Analysing Operational Performance of Few-Shot Learning Using Synthetic Data

Niclas Hansson[a*], Erik Persson[a*], Sidney Rydström[a*], Hannes Ovrén[a*], and Niclas Wadströmer[a]

[a]FOI, Swedish Defence Research Agency, SE-164 90 Stockholm, Sweden

ABSTRACT

Traditional machine learning techniques for vehicle detection and classification requires large amounts of annotated images. For applications in defence and public security, obtaining this is usually not possible, e.g., because it is difficult, or even impossible, to get access to the relevant environments and vehicles. Few-shot learning is a research area which aims to design models that can perform well with only a few training examples and can therefore be useful for these types of applications.

In this work, we evaluate how few-shot learning can be used to solve the problem of limited data for a UAV-based vehicle detection scenario. Two few-shot learning methods, Meta-DETR and CD-ViTO, are evaluated with respect to their performance given different numbers of training examples. Their performance is compared to a traditional baseline, Faster R-CNN, that is instead trained on large amounts of data. We use a synthetically generated dataset, and further describe how this dataset is designed.

We show that Meta-DETR has solid performance on our dataset given the small amounts of data, but does not reach the performance of the traditional baseline method Faster R-CNN. In contrast, CD-ViTO performed very poorly on our dataset and our analysis shows that this is likely because the DINOv2 features used for prototypes are not expressive enough to distinguish between the different vehicle classes.

Keywords: few-shot learning, synthetic data, operational performance.

1. INTRODUCTION

Object detection in images is one of the main uses of machine learning in defence and public security. Examples of applications are munitions guidance, autonomous navigation, and surveillance. Current deep learning-based approaches to image object detection can typically perform very well in a wide range of scenarios [1].

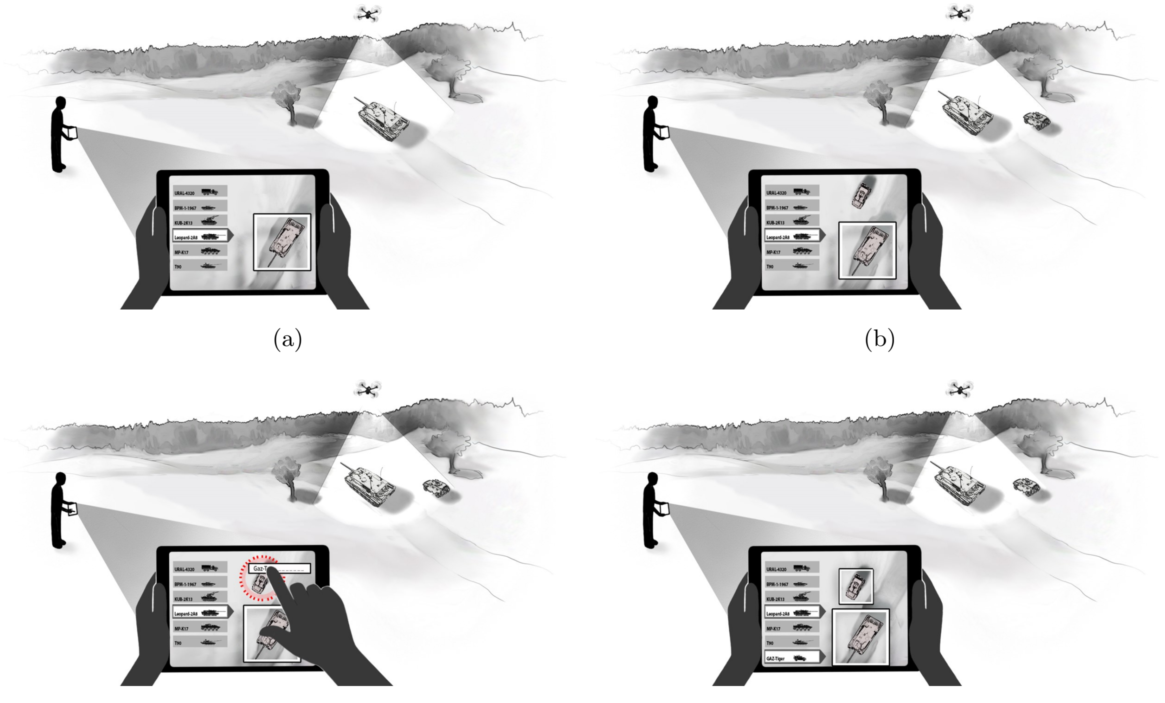
These object detection models are traditionally trained using large amounts of data which should cover the variations that are expected for the intended application, e.g., weather, sensor characteristics, and target distances. For defence applications it is often difficult or expensive to collect the amounts of data that is required to train robust object detection models. This is especially true when the objects of interest, e.g., military vehicles, are operated by an adversary. In defence, the reality is thus that machine learning models need to be trained using datasets that are very small compared to many civilian applications. Few-shot object detection attempts to tackle this issue by first training on many labelled samples on some classes, called base classes, and then leverage these to learn previously unseen, or novel, classes using only a few samples. The number of samples per novel class is called number of shots (e.g. 10 shots is equivalent to 10 samples per novel class) and this will be the naming convention used in this paper.

Few-shot object detection can be divided into three phases. First, the model is trained on the base classes using a dataset that is not constrained by the amount of data. Next, the model is fine-tuned on a more limited few-shot dataset which contains both the base classes and novel classes. This way, the model can utilise the base objects to compensate for the smaller amount of novel objects. Finally, the model is evaluated with a test set containing both base and novel classes [2].

Further author information: (Send correspondence to Niclas Hansson)
Niclas Hansson: E-mail: niclas.hansson@foi.se

[*]These authors contributed equally to the work.

Artificial Intelligence for Security and Defence Applications III, edited by Hugo J. Kuijf,
Radhakrishna Prabhu, Yitzhak Yitzhaky, Proc. of SPIE Vol. 13679, 1367904
2025 Published by SPIE · 0277-786X · doi: 10.1117/12.3070086



(a) (b)

(c) (d)

Figure 1: The UAV surveillance scenario.

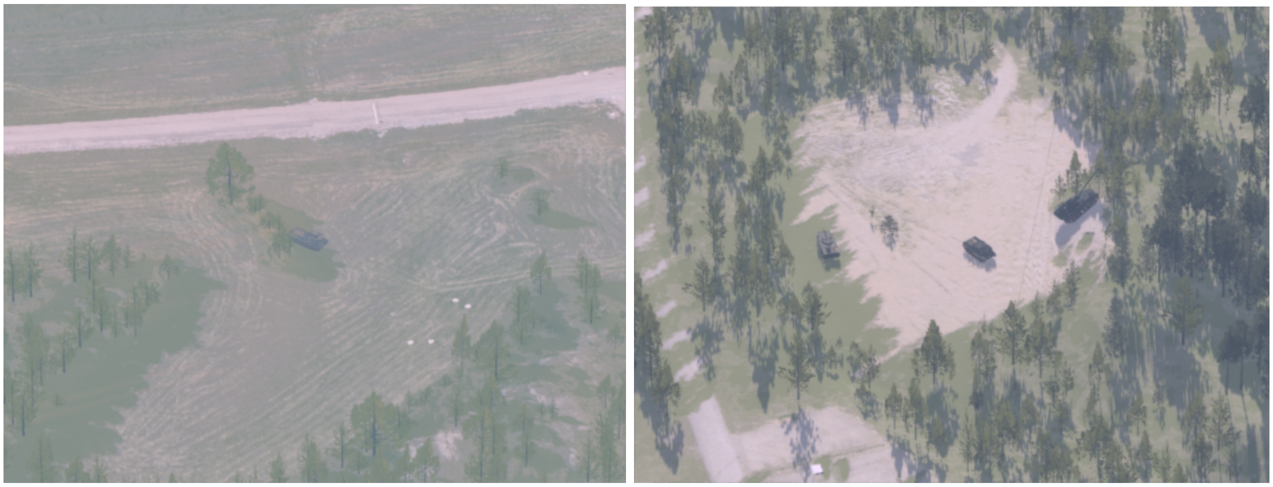
While few-shot object detections algorithms differ in model architecture, the methods can be divided into two primary paradigms: meta-learning and prototype-based learning. The former evolves around optimising a model into a task-agnostic predictor, so that it can more easily be fine-tuned for the specific data. In comparison, prototype-based learning relies on foundation models as its backbones. These are large models that have been trained on a large, diverse dataset to create meaningful image representations that can be used for, among many things, object detection [2].

This work concerns a scenario where an Unmanned Aerial Vehicle (UAV) is used for surveillance. The sensor operator can not only view output from the onboard sensor (more specifically, camera images in the visual spectrum), but is also supported by object detections provided by a machine learning model. The machine learning model has been trained on a set of vehicles which it is supposed to detect. Figure 1 shows a storyboard of how the system is operated: (a) The UAV is operating in an area and observes a vehicle which is part of the library of known vehicles; (b) A new vehicle which is not in the library, and is thus not recognized by the model, enters the field of view of the sensor; (c) The operator marks the new vehicle and captures a small set of training images that is then used to update the model; (d) After the model is updated the new vehicle type is detected along with the previously known vehicles. Here the time between discovery (c) of a new interesting vehicle type to deploying an updated model (d) is shown as an instant process, while in reality this could take anywhere between minutes to days depending on, e.g., the detection model, available compute infrastructure and policies regarding quality control.

In this work we investigate how few-shot learning can be used to solve the issue of limited data availability in this particular scenario. We evaluate two different approaches to few-shot learning and compare their performance to a more traditional baseline method that has access to large amounts of data. In order to perform a systematic evaluation, and since the traditional baseline requires large amounts of data, we choose to generate synthetic data. In addition to the object detection evaluation we also cover the design and generation of this dataset.

Table 1: Statistics for our synthetic data partitions, where some are reported as *mean* ± *one standard deviation*. Pixels per instance corresponds to the object size with the surrounding bounding box annotation.

Partition name	Images	Classes	Instances per image	Pixels per instance
Base training data	100 000	18	3.8 ± 1.9	$26.5^2 \pm 12.8^2$
Novel training data	2 426	9	1 ± 0	$27.9^2 \pm 14.4^2$
Test data	5 000	27	4.9 ± 1.4	$26.5^2 \pm 13.1^2$
Baseline training data	100 000	27	4.8 ± 1.4	$26.5^2 \pm 12.9^2$



(a) Example of *novel* training data.

(b) Example of *base* training data.

Figure 2: Example images of the training environments.

2. METHOD

The objective of this work was to evaluate the use of synthetic data and few-shot learning for the UAV-based vehicle detection scenario described in Section 1. This was performed by generating a synthetic dataset on which three different algorithms for object detection were evaluated: A traditional baseline to indicate the level of performance that can be reached when plenty of data is available, and two few-shot algorithms.

2.1 Synthetic Data

Collecting representative data for our application in volumes that constitute a solid foundation for evaluating object detection algorithms is not feasible. In a frugal data setting, the evaluation process is data-scarce since it constitutes a part of the development process. However, sufficient evaluation data is necessary for conducting algorithmic research.

Our data consists of images from a simulated environment rendered with physics-based models, e.g., scattering and reflectance, with vehicles inset using SE-Workbench [3]. The environment studied in this work consists of Swedish rural terrain with mainly fields, forests, and roads, as well as sparsely populated areas. Further variations in the simulations include, but are not limited to, the time of day and weather conditions such as varying degrees of cloudiness and sunlight. As a result of the simulation process, the data are accompanied by perfect ground-truth metadata and annotations. All images are annotated with COCO-format [4].

Although the rendered images are not perfect replicas of naturally collected images, we consider that our synthetic images are representative surrogates for the purpose of studying how different few-shot learning algorithms perform under varying conditions.

Our dataset consists of four partitions: *base* training data, *novel* training data, *baseline* training data, and *test* data. All images have a resolution of 640 × 480 pixels, and further specifications are listed in Table 1. Example images are provided in Figure 2. The *base*, *novel*, and *test* partitions are generated to support the evaluation of the few-shot object detection models.

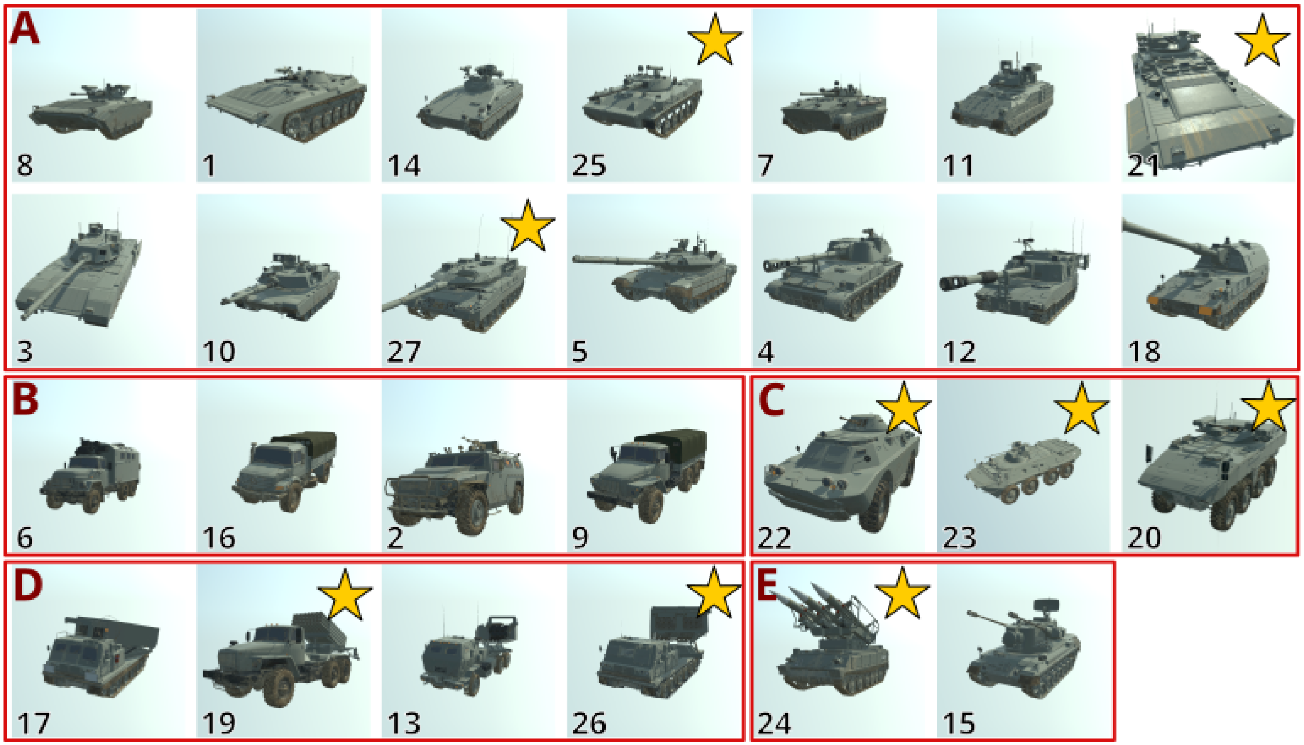


Figure 3: Dataset vehicle classes. Vehicles that are part of the novel classes are marked with a star. The vehicles are grouped into A, tracked armoured vehicles; B, trucks; C, wheeled armoured vehicles; D, rocket artillery; and E, air defence.

Table 2: Vehicle classes used in our synthetic dataset. Star indicates a novel class.

(1) BMP-1	(7) BMP-3	(13) M142	(19) ⋆ BM-21	(25) ⋆ BMD-3
(2) Tigr	(8) BMP-2	(14) Marder	(20) ⋆ VPK-7829	(26) ⋆ M270 MLRS
(3) T-14	(9) Ural-4320	(15) Flakpanzer Gepard	(21) ⋆ T-15	(27) ⋆ Leopard 2
(4) 2S3	(10) M1A2	(16) Mercedes-Benz Zetros	(22) ⋆ BRDM-2	
(5) T-90	(11) M2A3	(17) M270 MARS II	(23) ⋆ BTR-82	
(6) ZIL-131	(12) M109A6	(18) Panzerhaubitze 2000	(24) ⋆ 2K12	

To contextualize the few-shot learning results for our application, the separate *baseline* training data partition includes all classes to enable the use of traditional learning algorithms. All the training data partitions (*base*, *novel* and *baseline*) are simulated in a joint geographical environment, whereas the *test* data partition is simulated in a separate nearby area. In general, the training data partitions consist of variations (e.g., object placement, weather, and lighting conditions) for each generated image. For the *novel* training data partition, which is designed to replicate the surveillance scenario described in Section 1, multiple camera viewpoints are included for each variation to simulate observations during a UAV overflight.

The dataset consists of 27 different military vehicles which are listed in Table 2, and visualized in Figure 3. The vehicles were divided into a fixed training split of 18 base and 9 novel classes, the latter indicated by a star in both Table 2 and Figure 3. Given the division of the vehicles into categories A-E, the split was motivated as follows. All vehicles in group C are used as novel classes, which means that the few-shot algorithms need to be able to learn a whole new category of vehicles (wheeled armoured vehicles). The 2K12 vehicle in group E was chosen to represent a novel class which is very dissimilar from any of the base classes. The rest of the novel vehicles, in group A and D, are all examples of vehicles of which the same type are already present in the base classes.

2.2 Evaluation Protocol

To put results from the few-shot learning algorithms in context, we benchmarked our dataset using a traditional object detection method as a baseline. This is described in Section 2.2.1, followed by Section 2.2.2 where we introduce the evaluation metrics used and how they are applied.

2.2.1 Traditional Baseline

The role of the traditional baseline is to provide an indication on the upper bound of performance of the few-shot methods. This is achieved by training the traditional baseline model on the *baseline* training data partition, which contains a large amount of images for all classes, both base and novel.

As a traditional baseline we chose the Faster R-CNN [5] object detection model with a ResNet-101 [6] backbone pre-trained on ImageNet-1k [7]. It has a Region Proposal Network (RPN) which first detects regions of interest, which are then processed by a Fast R-CNN [8] module to produce the final detections. The entire model was fine-tuned on all classes for approximately 27 epochs, with a batch size of 12 and a learning rate of 0.001, on the *baseline* data. The traditional baseline was trained using the Detectron2 framework [9].

2.2.2 Metrics and Analysis

Average Precision (AP) [10] is a summarization metric for the precision-recall curve, and is the standard metric for object detection. In our experiments, we specifically report AP50, which correspond to a Intersection over Union (IoU) threshold at 50%. This choice is motivated with our dataset consisting of relatively small objects and the bounding-box (localization) accuracy is not critical for our application. According to the COCO standard, our dataset (based on bounding box sizes) contains only small objects (fewer than 32^2 pixels) and medium objects (at least 32^2 but fewer than 96^2 pixels). For size dependent evaluation, we report AP50 as AP50s for small objects and AP50m for medium objects.

Our analysis includes different number of shots, and to estimate how sensitive the few-shot learning models are to the images chosen for training, we repeat our experiments to report both mean and standard deviation. For more in-depth analysis of classification results, we use accuracy [11], the percentage of correct classifications, and confusion matrices, the distribution of predicted class versus actual class.

Reported confusion matrices are constructed using an IoU threshold and model confidence threshold of 50% each. For each ground truth box, the prediction with the highest confidence is selected, given the IoU and confidence thresholds. If there are no predictions given the thresholds, it is assumed that the model predicted the region as ”background”.

2.3 Few-Shot Algorithms

We chose to evaluate two different approaches to few-shot learning: meta-learning and prototype-based networks built on foundation models. The two models chosen are Meta-DETR [12] and CD-ViTO [13]. Both models have reported well-performing results and have explicit approaches to deal with fine-grained object detection. One major advantage of prototype-based methods like CD-ViTO over meta-learning methods is their ability to quickly adapt to novel classes. Their disadvantage is however that they are limited by the ability of the, typically frozen, feature extraction model that creates the prototypes. In comparison, meta-learning approaches like Meta-DETR learns and fine-tunes a feature extraction algorithm. Compared to prototype-based methods, this allows the model to more greatly adjust to the task specific data. The disadvantage, however, is that there is a higher risk of forgetting or neglecting base classes as the feature extraction is fine-tuned on novel classes. In our setup, all novel classes are added simultaneously from the *novel* training data for both models.

The two models also differ in that Meta-DETR is a one-stage detector, while CD-ViTO is a two-stage detector. Meta-DETR reports an increase in inter-class correlation and thus model generalization using their approach. However, the model is less modular compared to a two-stage detectors, where parts of a model can be reused or rebuilt. This is the case for CD-ViTO, where the authors use a traditional Faster R-CNN [5] and keep its core localization ability while replacing the classification module.

2.3.1 Meta-DETR

Meta-DETR [12] incorporates meta-learning into the Deformable End-to-end Detection Transformer (DETR) framework [14]. This is a one-stage detector that performs both classification and localization as one task rather than separating the two, dismissing the need of region proposals. The model takes one query image and multiple support images, extracting query features and support features. When the model performs feature correlation on a target, it uses multiple representations of the query and support features simultaneously and can thus exploit inter-class information better by mapping them to the same feature space. This allows Meta-DETR to increase its ability to perform fine-grained classification. Finally the model uses a transformer architecture to detect objects by predicting their location and feature encoding.

We trained Meta-DETR using a ResNet-101 backbone, pre-trained on ImageNet-1k. The hyperparameters for fine-tuning were chosen based on the results by the authors of Meta-DETR [12]. The algorithm was domain-adapted by fine-tuning on *base* training data for 600 epochs and a learning rate of 0.0001. Using 10 % of the *base* training data as validation data, the initial hyperparameters where determined to achieve satisfactory result and the optimal epoch was determined to be 199. After the pre-training phase, the model was adapted to the novel vehicle classes by fine-tuning using the *novel* training data. To prevent decreasing the performance for the base classes, the fine-tuning process also contains a random sample from the *base* training data as support instances to balance the novel classes.

Fine-tuning for 600 epochs, with a batch size of 16 and a learning rate of 0.0001, demonstrated well-balanced results. The experiments were repeated 20 times with identical settings apart from different random sampling of the training data partitions for novel and base classes.

2.3.2 CD-ViTO

CD-ViTO [13] is a two-stage detector built upon DE-ViT [15]. By separating the localization and classification, the model is able to perform few-shot object detection with minimal fine-tuning on the localization module for a Faster R-CNN model generates proposals bounding boxes around areas of interest. Next, these proposals are fed through DINOv2 [16] and are then compared to a gallery of support prototypes of both class objects and background objects that can be generated in advance. DINOv2 is a Vision Transformer (ViT) trained by self-supervision that is used to create meaningful representations based on image content. It can thus act as a foundation model to be used as a task-agnostic feature extractor suitable for few-shot learning. Thus, the prototypes for novel objects can be replaced with new ones and the model configuration and weights can be reused under the premise that the novel objects have similar shapes as the base objects and can thus be found using the RPN.

The authors of DE-ViT identified limitations regarding accurately localizing objects when using a region-based detector. This was addressed by adding a voting system for the different proposals, which enhanced the model's localization performance. Following that, the authors of CD-ViTO targetted the challenge of cross domain adaptation. They approached this by introducing learnable instance features to the class prototypes to improve fine-grained classification. This process involves first training on a general dataset such as COCO [4] and then fine-tuning the model on the target dataset.

We trained a Faster R-CNN [5] model with a ResNet-50 [6] backbone (pre-trained on ImageNet-1k [7]) for 16 epochs on the *base* training data. Next, the weights checkpoint from the Faster R-CNN model was used in combination with the weights of DINOv2 (ViT-large with a patch size of 14). The combined checkpoint was used as input to base train CD-ViTO on the *base* training data for one epoch. For this step, a subset of the *base* training data of 40 shots per class was used to generate prototypes for the model. To improve the classification, background prototypes were generated as in the original paper on CD-ViTO. This was done by taking randomly selected patches from the *base* training data and parse them through DINOv2.

For the few-shot scenario, novel and base classes were selected from their respective datasets. These were uniformly sampled, with an equal number of shots for novel and base classes. The same background prototypes were used as in the base training. The models were fine-tuned for 60 epochs for 1 shot, 30 epochs for 5 shots, 30 epochs for 10 shots, and 30 epochs for 20 shots, inspired by the authors of CD-ViTO and their fine-tuning for the DIOR [17] dataset. DIOR is an air-to-ground dataset containing 20 classes of various sizes and shapes and is benchmarked in CD-ViTO. The DIOR dataset was chosen due to having similar properties to our dataset in terms of object sizes, number of classes and domain.

3. RESULTS AND ANALYSIS

The trained models, as described in Section 2, were evaluated on the *test* data partition, as seen in Table 1, identifying strengths and weaknesses of each model in relation to our dataset.

3.1 Traditional Baseline

Faster R-CNN achieved AP50 of 90.9%, and AP50s and AP50m scores of 89.0% and 96.9%, respectively. When measuring foreground prediction the model achieves an AP50 score of 98.0%. Figure 4 shows the corresponding confusion matrix. The traditional baseline shows good performance on our dataset, however, the results are only used as an indication of an upper bound on the few-shot object detection performance.

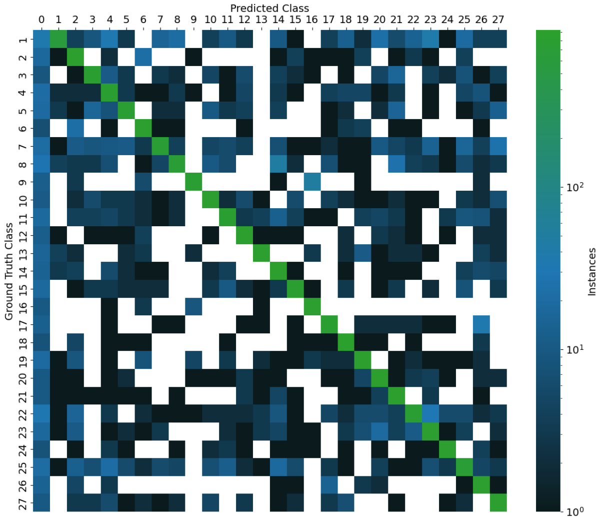


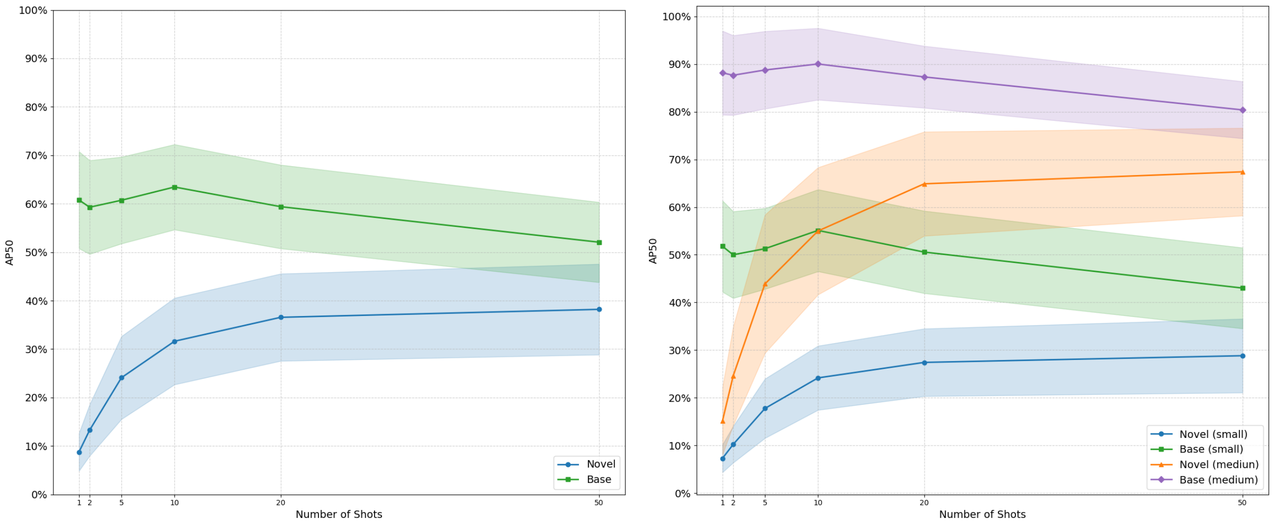
Figure 4: Confusion matrix for the traditional baseline (Faster R-CNN), IoU threshold 50%, score threshold 50%. White represents a count of zero, and *0* in the predicted class represents *Background*.

3.2 Meta-DETR

Meta-DETR achieved a maximum AP50 of 63.4% for base classes (10 shots) and 38.2% for Novel classes (50 shots). Figure 5a shows the averaged AP50 result where the shaded area shows the first standard deviation. It is evident that more data for the novel classes improves the performance for the novel classes on average. The most significant improvement is from 2 shots to 5 shots, and for more than 20 shots (i.e. 50 shots) the AP50 plateaus for novel classes and the performance for the base classes is reduced. Meta-DETR has solid performance on our dataset given the small amounts of data, but does not reach the performance of the traditional baseline.

When manually changing the predictions to foreground predictions to check localization accuracy, the model has an AP50 of 76.2% at 10 shots, indicating that the model struggles with classification rather than localization. The standard deviation in Figure 5a, and the difference between small and medium sized objects in Figure 5b, indicates a possible performance gain depending on the distance and view-angle of the objects in each selected shots. Investigating which distances and view-angles are more beneficial is left for future works.

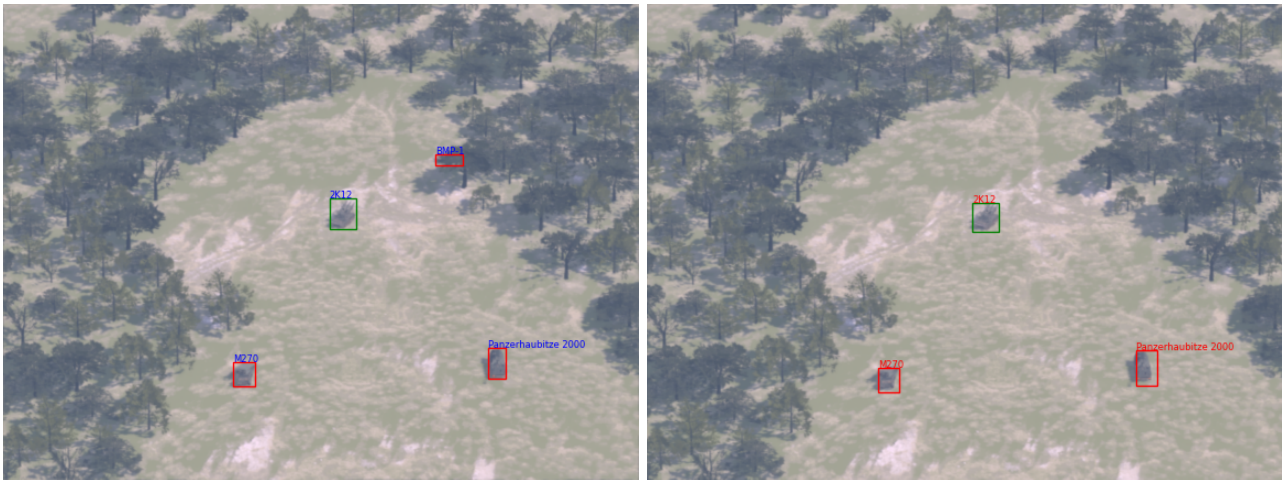
The dependence of the object sizes, measured as the pixels per image, is depicted in Figure 5b. The size-based partitioning generally reflects the overall trend (Figure 5a), but the difference in performance is significant between small and medium size instances. The model achieves at best AP50m of 90.0% for the base classes at 10 shots, and 67.4% for the novel classes at 50 shots. For small objects the best AP50s for the base classes is 55.1% at 10 shots, and 28.8% for the novel classes at 50 shots. Additionally, the performance for medium size objects



(a) All objects.

(b) Size-divided objects.

Figure 5: Meta-DETR AP50 score over 20 seeds. The shaded area indicates first standard deviation.



(a) Ground truth.

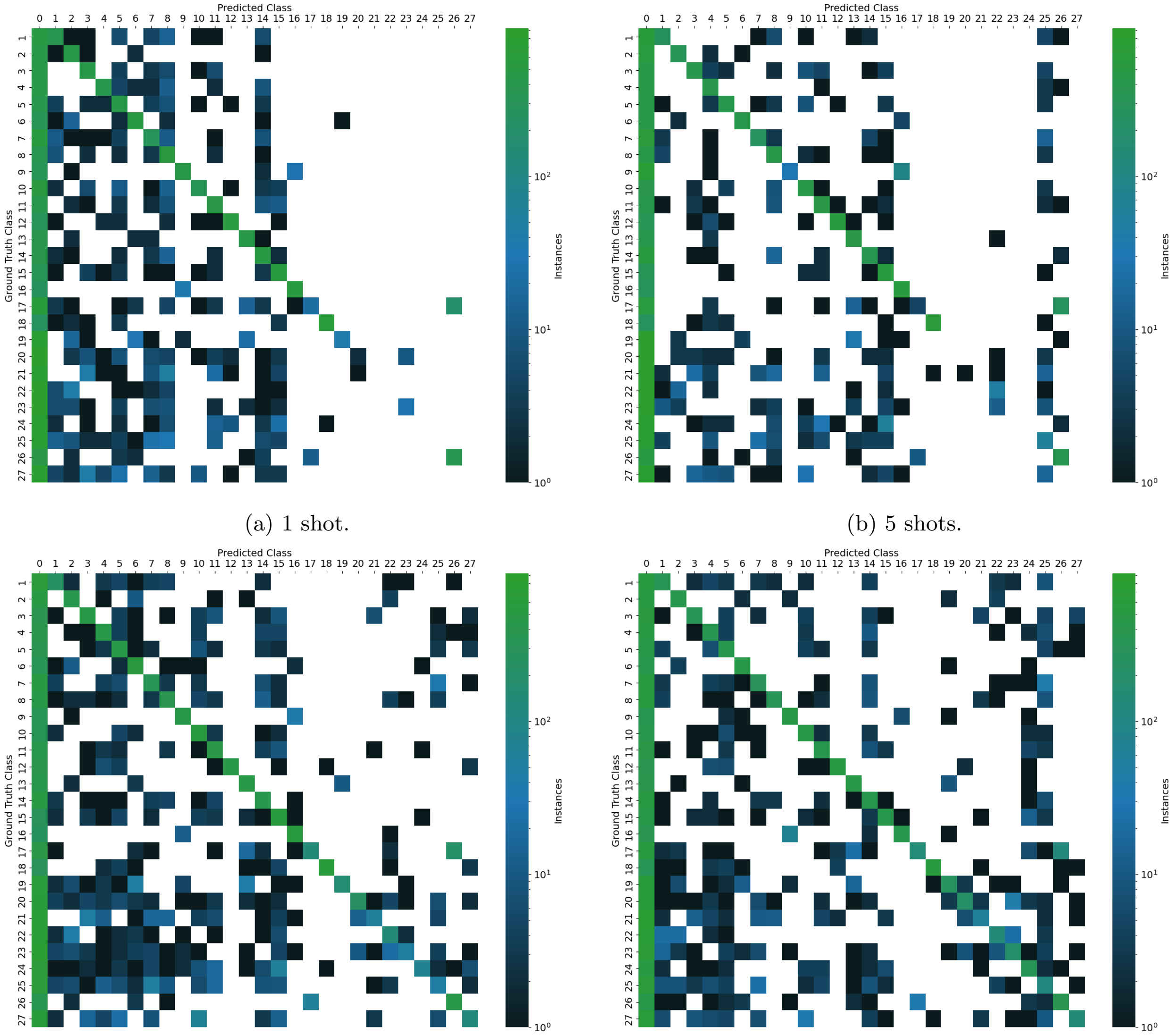
(b) Detections.

Figure 6: Qualitative object detection result from a Meta-DETR model fine-tuned with 20 shots. Ground truth labels are blue and prediction labels are red. Base classes (red box): M270 MARS II (17), BMP-1 (1), and Panzerhaubitze 2000 (18). Novel class (green box): 2K12 (24).

is not only notably better, the novel classes also improves more with additional shots. Notably, the maximum score of AP50m base classes is close to baseline performance.

Figure 6 shows an example of detection performance for a model trained on 20 shots. In the example, three out of four vehicles are successfully detected, and the missing detection is likely due to the vehicle being almost totally occluded by vegetation.

The confidence in correct detections increases with more shots, which is evident when comparing confusion matrices for models with different number of shots in Figure 7a-7d. Compared to the baseline model (Figure 4), the Meta-DETR models misclassify a significant portion of objects as background. This could be due to the poor performance on small objects. According to Figure 7, the weakest performing novel classes suffer from confusion with the base classes. Additionally, each grouping of classes, as described in Figure 3, have a single higher performing class with a lower rate of confusion. This is evident in Figure 7c, when looking at group C (20, 22, 23), where VPK7829 (20) is less likely to be confused with base classes than BRDM-2 (22) and BTR-82 (23)



(a) 1 shot.

(b) 5 shots.

(c) 10 shots.

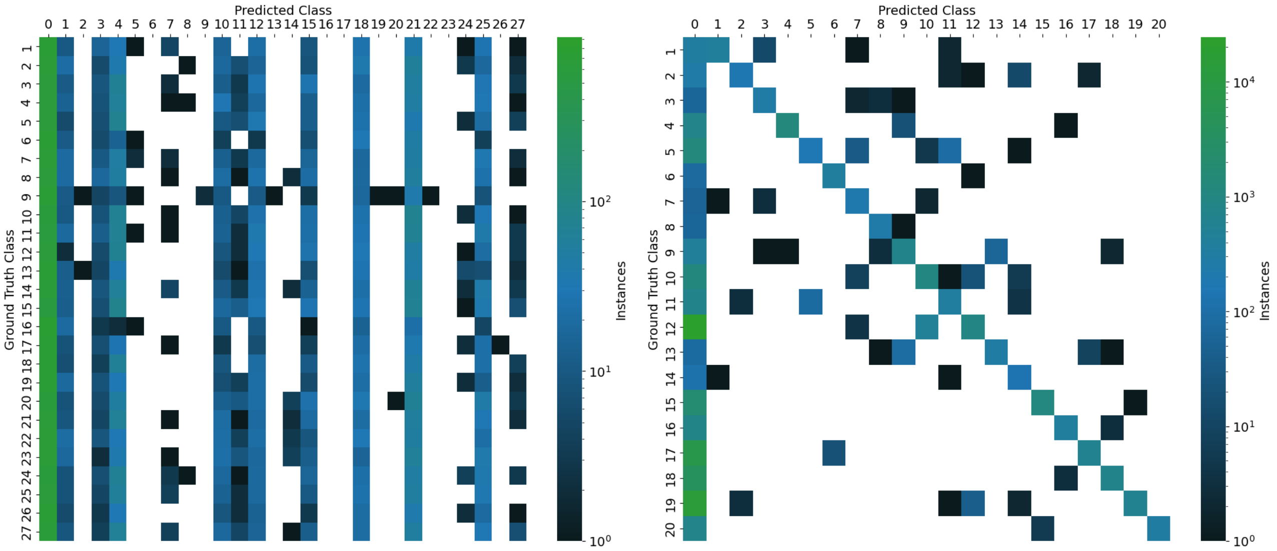
(d) 20 shots.

Figure 7: Meta-DETR confusion matrices for 1 – 50 shots, IoU threshold 50%, score threshold 50% for a random seed. White represents a count of zero, and *0* in the predicted class represents *Background*.

are. Further, the model is more likely to guess VPK7829 (20) when it sees the other two classes within the same group while the inverse is not true. This indicates that the model can differentiate one novel class from several similar base classes well, but has a worse performance overall if the novel classes are similar to each other.

3.3 CD-ViTO

Our experiments showed that CD-ViTO reached at best 1-2% in AP50 on our dataset. Since the results are too poor to be operationally useful, further analysis of performance, similar to what we did for Meta-DETR in Section 3.2, is irrelevant. Instead we opted for an extended analysis in order to investigate why the method failed on our dataset, while results on public datasets have been state of the art for few-shot object detection. Therefore, as a first experiment, the predictions were manually changed into foreground predictions to give an indication of the localization ability of the model. The 10 shots AP50 then reached 29.4%. This indicates an issue with the classification rather than the localization.



(a) Our dataset.

(b) DIOR.

Figure 8: CD-ViTO confusion matrices for the two datasets. IoU threshold 50%, score threshold 50% for a random seed. White represents a count of zero, and *0* in the predicted class represents *Background*.

Therefore, to further evaluate the results, the model was trained on the DIOR dataset as reference, with the model weights pre-trained on COCO as with our dataset. The DIOR dataset was chosen as it has already been benchmarked in the original CD-ViTO paper. To give comparable results, the DIOR dataset was split into 14 base classes and 6 novel classes. The novel classes were chosen arbitrarily and 20% of the dataset was set aside for testing while the rest was used for bulk and few-shot training. For the DIOR dataset, CD-ViTO reached 38.9% in AP50 for 10 shots and 36.1% in AP50 for 5 shots. However, it is clear that the AP differs between classes. Base classes may reach as high as 76.1% in AP50 (for class 6, ”chimney”) and as low as 3.6%, while novel classes range from 4.2% to 51.7% (for class 19, ”vehicle” and class 13, ”stadium” respectively). Moreover, when performing foreground predictions the model only reaches 16.1%. This means that the model is able to localize the objects more easily for our dataset, but classification is an easier task for the DIOR dataset.

This phenomenon is even more apparent when the misclassifications for the two datasets are compared using a confusion matrix. As seen in Figure 8, the model behaves differently for the two datasets. For DIOR even with the two worst performing classes, the base class 12 (”ship”) and novel class 19 (”vehicle”), the model is not likely to mistake them for another class. Rather, the model either misses the class completely or predicts it correctly, apart from class 12 (”ship”) sometimes being misclassified as class 10 (”harbor”) and ”vehicle” being interpreted as class 12 (”ship”). In comparison, the model appears to choose class at random for our dataset, with low confidence, on a few selected classes based on no obvious characteristic. Furthermore, there is no indication as to any bias to choose base over novel classes. The selected classes show no patterns of favour for specific groups (see Figure 3) and roughly half of the classes were simply ignored by the model.

The classification ability was further investigated by using linear probing with DINOv2, as it is closely tied to CD-ViTO's classification performance. In this setting, images were cropped using the ground truth bounding boxes and then fed to DINOv2 to extract the objects' feature embeddings. This allowed the analysis to be centred around the classification task while the overall object detection task remained unchanged. With the embeddings as input, a linear classification layer with a Softmax activation was trained using cross-entropy loss to perform classification among the 27 classes for our dataset and the 20 classes for DIOR. The model was trained for 12 500 iterations of mini-batch gradient descent with a batch size of 32, inspired by the linear probing proposed by the authors of DINOv2. The training procedure was repeated 10 times with different image samples. The inference, however, was performed with the full respective test data to ensure consistency between training runs. The results are shown in Figure 9 for the two datasets.

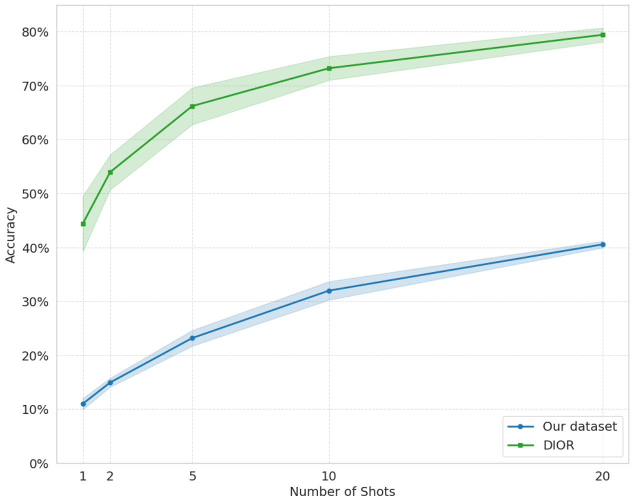


Figure 9: Classification accuracy of DINOv2 averaged over 10 seeds. The shadowed region represents one standard deviation.

As seen in Figure 9, the classification ability of DINOv2 is significantly higher for the DIOR dataset than for our own dataset. While they have a similar relative performance increase with the number of shots, our dataset needs more than 20 shots per class before attaining a result on par with the 1 shot performance on DIOR. Naturally, the datasets are not completely comparable due a differing number of classes, but even so it is apparent that DINOv2 lacks the ability to easily distinguish between the classes, in our dataset. This raises the question of whether this originates in the object sizes, the fine-grained difference between certain classes, or a domain shift to synthetic data. Due to our dataset being limited by the scenario, an analysis with data that uncover these characteristics is left for future work.

4. CONCLUSION

For fine-grained object detection, this work investigated the capability of few-shot algorithms on previously unseen military vehicles. The use of synthetic data enabled statistical results for algorithms on objects that are generally difficult to study otherwise.

Meta-DETR achieved competitive performance for the base classes while being robust to the performance of the novel classes, although it was sensitive to few-pixel objects. For the novel classes, the results are promising with very few examples, but the performance plateaus beyond 20 instances per novel class.

CD-ViTO underperformed in our application despite demonstrating strong localization capability. Extended experimentations implied performance limitations due to low classification accuracy with DINOv2 on our objects, including the base classes.

In summary, we have shown that few-shot object detection can be successfully applied to military relevant scenarios with fine-grained classes. However, future work should address the reduced performance on few-pixel instances, as this is especially common in the military domain.

ACKNOWLEDGMENTS

This work was performed within the project *Knowledge Extraction, Machine Learning and other AI approaches for secure, robust, frugal and explainable solutions in Defence Applications* (KOIOS), European Defence Fund (EDF) contract number 101103770.

The authors thankfully acknowledges Spanish Supercomputing Network (Red Española de Supercomputación, RES) resources provided by Barcelona Supercomputing Center (BSC) in MareNostrum to IM-2025-1-0019.

The authors would like to thank Esther Eriksson, Pär Glendor, and Jonas Allvar for their help with 3D-modelling, simulation setup, and storyboard illustrations.

REFERENCES

[1] Gallagher, J. and Oughton, E. J., "Transforming the Multidomain Battlefield with AI: Object Detection, Predictive Analysis, and Autonomous Systems," *Military Review* **September 2024 Online Exclusive Article** (2024).

[2] Huang, G., Laradji, I., Vazquez, D., Lacoste-Julien, S., and Rodriguez, P., "A Survey of Self-Supervised and Few-Shot Object Detection," *IEEE transactions on pattern analysis and machine intelligence* **45**(4), 4071–4089 (2023).

[3] OKTAL-SE, "SE-Workbench," (2025). https://www.oktal-se.fr.

[4] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L., "Microsoft COCO: Common Objects in Context," in [*Computer Vision – ECCV 2014*], Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., eds., 740–755, Springer International Publishing, Cham (2014).

[5] Ren, S., He, K., Girshick, R., and Sun, J., "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence* **39**, 1137–1149 (2017).

[6] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," in [*2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*], 770–778 (2016).

[7] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., "ImageNet: A large-scale hierarchical image database," in [*2009 IEEE Conference on Computer Vision and Pattern Recognition*], 248–255 (2009).

[8] Girshick, R., "Fast R-CNN," in [*2015 IEEE International Conference on Computer Vision (ICCV)*], 1440–1448 (2015).

[9] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R., "Detectron2." https://github.com/facebookresearch/detectron2 (2019).

[10] Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., and Zisserman, A., "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision* **88**(2), 303–338 (2010).

[11] Goodfellow, I., Bengio, Y., and Courville, A., [*Deep Learning*], MIT Press (2016). http://www.deeplearningbook.org.

[12] Zhang, G., Luo, Z., Cui, K., Lu, S., and Xing, E. P., "Meta-DETR: Image-Level Few-Shot Detection With Inter-Class Correlation Exploitation," *IEEE transactions on pattern analysis and machine intelligence* **45**(11), 12832–12843 (2023).

[13] Fu, Y., Wang, Y., Pan, Y., Huai, L., Qiu, X., Shangguan, Z., Liu, T., Fu, Y., Van Gool, L., and Jiang, X., "Cross-Domain Few-Shot Object Detection via Enhanced Open-Set Object Detector," in [*Computer Vision – ECCV 2024*], Leonardis, A., Ricci, E., Roth, S., Russakovsky, O., Sattler, T., and Varol, G., eds., 247–264, Springer Nature Switzerland, Cham (2025).

[14] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., and Dai, J., "Deformable DETR: Deformable Transformers for End-to-End Object Detection," in [*International Conference on Learning Representations*], (2021).

[15] Zhang, X., Liu, Y., Wang, Y., and Boularias, A., "Detect Everything with Few Examples," in [*8th Annual Conference on Robot Learning*], (2024).

[16] Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y., Li, S.-W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P., "DINOv2: Learning Robust Visual Features without Supervision," (2024).

[17] Li, K., Wan, G., Cheng, G., Meng, L., and Han, J., "Object detection in optical remote sensing images: A survey and a new benchmark," *ISPRS Journal of Photogrammetry and Remote Sensing* **159**, 296–307 (2020).