Monitoring rail-corridor water from onboard cameras using foundation-model patch descriptors

1st Marit Holden
Image analysis, machine learning and Earth observation
Norwegian Computing Center
Oslo, Norway
marit@nr.no

2nd Anders U. Waldeland
Image analysis, machine learning and Earth observation
Norwegian Computing Center
Oslo, Norway
andersuw@nr.no

3rd Fredrik A. Dahl
Image analysis, machine learning and Earth observation
Norwegian Computing Center
Oslo, Norway
fadahl@nr.no

4th Øivind Due Trier
Image analysis, machine learning and Earth observation
Norwegian Computing Center
Oslo, Norway
trier@nr.no

5th Amund Vedal
Image analysis, machine learning and Earth observation
Norwegian Computing Center
Oslo, Norway
amund@nr.no

***Abstract*—Flooding and standing water near the railway trackbed can compromise safety, increase maintenance needs and, in severe cases, cause derailment risk. We study detection and segmentation of track-adjacent water in forward-looking video frames captured by onboard train cameras under a few-shot supervision regime. Our approach uses a frozen DINOv3 vision foundation model as a dense feature extractor and trains a lightweight linear SVM head on patch-level descriptors trained from only K = 5 labeled images. We evaluate the method on a held-out set of 1500 negative and 1198 positive frames. Performance improves with K, reaching an image-level AUC of 98.27%, a mean Dice score of 0.755 and a global Dice score of 0.877 at $K = 5$. At a conservative decision threshold selected for low false alarms, the false positive rate is 0.8% with a 12.5% false negative rate. Qualitative analysis shows that most errors occur for small or distant water regions, partial occlusions by vegetation, and shallow water on wet surfaces. The results indicate that descriptors from strong pretrained foundation models enable practical few-shot water monitoring from onboard railway video with minimal task-specific training. This can enable scalable monitoring of track-adjacent water from onboard video, supporting maintenance planning and operational decision-making during heavy rainfall events.**

***Index Terms*—flood detection, railway safety**

I. Introduction

Securing the integrity of railway infrastructure is critical for infrastructure owners to provide safe and uninterrupted passenger and freight services. Flooding near the railway is one of several hazards that can compromise track integrity, and it is becoming increasingly relevant as climate change drives more frequent and extreme weather. Water accumulating alongside the trackbed may infiltrate and weaken the subgrade, reduce ballast stability, and trigger erosion and washout of embankment material; in severe cases it may increase derailment risk.

Detecting water accumulation along the railway is useful for two main reasons. First, it can reveal locations with inadequate drainage or blocked culverts intended to channel water beneath the track structure, enabling targeted maintenance to reduce the risk of erosion and washout during heavy rainfall. Second, train drivers may report observed water build-ups, supporting immediate mitigation actions such as reduced speed or temporary line closure. Today, we rely on inspectors being in the right place at the right time and on drivers having sufficient attention to observe and report such events. Since manual inspections are carried out only periodically and drivers must prioritize operational tasks, the current approach to detecting and monitoring flooding is insufficient.

Image foundation models (FMs), trained on large corpora of unlabeled images, have become a standard approach for robust computer vision. Trained on pretext objectives such as reconstructing masked image regions, they learn transferable representations and can often be adapted with minimal task-specific training, requiring substantially less labeled data than conventional supervision. A recent example is DINOv3 [1], released by Meta, which has demonstrated strong performance on segmentation and object detection in natural imagery. In this paper, we apply a frozen DINOv3 encoder and train a linear support vector machine (SVM) head to identify water in images, and apply this to detect and segment flooding near railways. We show that the SVM can be trained with

Funded by the Europe's Rail Joint Undertaking (EURail)

only a small number of labeled images and robustly identifies water near the trackbed. Because real images of flooding near or on the track are scarce, we also demonstrate synthetic data generation using commercially available LLM-based AI systems.

Our main contributions are:

- We propose a computer vision pipeline for detecting and segmenting track-adjacent water from onboard video frames, targeting scalable rail infrastructure monitoring.
- We adapt frozen DINOv3 patch descriptors with a lightweight linear SVM head for few-shot water segmentation in rail corridor imagery.
- We evaluate the approach using both image-level detection performance (AUC based on total predicted water signal) and pixel-level segmentation quality (Dice), and report performance as a function of the number of training images ranging from 1 to 5.

II. Related work

Computer vision has a long history in railway inspection, with a strong focus on detecting defects in rails, sleepers, fasteners, and rolling stock. Recent reviews summarize a broad set of sensing setups and learning-based pipelines, and point out recurring challenges such as varying lighting and weather, domain shift between different railway lines, and limited public datasets for benchmarking [2]. Complementary work also covers remote sensing as part of the monitoring toolbox, often trading local detail for wider coverage [3]. UAV-based inspection has become popular for targeted surveying, but it does not provide the same continuous coverage as train-borne sensing [4].

Flood and surface-water perception from images has been studied in road and urban settings, both as segmentation and as detection for operational decision support. Deep models can segment water from fixed cameras, but reflections, wet asphalt, and low visibility remain common failure modes [5]. For moving platforms, water hazard detection has also been addressed with dedicated architectures that try to handle reflections and appearance changes [6]. Several datasets support flood scene understanding, but many are aerial or scenario-specific, which complicates transfer to train-borne corridor imagery [7], [8].

Self-supervised vision transformers have made it practical to reuse pretrained representations as general-purpose features. DINO showed that strong semantic structure can emerge without labels [9], and DINOv2 scaled this idea to more robust visual features across domains [10]. DINOv3 further emphasizes high-quality dense features, which is attractive when training data is scarce and it is desirable to keep most of the model frozen [1]. In parallel, segmentation foundation models such as SAM have enabled prompt-based mask generation and are widely used as annotation support tools [11].

Few-shot semantic segmentation aims to produce pixel-wise masks from only a small number of labeled examples, and much of the literature focuses on episodic/meta-learning or matching-based architectures [12]. In our setting, we instead follow a representation-based approach: we keep a strong backbone frozen and learn only a lightweight classifier from the few available masks. This is consistent with how self-supervised ViT features are evaluated for dense prediction; for example, DINOv2 reports semantic segmentation results using frozen features with a linear classifier [10], and DINOv3 further targets high-quality dense features that transfer well with minimal adaptation [1]. Recent work on few-shot segmentation also argues that pre-trained ViTs already group patches with similar semantics, which can be exploited under limited supervision [13].

Fig. 1. Examples of image crops that were difficult to annotate accurately due to fragmented water regions.

III. Datasets and annotations

We used a large corpus of forward-looking onboard dashboard camera videos recorded from trains operating across most of Norway's railway network. From this corpus, we constructed (i) a labeled few-shot training set of five frames (Fig. 2), and (ii) a held-out test set of 1198 frames containing water and 1500 frames without water. The image size varied, with an average of about 4200 x 2000 pixels. Positive frames include puddles, trackside ditches with water, shallow standing water near the trackbed, and larger water bodies such as lakes and rivers.

To annotate the images, we manually inspected the video footage to identify frames containing water and sampled frames with sufficient temporal separation to avoid near-duplicates of the same scene. We used the base (ViT-B) version of the original SAM model [11] for annotating water in the images. Since the target water regions were identified by the annotators, we used SAM with point prompts (point-and-click), rather than other prompt types such as masks, bounding boxes, or text prompts.

Accurate annotation of water in natural scenes is often difficult, as vegetation and other foreground objects may partially occlude the water surface. In such cases, a perfect segmentation mask would be highly fragmented, as illustrated in Fig. 1.

To obtain high quality masks for all kinds of images in the training set in the interactive annotation process would have been too time-consuming. We instead adopted a minimalist annotation strategy by excluding heavily fragmented regions from the water masks. There is, however, a gray zone where partially fragmented areas could reasonably be included or excluded. As a result, some mask boundaries reflect subjective judgment, and the resulting ground truth is not expected to be perfectly reproducible. This also applies to the binary image

level classification of images where all, or almost all, of the water was partially obstructed. There were also cases where (not fragmented) specks of water were so small that it was hard to decide if they should count. Overall, the masks were considered sufficiently accurate for evaluating the proposed image classification and segmentation method.

For obtaining the best possible models, it is important to have water masks of high quality in the training set. In addition, the dataset should contain images with both deep and shallow water. As it turned out to be difficult to find such images in the corpus with real images, we instead generated synthetic images from some real images without water in the corpus by adding puddles to the image using ChatGPT. For the first training image, we added a puddle to an image with dense forest in the background. Since initial PCA analyses indicated that sky regions could have features similar to water, we chose a second template with a clear blue sky in the distance and a third one with clouds. The water areas in these images were easier to annotate accurately than those in the test set. We used OpenAI's GPT-image-1 for inpainting puddles into three images from our dataset.

Prompt:

> *Railway track through Nordic landscape. Add a puddle close to the rails in this image. Please leave light conditions as similar as possible to the original image.*

The generation process did not yield usable masks for the inserted puddles. We therefore annotated the synthetic puddles using SAM, following the same procedure as for natural images containing water.

In addition, we chose an image with a large lake with an irregular boundary, and prompted the model to re-create the image, without instructing it to add or remove anything. This gave a new version of the image with a smoother water boundary, which was beneficial to the SVM model. Although the lake in the image was too far away and too low below the track to pose any threat to the train, we included it to train the model to recognize a variety of water surfaces.

Lastly, we added a natural image where most of the terrain was covered with snow, since the initial model was confusing water and snow. This image, as well as those used as templates for the GPT model were chosen from a dataset that did not overlap with the test set.

The modified training images were 1536 x 1024 pixels, which is smaller and has a different aspect ratio than the test set images. The complete training set is shown in Fig. 2, where the first three are the images with added puddles, the fourth is the re-created image with the lake, and the bottom one is natural. The water bodies are shown with transparent red masks.

Fig. 2. The full training set with transparent red segmentation masks, top four are synthetic, while the bottom is a natural image.

Fig. 3. Example images with PCA map from the training set (top) and test set (bottom).

IV. Methods

DINOv3 is an openly available self-supervised vision foundation model [1]. In our setup, we use it as a frozen feature extractor that produces dense feature vectors on a regular grid over the image with patch size of 16×16 pixels. The resulting patch-level descriptors capture visual structure and can be used as input to lightweight downstream classifiers.

A common approach to illustrating localized feature maps is through principal component analysis (PCA). To this end, we have computed the first three principal components from our five training images, and mapped these to the RGB dimensions using the scikit-learn package. Fig. 3 shows the resulting transformations of a training image (top) and an image from the test set (bottom). Note that "training" in this context does not include the modification of any features; it only orients the three display axes in the feature space in the way that best captures the patch level variations in the training images.

We see that even without adaptation to segmentation masks, the model is able to segment different objects reasonably well. With only three PCA dimensions, it assigns similar colors

to water bodies and background pixels. However, when we inspect further PCA components (not shown), it also separates these well.

For adapting the model, we applied a form of linear probing by adding a linear classification on top of the patch-level features, and trained it by SVM, keeping the backbone frozen. This is particularly efficient for cases with only two classes, where a single output value can be used to separate them.

V. Experimental setup

A. Training protocol and evaluation metrics

Our few-shot setup used no more than five training images, and we evaluated the performance for $1, ..., 5$ images included in the order shown in Fig. 2.

The intended use of the model is a two-stage process, where we first classify an image as containing or not containing water, and present a water mask when appropriate. We therefore define separate evaluation metrics for classification and segmentation.

1) Classification: To map the model output to a real-valued statistic, we aggregate the positive part of the SVM patch output scores over the corresponding pixels, and label it $\Sigma(\text{image})$. In cases where all the patches have negative SVM scores, we define $\Sigma(\text{image})$ as the maximum (least negative) over these, as a tie breaker. We measure the classification performance by the area under the receiver operating characteristic curve (AUC) for $\Sigma(\text{image})$. AUC has the equivalent definition of 'the fraction of positive-negative pairs that the model orders correctly', which may be more intuitive. It has the advantage of measuring the model's discriminative power independently of the relative frequencies of the classes, while a practical application would require us to trade off sensitivity for specificity by setting a decision threshold.

2) Segmentation: To measure the model's segmentation quality, we use the Dice score, which is equivalent to a pixel-level F1-score. We include both average Dice score over the positive images (Mean Dice) and a pooled Dice score for all pixels in all images, including the negative ones (Global Dice).

B. Implementation details

We used the DINOv3 backbone checkpoint `dinov3-vitl16-pretrain-lvd1689m`. Patch descriptors of dimension 1024 were taken from the DINOv3 patch tokens. The patch classifier was a linear SVM implemented with the Python package `sklearn.svm.SVC` using a linear kernel.

In the training set, we labeled a patch as water if all pixels in the patch had been labeled as water pixels and as background if all pixels in the patch had been labeled as background. Patches with mixed labels were defined as unlabeled, and disregarded in the SVM optimization. In order to support the analysis of large images without down-sampling, we implemented a pipeline where we cropped the image into four partially overlapping parts, which were evaluated individually by the model. For the overlapping parts, we split the SVM outputs along the middle of overlaps, which reduced undesirable edge effects.

VI. Results

Table I gives the AUC and Dice scores as a function of the number of images included in the training set, K. The subsequent results are all for $K = 5$.

TABLE I
Performance as a function of the number of labeled training images K.

K	AUC (%)	Mean Dice	Global Dice
1	94.77	0.643	0.727
2	97.28	0.708	0.809
3	97.47	0.704	0.805
4	98.05	0.751	0.871
5	98.27	0.755	0.877

We used a decision threshold of $\Sigma(\text{image}) > 4000$, which yielded a false positive rate of 12/1500 = 0.8% and a false negative rate of 150/1198 = 12.5%.

Fig. 4 shows how the true positive rate and average Dice scores for positive images vary with the size of the water mask. For image groups where water covers more than 0.3% of the image area, the true positive rate exceeds 0.8 and the mean Dice score exceeds 0.6.

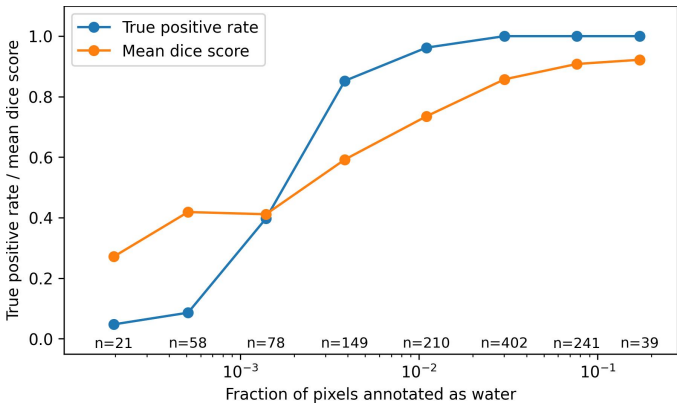

Fig. 4. True positive rate and average Dice scores for positive images, as a function of the water mask size.

Fig. 5 shows histograms of Dice scores for the image groups above (blue) and below (orange) the 4000 threshold. Positive images that fell below the 4000 limit had lower Dice scores (average 0.33), while those above the threshold had higher scores (average 0.82).

A. Qualitative error analysis

To complement the quantitative results, we show representative examples of failure modes. We overlay the annotation mask in red and the model's output mask in blue (purple where they overlap).

Fig. 6 shows three examples with sub-optimal Dice scores, with the original images to the left and the overlay visualization to the right.

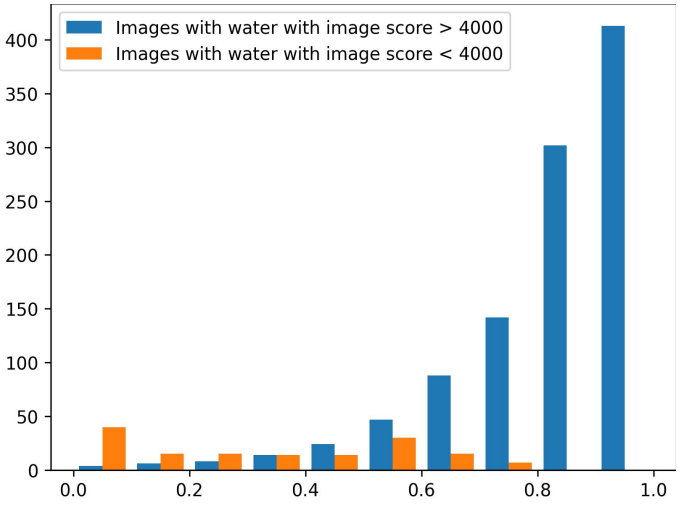

Fig. 5. Frequencies of Dice scores for positive images, below and above the classification threshold. The X-axis is grouped into Dice score bins of width 0.1, while the Y-axis gives the corresponding numbers of images with score below 4000 (orange) and above 4000 (blue).

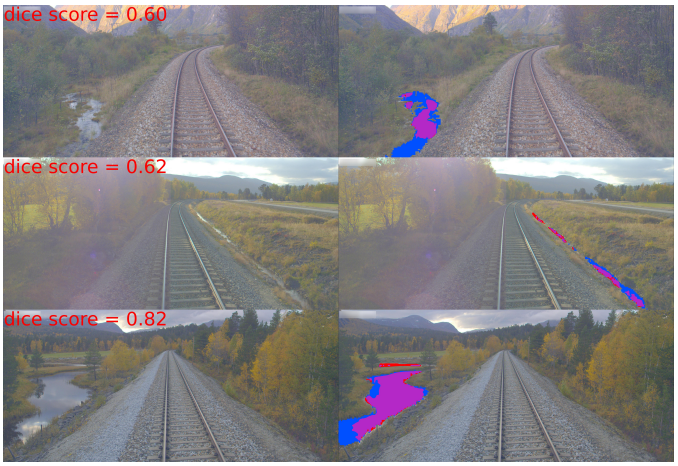

Fig. 6. Examples with sub-optimal Dice scores. Original images to the left, and to the right the annotation masks (red) and the model's output masks (blue), with overlap shown in purple.

In the top image, the model may arguably have identified parts of the water that was overlooked during annotation, resulting in a Dice score of 0.60. In the second one, the model has marked only the closer parts of water in a ditch, for a score of 0.62. In the third image, parts of the water that showed reflections of trees were identified by the model and not the annotator, for a Dice score of 0.82.

We next consider classification errors at the selected decision threshold. The 150 positive images that were misclassified were mainly images with small or narrow water areas, water partially hidden by vegetation like grass or trees, water far away from the train, and water areas with very shallow water like water on asphalt or wet soil. Fig. 7 shows three examples (same visualization as in Fig. 6). Note that in these false-negative cases, the predicted water regions are typically small, so only a few blue/purple pixels are visible.

Fig. 7. Examples of positive images that were misclassified (false negatives). Original images to the left, and to the right the annotation masks (red) and the model's output masks (blue), with overlap shown in purple.

In the top image in Fig. 7, the annotated water appears to be wet mud. In the second, the annotated water is a thin line of a river in the distance. In the third, the annotated water consists of fragments seen through vegetation.

For the 12 images without water that were classified as positive, we observed that in 5 cases there appeared to be water present, while in 3 cases it was difficult to determine whether water was present. In the remaining 4 images, water was predicted on a masonry wall at the tunnel entrance, on a trackside masonry strip, on a portion of a road surface, and in a saturated area.

VII. Discussion

Overall, the results show strong performance with very limited supervision. The largest gain is obtained when increasing the number of labeled images from $K = 1$ to $K = 2$, while subsequent improvements are smaller (Table I). This suggests diminishing returns once the few-shot set covers the main visual modes of water in the corridor imagery, for example when additional synthetic puddles are visually similar. Performance was high even though the training images differed in aspect ratio and resolution from the test set.

In a practical monitoring setting, most frames will be negative, and a given water body will typically be observed across multiple frames and viewing conditions. This motivates selecting a conservative decision threshold with low false alarm rate. With $\Sigma(\text{image}) > 4000$, we obtain a false positive rate of 0.8% at the expense of a higher false negative rate (12.5%), indicating that the threshold is primarily tuned to avoid frequent alarms.

The error patterns in the false negatives are consistent with a small-signal regime: narrow or distant water regions and water partially obscured by vegetation, as well as very shallow water that is hard to separate from wet ground or asphalt. These cases also align with the mask-size analysis in Fig. 4, where both true positive rate and Dice score improve with increasing annotated water area. The overlap seen in the Dice

distributions above and below the threshold (Fig. 5) further indicates that the decision threshold separates clear water cases from more marginal detections.

Finally, the Dice score is a strict criterion that is reduced for each pixel that is incorrectly classified. Since the qualitative examples suggest that part of the apparent error can be due to annotation ambiguity, a perfect Dice score should not be expected.

VIII. Future work

The planned application of our model is to run a real-time or close to real-time analysis of forward-looking railway videos and perform a running comparison to previous analyses of the same track. Video analyses will allow temporal aggregation across consecutive frames that is likely to reduce the number of missed small water regions. The current model is applied to full frames and may therefore respond to water anywhere in the field of view. In deployment, we will restrict inference to a region of interest around the track and trackside drainage, reducing spurious alarms from distant lakes or rivers.

The main goal will be to create warnings in areas where the build-up of water bodies is likely to create problems in the near future, while in extreme cases it may also trigger alarms that can stop the train immediately.

We also plan to extend the model to detect developing ice patches and icicles along the railway tracks.

IX. Conclusion

The proposed few-shot setup uses DINOv3 as a frozen patch-level feature extractor and a lightweight SVM-based patch classifier trained from only five segmented images. On the held-out test set, we obtained an image-level AUC of 98.27%, a mean Dice score of 0.755, and a global Dice score of 0.877. These results indicate that pretrained patch descriptors can support practical few-shot water segmentation from onboard railway imagery despite ambiguity in the ground truth annotations. This can enable scalable monitoring of track-adjacent water from onboard video, supporting maintenance planning and operational decision-making during heavy rainfall events.

X. Acknowledgement

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Europe's Rail Joint Undertaking. Neither the European Union nor the granting authority can be held responsible for them. The project R2DATO is supported by the Europe's Rail Joint Undertaking and its members.

References

[1] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski, "DINOv3," arXiv:2508.10104, 2025. [Online]. Available: https://arxiv.org/abs/2508.10104

[2] A. Kumar and S. P. Harsha, "A systematic literature review of defect detection in railways using machine vision-based inspection methods," *International Journal of Transportation Science and Technology*, vol. 18, pp. 207–226, 2025.

[3] W. Helmi, R. Bridgelall, and T. Askarzadeh, "Remote sensing and machine learning for safer railways: A review," *Applied Sciences*, vol. 14, no. 9, p. 3573, 2024.

[4] P. Aela, H.-L. Chi, A. Fares, T. Zayed, and M. Kim, "UAV-based studies in railway infrastructure monitoring," *Automation in Construction*, vol. 167, p. 105714, 2024.

[5] Y. Wang, Y. Shen, B. Salahshour, M. Cetin, K. Iftekharuddin, N. Tahvildari, G. Huang, D. K. Harris, K. Ampofo, and J. L. Goodall, "Urban flood extent segmentation and evaluation from real-world surveillance camera images using deep convolutional neural network," *Environmental Modelling & Software*, vol. 173, p. 105939, 2024.

[6] X. Han, C. Nguyen, S. You, and J. Lu, "Single image water hazard detection using FCN with reflection attention units," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 105–121.

[7] M. Rahnemoonfar, T. Chowdhury, A. Sarkar, D. Varshney, M. Yari, and R. Murphy, "FloodNet: A high resolution aerial imagery dataset for post flood scene understanding," *IEEE Access*, vol. 9, pp. 89 644–89 654, 2021.

[8] O. Lee and H. Joo, "AlleyFloodNet: A ground-level image dataset for rapid flood detection in economically and flood-vulnerable areas," *Electronics*, vol. 14, no. 10, p. 2082, 2025.

[9] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9650–9660.

[10] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "DINOv2: Learning robust visual features without supervision," arXiv:2304.07193, 2023. [Online]. Available: https://arxiv.org/abs/2304.07193

[11] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 4015–4026.

[12] N. Catalano and M. Matteucci, "Few-shot semantic segmentation: A review of methodologies, benchmarks, and open challenges," arXiv:2304.05832, 2023. [Online]. Available: https://arxiv.org/abs/2304.05832

[13] Z. Zhou, H.-M. Xu, Y. Shu, and L. Liu, "Unlocking the potential of pre-trained vision transformers for few-shot semantic segmentation through relationship descriptors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 3817–3827, [Online]. Available: https://openaccess.thecvf.com/content/CVPR2024/html/Zhou_Unlocking_the_Potential_of_Pre-trained_Vision_Transformers_for_Few-Shot_Semantic_CVPR_2024_paper.html.