Supplementary Material:
Active Data Collection and Management for Real-World Continual Learning via Pretrained Oracle

Vivek Chavan*[1] Paul Koch[1] Marian Schlüter[1] Clemens Briese[1] Jörg Krüger[1,2]

[1] Fraunhofer IPK [2] Technical University of Berlin, Germany

1. Setup Details for the Experiments

Table 1 gives the details of the setup for the experiments. We use the same setup for all our ML trainings for a fair and unbiased comparison. Please note that we use 20 exemplars per class for all tasks when comparing ADCM with Herding [13] and RMM [11] to enable a clearer comparison between the approaches.

Parameter	Value
Train-Val Split	80/20
Optimizer	SGD
lr start	0.1
lr end	0.0001
weight decay	0.0005
Batch Size	64
Transforms: Train	Resize: (224, 224), RandomHorizontalFlip
Transforms: Val	Resize: (256, 256), CenterCrop
System Memory	48GB
CPU Cores	12
GPU Count	1
GPU type	NVIDIA RTX A6000
Python version	3.8.13

Table 1. Hyperparameter and workstation details for the ML Experiments

2. Traditional Continual Learning Framework

We start with the classical CIL implementation, where an untrained ResNet18 model is trained incrementally with a limited rehearsal memory budget. At the end of each IL task, the model is used as an encoder for selecting exemplars from old data. We use the InVar-100 dataset for this investigation and train RN18 over 12 tasks using POD-AANet implementation [10] and use RMM [11] for memory management.

Our analysis shows that in continual learning scenarios, the learnt feature representations are distorted as new tasks are introduced. The problem is exacerbated for fine-grained objects and data with clutter, where the background features dominate the embeddings as the data distribution gets updated. We show three pairs of images from the data and see how the distance between the images within each pair *diminishes* as the model is incrementally trained. Each pair has a similar background. A visualisation of the feature encodings is also shown. Figure 1 shows Class Activation Maps (CAMs, as proposed by [12] and [18]) of test images from other classes in the dataset, which are misclassified during the newer tasks. We break this classical implementation into two separate problems: incremental training (plasticity-rigidity dilemma) and feature encoding (for exemplars).

3. Feature Representation Distortion during Incremental Learning

Figure 2 compares the feature embeddings from classes introduced during Task 0 and compares them against Task 12 (embeddings downsampled using Principal Component Analysis (PCA)). We use the POD-AANet implementation.

4. Feature Distribution Comparison for Pretraining Methods

As stated in the paper, we analyse the features learnt by different state-of-the-art SSL approaches including MoCoV3 [5], SwAV [3], Barlow Twins [17], DINO [4], VICReg [1] and VICRegL [2] on the datasets. Figure 3 gives the downscaled intra-class PCA distribution obtained from one of the classes in the InVar-100 dataset. We notice sensitivity to the object orientation in DINO and VICRegL embeddings. Please note that Bardes et al. [2] pretrain ConvNeXt-XL on ImageNet-22K. We have included it here

Code: https://github.com/Vivek9Chavan/ADCM
*Correspondence: vivek.chavan@ipk.fraunhofer.de

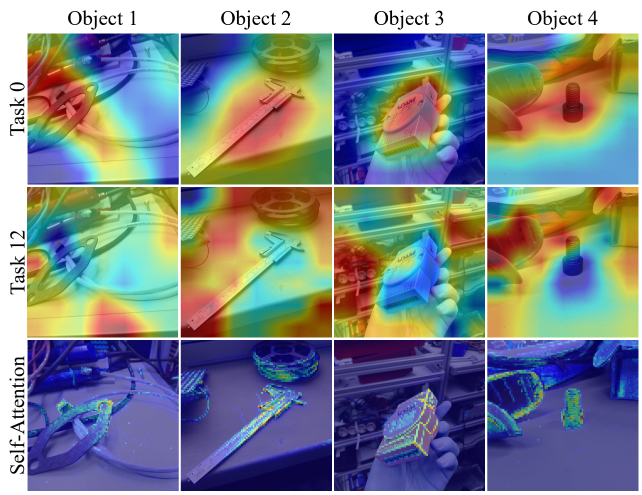


Figure 1. **Top, Middle:** CAMs [18] for test images for an incrementally trained model with POD-AANet on InVar-100, exhibiting the consequences of catastrophic forgetting. The images were correctly classified during Task 0 but were misclassified during Task 12. **Bottom:** For contrast, corresponding attention maps (taken from first Head) from *frozen* DeiT-S trained using DINO [4].

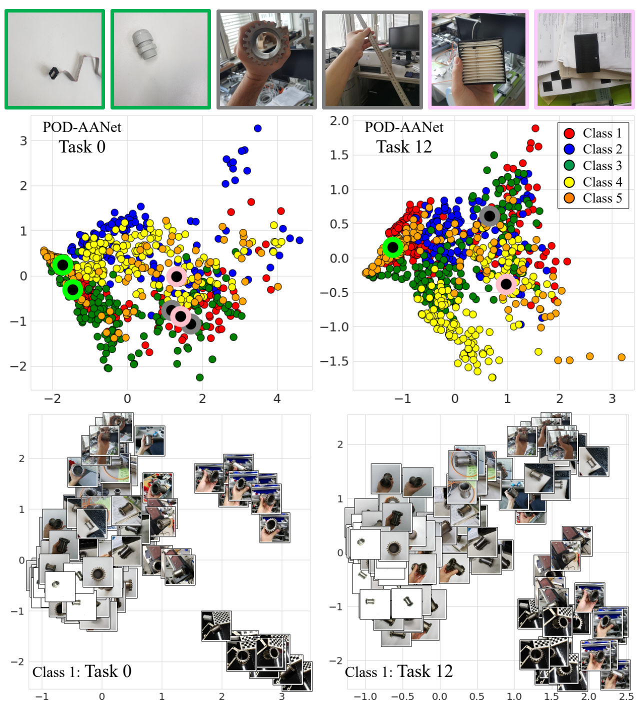


Figure 2. **Top:** Sample image pairs from InVar-100. **Middle:** PCA distribution from 5 classes introduced during Task 0 (left) and Task 12 (right). The feature representations get distorted as the model is retrained. **Bottom:** Visualisation of the corresponding intra-class distribution from one of the classes.

for comparison. However, it is not a part of our study, due to our defined scope (§3 in the paper).

Figure 4 shows the distribution of embeddings from different views of the MVIP dataset, extracted from uncropped images. Each colour represents a different camera view. As stated in the paper, we observe that the cameras with similar view perspectives are placed closer together. Additionally, a more uniform cluster represents homogeneity in data, i.e. the object rotations do not add significant additional *information* about the object features. On the other hand, loosely collated clusters denote camera views where the features of the object are better captured.

Figure 5 supplements Figure 5 and §3.3 in the paper and shows the feature distribution for the DIMO dataset, extracted using DeiT-S + DINO, ResNet50 + DINO and Supervised ResNt50. We observe that the pertaining method has a greater influence on the distribution compared to the model architecture. The distribution obtained from DeiT-S and ResNet50 (pretrained using DINO) are similar, however, the embeddings from DeiT-S are more sensitive to object orientation, shape and lighting.

5. ADCM: Active Data Collection and Management

This section expands on the applications of our implementation for analysing real-world and industrial data.

5.1. ADCM$_0$

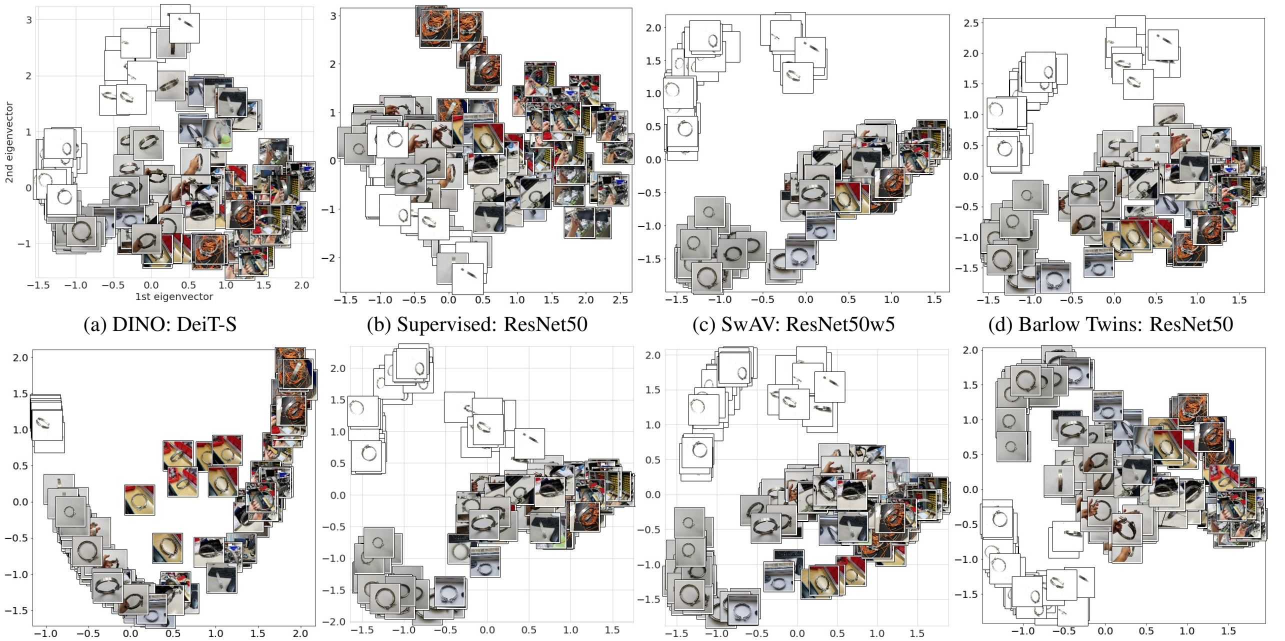
The memory policy and pruning policy of ADCM$_0$ are implemented as given in Algorithm 1. We take DeiT-S pretrained using DINO as the encoder.

We emphasize multiview photo or video based digitisation in our work since such a stationary setup can capture the features of the objects better and enable downstream applications w.r.t. ML [6, 7, 15]. ADCM proves useful for data pruning and management for such applications. Miscellaneous challenges for multiview part identification are out of the scope of this paper; we refer to [9] for more details.

5.2. Data Pruning and Analysis

Figure 7 shows additional examples of outlier and redundant data identification from the MVIP and InVar-100 datasets. The outlier image for MVIP is incorrectly segmented, which was correctly flagged. Figure 8 supplements Figure 11 in the paper (also Figure 4), showing images from the two camera views. Camera 1 is positioned such that a change in object orientation adds more information, whereas, Camera 9 does not.

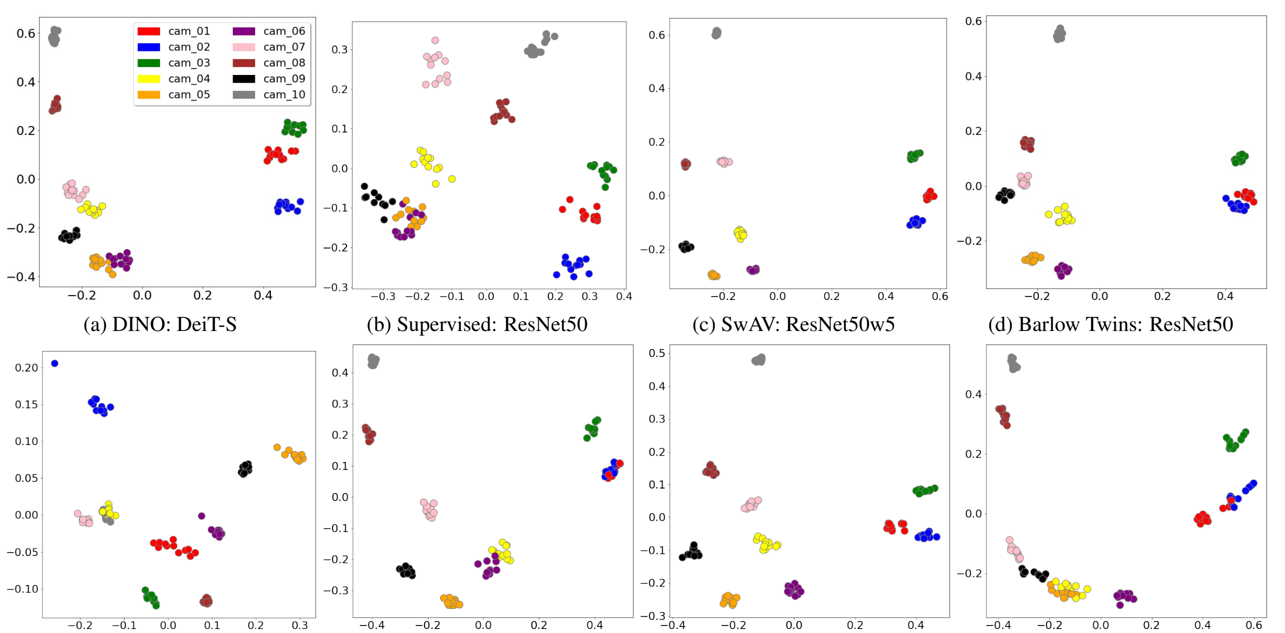
As mentioned in the paper, an alternative approach to identifying outliers is via intra-class clusters. To define statistical outliers based on the Z-score, we use k-means clustering to identify intra-class clusters and cluster centroids



(a) DINO: DeiT-S (b) Supervised: ResNet50 (c) SwAV: ResNet50w5 (d) Barlow Twins: ResNet50

(e) MoCo v3: ViT-B (f) VICReg: ResNet200x2 (g) VICRegL: ResNet50 (h) VICRegL: ConvNeXt-XL

Figure 3. Visualisation of intra-class feature distribution obtained from the eight different approaches. We notice that the images with approximately similar backgrounds are placed closer together for most approaches. However, the embeddings for DINO and VICRegL are also grouped based on object orientation and shape.



(a) DINO: DeiT-S (b) Supervised: ResNet50 (c) SwAV: ResNet50w5 (d) Barlow Twins: ResNet50

(e) MoCo v3: ViT-B (f) VICReg: ResNet200x2 (g) VICRegL: ResNet50 (h) VICRegL: ConvNeXt-XL

Figure 4. MVIP Dataset clustering based on camera views, obtained from the eight different approaches. We notice a similar pattern to Figure 3, in that the embeddings from DINO and VICRegL are more contextually sound.

(μ). The number of clusters may either be dictated by the silhouette score [14] or by M_p. For instance, we fine-tune the outlier policy on the DIMO data to flag image embeddings that lie over $\zeta = 4.5$ standard deviations (σ) from each centroid within the class. This parameter is not learnt, but instead is controlled by human supervision based on ini-

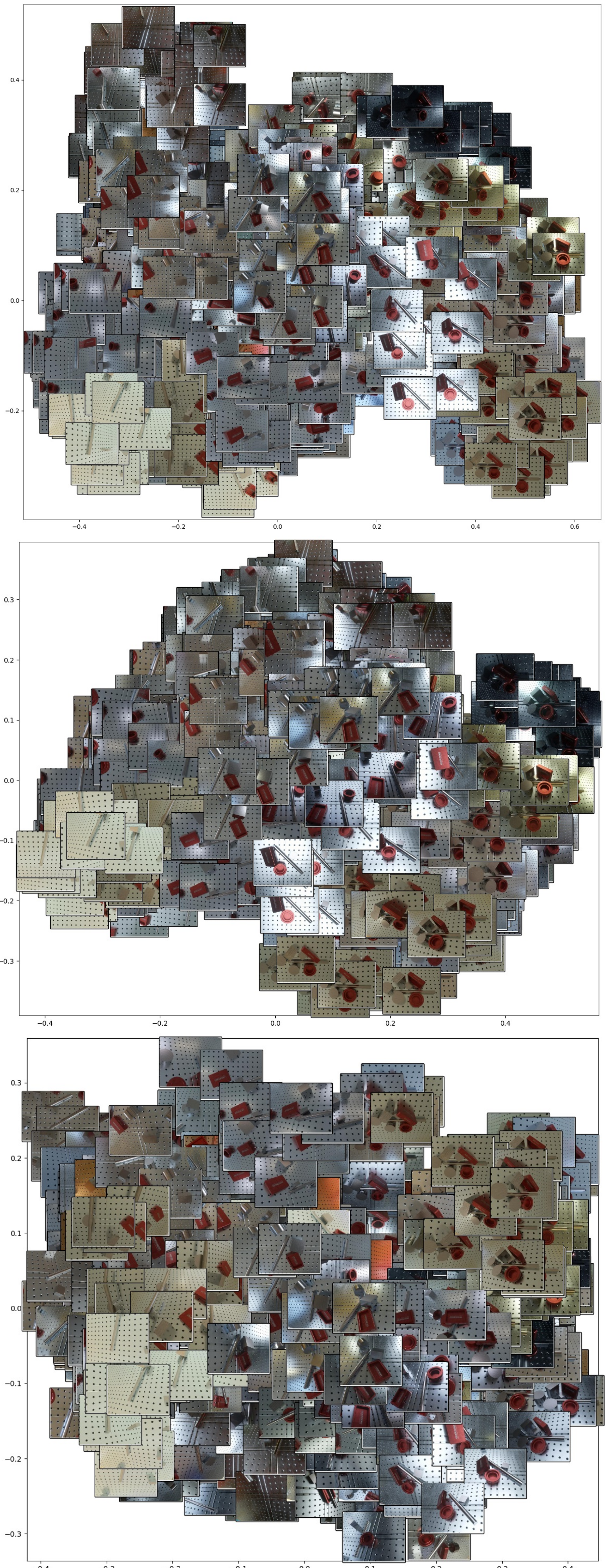


Figure 5. An analysis of the feature distribution from the embeddings offered from DeiT-S + DINO (**Top**), ResNet50 + DINO (**Middle**), and Supervised ResNet50 (**Bottom**).

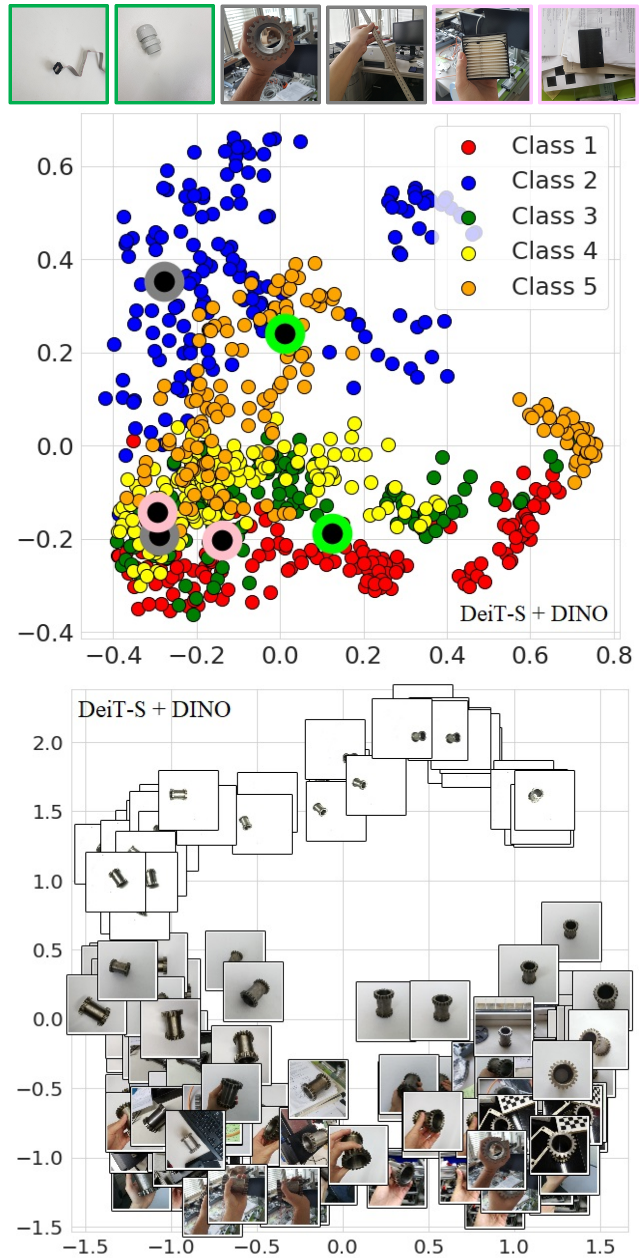


Figure 6. **Top:** A supplement to Figure 2. Sample image pairs from InVar-100. **Middle:** PCA distribution from 5 classes introduced during Task 0, obtained from *frozen* DeiT + DINO encoding. **Bottom:** Visualisation of the corresponding intra-class distribution from one of the classes.

tial fine-tuning on a few classes. For instance, the default value is 3, based on which the user can visualise the selected outliers and the resulting pruned dataset. This was estimated to remove some *good* data points, after which ζ was changed to 6- which was too conservative. Finally, the value of 4.5 was selected as the optimal point. We found this

Algorithm 1: Implementation of ADCM$_0$ for practical continual learning scenarios

```
Input : Full dataset:  D_cloud, Memory Policy:  M_p
        Pruning policy:  D_p, Threshold & other
        values
Output: Pruned Dataset:  D_i, Continual Learning
        Output
import Pretrained Encoder (DeiT-S)
Task = T_0 //Initial Joint Learning
//Analyse and Prune Data
load image data
for i ← 0 to i_0 do
    //i_0 initial classes
    for j ← 0 to n do
        //n images for class i
        θ = Feature Vector [1, dim encoder]
         //process image via Encoder
    end for
    Class Feature Vectors = [n, dim encoder]
    Feature distribution analysis
        θ_a = Σ_0^n θ_j
        k_i = Weighted class exemplar count
         //proportional to θ_a //D_p
    Downsample:
        Principal Component analysis
        Modified Feature Vectors = [n, 32]
    Prune Data:
    K-means Clustering
        Sampled Feature Vectors = [k_i, 32]
        D ←Sampled Feature Vectors//D_p
end for
    ML Training:  Model M_0
    Deploy Model M_0
for Task ← 1 to T do
    for i ← 0 to i_0 do
        //Old classes
        Re-sample dataset for old classes
        Exemplars List ←Sampled Feature
         Vectors//M_p
    end for
    for i ←i_0 to i_Task do
        //New classes
        Analyse and prune new class data
        D ←Sampled Feature Vectors//D_p
    end for
    D = Σ_0^{i_0} Exemplars List+ Σ_{i_0}^{i_Task} D
    //Updated D_p and M_p
end for
    Incremental ML Training:  Model M_Task //CIL or
     or Domain-IL or Online Learning
    Deploy Model M_task
if Accuracy_model < Threshold then
 | Re-initiate Joint Training //Reset D_p and M_p
Repeat
```

approach to generalise well to the complete dataset in different scenarios, as long as substantial data is available (at least 40 images per class). Other approaches such as Herding [13] or K-NN search [8] may also be used, depending on the application. Depending on the scope of the project and the image data, outliers may be unsuitable for training or may represent a different image context that is not accounted for by the clusters. Hence, human supervision is necessary.

$$|\boldsymbol{\theta}_{out} - \bar{\boldsymbol{\mu}}| > \zeta \cdot |\boldsymbol{\sigma}| \tag{1}$$

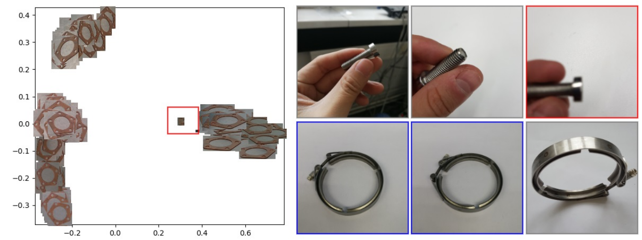


Figure 7. Examples from MVIP dataset (left) and InVar-100 dataset (right) with identified outliers and redundant image pair.

(a) Cam 01

(b) Cam 09

Figure 8. A supplement to Figure 11 from the paper. **Left:** Images from the camera view cluster with high variance. **Right:** Images from the camera view cluster with low variance.

5.3. Exemplar Selection

We use the ADCM implementation for coreset selection prior to ML training and for selecting exemplars at the end of each incremental task. As a supplement to Figure 10 and Equations (3) and (4) in the paper, Algorithm 2 elaborates the approach for data sampling based on the feature imbalance. We use DeiT-S as the pretrained encoder.

5.4. Applicability to Large Datasets

We demonstrate the general applicability of our approach to large industrial datasets with objects captured in different contexts and use cases. We explore data pruning and coreset selection with and without weak supervision and observe superior results compared to the baseline. With MVIP, we also explore the visual inspection and analysis of the data based on meta-labels and descriptors. Our approach is particularly useful for industrial and stationary-setup applications, where the objects are often digitised using a fixed setup and the variance in intra-class distribution may largely result from changing perspectives and object orientation[9]; with background clutter and occlusion as additional factors. The approach scales to larger datasets. Our approach outperforms Herding by a statistically significant margin in all tested use cases. ADCM outperforms Herding by 1.4% on the DER implementation and by 4.1% on PODNet. It outperforms RMM by 0.8% w.r.t. performance on old classes. The gain in accuracy is a result of the exemplars being more representative of the underlying class distribution. As discussed in the paper, the frozen feature encoder provides more accurate feature embeddings throughout the incremental tasks.

5.5. Data Quality

Our analysis shows there is a substantial overlap between CAM and self-attention map regions for *good data*. Additionally, data that is misclassified is passed through the pretrained encoder and resampled to be included in the memory storage M_i. This approach maintains the most representative as well as the most *challenging data* points that are relevant to continual learning. Weak supervision is optional. For *good and clean data*, there is an overlap between the CAMs and the corresponding self-attention maps from SSL-pretrained ViT, which may be necessitated by thresholding the overlap region. The normalised overlap region can be calculated as follows.

$$\text{Overlap} = \frac{|A \cap B|}{\min(|A|, |B|)} \tag{2}$$

A threshold overlap value can be set for additional control. However, this threshold needs to be carefully selected and tested on different subsets of the data. Alternatively, it can be parameterised and learned. This approach hasn't been thoroughly tested and is included here as a *rough concept*.

6. Weak Supervision

In the context of the applications presented in our work, weak supervision plays a key role, in that it puts the human operators in control of the data management. One of the challenges in developing this solution was to address the repetitive and cumbersome aspects of the process of data acquisition without replacing the human experts. Using human supervision, the data pruning and management operate much faster. For instance, the pruning policy (D_p) is learnt by finetuning the ζ parameter on the given set of data and getting user input on whether a given image is a *good* or a *poor* data point.

Algorithm 2: Variable coreset/exemplar selection for Continual Learning with ADCM

```
Input : Old Class data, j classes, n images per
        class
Output: Exemplars, k < n sampled images (varies
        according to class feature distribution)
Exemplar List = [ ]
import Pretrained Encoder (DeiT-S)
for i ← 0 to j do
    for j ← 0 to n do
        θ = Feature Vector [1, dim encoder]
         //process image via Encoder
    end for
    Class Feature Vectors = [n, dim encoder]
    Feature distribution analysis
        θa = Average feature variance for all
         vectors within the class
        n = Weighted class exemplar count
         //proportional to θa
    Downsample:
        Principal Component analysis
        Modified Feature Vectors = [k, 32]
    Prune Data:
    K-means Clustering
        Sampled Feature Vectors = [k, 32]
        Exemplar List ←Sampled Feature Vectors
end for
return Exemplar List
Repeat for the next incremental Task
```

7. Downscaling of Feature Embeddings

Caron et al. [4] downscaled ImageNet features using PCA (384 to 30) and t-SNE (30 to 2) [16] to represent class means in 2D and their interrelations. We find that PCA is sufficient for our application. Moreover, since PCA aggregates the global features of the data, it is better able to retain the unique features of the target objects. Figure 11 shows the downscaled feature distributions using PCA (left) and t-SNE (right) from our previous experiments.

References

[1] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *CoRR*, abs/2105.04906, 2021. 1

[2] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicregl: Self-supervised learning of local visual features. In *NeurIPS*, 2022. 1

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. 2020. 1

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 6, 7

[5] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9620–9629, Los Alamitos, CA, USA, 2021. IEEE Computer Society. 1

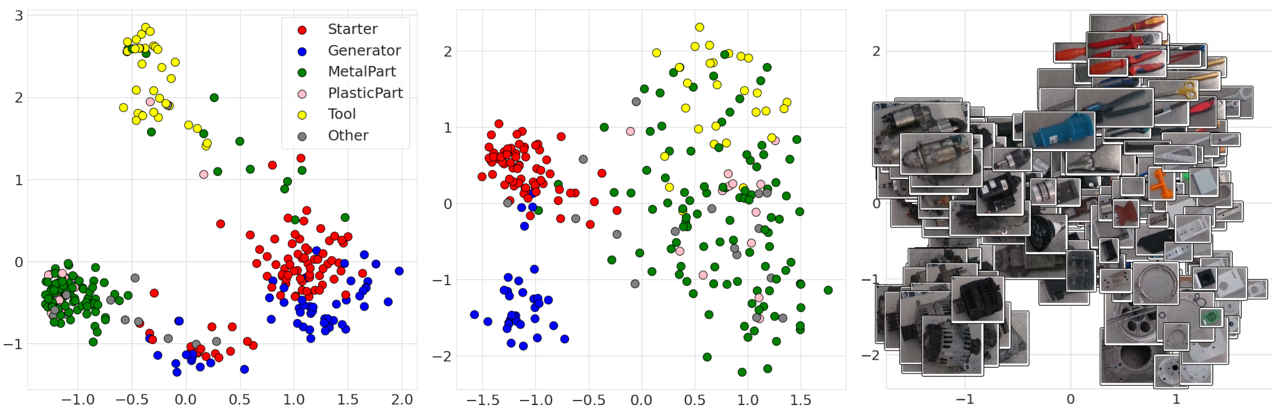


Figure 9. Example of ADCM$_0$ implementation used for MVIP dataset superclass analysis. Keeping the camera view fixed allows us to use the approach for comparing different object classes. **Left:** Clustering using the complete image embeddings. **Middle:** Clustering using ROI crop embeddings. **Right:** Embedding visualisation for the ROI crops.

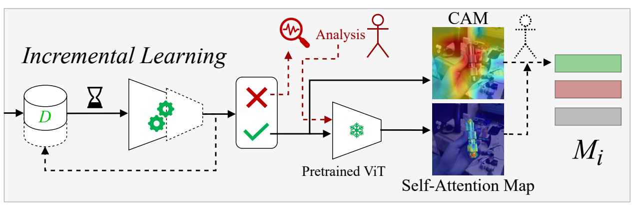


Figure 10. An alternative approach to classify and sort image data during long project timelines. The memory budget M_i comprises of an ensemble of *good data*, *challenging data*, and *sampled exemplars* based on part identification accuracy, clustering and analysis via the pretrained encoder.

[6] Abdullah Hamdi, Silvio Giancola, Bing Li, Ali K. Thabet, and Bernard Ghanem. MVTN: multi-view transformation network for 3d shape recognition. *CoRR*, abs/2011.13244, 2020. 2

[7] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15524–15533, 2022. 2

[8] Ahmet Iscen, Thomas Bird, Mathilde Caron, Alireza Fathi, and Cordelia Schmid. A memory transformer network for incremental learning, 2022. 5

[9] Paul Koch, Marian Schlüter, Clemens Briese, and Vivek Chavan. Mvip: A dataset for industrial part recognition, 2023. Fraunhofer Fordatis. 2, 6

[10] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2544–2553, 2020. 1

[11] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. In *Advances in Neural Information Processing Systems*, pages 3478–3490. Curran Associates, Inc., 2021. 1

[12] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic.

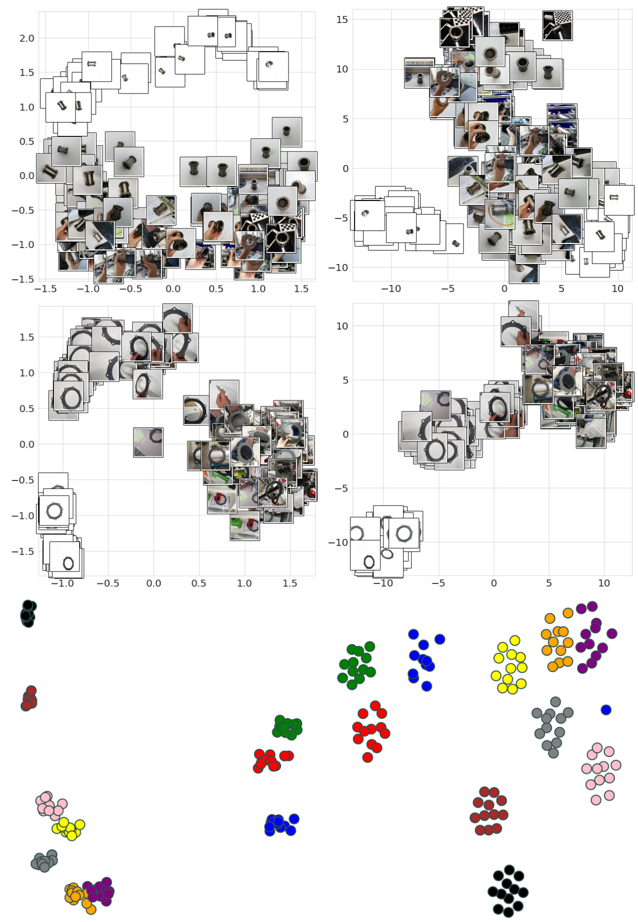


Figure 11. Feature vectors from DeiT + DINO [4] downsampled using PCA (left) and t-SNE (right) [16]. **Top, Middle:** Objects from the InVar-100 dataset. **Bottom:** Multi-view clustering for the MVIP dataset. Each colour represents a separate camera view

Is object localization for free? - weakly-supervised learning with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1

[13] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 1, 5

[14] Ketan Rajshekhar Shahapure and Charles Nicholas. Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, 2020. 3

[15] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proc. ICCV*, 2015. 2

[16] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 6, 7

[17] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, 2021. 1

[18] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 1, 2