines [38, 68, 69, 72].

B.3 Image nearest neighbor retrieval

Following LOD [69], we use LOST box descriptors to find images that are similar to each other (image neighbors) in the image collection.

To this end, each image is represented by the `CLS` descriptors of its LOST box and the cosine similarity between these descriptors is used to define a similarity between the images. Then, for each image, the top τ images with the highest similarity are chosen as its neighbors. Similar to LOD [69], we choose $\tau = 10$ and use CorRet [15] as the evaluation metric, defined as the average percentage of the retrieved image neighbors that are actual neighbors (i.e., that contain objects of the same category) in the ground-truth image graph over all images.

We compare the performance of our method in this task with rOSD [68] and LOD [69] in Table 10. We see that LOST boxes, when represented by DINO [13] features, yield the better CorRet score compared to [68, 69]. When VGG16 [55] features are used, LOST is behind LOD [69] but better than rOSD [68].

Method	Features	CorRet (%)
rOSD [68]	VGG16 [55]	64
LOD [69]	VGG16 [55]	70
LOST (ours)	VGG16 [55]	68
LOST (ours)	DINO [13]	**72**

Table 10: Image neighbor retrieval performance (CorRet) of different methods.

Method	Features	CorLoc (%)		
		VOC07_trainval	VOC12_trainval	COCO20k_trainval
LOD [69]	VGG16 [55]	53.6	55.1	48.5
LOD [69]	DINO [13]	43.2	45.9	33.7
LOST (ours)	VGG16 [55]	42.0	47.2	30.2
LOST (ours)	DINO [13]	**61.9**	**64.0**	**50.7**

Table 11: Single-object discovery performance in CorLoc of LOD [69] and LOST with different types of features.

B.4 Using DINO features

We are aware that, in Table 1 of the main paper, we compare our method using a transformer backbone to methods based on a VGG16 pre-trained on ImageNet models. For a fair comparison, we investigate here the state-of-the-art LOD [69] method when adapted to use the transformers features.

LOD [69] uses the algorithm from rOSD [68] to generate region proposals from CNN features for their pipeline, but we observe that this algorithm does not yield good proposals with transformer features. We therefore run LOD with edgeboxes [84] and use DINO [13] features, extracted with ROIPool [27], to represent these proposals. We present the results on VOC07_trainval, VOC12_trainval and COCO20k_trainval dataset in Table 11.

Our results in Table 2 of the main paper show that a direct adaption of LOST, designed by analysing the properties of transformers features, to CNN features yields worse performance. Conversely, as we see in Table 11 here, adapting algorithms developed using properties of

CNN features to transformer features is also not direct. Nevertheless, the number of design choices to adapt these algorithms to new types of features is vast and we do not exclude that some design choices might improve the results even further, e.g., by exploiting together CNN and transformer features.

B.5 Using supervised pre-training.

We test LOST but this time using a transformer pre-trained under full supervision on ImageNet. We use the model provided by DeiT [62].

With this model, LOST achieves a CorLoc of 16.9% which is significantly worse than the results obtained with the DINO self-supervised pre-trained model. We remark that a similar observation was made for DINO [13], where the segmentation performance obtained with the model trained under full supervision yields significantly worse results than when using DINO's model. It is unclear, however, if this difference of performance can be attributed to the properties of the self-supervision loss or to the more aggressive data augmentation used during DINO pre-training.

C More visualizations (single- and multi-object discovery)

We present in Figures 4-9 additional qualitative results of our method.

Figure 4 and Figure 5 are discussed in the subsection A.2.

Figure 6 and Figure 7 show successful examples of LOST + CAD in VOC07_trainval and COCO20k_trainval datasets. It can be seen that it is able to localize multiple objects in the same image.

Figure 8 and Figure 9 present results obtained with LOST + OD on the VOC07 and COCO datasets respectively. They show the localization predictions with their predicted pseudo-classes. Each pseudo-class is assigned a different color. In Figure 9, the "person" objects are assigned three different pseudo-classes; those failures show the difficulty to assign the same class to "person" in very different positions.

D Training details of the Faster R-CNN detection models

In the main paper, we explore the application of LOST in unsupervised object detection by using its pseudo-boxes as ground truth for training Faster R-CNN detection models.

For the implementation of the Faster R-CNN detector, we use the `R50-C4` model of Detectron2 [73] that relies on a ResNet-50 [32] backbone. In our experiments, this ResNet-50 backbone is pre-trained with DINO self-supervision. Then, to train the Faster R-CNN model on the considered dataset, we use the protocol and most hyper-parameters from He *et al*. [34].

In details, we train with mini-batches of size 16 across 8 GPUs using `SyncBatchNorm` to finetune BatchNorm parameters, as well as adding an extra BatchNorm layer for the RoI head after `conv5`, i.e., `Res5ROIHeadsExtraNorm` layer in Detectron2. During training, the learning rate is first warmed-up for 100 steps to 0.02 and then reduced by a factor of 10 after 18K and 22K training steps. We use in total 24K training steps for all the experiments, except when training class-agnostic detectors on the pseudo-boxes of the VOC07 trainval set, in which case we use 10K steps. For all experiments, during training, we freeze the first two convolutional blocks of ResNet-50, i.e., `conv1` and `conv2` in Detectron2.

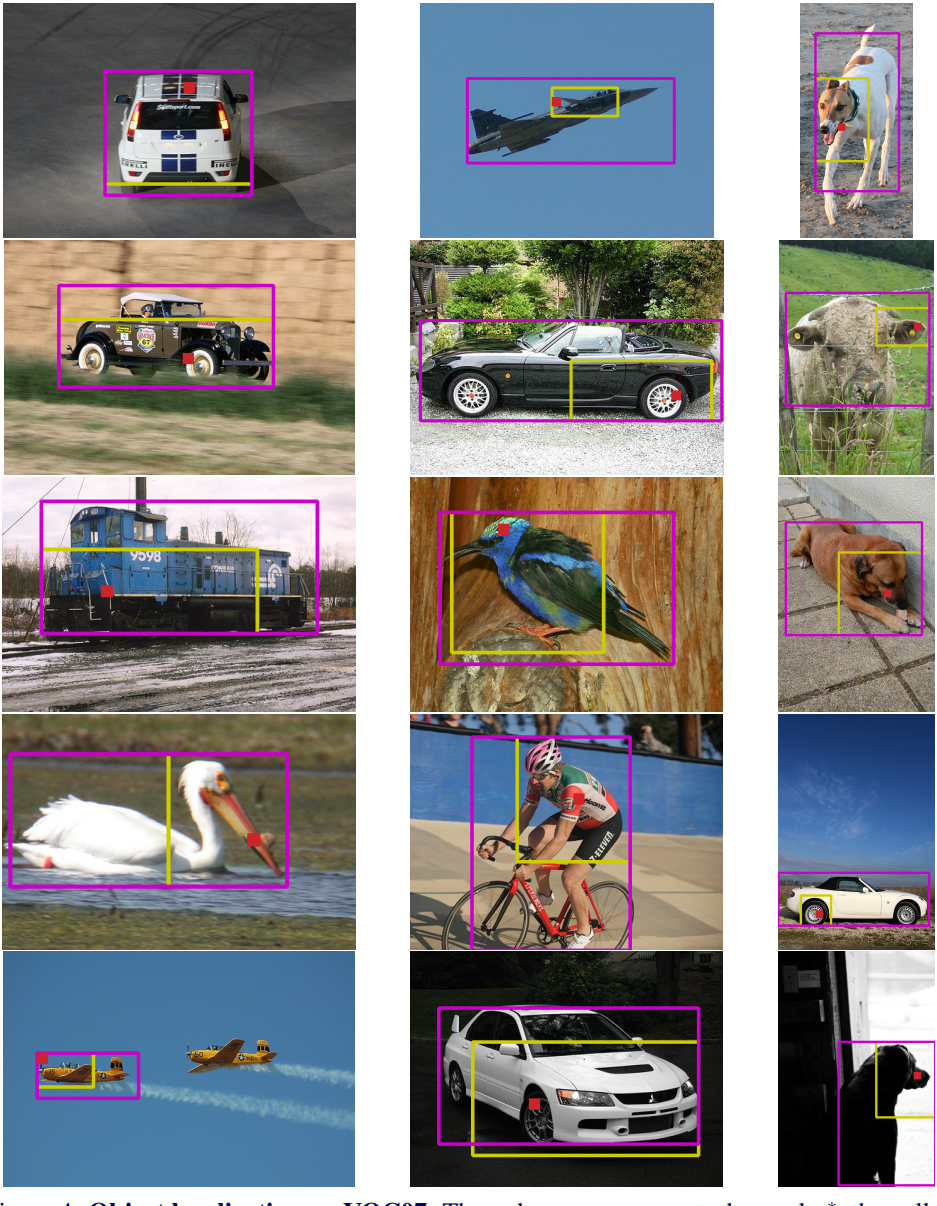

Figure 4: **Object localization on VOC07.** The red square represents the seed p^*, the yellow bos is the box obtained using only the seed p^*, and the purple box is the box obtained using all the seeds $\mathcal{S}$ with $k = 100$.

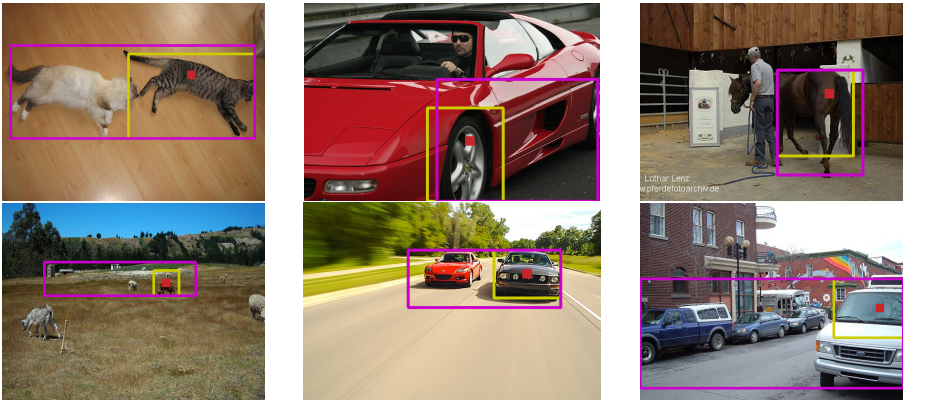

Figure 5: **Cases of localization failure on VOC07.** The red square represents the seed p^*, the yellow box is the box obtained using only the seed p^*, and the purple box is the box obtained using all the seeds $\mathcal{S}$ with $k = 100$.

Figure 6: **Multi-object discovery on VOC07 (LOST + CAD).** Predictions performed by the class-agnostic detector on VOC07.

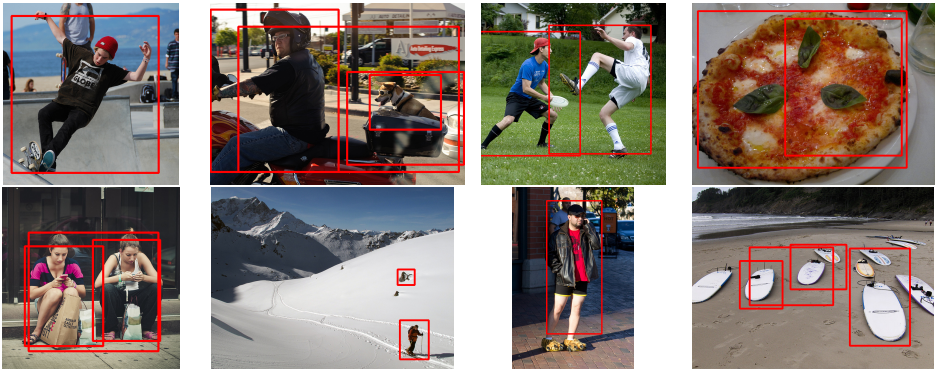

Figure 7: **Multi-object discovery on COCO (LOST + CAD).** Predictions performed by the class-agnostic detector on COCO.

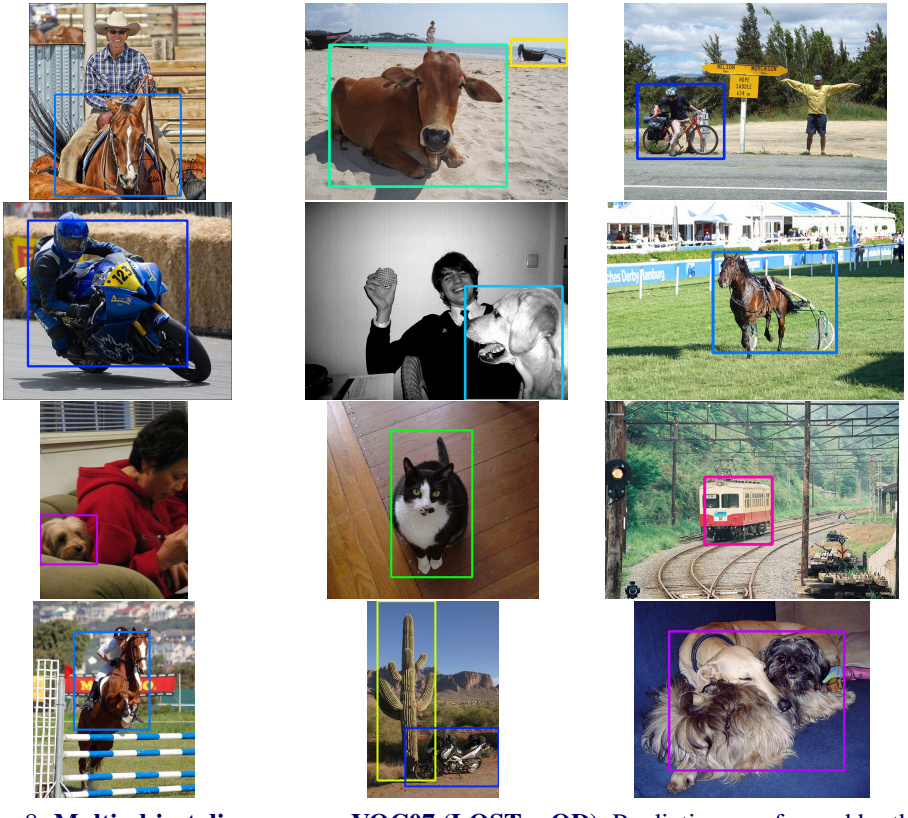

Figure 8: **Multi-object discovery on VOC07 (LOST + OD).** Predictions performed by the class-aware detector on VOC07 (a different color per class).

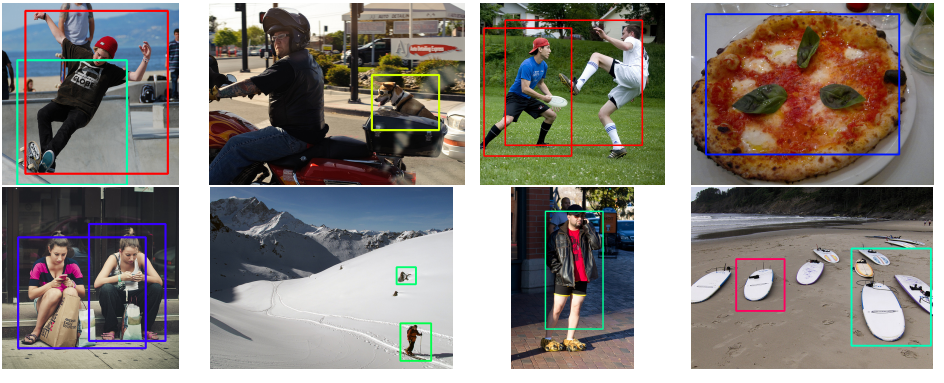

Figure 9: **Multi-object discovery on COCO (LOST + OD).** Predictions performed by the class-aware detector on COCO (a different color per class). The actual "person" class is assigned three different pseudo-classes, illustrating the difficulty to see a single category for a "person" in very different positions.