RICO: Two Realistic Benchmarks and an In-Depth Analysis for Incremental Learning in Object Detection

Supplementary Material

A. Using the RICO Benchmark

Domain RICO (D-RICO) and Extending-Classes RICO (EC-RICO) are developed as a platform to advance research in Incremental Learning (IL) within more realistic settings. We propose this benchmark to serve dual purposes: (1) as a challenging evaluation framework for future IL methods and (2) as a resource for deriving novel insights into IL mechanisms. We have made the benchmark set up and evaluation framework openly accessible to facilitate broader research engagement. This section provides an overview of the benchmark utilization process, with additional technical specifications available in the GitHub repository. Details on using our training and evaluation framework are given in Section E.

Since we do not hold rights to the constituent datasets, we cannot distribute the complete benchmark as a unified package. Utilizing the benchmark therefore requires the following steps:

1. Downloading the original datasets from their respective sources;
2. Processing the images and annotations;
3. Combining the processed images and annotations into D-RICO and EC-RICO.

The following subsections elaborate on each of these steps.

1. Downloading the Datasets The initial step requires acquiring all constituent datasets from their original sources.

- nuImages [7]
- BDD100K [98]
- WoodScape [97]
- FishEye8K [32]
- DENSE [4]
- Sim10k [38]
- TIMo [74]
- SHIFT [83]
- Teledyne FLIR [26]
- LOAF [96]
- SMOD [13]
- VisDrone [111]
- Synscapes [92]
- DSEC [28, 29]

Due to potential storage location changes, we do not provide direct download links. However, all information can be accessed through the cited papers, respective dataset homepages, and associated GitHub repositories. Some datasets require access requests, which are typically granted for research purposes. While the complete raw data across all datasets comprises several terabytes, the benchmark-specific subset is substantially smaller.

2. Processing the images and annotations Section B provides detailed descriptions of all datasets and their required processing protocols. To get the raw annotations and images into the correct format required for D-RICO and EC-RICO, we provide a script for each dataset. For some datasets, it is enough to just merge the classes in the RICO classes; for other datasets, extensive calculations are required.

3. Combining the processed images and annotations into D-RICO and EC-RICO We provide a template file for the final D-RICO and EC-RICO annotation file. These files miss the actual annotations but hold all other relevant information like file name and image size. The missing annotations can either be filled in with a custom script or be reproduced using one of the provided scripts.

The D-RICO and EC-RICO benchmark can then be used. We provide our training and evaluation framework that is based on Detectron2 [93]. See Section E for further details.

B. More Details on the D-RICO and EC-RICO

Section 3 introduces the Domain-RICO and Expanding-Classes RICO benchmarks. In this section, we present additional statistics and comparisons regarding the tasks, as well as a comprehensive explanation of the construction of each task and the data processing and preparation.

B.1. Additional Statistics

B.1.1. Detailed Class Distribution

Tables S.1 and S.2 show the relative values and total label counts for each task, with visual distributions in Figure 2 e) and j). Intra-task differences vary significantly, and annotations are unevenly distributed. For D-RICO, *thermal fisheye indoor* holds less than 1% of labels, while *photorealistic simulation* and *drone* each contribute 22%. In EC-RICO, *fisheye car* has 3%, whereas *drone* accounts for 33%. This variation enhances task diversity, as class distributions and object counts differ across tasks.

B.1.2. t-SNE and Nearest Mean Classifier of Image Features

We project extracted image features into a two-dimensional space using t-SNE to analyze the differences between tasks in image space. The features are obtained from a model trained following the setup in Section 4.1.

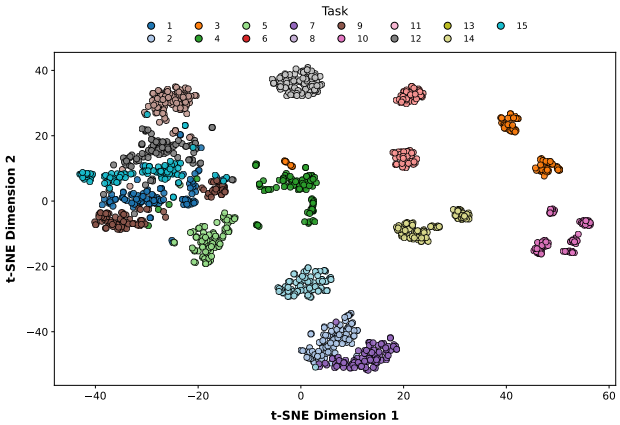
Figure S.1 presents the t-SNE visualization. The image features are well-separated for EC-RICO, whereas in D-

Table S.1. Annotation counts per task and class as percentages, including total dataset contribution. A dash (-) indicates that there are no available annotations for that class in the respective dataset.

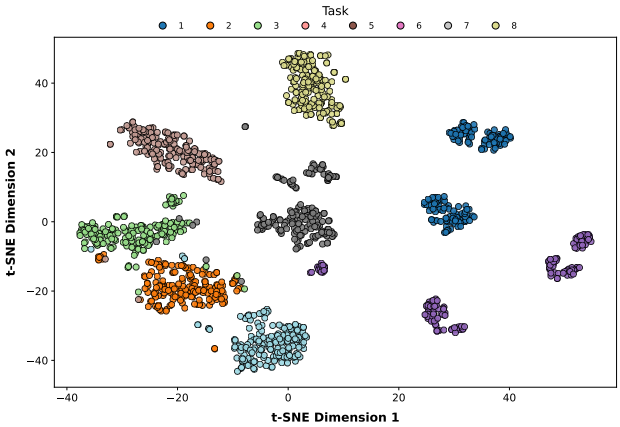
Task	person	vehicle	bicycle	Total	Total in %
daytime	37.8	57.86	4.34	28255	3.24
thermal	35.14	59.76	5.1	58337	6.70
fisheye fix	8.58	91.42	-	99688	11.45
drone	29.91	67.66	2.42	187425	21.52
simulation	42.48	54.55	2.97	42698	4.90
fisheye car	25.15	74.85	-	54298	6.24
RGB + thermal fusion	26.14	20.12	53.73	20686	2.38
video game	-	100.0	-	27038	3.10
nighttime	6.52	92.91	0.58	48139	5.53
fisheye indoor	100.0	-	-	21752	2.50
gated	39.56	60.44	-	35050	4.02
photoreal. simulation	62.5	37.5	-	195263	22.42
thermal fisheye indoor	100.0	-	-	5747	0.66
inclement	29.34	70.66	-	23022	2.64
event camera	20.2	77.33	2.48	23418	2.69

Table S.2. Annotation counts per task and class as percentages, including total dataset contribution. A dash (-) indicates that there are no available annotations for that class in the respective dataset.

Task	person	car	bicycle	motor-cycle	truck	bus	traffic light	traffic sign	Total	Total in %
fisheye car	100.00	-	-	-	-	-	-	-	18777	3.20
gated	41.95	58.05	-	-	-	-	-	-	35963	6.13
daytime	39.62	40.74	19.64	-	-	-	-	-	27461	4.68
fisheye fix	8.91	33.11	-	57.97	-	-	-	-	102594	17.48
simulation	43.95	39.82	3.1	4.31	8.82	-	-	-	44100	7.51
drone	27.65	53.93	2.47	9.47	4.01	2.47	-	-	196355	33.45
thermal	33.15	39.41	4.52	0.87	0.54	1.24	20.28	-	79521	13.55
nighttime	4.31	56.22	0.28	0.15	1.18	0.47	17.82	19.57	82285	14.02



(a) D-RICO



(b) EC-RICO

Figure S.1. t-SNE features

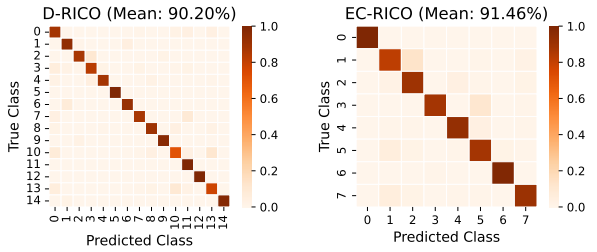


Figure S.2. Confusion Matrix of the Nearest Mean Classifier based on image features.

RICO, some tasks are closer while remaining distinguishable. This indicates that tasks differ in terms of image representations.

Furthermore, a simple nearest mean classifier (as used in LDB [81]) can already achieve good task classification based on these features. Figure S.2 shows the corresponding confusion matrix, reinforcing the distinctiveness of tasks in feature space.

B.1.3. Image sizes

The input to the network is always $1536 \times 1536 \times 3$, while the original dataset images vary in size, as shown in Table S.3. Except for *fisheye fix* and *drone*, all tasks maintain a constant intra-task resolution. The *thermal*, *gated*, and *thermal fisheye indoor* tasks contain grayscale images, whereas all others are RGB.

Since none of the original image sizes match 1536×1536, all images are padded, resized, or cropped accordingly. Grayscale images are duplicated across channels to match the required three-channel format. This highlights the benchmark's diversity, in contrast to others that typically maintain uniform image resolutions.

B.1.4. Labeling Policy

Each dataset follows a specific labeling policy that defines how objects in an image are labeled. In Section B.2 the labeling policies and how we processed them are stated in more detail. Additional details can be found in the respective publications.

Figures S.3 and S.4 illustrate randomly sampled objects for each class of D-RICO and EC-RICO. The visualizations

Table S.3. Image sizes for each task (mean ± standard deviation).

Task	Width × Height
daytime	$1600_{\pm 0} \times 900_{\pm 0} \times 3$
thermal	$640_{\pm 0} \times 512_{\pm 0} \times 1$
fisheye fix	$1404_{\pm 312} \times 1232_{\pm 310} \times 3$
drone	$1484_{\pm 229} \times 951_{\pm 208} \times 3$
simulation	$1280_{\pm 0} \times 800_{\pm 0} \times 3$
fisheye car	$1280_{\pm 0} \times 966_{\pm 0} \times 3$
RGB + thermal fusion	$640_{\pm 0} \times 512_{\pm 0} \times 3$
video game	$1914_{\pm 0} \times 1052_{\pm 0} \times 3$
nighttime	$1280_{\pm 0} \times 720_{\pm 0} \times 3$
fisheye indoor	$1024_{\pm 0} \times 1024_{\pm 0} \times 3$
gated	$1280_{\pm 0} \times 720_{\pm 0} \times 1$
photoreal. simulation	$1440_{\pm 0} \times 720_{\pm 0} \times 3$
thermal fisheye indoor	$512_{\pm 0} \times 512_{\pm 0} \times 1$
inclement	$1920_{\pm 0} \times 1024_{\pm 0} \times 3$
event camera	$640_{\pm 0} \times 480_{\pm 0} \times 3$

highlight substantial differences in labeling policies and annotation quality. The differences concern tight vs. loose bounding boxes, amodal vs. visible bounding boxes, background objects, small objects, groups of objects, and class definitions. In addition to these differences, there are also differences in terms of objects that are not being labeled.

B.2. Detailed Task and Dataset Description

B.2.1. Daytime (nuImages [7])

NuImages [7] is a large-scale dataset derived from nuScenes that contains 93,000 annotated images captured in Boston and Singapore under various weather conditions and times of day. The dataset encompasses diverse urban driving scenarios, emphasizing autonomous perception tasks such as object detection and scene understanding.

Dataset Processing. We chose daytime images exclusively from the Singapore One-North district to make the task more distinct. We exclude images with *vehicle.bus.bendy* instances due to inconsistent annotation protocols wherein bus segments are individually labeled, contrary to standard buses' unified labeling. Merging these segmented annotations poses difficulties, especially in multi-bus scenes. Additionally, we exclude images featuring the class *static_object.bicycle_rack* as it contradicts with the *vehicle.bicycle* class, as the bicycles in the rack are not labeled to the bicycle class. Examples are given in Figure S.5.

Train, Val, and Test Split. To prevent data leakage, we enforce scene-level integrity by ensuring all images from the same scene ID remain in a single partition. We employ a stochastic optimization approach to achieve the target distribution of 60% train, 10% validation, and 30% test. We repeatedly generate random scene-to-partition assignments and evaluate how closely they match the desired proportions. Among these, we select the configuration that minimizes deviation while preserving scene consistency. This

Figure S.3. Random example objects to visualize the D-RICO benchmark's different annotation policies and qualities.

Figure S.4. Random examples of objects to visualize the various annotation policies and qualities for the EC-RICO benchmark.

Figure S.5. Example images of the daytime task with labels for D-RICO.

Monte Carlo-based strategy ensures a balanced partitioning while maintaining contextual coherence within each split.

Labeling Policy We use this dataset's labeling policy as a reference for the other datasets and compare them to this one. Objects are labeled tightly, and their bounding boxes cover only the visible parts. Bicycles and motorcycles include the rider within the bounding box. Small visible objects in the background are labeled.

D-RICO Classes.

- **person:**
  - *human.pedestrian.adult*
  - *human.pedestrian.child*
  - *human.pedestrian.construction_worker*
  - *human.pedestrian.personal_mobility*
  - *human.pedestrian.police_officer*
- **bicycle:** *vehicle.bicycle*
- **vehicle:**
  - *vehicle.bus.rigid*
  - *vehicle.car*
  - *vehicle.construction*
  - *vehicle.emergency.ambulance*
  - *vehicle.emergency.police*
  - *vehicle.motorcycle*
  - *vehicle.trailer*
  - *vehicle.truck*

EC-RICO Classes.

- **person:**
  - *human.pedestrian.adult*
  - *human.pedestrian.child*
  - *human.pedestrian.construction_worker*
  - *human.pedestrian.personal_mobility*
  - *human.pedestrian.police_officer*
- **Car:** *vehicle.car*

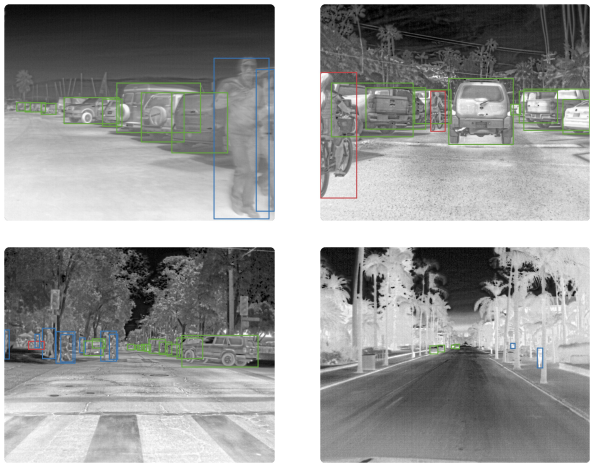


Figure S.6. Example images of the thermal task.

Figure S.7. Example images of the fisheye fix task.

- **Bicycle:** *vehicle.bicycle*

B.2.2. Thermal (Teledyne FLIR [26])

The Teledyne FLIR [26] dataset consists of 26,000 thermal images collected in various urban environments, covering a range of lighting and weather conditions. The dataset is used for thermal-based object detection and multispectral perception in autonomous driving applications.

Dataset Processing. The dataset labels the rider and the bicycle separately, as we define the bicycle class as the bicycle with the rider, we need to merge the two. The rider is labeled as *person*, so we calculate the intersection over union (IoU) of all person and bicycle bounding boxes. If the IoU exceeds 25%, the two boxes are merged and labeled bicycle. This is not always perfect; it also merges if someone is standing next to a bicycle. See Figure S.6 for four example images.

Train, Val, and Test Split. The dataset provides a scene ID from which the train, validation, and test split is created like described in Section B.2.1.

Labeling Policy The labeling policy is close to that of nuImages (see Section B.2.1)

D-RICO Classes.

- **person:** *person*
- **bicycle:** *bicycle*
- **vehicle:** *car*, *truck*, *bus*

EC-RICO Classes. The naming of the classes in the dataset matches the naming of EC-RICO, hence, we select for the EC-RICO classes the following labels: *person*, *car*, *bicycle*, *motorcycle*, *truck*, *bus*, and *traffic light*. All other available labels are not used.

B.2.3. Fisheye Fix (FishEye8K [32])

FishEye8K [32] is a dataset comprising 8,000 fisheye images collected from 18 surveillance cameras at road intersections in Taiwan. It features over 157,000 annotated objects across five categories, capturing a range of lighting conditions and object scales to support fisheye-based perception research.

Dataset Processing. No further processing is required for this dataset. Examples are shown in Figure S.7.

Train, Val, and Test Split. The dataset provides a scene ID from which the train, validation, and test split is created like described in Section B.2.1.

Labeling Policy Compared to the reference labeling policy (see Section B.2.1), the *bike* class consists of motorcycles and bicycles. However, as the number of motorcycles in this dataset is much greater than that of bicycles, we decided to label all as vehicles.

D-RICO Classes.

- **person:** *Pedestrian*
- **vehicle:** *Car*, *Bus*, *Truck*, *Bike*

EC-RICO Classes.

- **person:** *Pedestrian*
- **car:** *Car*
- **motorcycle:** *Bike*

B.2.4. Drone (VisDrone [111])

VisDrone [111] is a large-scale dataset comprising over 10,000 images and 263 video clips captured by drones in 14 cities across China. It includes dense urban and suburban scenes with millions of object annotations, facilitating aerial-based object detection and tracking research.

Dataset Processing. We remove images with regions labeled as *ignore*. These regions have many small objects that are not individually labeled. Bicycle and rider are merged into a combined bicycle class, as described in Section B.2.2. Figure S.8 depicts example images.

Figure S.8. Example images of the drone task.

Figure S.9. Example images of the simulation task.

Train, Val, and Test Split. The dataset provides a scene ID from which the train, validation, and test split is created like described in Section B.2.1.

Labeling Policy Compared to the reference labeling policy (see Section B.2.1), the rider of a motorcycle is labeled as a person and is not part of the motorcycle class. Some objects are left unlabeled; however, since the total number of labels is large, this is only a small fraction. The objects have amodal bounding boxes, but due to the bird's-eye view, these are generally not much different from visual bounding boxes.

D-RICO Classes.

- **person:** *pedestrian*, *people*
- **bicycle:** *bicycle*
- **vehicle:** *car*, *van*, *truck*, *tricycle*, *awning-tricycle*, *bus*, *motor*, *others*

EC-RICO Classes.

- **person:** *Pedestrian*
- **car:** *car*, *van*,*tricycle*, *awning-tricycle*,
- **bicycle:** *bicycle*,
- **motorcycle:** *motor*,
- **truck:** *truck*
- **bus:** *bus*

B.2.5. Simulation (SHIFT [83])

SHIFT [83] is a synthetic dataset created using the CARLA simulator, which contains 2.5 million annotated frames across 4,800 driving sequences. It encompasses a variety of environmental conditions, including weather changes, time of day, and traffic density, to investigate robustness in autonomous driving models.

Dataset Processing. As the frames are samples at a high frequency, we keep only every 50th frame. Example images are given in Figure S.9.

Train, Val, and Test Split. The dataset provides a scene ID from which the train, validation, and test split is created like described in Section B.2.1.

Labeling Policy The labeling policy matches the reference of Section B.2.1.

D-RICO Classes.

- **person:** *pedestrian*
- **bicycle:** *bicycle*
- **vehicle:** *car*, *truck*, *bus*, *motorcycle*

EC-RICO Classes.

- **person:** *Pedestrian*
- **car:** *car*
- **bicycle:** *bicycle*,
- **motorcycle:** *motorcycle*,
- **truck:** *truck*

B.2.6. Fisheye Car (WoodScape [97])

WoodScape [97] is a fisheye dataset created for autonomous driving, featuring over 100,000 images captured with surround-view cameras. It offers multi-task annotations for object detection, depth estimation, and semantic segmentation, emphasizing addressing fisheye distortion in perception models.

Dataset Processing. We recalculate the bounding boxes from the semantic and instance segmentation masks, as the provided bounding boxes are low-quality. We discard annotations with bounding boxes smaller than 25×25. To remove nighttime images from the dataset, we calculate each image's mean grayscale value and discard those below 0.3. This value was found by manual inspection of the images. See Figure S.10 for examples.

Train, Val, and Test Split. First, we discard two large sections of the image sequence to ensure a proper dataset split without overlap. This is necessary because no scene ID is provided, and we assume the data consists of a single continuous sequence. By removing sufficiently large portions, we break the sequence into three distinct parts while ensuring that the remaining data is split into approximately 60% for training, 10% for validation, and 30% for testing. This approach ensures that each subset represents a separate segment of the sequence, preventing consecutive frames from

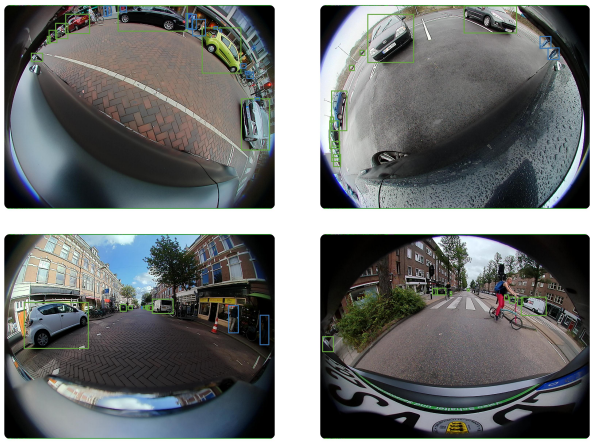


Figure S.10. Example images of the fisheye car task.

Figure S.11. Example images of the RGB + thermal fusion task.

Figure S.12. Example images of the video game task.

appearing across different splits and improving the robustness of the evaluation.

Labeling Policy Creating bounding boxes from segmentation masks can result in some individual objects being labeled with multiple bounding boxes due to interruptions in the segmentation. Aside from that, the labeling policy aligns with the reference in Section B.2.1.

D-RICO Classes.

- **person:** *person*
- **vehicle:** *car*, *train/tram*, *truck*, *trailer*, *van*, *caravan*, *bus*, *motorcycle*

B.2.7. RGB + Thermal Fusion (SMOD [13])

The SMOD dataset [13] comprises 8,676 well-aligned RGB and thermal image pairs, gathered for pedestrian detection and multispectral object recognition. The dataset features four object categories with comprehensive occlusion annotations, highlighting challenging low-light scenarios.

Dataset Processing. We first select low visibility images by calculating the mean gray value of each image and keep only those with a value below 0.35. For fusing the two modalities we use the neural network approach by [105]. It inputs the normalized RGB and thermal image and outputs a fused RGB image. See Figure S.11 for examples.

Train, Val, and Test Split. As no scene IDs are available, the dataset is split into train, val, and test set like described in Section B.2.6

Labeling Policy In this dataset, motorcycles are labeled as bicycles and, hence, are not part of the vehicle class. There are also numerous objects not labeled.

D-RICO Classes.

- **person:** *person*
- **bicycle:** *bicycle*, *rider*
- **vehicle:** *car*

B.2.8. Video Game (Sim10k [38])

SIM10K [38] is a synthetic dataset created using the *Grand Theft Auto V* game, consisting of 10,000 images annotated with vehicle information. It is used to investigate domain adaptation and transfer learning from synthetic to real-world driving datasets.

Dataset Processing. We select high visible images like described in Section B.2.6 with a gray value threshold of 0.4. Examples are presented in Figure S.12.

Train, Val, and Test Split. As the dataset does not include scene IDs, we split it up like in Section B.2.6 described.

Labeling Policy Compared to Section B.2.1, the rider on the motorcycle is not included in the bounding box. The rider, however, is also not labeled as a person.

D-RICO Classes. We only use the vehicle class, constructed from *car* and *motorbike* classes.

B.2.9. Nighttime (BDD100K [98])

BDD100K [98] is a large-scale driving dataset of 100,000 videos captured under various conditions, including nighttime driving. It offers extensive annotations for multiple perception tasks, making it one of the most comprehensive

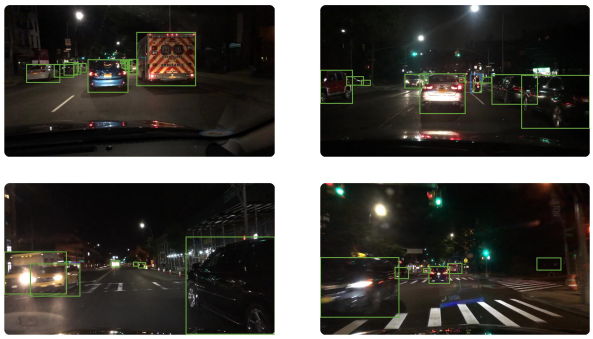


Figure S.13. Example images of the nighttime task.

datasets in autonomous driving.

Dataset Processing. Based on the provided meta-data, we select only those images from the dataset that are labeled as night and clear weather. Rider and bicycle are merged like in Section B.2.2 described. Figure S.13 shows example images.

Train, Val, and Test Split. Based on the available scene IDs, the data is split into training, validation, and test like in Section B.2.1 described.

Labeling Policy The labeling policy aligns with the reference in Section B.2.1.

D-RICO Classes.

- **person:** *person*, *rider*
- **bicycle:** *bike*
- **vehicle:** *car*, *truck*, *motor*, *train*, *bus*

EC-RICO Classes.

- **person:** *person*, *rider*
- **car:** *car*
- **bicycle:** *bike*,
- **motorcycle:** *motor*,
- **truck:** *truck*
- **bus:** *bus*
- **traffic light:** *traffic light*
- **street sign:** *street sign*

B.2.10. Fisheye Indoor (LOAF [96])

LOAF [96] is a dataset designed for person detection in fisheye images, comprising 43,000 frames extracted from 70 videos filmed in indoor and outdoor surveillance environments. It features radius-aligned bounding boxes specifically adapted for fisheye distortion, emphasizing person localization from overhead camera perspectives.

Dataset Processing. The dataset only provides rotated bounding boxes. We transform them into non-rotated bounding boxes using the given angle. However, this results in boxes that aren't tightly fitted around the object. To narrow the task, we manually choose only indoor scenes. Example images are illustrated in Figure S.14.

Train, Val, and Test Split. Based on the available scene IDs, the data is split into training, validation, and test like in Section B.2.1 described.

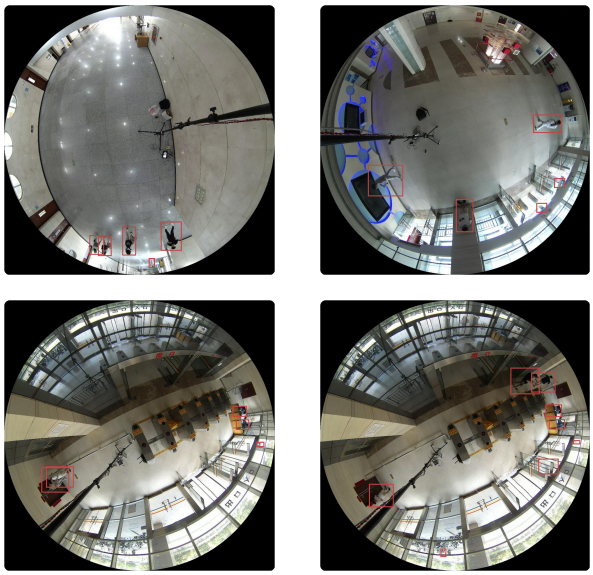


Figure S.14. Example images of the fisheye indoor task.

Labeling Policy The labeling policy corresponds with the reference in Section B.2.1.

D-RICO Classes. The dataset only provides labels for the *person* class.

B.2.11. Gated (DENSE [4])

DENSE [4] is a multimodal dataset collected over 10,000 kilometers of driving, featuring data from gated cameras, LiDAR, and radar. It focuses on autonomous perception in adverse weather conditions like fog, rain, and snow.

Dataset Processing. We select all non-inclement images from the dataset. We further remove all images that include one of the following classes: *DontCare*, *LargeVehicle_is_group*, *Vehicle_is_group*, *PassengerCar_is_group*, *RidableVehicle_is_group*, and *Pedestrian_is_group* This is because the objects of the group could be separated and therefore create contradictions. We remove all objects smaller than 7×7, as such small objects are not labeled in other datasets.

Figure S.15 provides example images.

Train, Val, and Test Split. The data is divided into training, validation, and test sets based on the available scene IDs, as outlined in Section B.2.1.

Labeling Policy The labeling policy aligns with Section B.2.1.

D-RICO Classes.

- **person:** *Pedestrian*,
- **vehicle:** *PassengerCar*, *Vehicle*, *LargeVehicle*

EC-RICO Classes.

- **person:** *Pedestrian*

Figure S.15. Example images of the gated task.

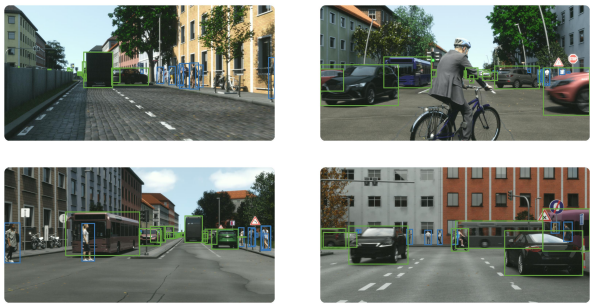


Figure S.16. Example images of the photorealistic simulation task.

- **car:** *PassengerCar*

B.2.12. Realistic Simulation (Synscapes [92])

Synscapes [92] is a synthetic dataset of 25,000 photorealistic urban driving images created using physically based rendering techniques. It provides per-pixel semantic segmentation and instance annotations, which are designed for training perception models with realistic lighting and material properties.

Dataset Processing. The dataset provides bounding boxes; however, these also include labels for objects that are completely hidden. We calculate the bounding boxes for each visible object based on the semantic and instance segmentation masks. We only select images with mean gray values above 0.38 to exclude low visibility and night images (see Section B.2.6). Refer to Figure S.16 for examples.

Train, Val, and Test Split. Each image is rendered from a unique scene, and the images can, therefore, be randomly assigned into training, validation, and test splits.

Labeling Policy Multiple objects are not labeled, and some bicycles are labeled as vehicles.

D-RICO Classes.

- **person:** *person*,
- **vehicle:** *car*, *truck*, *bus*, *train*, *motorcycle*, *rider*

B.2.13. Thermal Fisheye Indoor (TIMo [74])

TIMo [74] is a thermal imaging dataset designed for indoor person detection, comprising over 612,000 frames captured with infrared cameras. It features detailed annotations for tracking individuals and detecting anomalies in low-visibility environments.

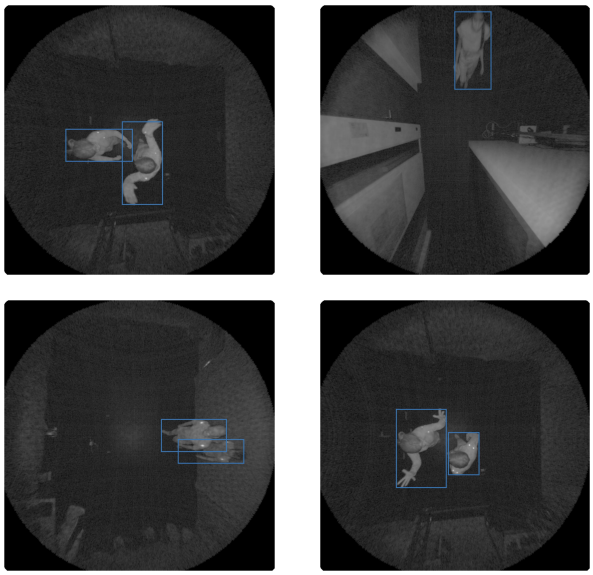


Figure S.17. Example images of the thermal fisheye indoor task.

Dataset Processing. We use only the infrared images from the dataset and keep every sixth frame. The images provided in the dataset are in signal strength. We calculate the logarithm of the images and normalize them. Examples are displayed in Figure S.17.

Train, Val, and Test Split. The dataset is separated into training, validation, and test sets according to the scene IDs as detailed in Section B.2.1.

Labeling Policy The labeling policy is consistent with Section B.2.1.

D-RICO Classes. The dataset exclusively offers labels for the *person* class.

B.2.14. Inclement (DENSE [4])

This subset of DENSE [4] includes images taken in non-clear weather conditions like heavy fog, snow, and rain. It offers a challenging testbed for assessing robustness in perception systems under limited visibility.

Dataset Processing. We manually sort each image according to the weather conditions. We select those that depict intense fog, snow, and heavy rain. Additionally, we eliminate small objects and images based on the group classes, as described in Section B.2.11. Example images are shown in Figure S.18.

Train, Val, and Test Split. The dataset is divided into training, validation, and test sets based on the scene IDs outlined in Section B.2.1.

Labeling Policy The labeling policy is consistent with Section B.2.1.

Figure S.18. Example images of the inclement task.

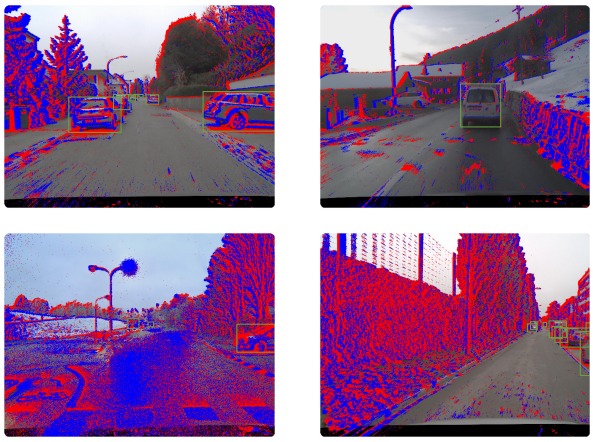


Figure S.19. Example images of the event camera task.

D-RICO Classes.

- **person:** *Pedestrian*,
- **vehicle:** *PassengerCar*, *Vehicle*, *LargeVehicle*

B.2.15. Event Camera (DSEC [28, 29])

DSEC [28, 29] is a multimodal dataset that features high-resolution stereo event cameras, as well as RGB and LiDAR data. It captures 53 driving sequences in urban and rural settings, facilitating research on event-based perception in dynamic environments.

Dataset Processing. The gated and RGB images are overlaid according to the provided algorithm [28, 29]. Since this creates bounding boxes that extend beyond the images, we adjust them to fit the image size. As detailed in Section B.2.2, the rider and bicycle are combined. Figure S.19 showcases example images.

Train, Val, and Test Split. Since the dataset lacks scene IDs, we divided it as outlined in Section B.2.6.

Labeling Policy The labeling policy matches with Section B.2.1, except for the missing labels for trams.

D-RICO Classes.

- **person:** *pedestrian*, *rider*
- **bicycle** *bicycle*
- **vehicle:** *car*, *bus*, *truck*, *motorcycle*, *train*

EC-RICO Classes.

- **person:** *Pedestrian*
- **car:** *PassengerCar*
- **bicycle:** *bicycle*
- **motorcycle:** *motorcycle*

C. Additional Details on the Experimental Setup

In this section, we provide additional details on the experimental setup, including the hyperparameters and the adaptations made during the implementation of the IL methods.

C.1. Hyperparameters

We mostly used the hyperparameters as specified in [24]. Table S.4 lists the general hyperparameters we used for all experiments that differ from [24]. In Table S.5 to S.14, the experiment-specific hyperparameters are listed. For all trainings, we conducted a small random hyperparameters search on the learning rate and the method-specific hyperparameters.

C.2. Implementation Details of IL Methods

In this section we elaborate on the implementation details of the Replay approach and ABR[55], Meta-ILOD [39], BPF [66] and LDB [81].

C.2.1. Replay

We employ a growing memory buffer that is initialized empty. For each new task, a subset of training data is randomly selected and added to the buffer. During training, the remaining task data is merged with the memory buffer.

A growing buffer is preferable to a static one, especially for long task sequences. A fixed-size buffer either over-replays early task examples, wasting compute, or retains too few examples as tasks accumulate. If memory constraints are not strict, an expanding buffer ensures stability by maintaining consistent replay examples per task.

To keep the number of training iterations on new task data constant, the total iterations must increase over time. Consequently, both memory and compute requirements grow with the number of tasks. This approach is feasible only if resource constraints are not too restrictive.

C.2.2. ABR

In the original ABR implementation, the memory buffer accumulates boxes from previous tasks but is only divided by class. We introduce an additional split by task, motivated by the feature dissimilarity across objects, which prevents defining a uniform proximity metric to the class mean.

Our approach retains a fixed-size memory buffer, as in the original method, but after each task, the half containing the least representative boxes for each class and task is discarded. We compute each object's feature vector without

Table S.4. General hyperparameters used in all experiments if not specified differently.

Hyperparameter	Value
batch size	20
input image size	$1536 \times 1536 \times 3$
training iterations	696
evaluation period	500
warm-up length	0.1
warm-up factor	0.001
learning rate	0.001
learning rate scheduler	cosine
learning rate scheduler end value	0
Aug.: rnd. flip	True
Aug.: resize scale	True
Aug.: fixed size crop	True
Aug.: rnd. brightness	True
Aug.: rnd. contrast	True
Aug.: rnd. saturation	True
Aug.: rnd. lightning	True
Aug.: resize scale, min scale	0.1
Aug.: resize scale, max scale	2
Aug.: rnd. brightness, min. intensity	0.6
Aug.: rnd. brightness, max. intensity	1.4
Aug.: rnd. contrast, min. intensity	0.6
Aug.: rnd. contrast, max. intensity	1.4
Aug.: rnd. saturation, min. intensity	0.6
Aug.: rnd. saturation, max. intensity	1.4
Aug.: rnd. lightning, scale	0.1
Aug. for evaluation and testing	False

Abbreviations: Aug. = augmentation, min. = minimum, max. maximum, rnd. = random

Table S.5. Hyperparameters for joint training on D-RICO.

Hyperparameter	Value
training iterations	$10,440$
evaluation period	696

Table S.6. Hyperparameters for joint training on EC-RICO.

Hyperparameter	Value
training iterations	$5,568$
evaluation period	696

Table S.7. Hyperparameters for ABR [55] training on D-RICO.

Hyperparameter	Value
learning rate	0.001
α for class distillation	1
α for box distillation	1
β for box distillation	0
γ for box distillation	0
memory buffer size	$2,000$
training iterations	870

Table S.8. Hyperparameters for ABR [55] training on EC-RICO.

Hyperparameter	Value
learning rate	0.001
α for class distillation	1
α for box distillation	1
β for box distillation	0
γ for box distillation	0
memory buffer size	$2,000$
training iterations	870

Table S.9. Hyperparameters for Meta-IOD [39] training on D-RICO.

Hyperparameter	Value
learning rate	0.0001
Number of features per class	5000
Number of images per class	5000
Distillation weight	1
Warp Iteration	20
Warp Layer	box_head.2.conv4

augmentations to ensure representative features. Stability is improved by selecting only non-overlapping boxes and excluding those smaller than 20×20 pixels [88].

For mixup and mosaic, we closely follow the ABR implementation. Since we apply resize augmentation, we also scale replayed boxes in the mixup to match the resized image.

Using the Cascade Faster-RCNN [8] detection head prevents direct application of the same distillation approach. Instead, we generate soft labels from the last (*i.e.* third) cascade stage without filtering empty boxes. Given our large input image size of 1536×1536, we stabilize the bounding box distillation loss by computing it on the logarithm of the

Table S.10. Hyperparameters for Meta-IOD [39] training on EC-RICO.

Hyperparameter	Value
learning rate	0.0001
Number of features per class	5000
Number of images per class	5000
Distillation weight	1
Warp Iteration	20
Warp Layer	box_head.2.conv4

Table S.11. Hyperparameters for BPF [66] training on D-RICO.

Hyperparameter	Value
learning rate	0.001
α for class distillation	0.1
α for box distillation	0.15
memory buffer size	2000

Table S.12. Hyperparameters for BPF [66] training on EC-RICO.

Hyperparameter	Value
learning rate	0.001
α for class distillation	0.15
α for box distillation	0.15
memory buffer size	2000

Table S.13. Hyperparameters for LDB [81] training on D-RICO.

Hyperparameter	Value
learning rate	0.001
batch size	5
training iterations	2784

Table S.14. Hyperparameters for LDB [81] training on EC-RICO.

Hyperparameter	Value
learning rate	0.001
batch size	5
training iterations	2784

box coordinates.

Since not all objects are labeled for every class in each task, we apply a mask during distillation to ensure it only affects classes seen by the teacher during training.

C.2.3. Meta-ILOD

We follow the original implementation for image and feature storage. The *warping* operation is applied to the last convolution layer of the final (*i.e*. third) cascade stage.

Since the backbone is fixed and not trained, we omit the backbone feature distillation loss. The *warp* loss is computed using the last cascade stage.

In all other aspects, we adhere closely to the original implementation.

C.2.4. BPF

BPF involves generating pseudo labels using the previous model. However, since we do not operate in a CIL setting, we disable this feature.

For distillation, we obtain soft predictions from the final cascade stage, similar to ABR. Additionally, we apply masking in the distillation loss to ensure learning only from predictions the teacher model was trained on. As the *fine-tuning* teacher, we use the model trained for the individual training baseline.

C.2.5. LDB

Since we apply augmentations during training but not during inference, we cannot directly collect image features for the nearest mean classifier during training. Instead, we compute the image features and class means separately, storing the means for use during IL.

Our IL framework enables task iteration without requiring separate training for each task. To improve efficiency, we store domain bias terms and output layers as matrices, selecting the corresponding row based on the determined task ID. This setup allows a single model to be used for inference, automatically selecting the appropriate model components based on the task ID.

C.3. Mean Average Precision

We report the COCO-style mean Average Precision (mAP), computed by averaging the AP at ten intersection over union (IoU) thresholds from 0.50 to 0.95 in increments of 0.05, denoted as mAP@$[0.50 : 0.05 : 0.95]$.

C.4. Task Affinity

For the task affinity metric, only the output layers of the backbone are adapted, meaning the last layer in both the classification and bounding box regression parts.

D. Additional Results

The main paper holds the primary results of this work. We included here some additional results.

D.1. Quantitative Comparison to Existing Benchmarks

In Table 1, we qualitatively compare the proposed benchmarks to existing ones. To highlight these differences

quantitatively, we conduct a comparative analysis with the CLAD-D [85] and VOC Series [81] benchmarks (see Section 2 for brief descriptions). Specifically, we evaluate joint training, individual training, naive FT, and replay with 10% of the training data. The results for CLAD-D and VOC Series are presented in Table S.15, while the results for D-RICO and EC-RICO are shown in Table 3.

We observe the most significant difference in naive FT forgetting rates: 19% for D-RICO compared to just 4.33% and 4.21% for CLAD-D and VOC Series, respectively. A similar pattern emerges in the 1% replay scenario, where forgetting is minimal for CLAD-D and VOC Series but remains relatively high (8.91%) for D-RICO. Additionally, the $\overline{\mathbf{mAP}}$ is lower for D-RICO, indicating that a larger, more comprehensive model is necessary to achieve strong overall performance.

Beyond incremental learning experiments, we also evaluate task affinity (see Section 3.5) across these benchmarks. We report mean TA scores of 94% for CLAD-D, 98% for VOC Series, and notably lower at 76% for D-RICO. This indicates that the output layer alone is insufficient for adapting to new tasks in D-RICO, underscoring greater task diversity and dissimilarity.

Overall, these experiments clearly demonstrate that the diversity present in D-RICO and consequently EC-RICO significantly exceeds that of the existing benchmarks, CLAD-D and VOC Series.

Table S.15. Results on the CLAD-D and VOC Series benchmarks.

Benchmark	Method	$\overline{\mathbf{mAP}}\uparrow$	FM $\downarrow$	FWT $\uparrow$	IM $\uparrow$
CLAD-D	Joint Training	59.96			
	Individual Training	55.24			
	Naive FT	49.88	4.33	−1.05	−2.24
	Replay 1%	50.36	3.79	−0.36	−2.17
VOC Series	Joint Training	62.54			
	Individual Training	64.49			
	Naive FT	59.86	4.21	−1.06	0.46
	Replay 1%	61.45	2.24	−0.49	0.59

D.2. Unfrozen Backbone

In addition to the experiments with a frozen backbone, we analyze the impact of making the backbone trainable. For simplicity, we use a subset of D-RICO (tasks: $1 \to 2 \to 3 \to 11 \to 15$) and compare joint and individual training, naive fine-tuning (FT), and 1% replay in this reduced setting, both with and without a frozen backbone.

Table S.16 presents the results, showing that unfreezing the backbone improves model plasticity and overall performance. However, in the naive FT scenario, this comes at the cost of increased forgetting. Since 1% replay is generally effective at mitigating forgetting, it also helps reduce forgetting when the backbone is unfrozen. These findings suggest that while an unfrozen backbone can offer performance benefits, it also amplifies forgetting, making the choice between frozen and unfrozen non-trivial. In such cases, more robust IL methods become necessary to counteract the negative effects of increased plasticity.

Table S.16. Un/frozen backbones (tasks: $1 \to 2 \to 3 \to 11 \to 15$).

Benchmark	Method	$\overline{\mathbf{mAP}}\uparrow$	FM $\downarrow$	FWT $\uparrow$	IM $\uparrow$
frozen backbone	Joint Training	37.30			
	Individual Training	42.15			
	Naive FT	26.77	18.99	−2.76	6.94
	Replay 1%	38.53	3.33	−0.16	6.18
unfrozen backbone	Joint Training	38.33			
	Individual Training	44.80			
	Naive FT	29.03	21.57	2.94	9.38
	Replay 1%	42.03	2.58	1.21	6.57

D.3. Detailed Run Results

Table 3 presents the AA, FM, FWT, and IM metrics, summarizing each method's performance. To provide a more detailed view, Figure S.20 illustrates the progression of each task after its initial learning, along with the evolution of these metrics.

Performance generally declines after learning a new task. With increasing replay size, IL metrics become more parallel and stable. The ABR and BPF methods closely resemble naïve FT, highlighting their underperformance in IL settings. In contrast, Meta-ILOD and LDB exhibit greater stability.

Qualitatively, D-RICO and EC-RICO behave similarly, with weaker methods showing more unstable learning curves. Both benchmarks exhibit a decreasing AA trend. The IM drops across all runs, while FWT increases, indicating that at the start of training, all approaches outperform joint training but still lag behind individual training. Interestingly, FWT continues to rise, suggesting increased model plasticity, though this effect is minor and stabilizes after one or two tasks.

As expected, FM increases for D-RICO, reflecting the growing challenge of knowledge retention with additional tasks. However, unexpectedly, FM decreases in EC-RICO. We hypothesize that this results from the increasing number of labels, reducing the relative contribution of early tasks.

Analyzing individual runs reveals that some tasks follow similar trends, suggesting they rely on overlapping features.

D.4. Joint and Individual Training Results

Joint and individual training represent two kinds of upper bounds, with joint training (D-RICO: 43.75, EC-RICO: 38.46) serving as a practical upper bound for single model configurations. The individual training (D-RICO: 49.37, EC-RICO: 45.54) demonstrates what would be possible

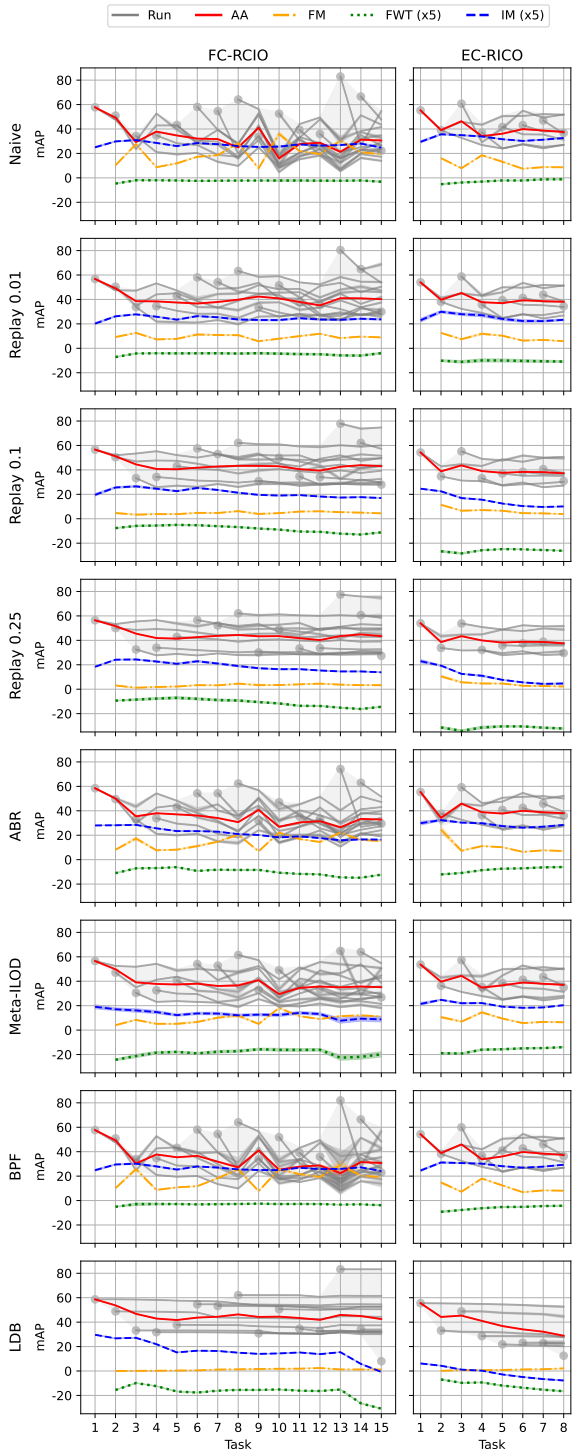


Figure S.20. Test performance on D-RICO and EC-RICO for the tested methods. The value of FWT and IM (except for LDB for EC-RICO) are scaled by a factor five to enhance visibility.

without the restriction of a single model. To analyze this discrepancy further, we compare individual and joint training using the relative difference to individual training.

Table S.17 quantifies the performance gap between joint and individual training. Smaller objects have the highest differences, with $\text{mAP}_{\text{small}}$ dropping most (D: 0.25, EC: 0.23), highlighting the challenge of learning fine-grained details in a joint setting.

For D-RICO, bicycles show the highest discrepancy (0.39) due to inconsistent labeling, often misclassified as motorcycles or treating bicycle racks differently. In EC-RICO, motorcycles (0.27) and bicycles (0.19) also exhibit high gaps due to annotation policies, where motorcycles are sometimes only labeled when ridden. Traffic lights (0.24) and street signs (0.35) show notable differences, driven by noisy annotations and small bounding boxes.

As street signs appear in only one task, this suggests that intra-class contradictions alone do not fully explain the discrepancy, making the benchmark inherently challenging for both IL methods and joint training. The reason could be that the street sign class is introduced during nighttime tasks, making it not always clearly visible and somewhat ambiguous.

Our further analysis in this regard yields the following concrete examples, which demonstrate contradictions and false negatives during both training and evaluation:

1. *Thermal+RGB:* The object labeled as a *bicycle* in the ground truth is actually a motorbike, which is instead predicted as a *vehicle* by the model.
2. *Drone:* In this case, the ground truth labels for bicycles are annotated as *vehicle*, whereas the prediction correctly identifies them as *bicycle*. Additionally, the amodal ground truth bounding boxes do not match with the predictions.
3. *Video game:* The ground truth annotations exclude the rider from the motorbike bounding boxes, in contrast to the predictions, which include the rider as part of the object.
4. *Fisheye car* and *Fisheye Fixed:* In these settings, small and distant objects present in the ground truth are frequently missed by the model's predictions.

D.5. Task Order

In Section 4.7, we analyze the effect of different task orders on performance. The following task orders were evaluated:

A) $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow 14 \rightarrow 15$
B) $15 \rightarrow 14 \rightarrow 13 \rightarrow 12 \rightarrow 11 \rightarrow 10 \rightarrow 9 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$
C) $1 \rightarrow 15 \rightarrow 8 \rightarrow 6 \rightarrow 13 \rightarrow 10 \rightarrow 14 \rightarrow 11 \rightarrow 12 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 9 \rightarrow 4 \rightarrow 7$
D) $13 \rightarrow 10 \rightarrow 14 \rightarrow 11 \rightarrow 8 \rightarrow 6 \rightarrow 4 \rightarrow 15 \rightarrow 9 \rightarrow 2 \rightarrow 5 \rightarrow 12 \rightarrow 3 \rightarrow 1 \rightarrow 7$
E) $15 \rightarrow 4 \rightarrow 7 \rightarrow 13 \rightarrow 2 \rightarrow 3 \rightarrow 9 \rightarrow 11 \rightarrow 1 \rightarrow 12 \rightarrow 5 \rightarrow 10 \rightarrow 8 \rightarrow 14 \rightarrow 6$

To evaluate robustness to task ordering, we extend this with five additional randomly sampled orders (ten total). These experiments confirm that performance is largely insensitive to task order for replay-based methods, with average performance $\overline{\mathbf{mAP}} = 42.57_{\pm 0.46}$ and forgetting measure $\text{FM} = 4.41_{\pm 0.65}$. In contrast, naïve fine-tuning (FT)

Table S.17. Results for D-RICO and EC-RICO.

D-RICO		EC-RICO	
Metric	**Rel. Diff.**	**Metric**	**Rel. Diff.**
mAP	$0.13_{\pm 0.05}$	mAP	$0.16_{\pm 0.05}$
$\text{mAP}_{\text{large}}$	$0.08_{\pm 0.06}$	$\text{mAP}_{\text{large}}$	$0.12_{\pm 0.06}$
$\text{mAP}_{\text{medium}}$	$0.13_{\pm 0.05}$	$\text{mAP}_{\text{medium}}$	$0.16_{\pm 0.06}$
$\text{mAP}_{\text{small}}$	$0.25_{\pm 0.19}$	$\text{mAP}_{\text{small}}$	$0.23_{\pm 0.09}$
$\text{mAP}_{\text{person}}$	$0.14_{\pm 0.06}$	$\text{mAP}_{\text{person}}$	$0.16_{\pm 0.06}$
$\text{mAP}_{\text{bicycle}}$	$0.39_{\pm 0.48}$	mAP_{car}	$0.11_{\pm 0.07}$
$\text{mAP}_{\text{vehicle}}$	$0.11_{\pm 0.06}$	$\text{mAP}_{\text{bicycle}}$	$0.19_{\pm 0.10}$
		$\text{mAP}_{\text{motorcycle}}$	$0.27_{\pm 0.22}$
		$\text{mAP}_{\text{truck}}$	$0.17_{\pm 0.09}$
		mAP_{bus}	$0.14_{\pm 0.02}$
		$\text{mAP}_{\text{traffic light}}$	$0.24_{\pm 0.02}$
		$\text{mAP}_{\text{street sign}}$	$0.35_{\pm 0.01}$

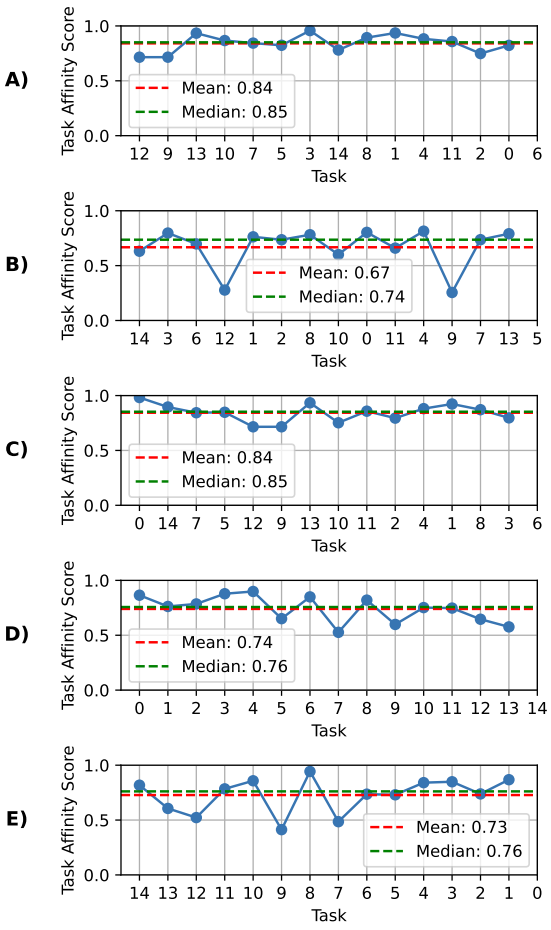


Figure S.21. Task affinity to the next task for different task orders.

exhibits moderate sensitivity, with $\overline{\text{mAP}} = 32.33_{\pm 3.39}$ and $\text{FM} = 17.85_{\pm 3.75}$.

The TA score, as defined in Section 3.5, is visualized for each order in Figure S.21. Interestingly, task order **B)** results in the best outcome for naïve FT, despite having the lowest average TA and exhibiting several sharp drops in affinity. This reinforces our finding that TA does not correlate with IL performance for either FT or replay. However TA remains a valuable unified metric for quantifying task (dis)similarity—even if it does not predict continual learning performance directly or capture all influencing factors.

D.6. Class imbalance

We evaluated *federated loss* (EVA-02 implementation) for EC-RICO, but see no gains (Tab. S.18). Improving balancing further could boost future methods, but that lies beyond this benchmark's scope.

Table S.18. Results on EC-RICO with federated loss.

Method	$\overline{\text{mAP}}\uparrow$	FM $\downarrow$	FWT $\uparrow$	IM $\uparrow$
joint training	37.48			
individual training	45.40			
naive finetuning	37.86	7.51	−0.67	6.96
replay 10%	37.69	3.39	−4.90	3.18

E. Detectron2 for IL

Incremental Learning (IL) for classification typically relies on the Avalanche library [58]. However, the model architecture and evaluation protocols for object detection differ substantially from classification, necessitating extensive modifications to the Avalanche framework. Most prior research on IL for object detection utilizes established object detection frameworks such as Detectron2 [93], restarting the entire codebase for each task while preserving and transferring essential information between tasks [39, 55, 66, 81].

While this approach is functional and existing templates can facilitate future work, we opted to refactor Detectron2 to better accommodate IL requirements. The primary enhancement is the implementation of task-iterative processing, allowing seamless storage of data, models, and images as variables between consecutive tasks. This refactoring required the following modifications:

- **Data Sampler:** We developed a specialized data sampler that processes the complete training, validation, and testing annotation files while allowing selective loading of task-specific data.
- **Evaluator:** IL frameworks necessitate evaluation across both current and previous tasks. We modified the evaluator to systematically iterate through and assess all tasks.
- **Event Storage:** When training and evaluating tasks within iterative loops, metrics must be stored in task-specific contexts. Our custom event storage mechanism preserves metrics separately for each task.

- **IL Metrics:** After each task, all test sets are evaluated and key IL metrics are calculated, including *Average Accuracy*, *Average Incremental Accuracy*, *Forgetting Measure*, and *Backwards Transfer*. Plasticity metrics require subsequent computation using results from both joint and individual training paradigms.
- **Joint and Individual Training:** Both joint and individual training runs must be conducted to calculate plasticity metrics. These results can be generated using a designated flag in our framework.
- **Replay:** Using a configurable percentage parameter, historical data can be incorporated into a replay buffer and utilized in subsequent tasks.

Beyond these core modifications, numerous additional changes were implemented regarding hooks, model persistence mechanisms, task ordering protocols, and other system components.

The advantages of our customized Detectron2 version include:

- Accelerated integration of novel methodologies
- More coherent experimental configuration
- Enhanced training efficiency

Detailed installation instructions for `Detectron2-IL` are provided in the corresponding GitHub Repository.