Towards Open-World Object-based Anomaly Detection via Self-Supervised Outlier Synthesis Supplementary Material

Brian K. S. Isaac-Medina*, Yona Falinie A. Gaus*,
Neelanjan Bhowmik*, Toby P. Breckon*,†

Department of {*Computer Science, †Engineering}, Durham University, UK

A Dataset Details

This work uses five different datasets for anomaly detection. While these datasets are focused on object detection and instance segmentation, we can adapt them for our anomaly detection task. The details for the dataset splits and their statistics are described next.

A.1 PASCAL-VOC 2007/12 and BDD100k

We use the **PASCAL-VOC 2007/12** [16] dataset as in-distribution (normal) data, containing 20 normal object categories (person, bird, cat, cow, dog, horse, sheep, airplane, bicycle, boat, bus, car, motorcycle, train, bottle, chair, dining table, potted plant, couch and tv). The training partition has 16,551 images and 47,223 objects, while the test set has 4,952 images and 14,997 objects. Additionally, we also use the **BDD100K** [68] dataset as the in-distribution data. This dataset consists of 69,863 images and 1,273,707 objects in the training set and 10,000 images and 185,945 object annotations in the test set, spanning across 10 normal classes (pedestrian, rider, car, truck, bus, train, motorcycle, bicycle, traffic light and traffic sign). For both datasets, MS-COCO is used as the OoD dataset removing the images with overlapping in-distribution objects. In this regard, the COCO test partition without VOC objects consists of 930 images and 2,824 annotations, while the COCO test set without BDD100k in-distribution data has 1,880 images and 8,980 object instances. We use the annotation files provided by Du et al. [15].

A.2 Durham Baggage Full Image (DBF6)

The DBF6 [2] is an X-ray security imagery dataset containing 6 object classes (firearm, firearm part, laptop, camera, knife and ceramic knife). The experiments involving this dataset use an in-distribution training partition without firearms and firearm parts containing 4,588 images and 5,396 objects, an in-distribution testing set with 1,114 images and 1,340 objects, and an OoD test partition of 692 images and 692 objects (with one single firearm/firearm part per image). The DBF6 dataset is the only dataset with manually annotated instance segmentation masks.

A.3 SIXRay10

The **SIXRay** [39] dataset consists of 1,059,231 X-ray images with 6 prohibited items (gun, knife, wrench, pliers, scissors and hammer), although it only has 5 annotated classes (hammer is not annotated). In this dataset, we consider the gun as an anomaly and trained on the other classes. We use the SIXRay10 subset, containing 10,000 images, resulting in a training partition with 7,496 normal images and 11,116 objects, and a testing in-distribution partition of 988 images and 1,422 anomaly instances. The OoD test set consists of 352 images with 553 anomalies (guns).

A.4 LTDImaging

Finally, the **LTDImaging** [42] is an infrared video-surveillance dataset with four classes (person, bicycle, motorcycle and vehicle). We train on a week's worth of data considering the vehicle class as an anomaly, giving a training set of 10,108 images and 50,924 objects. For testing, we use a one-day partition (outside the training week) without vehicles, giving a total of 1,570 images and 12,214 objects. For testing OoD detection, we use data from the same training week containing only vehicles (since their occurrence is small compared with the other classes), having a test set of 284 images and 298 objects.

B Pseudo-code

The pseudo-code for training OLN-SSOS/FFS is given in Algorithm 1, while the inference pipeline is described in Algorithm 2.

C Training Regime

We train our models using the MMDetection [9] framework, with slight variations for each dataset. For all of our datasets, except BDD100K, we initialize our models with the OLN [26] pre-trained on the VOC dataset, as per the original implementation (see [26] for the details). Since some classes in VOC are considered anomalies for the BDD100K experiment, we trained an OLN on the BDD100k dataset and used it to initialize the BDD100K experiments. For all the experiments except the LTDImaging, we resize the images to have a maximum side length of 1,333 pixels and variable minimum side length to allow for multi-scale training. Since LTDImaging images come from the same camera, we kept the same image size for training and testing, *i.e.*, 384×288 pixels. For all the experiments, random horizontal flip is used during training and 0-padding is added so the images are exactly divisible by 32.

For all our experiments, OLN-SSOS and OLN-FFS are trained for 8 epochs with an initial learning rate of 0.001 with a linear warmup for the first 100 iterations, decaying by a factor of 10 after epoch 4. All our training is carried out using stochastic gradient descent with a weight decay of 1×10^{-4}. Considering

Algorithm 1: OLN-SSOS/FFS Train Pipeline.

Data: Input images $\{\mathbf{x}\}_{i=1}^{N}$, ground truth $\{y_{ij}\}_{i=1,j=1}^{N,N_i}$, where N_i is the number of objects in $\mathbf{x}_i$, pseudo-labels K and OLN-SSOS model M.

begin

Randomly initialise the pseudo-label centres $\mathbf{p} \leftarrow \mathcal{N}(\mathbf{0}, I)$

foreach *epoch* **do**

Ground truth boxes clustering

$fts \leftarrow List$ `/* Object features list */`

for $i \leftarrow 1$ **to** N **do**

$\mathbf{f}_i = M.Backbone(\mathbf{x}_i)$

for $j \leftarrow 1$ **to** N_i **do**

$\mathbf{z}_{ij} = RoIAlign(\mathbf{f}_i, y_{ij})$

fts.append($\mathbf{z}_{ij}$)

Initialise kmeans with $\mathbf{p}$.

$kmeans$ = MiniBatchKMeans(initial = $\mathbf{p}$)

$kmeans.fit(fts)$

$\mathbf{p} = kmeans.centres$

$\mathbf{c} = kmeans.labels$`/* Assigned pseudolabels */`

Model Training

for $i \leftarrow 1$ **to** N **do**

Extract image features

$\mathbf{f}_i \leftarrow M.Backbone(\mathbf{x}_i)$

Get Proposals

$P_i \leftarrow M.RPN(\mathbf{f}_i)$

Pool proposal features

$\mathbf{u}_i \leftarrow RoIAlign(\mathbf{f}_i, P_i)$

$\mathbf{v}_i \leftarrow M.g(\mathbf{u}_i)$ `/* Object features. g: Shared head in Fig. 2 */`

Predict Pseudo-classes

$l_i \leftarrow \text{MLP}(\mathbf{v}_i)$

Outlier Synthesis

$\tilde{\mathcal{V}} \leftarrow List$

$\boldsymbol{\Sigma} \leftarrow Cov(\{\mathbf{v}_i\}, \{l_i\})$`/* From Eq. (5) */`

for $k \leftarrow 1$ **to** K **do**

$\mathcal{V}^{(k)} \leftarrow \{\mathbf{v}_i | \forall(\mathbf{v}_i, l_i), l_i = k\}$

$\boldsymbol{\mu}_k \leftarrow Mean(\mathcal{V}^{(k)})$ `/* From Eq. (4) */`

$G_k \leftarrow \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$

Sample virtual outliers

$\tilde{\mathbf{v}}_k \leftarrow \{\tilde{\mathbf{v}}_j | p(\tilde{\mathbf{v}}_j \sim G_k) < \epsilon\}$

$\tilde{\mathcal{V}}.append(\tilde{\mathbf{v}}_k)$

Get Energies

$E_i \leftarrow Energy(\{\mathbf{v}_i\})$ `/* Normal energies, from Eq. (6) */`

$\tilde{E} \leftarrow Energy(\tilde{\mathcal{V}})$ `/* Abnormal energies, from Eq. (6) */`

Predict uncertainty score using the anomaly MLP ϕ

$\lambda_i \leftarrow \phi(E_i)$

$\tilde{\lambda} \leftarrow \phi(\tilde{E})$

Get loss values from Eqs. (1), (2) and (7). For FFS, use Eq. (9). For mask versions, use the error functions from Mask R-CNN [21] and Mask Scoring [23]

$\mathcal{L} = \mathcal{L}_{RPN} + \mathcal{L}_{bbox} + \mathcal{L}_{mask} + \alpha\mathcal{L}_{pcls} + \beta\mathcal{L}_{anomaly} + \gamma\mathcal{L}_{nll}$

Algorithm 2: OLN-SSOS/FFS Inference Pipeline.

Data: Input image $\{\mathbf{x}\}$ and OLN-SSOS model M.

begin

Extract image features
$\mathbf{f} \leftarrow M.Backbone(\mathbf{x})$
Get Proposals
$P \leftarrow M.RPN(\mathbf{f})$
Pool proposal features
$\mathbf{u} \leftarrow RoIAlign(\mathbf{f}, P)$
$\mathbf{v} \leftarrow M.g(\mathbf{u})$ `/* Object features. g: Shared head in Fig. 2 */`
Get object energies
$E \leftarrow Energy(\mathbf{v})$ `/* From Eq. (6) */`
Predict uncertainty score using the anomaly MLP ϕ
$\lambda \leftarrow \phi(E)$
All objects below a threshold uncertainty score are anomalies.

the pertaining time of OLN, our models are trained for a similar number of epochs as in VOS [15] (in total, we train for 16 epochs while VOS is trained for $\sim$ 18 epochs). All our models are trained with a batch size of 2 except for LTDImaging, which uses a batch size of 64. Pseudo-class training starts from the beginning, reclustering before each epoch.

D Analysis of learned pseudo-classes

Figs. 6 to 10 show a t-SNE [S1] projection of the object features, clustered by their learned pseudo-classes. It is seen in Fig. 6 that the VOC dataset is not properly clustered when using only a few pseudo-labels. For instance, for pseudo-labels $k = 5$ and $k = 10$, no significant difference can be observed among the pseudo-labels, specially when clustering people instances. Although some semantic separation can be observed between vehicles and people, there is still some confusion for $k = 10$. On the other hand, when using a large number of pseudo-classes, such as $k = 100$, it is seen that the learned labels dive the objects into more semantically meaningful clusters, such as animals (green), seated people (purple) or indoor objects (red). However, this clustering is still challenging and it demonstrates why our method does not match the state of the art for this dataset. A similar trend is observed in Fig. 7 for the BDD dataset, although the pseudo-labels cluster the objects better for smaller k compared with VOC. Pseudo-clusters for the DBF6 dataset is shown in Fig. 8. Given the more balanced distribution of categories, clusters seem to capture semantically similar objects, even for a small number of pseudo classes. It is also observed that our method seems to differentiate between different orientations of knives, while keeping all laptops in a similar class. Additional results for X-Ray imagery is presented for the SIXRay10 dataset in Fig. 9, showing that while some objects might look similar (scissors and tweezers), they can still be separated into

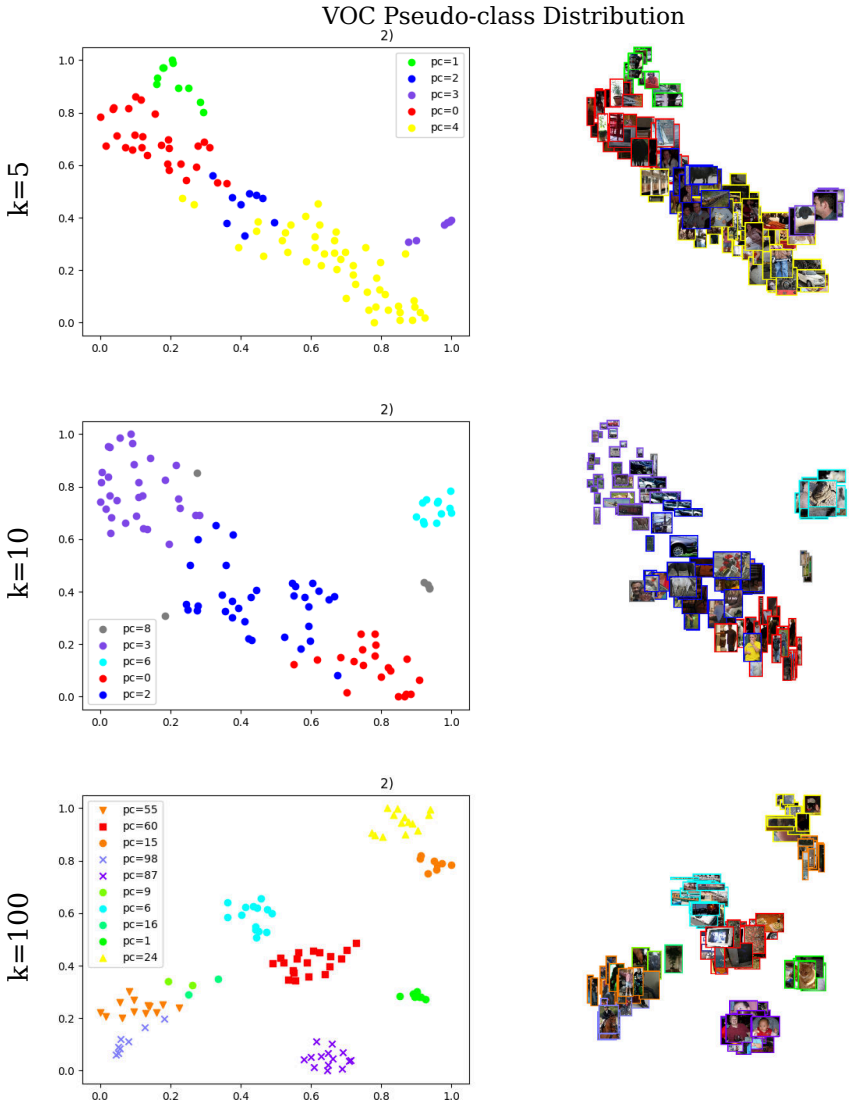

Fig. 6: Learned pseudo-labels for the VOC dataset.

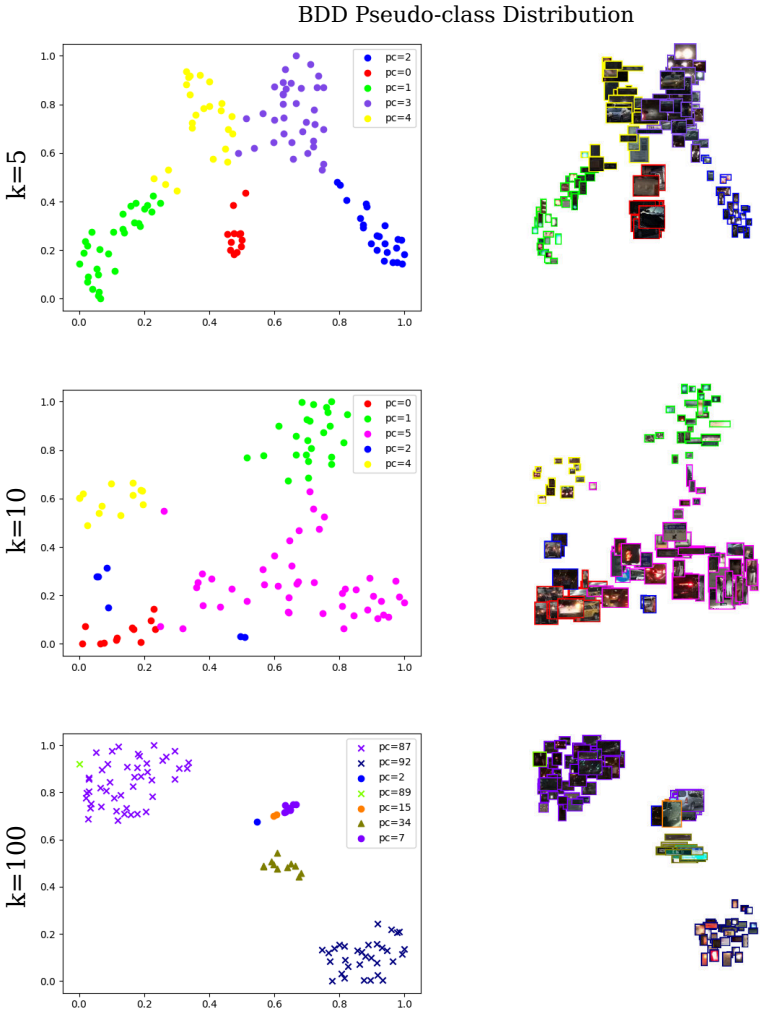

Fig. 7: Learned pseudo-labels for the BDD dataset.

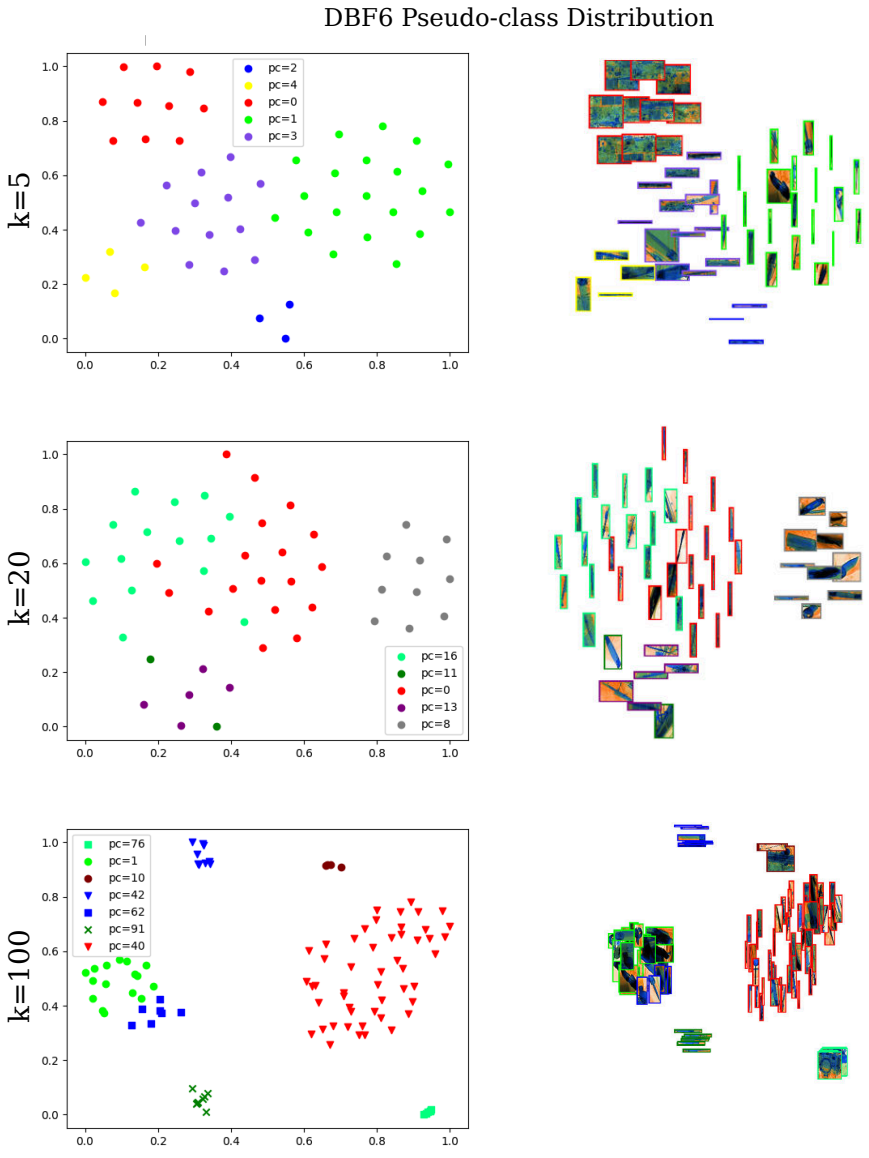

Fig. 8: Learned pseudo-labels for the DBF6 dataset.

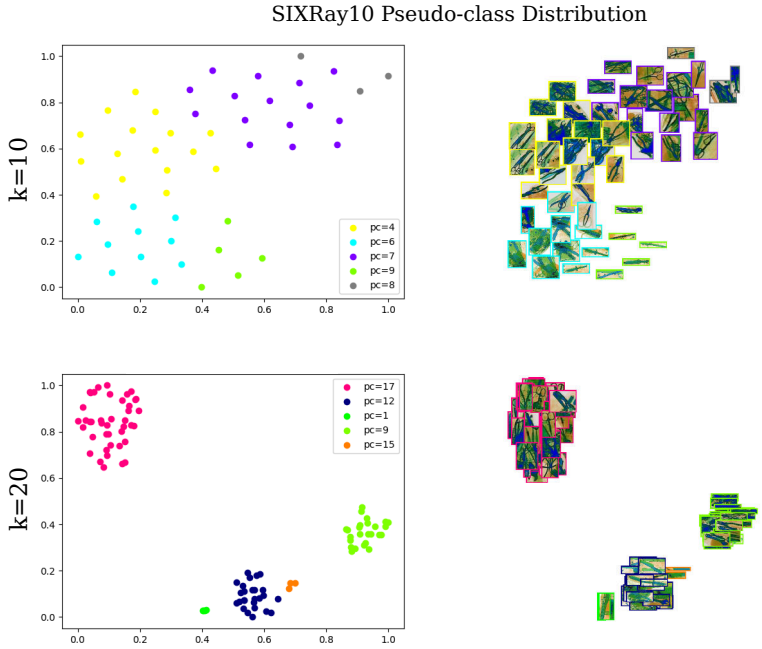

Fig. 9: Learned pseudo-labels for the SIXRay10 dataset.

different classes with no class training. Finally, Fig. 10 shows the pseudo-label analysis for the LTDImaging dataset. While some semantic separation can be observed (for instance, people in similar poses are clustered together), the low object variability makes it difficult to separate them into meaningful clusters, meaning that over-segmentation might negatively impact the performance, as seen in Fig. 5.

E Comparison against other open-world detectors

While OLN-SSOS focuses on localising objects and labelling them as anomalies, our goal can be seen as similar to OW-DETR [20] and PROB [73]. Therefore, we include a comparison of our approach against such open-world object detectors in Tab. 1. We compare PROB with the best results for each dataset in our work (excluding BDD). Unknown recall at 0.5 IoU ($UR_{0.5}$) and COCO AR@100 are reported. Both metrics are for 100 detections (PROB uses 100 object queries). A similar trend is observed, i.e., it performs better in VOC/COCO (our method being competitive) but our method is superior in the other tasks. It is worth noting that PROB also uses class supervision to detect unknown classes. Specifically, an unknown object is detected if they have a high objectness score but a low known class probability. Also, our evaluation results trend mirror the single

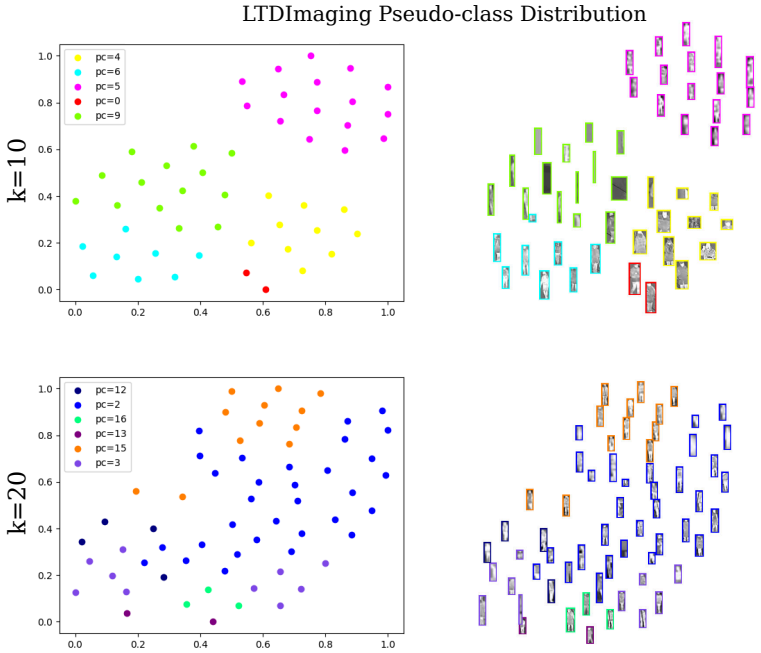

Fig. 10: Learned pseudo-labels for the LTDImaging dataset.

Table 1: PROB [73] vs Ours.

	VOC/COCO		DBF6		SIXRay10		LTD	
	$UR_{0.5}$	AR@100	$UR_{0.5}$	AR@100	$UR_{0.5}$	AR@100	$UR_{0.5}$	AR@100
PROB	**53.2**	**34.0**	26.9	6.2	57.5	13.0	3.29	1.0
Ours	40.4	17.8	**90.5**	**48.8**	**92.4**	**35.6**	**38.6**	**18.2**

task for OW-DETR/PROB. These results further showcase the ability of our method for anomaly detection and localisation without class supervision.

F Further Ablations

Figs. 11 to 15 show further ablation studies for varying sampling sizes. In particular, Fig. 11 show the average recall for different sampling sizes and different numbers of pseudo-classes; Figs. 12 and 13 show the ablations for the OoD sampling size for DBF6 Box and DBF6 Mask; and Figs. 14 and 15 show the ablations for SXIRay10 and LTDImaging. While the best results for VOC/COCO are obtained for a sampling size of 300 images and with 100 pseudo-classes, being the reason why we used this sampling size in our experiments, this might not be the same for different datasets, therefore indicating that a proper evaluation must be carried out for each dataset.

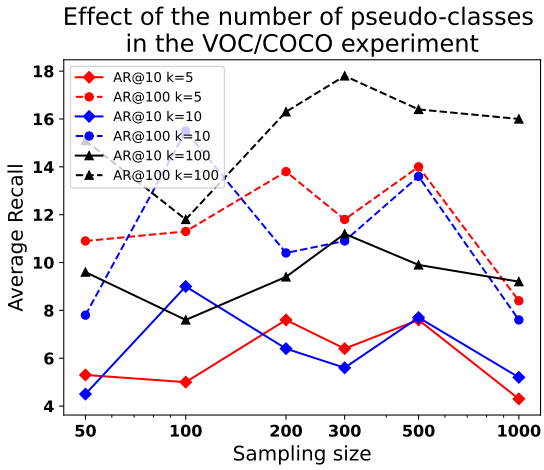

Fig. 11: Ablations for VOC/COCO. Maximum performance is achieved for k = 100 and 300 samplings for virtual outlier synthesis.

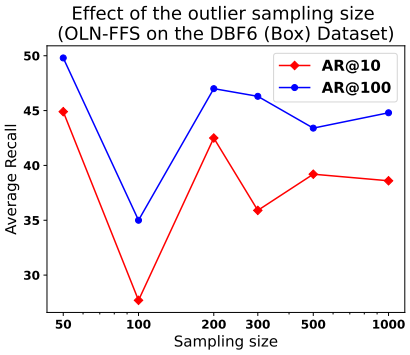

Fig. 12: Ablations for DBF6 (Box).

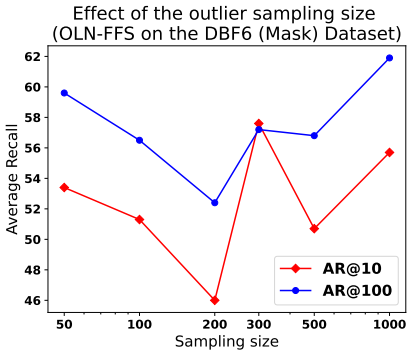

Fig. 13: Ablations for DBF6 (Mask).

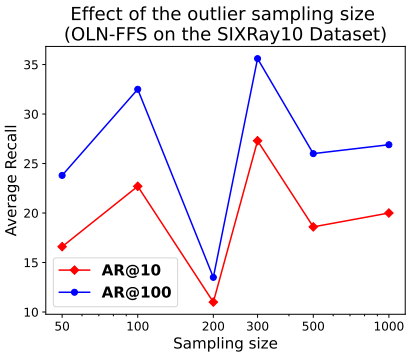

Fig. 14: Ablations for SIXRay10.

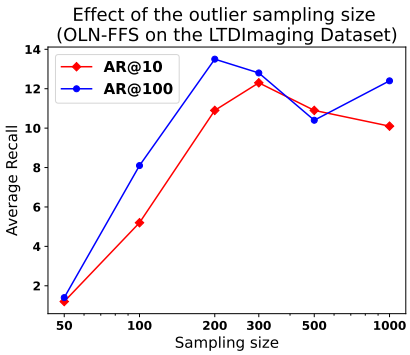

Fig. 15: Ablations for LTDImaging.

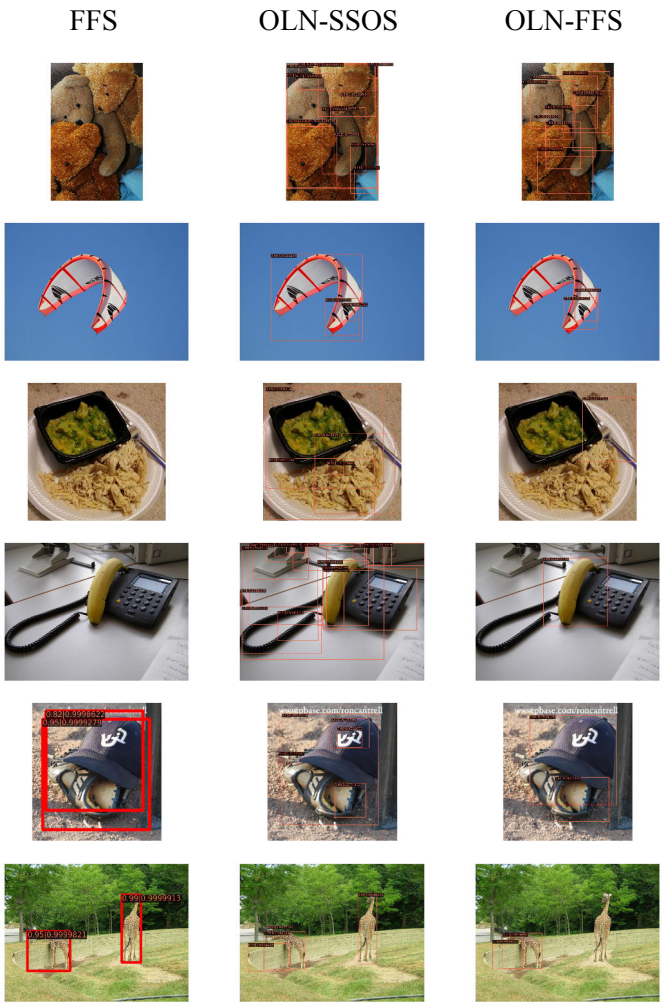

Fig. 16: Qualitative results for VOC/COCO. It is observed that while OLN-SSOS gets more objects, OLN-FFS can get objects with more quality. In some instances. OLN-FFS misses objects of interest. It is also observed that FFS only detect objects closer to the training set, like animals or a cap (similar to a human head).

G Qualitative Results

Figs. 16 to 20 show more qualitative results for the bounding box models (only the baseline FFS [28] is included). Fig. 21 shows additional qualitative results for our mask models.

Fig. 17: Qualitative results for BDD/COCO. Similar to VOC/COCO, OLN-FFS gets less objects but with more quality (see the last row). In this example, FFS gets less images since it has fewer training classes.

Supplementary Material References

1. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)

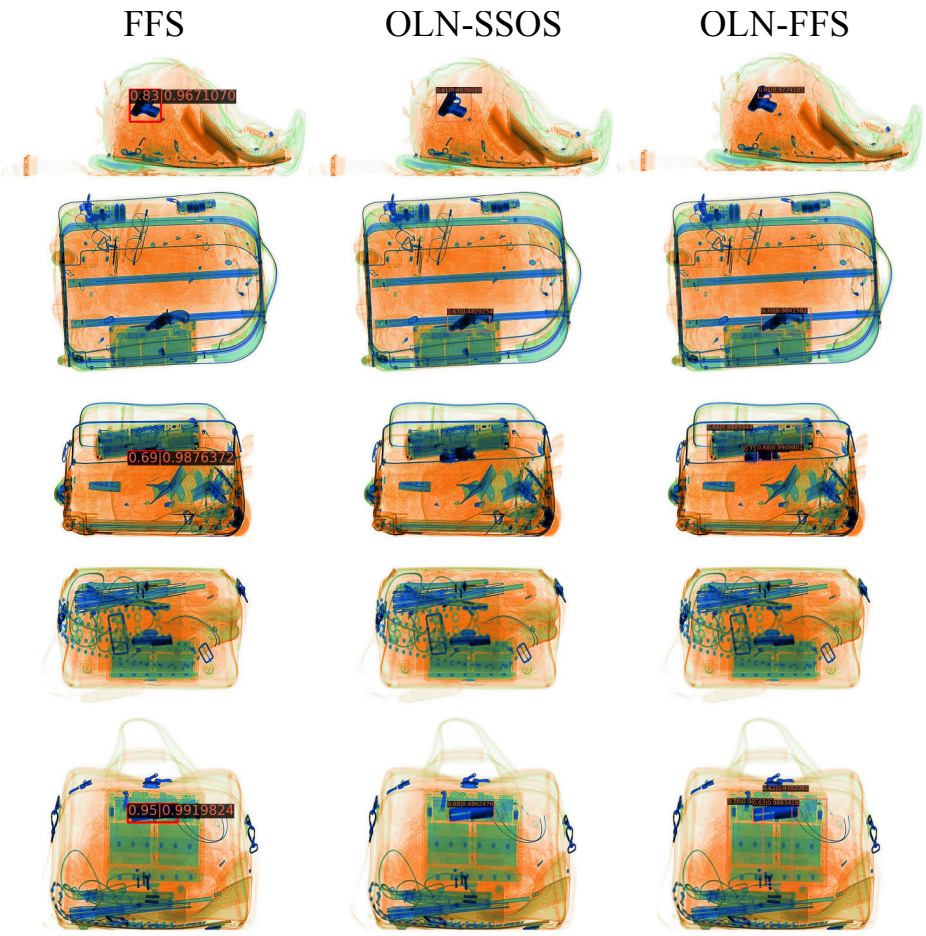

Fig. 18: Qualitative results for DBF6 (Box). While FFS has a relative good performance, it sometimes misses objects like the firearm in the second row. Additionally, OLN-FFS detects other anomalies that are not in the test set, like the tablet in the fourth row. There are some cases where none of the models can detect the anomaly, like the fifth row.

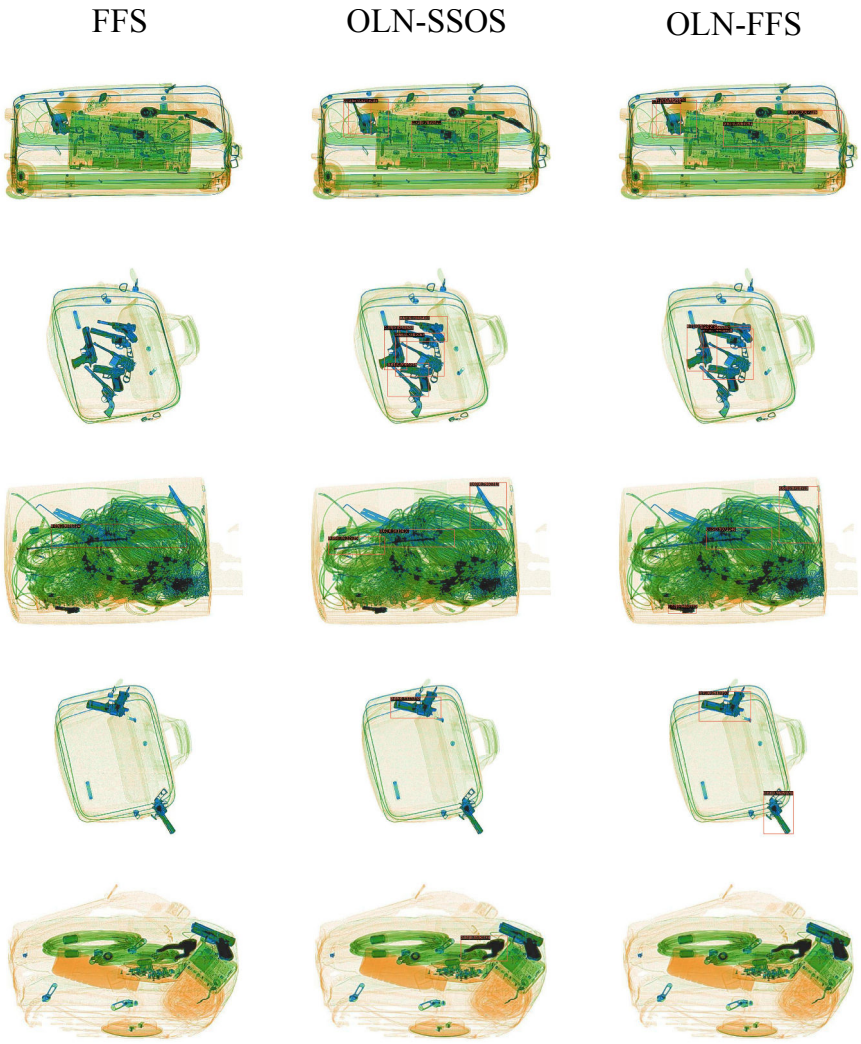

Fig. 19: Qualitative results for SIXRay10. In all of the examples, FFS misses the anomaly, while OLN-SSOS and OLN-FFS get most of the anomalies.

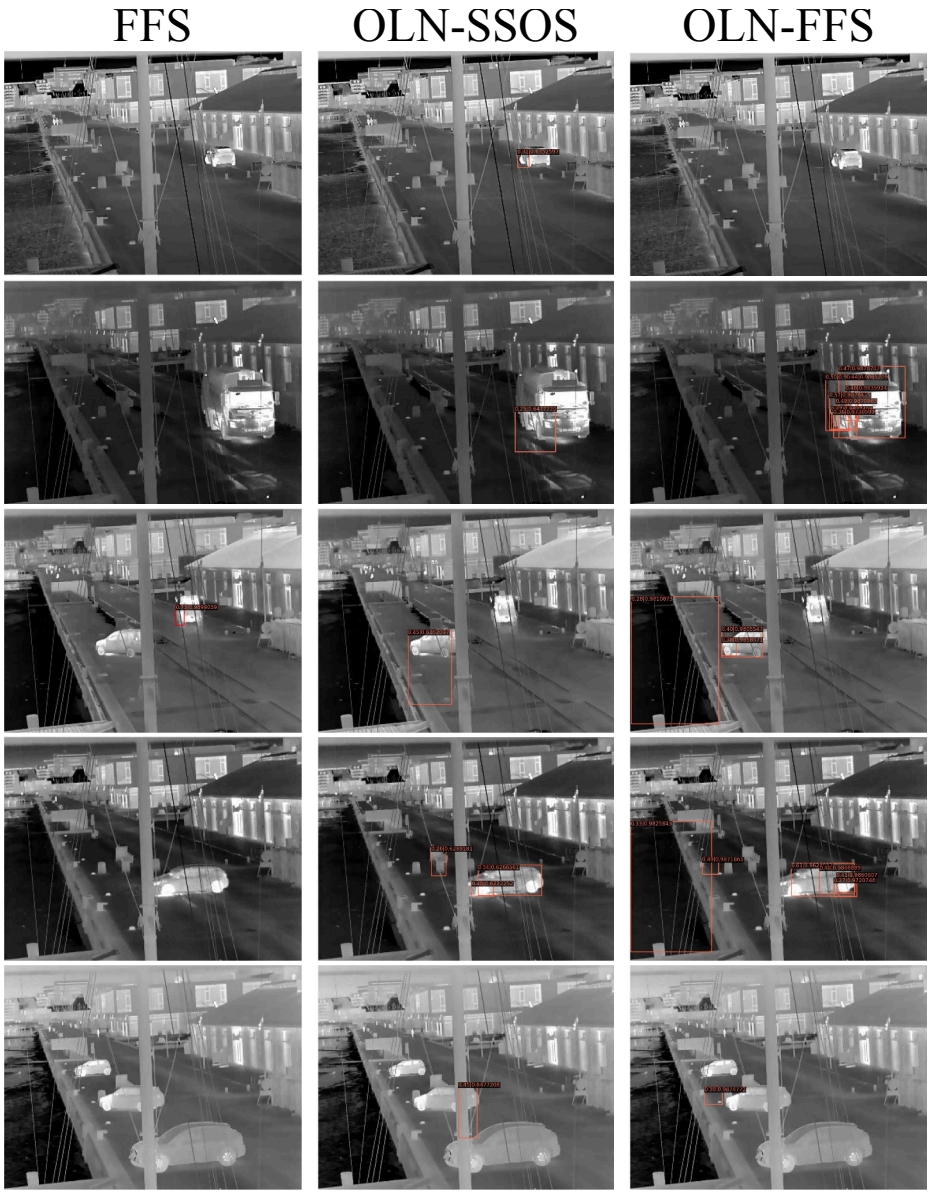

Fig. 20: Qualitative results for LTDImaging. Similar to SIXRay10, FFS misses almost all the anomalies. It can be seen that OLN-FFS detects more anomalies than OLN-SSOS, although they both fail in some instances (last row).

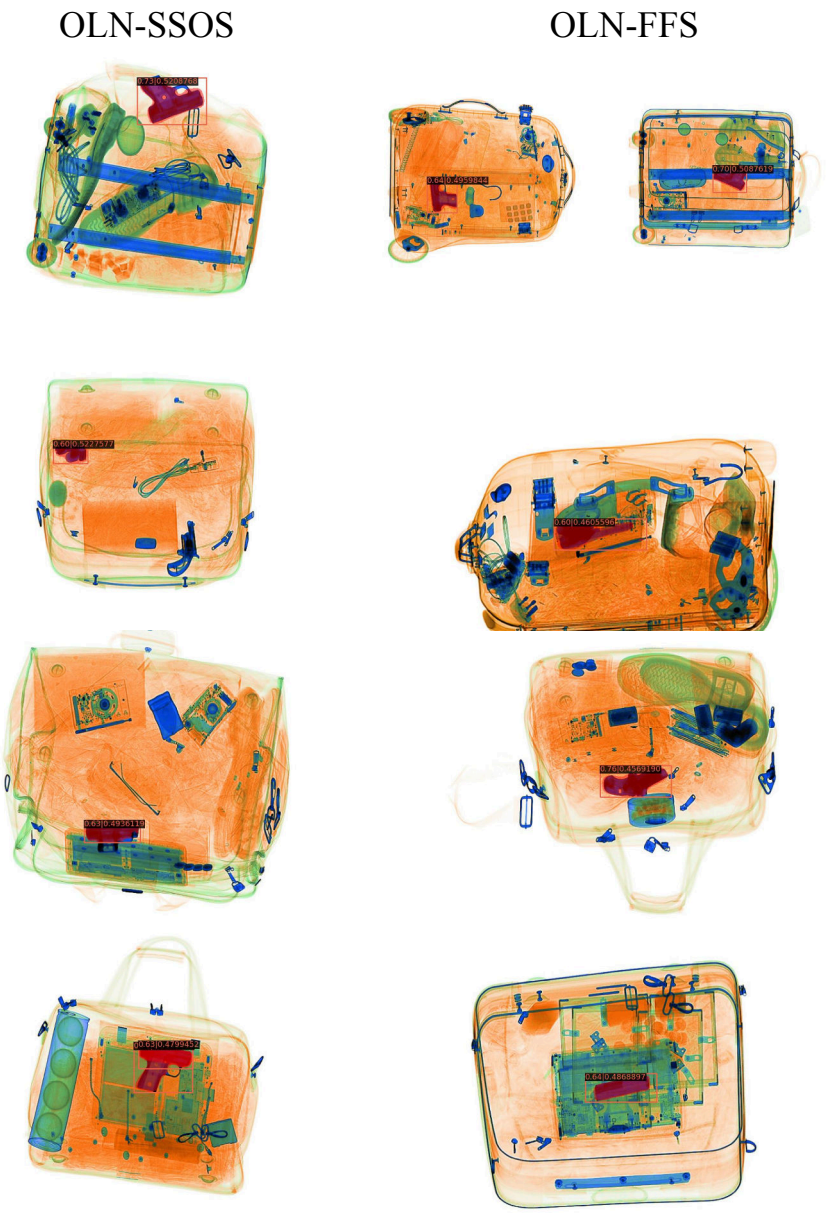

Fig. 21: Qualitative results for DBF6 (Mask). No baseline is presented since there is no baseline for instance segmentation. In all of the examples, it can be noted that both methodologies get the correct segmentation mask, with the exception of the missed gun for OLN-SSOS in the second row.