



ARTICLE

ReMiss: Missed-Object History for Data Scheduling in Driving-Scene Object Detection

Seongwook Lee¹ , Seungyeong Kim¹ , Jaemin Oh¹ , L. Minh Dang² and Hyeonjoon Moon^{3,*}

¹Department of Convergence Engineering for Artificial Intelligence, Sejong University, Seoul, Republic of Korea

²Ho Chi Minh City Open University, Ho Chi Minh City, Vietnam

³Department of Computer Science and Engineering, Sejong University, Seoul, Republic of Korea

*Corresponding Author: Hyeonjoon Moon. Email: hmoon@sejong.ac.kr

Received: 24 June 2026; Accepted: 02 September 2026; Published: 22 October 2026

ABSTRACT: Average precision (AP) is an aggregate detector metric and does not identify which annotated ground-truth instances remain unmatched across repeated evaluations during training. Existing hard example mining and sampling strategies usually define difficulty at the image, proposal, class, or loss level and do not retain recurrent object-level failures after post-processing as a data-scheduling signal. ReMiss uses missed-object history as a training-time data scheduling signal without changing detector architecture or loss. MissBank records each ground-truth instance's miss status, miss-count history, and localization and confidence diagnostics. Hard Replay (HR) assigns priorities to full images containing currently missed objects based on miss counts and matching gaps, then schedules those images into later epochs under a replay budget. We evaluate ReMiss on KITTI, BDD100K, Cityscapes, and nuImages with fully convolutional one-stage object detection (FCOS), Faster R-CNN, and detection transformer with improved denoising anchor boxes (DINO), yielding 12 dataset–detector pairs. Across the full replay-ratio sweep, all 36 dataset–detector–ratio configurations improve upon the No-HR diagnostic baseline, with a minimum gain of +0.22 AP (sign test, $p < 10^{-10}$); under a replay ratio fixed in advance at 5.0%, all 12 pairs improve by +0.64–+2.17 AP. Selecting the HR setting by validation mean average precision (mAP), reported at mAP_{50:95}, yields +1.03–+2.23 AP. These results support missed-object history as an architecture- and loss-agnostic scheduling signal across the evaluated detector families, with a measured 13%–27% increase in training time over the training-only baseline and no additional inference cost.

KEYWORDS: Object detection; missed detection; ReMiss; MissBank; hard replay; hard example mining; autonomous driving; DINO

1 Introduction

Object detection is a core perception function for autonomous driving and intelligent transportation systems [1–3]. Driving-scene detectors must continuously localize perception-relevant objects, including vehicles, pedestrians, cyclists, and traffic infrastructure. This task is complicated by long-range viewpoints, small objects, partial occlusion, dense layouts, motion blur, illumination changes, and class imbalance [4]. Although modern detectors achieve strong average benchmark performance, aggregate metrics alone do not fully describe perception-level failure patterns [5–7]. Two detectors with similar mean average precision (mAP) can differ substantially in which ground-truth objects they detect reliably and which annotated instances are absent from the final predictions.

This paper defines recurrent missed detection as an annotation-instance history rather than video tracking. Across offline mining snapshots, the same annotated training instance is recurrently missed when no same-class prediction in the detector’s final prediction interface matches it after post-processing. The final prediction interface is the set of boxes, labels, and scores used for evaluation; a miss occurs when no same-class prediction satisfies both the confidence-score and intersection-over-union conditions. Repeated misses for small pedestrians, distant vehicles, or partially occluded targets expose perception-level failures that aggregate average precision (AP) can obscure [4,6]. Thus, driving-scene training benefits from tracking not only image- or class-level difficulty, but also object-level failures that persist after final post-processing.

Many existing training strategies address this gap only indirectly. Sampling, reweighting, and hard example mining help emphasize difficult samples or imbalanced classes, but they usually define difficulty at the proposal, loss, or class level [8–10]; Section 2.3 reviews this family in detail. These signals are useful for optimization, but they do not preserve, across epochs, whether a particular ground-truth instance remains unmatched after final post-processing. The signal considered in this work is object-level persistence: the same annotated instance stays missed across multiple mining stages. This leaves a practical gap between object-level failures observed in final predictions and the construction of the next epoch’s training data.

We propose ReMiss to carry this signal into training. ReMiss is a training-time framework that uses missed-object history as a data scheduling signal without changing detector architecture or loss. MissBank scans the base training data at offline mining stages and stores each ground-truth instance’s missed state and temporal miss statistics. Hard Replay (HR) selects full images containing currently missed objects, prioritizes them by repeated misses and matching gaps, and inserts replay samples into the next epoch under an epoch-level budget. Because replay is image-level, scale, occlusion, surrounding clutter, and contextual ambiguity are retained. Accordingly, missed-object history guides full-image exposure, while optimization remains the detector’s standard objective and the Hard Replay settings used for the main comparisons are selected by validation $mAP_{50:95}$.

This design allows the same data-centric mechanism to be applied to the three detector families evaluated in this study: fully convolutional one-stage object detection (FCOS), Faster R-CNN, and detection transformer with improved denoising anchor boxes (DINO). MissBank and Hard Replay do not depend on detector-internal features, auxiliary heads, or new optimization objectives. They use only final boxes, labels, scores, and a MissBank matching rule. We evaluate this setting on four driving-scene datasets: KITTI, BDD100K, Cityscapes, and nuImages. The validation-selected results show that missed-object history can serve as a detector-agnostic data scheduling signal, while the replay-ratio sweep shows that the effect varies by dataset, detector, replay ratio, missed-object pool, and replay budget. Runtime analysis measures the cost of adding offline mining and replay.

The contributions of this work are fourfold. First, we formalize recurrent missed detection as a missed-object history signal: a ground-truth instance is not matched to any final prediction across multiple mining snapshots. Second, we implement this signal through MissBank, an instance-level memory that records missed states and matching diagnostics during offline mining. Third, we design Hard Replay, a data-layer sampler that assigns persistence-aware priorities to full images containing currently missed objects and schedules them into the next epoch’s training stream. Fourth, we evaluate ReMiss on KITTI, BDD100K, Cityscapes, and nuImages with FCOS, Faster R-CNN, and DINO, characterizing validation-selected gains, replay-ratio sensitivity, diagnostic MissBank behavior, and runtime overhead. The first contribution is what separates ReMiss from existing difficulty-aware and replay-based training: the signal is defined at the final prediction interface and attached to an individual ground-truth instance, rather than to a loss value, a proposal, a class, or an example. Table 1 in Section 2.5 sets this against the alternatives.

2 Related Work

2.1 Driving-Scene Object Detection and Benchmarks

Object detection for autonomous driving and intelligent transportation is directly tied to localizing perception-relevant objects in real road scenes. KITTI established an early benchmark for autonomous-driving perception using images and sensor data collected from a driving platform [1]. Cityscapes highlighted the complexity of urban scene understanding through finely annotated street scenes [11]. BDD100K expanded this setting with large-scale driving videos across diverse locations, weather, and time-of-day conditions [2]. nuScenes and nuImages further broadened perception research by providing multimodal sensor data and image-level 2D annotations under diverse driving conditions [3,12].

These benchmarks enable detector comparison through AP and recall-oriented metrics, while diagnostic analyses explain how detection errors arise after evaluation [6,7]. Such diagnostics are valuable for interpreting false positives, localization errors, and missed objects, but they are typically retrospective rather than part of the training data schedule. ReMiss adopts this diagnostic viewpoint at training time: it records which annotation instances remain unmatched under the detector's final prediction interface and uses that history as a scheduling signal, without redefining the benchmark metric.

2.2 Detector Families and Post-Processed Prediction Interfaces

Modern object detectors have developed across two-stage, one-stage, and transformer-based families. Faster R-CNN combines a region proposal network with a detector head and remains a representative two-stage detection pipeline [13]. FCOS is an anchor-free and proposal-free one-stage detector that provides a simple dense-prediction formulation [14]. The detection transformer (DETR) reformulates detection as set prediction with a transformer encoder-decoder and bipartite matching [15]. Deformable DETR improves convergence and spatial resolution through deformable attention [16], and DINO improves DETR-style training with denoising training, query selection, and box refinement [17]. Subsequent query-based detectors continue along the same interface: Co-DETR adds collaborative one-to-many assignments [18], and RT-DETR removes non-maximum suppression from the real-time regime [19]. These developments concern assignment, query construction, and efficiency, and leave unchanged the property ReMiss relies on, namely that the detector exposes a final set of boxes, labels, and scores. DINO is therefore used here as the representative of this family, not as its most recent member.

These detector families differ in architecture, candidate generation, matching, and optimization. Two-stage and one-stage detectors produce final box sets after score thresholding, non-maximum suppression, and top-k filtering, whereas DETR-style detectors expose final boxes, labels, and scores after set-prediction training and matching. ReMiss does not attempt to make these internals identical. Instead, it observes the final prediction interface used for evaluation and downstream use. In this sense, ReMiss is not a new detector head, backbone, query mechanism, or loss objective; it is a data scheduling procedure attached to detectors that provide final boxes, labels, and scores.

2.3 Difficulty-Aware Training and Data Scheduling

Handling difficult and imbalanced samples has long been central to detector training. Online hard example mining selects high-loss proposals during region-based detector training to focus optimization on more informative examples [8]. Focal Loss down-weights well-classified examples so that dense one-stage detectors are not dominated by large numbers of easy negatives [9]. Class-Balanced Loss uses the effective number of samples for reweighting in long-tailed distributions [10], and LVIS emphasizes the long-tailed visual recognition setting where rare categories have few training examples [20]. Gradient Harmonizing

Mechanism and Libra R-CNN address detector-training imbalance through gradient-distribution balancing and through sample-, feature-, and objective-level balancing [21,22]. PISA focuses training on prime samples under the premise that not all samples contribute equally to mAP [23].

These methods share an important motivation with ReMiss: detector training benefits from emphasizing difficult cases. The contrast is that conventional hard-example and imbalance-aware methods usually define difficulty through losses, gradients, proposals, class frequencies, mini-batch composition, or estimated sample contribution. These signals act inside the optimization process, whereas ReMiss observes the detector after final post-processing and matching. It therefore preserves a different kind of difficulty signal: recurrence across mining snapshots for annotation instances that remain unmatched at the final prediction interface. ReMiss is complementary to hard example mining rather than a replacement for it.

Curriculum learning and self-paced learning are related because they adjust sample order or presentation frequency [24,25]. Example forgetting and dataset cartography further show that model behavior observed during training can reveal which examples are unstable, easy, or ambiguous [26,27]. These works motivate using training dynamics as data signals, but their typical unit of analysis is the example or classification behavior rather than a post-processed detection miss for a specific ground-truth box. The reported evaluations are correspondingly on classification benchmarks, and the intervention runs in the opposite direction to replay: forgetting statistics are used to *remove* examples rather than to reschedule them [26].

A more recent data-centric line makes the selection itself the object of study, pruning training data by difficulty scores balanced against coverage of the data distribution [28–32]. This line is close to ReMiss in spirit, since both intervene on the data rather than on the objective. Its decision variable is which examples to keep in a reduced dataset, whereas ReMiss keeps the dataset fixed and decides how often to re-expose images. Its scores are also attached to whole examples and recomputed at each selection step instead of accumulated, so no record persists of which annotated object the detector failed to produce.

Within detection specifically, coreset selection has been adapted to the fact that one image carries many annotated objects of differing value [33], and active-learning work selects images for annotation under uncertainty and label-noise criteria [34]. These are the closest existing counterparts, but their image score is a property of the current model state, not a history, and both target annotation cost or dataset size, not re-exposure within a fully labeled dataset.

Stated in terms of what each family retains and what it discards, the gap becomes concrete. Curriculum and self-paced methods keep no record of which objects remained undetected once an example has been consumed. Loss-, gradient-, and proposal-level methods retain a difficulty value only for the update in which it is computed, and define it before post-processing, so an instance suppressed by non-maximum suppression need not register as difficult. Forgetting-based analyses and dataset cartography do retain a temporal record, the closest prior signal to ours, but it describes classification behavior for an example and does not identify which object was lost. What ReMiss retains is exactly the quantity these methods discard: for each annotated instance, whether the detector’s own final prediction set explained it, accumulated over mining snapshots.

Active learning and subset selection are related through informative sample selection, but they usually target annotation acquisition from an unlabeled pool or select a labeled subset for training efficiency [35,36]. From a data-centric AI perspective, model behavior is affected not only by architecture but also by training-data construction, quality, and use [37]. ReMiss assumes a fixed labeled detection dataset. Its scheduling signal is not annotation utility or global sample representativeness, but the history of ground-truth instances that repeatedly remain unmatched under the detector’s own final prediction interface.

The difference is one of operating regime, and it is visible in the protocols these methods report. Active learning for detection is evaluated while the labeled budget is still growing [36,38], and an acquisition function has no residual definition once every image is labeled; subset selection runs in the same direction [33]. ReMiss operates where those methods terminate, with the full labeled split already in use, and reallocates exposure within it rather than choosing what to label or what to keep.

2.4 Replay and Memory-Guided Training

Replay and memory-guided training are broad strategies that re-expose informative samples to improve learning efficiency or stability. Prioritized Experience Replay reuses reinforcement-learning transitions according to importance within a replay memory [39]. iCaRL uses exemplar memory for class-incremental learning to mitigate forgetting of previous classes while learning new ones [40]. These studies show that memory or replay buffers can act not only as storage, but also as training signals that determine which samples are revisited.

In object detection, replay and memory mechanisms have been studied most often under continual or incremental detection, where new classes or tasks arrive over time and the model must preserve performance on previously learned categories. Early incremental object detectors use distillation to reduce catastrophic forgetting when old training data are unavailable [41]. Later studies analyze detection-specific complications such as missing annotations and background conflicts [42], and adapt distillation and exemplar replay to transformer-based detectors [43]. Continual object detection surveys therefore frame replay around stability–plasticity trade-offs, task boundaries, old/new class balance, and annotation incompleteness [44].

Replay-oriented detection methods also differ in what they store and revisit. RODEO uses compressed scene-level representations for online object detection [45], while Augmented Box Replay stores foreground boxes to address foreground shift in incremental detection [46]. Other incremental detection methods use pseudo-object replay, mining, or diffusion-based generative replay to reduce forgetting without retaining all original data [47,48]. These approaches treat replay partly as a memory-compression problem under continual or incremental learning constraints.

Although ReMiss uses the term replay, it does not adopt the continual-learning or reinforcement-learning setting. It does not address task sequences, old-class retention, new-class plasticity, policy learning, or compressed exemplar storage. MissBank stores lightweight annotation-instance records, and Hard Replay revisits the original full images that contain currently missed objects. ReMiss replay is therefore better viewed as memory-guided data scheduling within a fixed supervised detection task: its contribution is the conversion of missed-object history observed at the final prediction interface into next-epoch sampling decisions.

The same retain-and-discard reading separates ReMiss from the replay literature. Prioritized experience replay retains a priority-weighted buffer, but it is defined over reinforcement-learning transitions whose temporal-difference error has no counterpart in supervised detection. Exemplar- and distillation-based incremental detectors retain a compressed subset of previously seen data, protecting knowledge that was acquired and is at risk of being overwritten when the task distribution shifts. Our setting contains no task sequence and no distribution shift: the dataset is fixed and fully labeled throughout training, and the objects ReMiss targets were never acquired in the first place, so the failure is persistent non-acquisition, not forgetting. This is why the memory it maintains is an instance-level miss record, not an image buffer.

2.5 Positioning of ReMiss

Table 1 summarizes the preceding comparison along three axes: where the difficulty signal is defined, what unit the signal is attached to, and where the method intervenes in training. Under those axes ReMiss is

the only entry that combines a post-processed final-prediction signal, a ground-truth instance miss history, and epoch-level data scheduling while leaving the detector architecture and the loss function unchanged.

Table 1: Positioning of ReMiss relative to difficulty-aware training, curriculum, and replay methods. ReMiss uniquely combines a post-processed final-prediction signal, an instance-level miss-history unit, and epoch-level data scheduling without modifying detector architecture or loss.

Method	Signal Location	Signal Unit	Recurrence History	Intervention Point	Arch/Loss Change
OHEM [8]	Loss (proposal)	Proposal	No	Mini-batch selection	Loss
Focal Loss [9]	Loss (dense)	Anchor	No	Loss reweighting	Loss
GHM [21]	Gradient density	Anchor/proposal	No	Loss reweighting	Loss
Libra R-CNN [22]	Multi-level imbalance	Proposal	No	Training pipeline	Arch + Loss
PISA [23]	Est. mAP contribution	Proposal	No	Loss reweighting	Loss
Class-Balanced [10]	Class frequency	Class	No	Loss reweighting	Loss
Curriculum/SPL [24,25]	Sample difficulty	Example	Ordering only	Presentation order	None
Forgetting/Cartography [26,27]	Training dynamics	Example	Yes	Data analysis/selection	None
Data pruning/Coreset [28–30]	Difficulty + coverage	Example	No	Dataset subset selection	None
Coreset for detection [33]	Aggregated object value	Image	No	Dataset subset selection	None
Active learning [34–36]	Uncertainty/loss estimate	Image	No	Annotation acquisition	None
Prioritized Replay [39]	TD-error	RL transition	Yes (buffer)	Replay sampling	(RL setting)
iCaRL/Incremental [40,41]	Class/task boundary	Exemplar	Yes (memory)	Replay sampling	Distillation
ReMiss (ours)	Final prediction interface	GT-instance miss history	Yes	Epoch-level data schedule	None

Two consequences follow. First, the comparison is one of complementarity rather than substitution: because ReMiss does not modify the objective, it can be applied on top of loss-level hard example mining or reweighting, for which [Section 4.7](#) reports the evidence the evaluation carries. Second, the distinguishing property is the combination of the three axes, not any single column of [Table 1](#); individual entries share one or two axes with ReMiss, but none shares all three.

3 Method

3.1 Overview

ReMiss is a training-time framework that converts recurrent missed detections observed through a detector’s final prediction interface into an epoch-level data scheduling signal. Let the training dataset be $\mathcal{D} = \{(x_i, G_i)\}_{i=1}^N$, where x_i is an image and G_i is the set of ground-truth instances in image i . Each ground-truth instance $g \in G_i$ has a bounding box b_g and class label c_g . After detector-specific post-processing, the detector wrapper exposes a final prediction set $P_i = f_\theta(x_i)$, where each prediction $p \in P_i$ has bounding box b_p , class label c_p , and confidence score s_p . We refer to this wrapper output as the final prediction interface. ReMiss observes only this interface: MissBank does not inspect raw logits, candidates before final post-processing, intermediate features, proposals, queries, or loss terms.

The training workflow has three stages. First, the detector is trained on the base training loader under the standard supervised detection objective. Second, at a scheduled mining stage, the detector is switched to inference mode, gradient computation is disabled, and the full base training set is scanned to compare P_i with G_i . We use three distinct terms for offline mining: a mining stage is the configured scheduler point at which mining is triggered, a mining pass is the no-gradient inference scan of the base training loader at that stage, and a mining snapshot is the MissBank state immediately after that pass. The detected/missed state and matching diagnostics for each ground-truth instance are stored in MissBank. Third, before the next training epoch begins, Hard Replay selects images containing ground-truth instances that are currently missed in the latest MissBank snapshot, computes priority from miss-count history and matching gaps, and adds full-image replay samples to the training stream under an epoch-level replay budget. These replay samples are additional image exposures in the next epoch’s data stream rather than replacements for base samples. Because this study uses only the offline mining path, MissBank is not updated inside optimizer steps, and neither the detector architecture nor the training loss is changed.

[Fig. 1](#) summarizes the epoch-level loop of base training, no-gradient offline mining, MissBank update, and Hard Replay scheduling.

Algorithm 1 describes the same loop as an epoch-level training procedure. In the method description, epoch denotes the ReMiss scheduling unit associated with the base-loader pass used to define N_{load} , mining snapshots, and replay budgets. Detector optimization may still be configured in optimizer iterations or in epochs; ReMiss does not convert learning-rate milestones or stopping criteria between these units. When replay samples are appended to the data stream, the resulting number of consumed mini-batches follows the loader protocol used by the corresponding detector run. The matching rule, eligibility rule, and priority score referenced in the algorithm are defined in the following subsections.

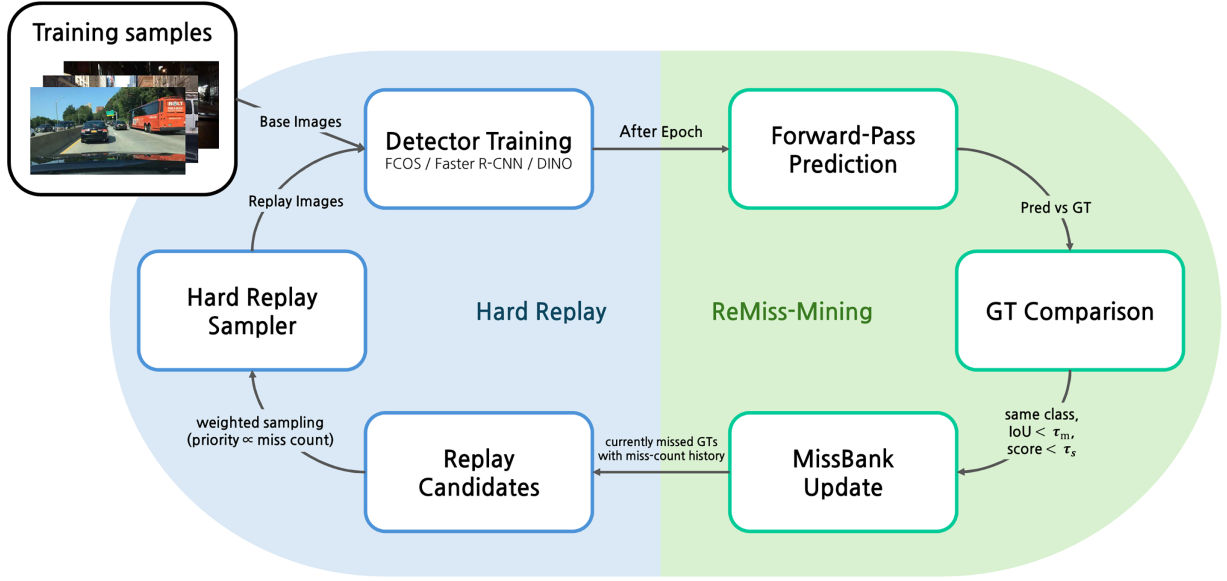


Figure 1: Epoch-level ReMiss loop. After training with base and replay images, no-gradient mining evaluates the base data, updates MissBank from final prediction–ground-truth matching, and selects replay candidates for the next epoch.

Algorithm 1: ReMiss epoch-level training loop

1. Initialize detector f_θ and an empty MissBank $\mathcal{B}$ indexed by annotation-level ground-truth instance keys.
 2. For each training epoch e , train f_θ with the current data stream. The stream contains the base training data and, after replay candidates become available, the scheduled full-image replay samples.
 3. If e is a mining epoch, freeze f_θ for no-gradient inference and evaluate every base training image x_i . Compare the final prediction set P_i^e with G_i using the matching rule in Eq. (1); update $\mathcal{B}$ for every ground-truth instance.
 4. Before epoch $e + 1$, select eligible missed instances from the latest MissBank snapshot, aggregate them into image-level replay candidates, compute replay priorities with Eq. (2), and sample up to the replay budget in Eq. (3) subject to the exposure cap.
 5. Interleave the selected full-image replay samples into the next epoch's data stream while keeping the detector forward path and loss unchanged.
-

If MissBank is empty, or if no ground-truth instance satisfies the replay eligibility rule, the current data stream is identical to the base training loader. ReMiss inserts additional replay samples only after eligible replay targets have been observed in the latest mining snapshot.

3.2 MissBank and Offline Miss Mining

MissBank $\mathcal{B}$ is an instance-level memory indexed by annotation-level ground-truth keys rather than by images. For a ground-truth instance key g , its missed-object history is the ordered record, over completed mining snapshots, of its detected/missed state and derived miss-count statistics. This is annotation-instance history, not temporal tracking of the same physical object across video frames. For each ground-truth instance g , MissBank stores static identifiers and annotation fields: image identifier, ground-truth identifier, class label, and bounding box. It also stores the latest observed epoch, current missed state, consecutive miss count, cumulative seen count, cumulative missed count, historical maximum consecutive miss count, and confidence-score and IoU diagnostics for a representative same-class candidate chosen in the latest mining

pass. This allows ReMiss to track whether an annotation-level instance, defined by image and ground-truth identifiers, remains repeatedly missed, rather than globally reselecting “hard images.”

When a converted dataset does not provide persistent annotation identifiers, we use a tuple of image identifier, class label, and the file-order index of the annotation within the same image in the fixed converted annotation file. This key is defined before training and is used only to refer consistently to the same annotated object across offline mining stages.

Offline miss mining is performed on the base training data $\mathcal{D}$, not on the replay-augmented stream. This design keeps each MissBank snapshot tied to the same base-data distribution rather than to the distribution induced by replay. During the mining pass at stage e , a ground-truth instance $g \in G_i$ in image i is marked as detected when a final same-class prediction satisfies both the confidence-score threshold and the ground-truth IoU matching condition:

$$\text{detected}_e(g) = \mathbf{1} \left[\begin{array}{l} \exists p \in P_i^e \text{ such that } c_p = c_g, \\ s_p \geq \tau_s \text{ and } \text{IoU}(b_p, b_g) \geq \tau_m \end{array} \right], \quad (1)$$

where P_i^e is the final prediction set for image i in the mining pass at epoch e , τ_s is the confidence threshold used for MissBank matching, and τ_m is the IoU threshold used by MissBank matching. The threshold τ_m does not replace the detector’s internal non-maximum suppression behavior. It is the MissBank matching threshold used to decide whether a final prediction explains a particular ground-truth instance. Eq. (1) also does not replace the one-to-one assignment used in official COCO-style evaluation [5–7]. We evaluate Eq. (1) independently for each ground-truth instance, using the rule as a training-time diagnostic for recording which instances remain repeatedly unmatched under the detector’s final prediction interface.

Independent evaluation means that in a crowded scene one prediction can satisfy the condition for several neighboring ground-truth boxes, which a one-to-one assignment would not permit. The deviation is conservative in the direction that matters for scheduling: *more* instances are marked detected than a one-to-one rule would mark, so misses are under-recorded and the eligible replay pool shrinks. The independent rule can therefore only omit a genuinely missed instance, never inject a falsely missed one, and since no AP value in this paper is computed with Eq. (1), the consequence is a smaller replay pool, not a distorted evaluation. A duplicate match moreover changes the MissBank state only for an instance that has *no* individual matching prediction of its own; when such a prediction exists, Eq. (1) is satisfied by it and the duplicate is immaterial. How often the situation is geometrically admissible at all is bounded by the annotations alone: testing every same-class pair in the four training splits, the fraction of annotated instances belonging to at least one pair that admits a common box at $\text{IoU} \geq \tau_m$ ranges from 11.1% on BDD100K to 35.6% on Cityscapes, which is a ceiling on what is admissible rather than an observed duplicate-match rate.

The divergence from a one-to-one assignment is correspondingly one-signed: such a rule can only enlarge the missed-GT pool, never introduce a spurious missed instance. Measurement shows the divergence to be small. Reconstructing the missed-GT pool under greedy and under Hungarian one-to-one assignment from the same stored predictions enlarges it on every dataset–detector pair, as the argument requires. The largest increase over any pair is 4.51 percentage points of annotated instances under greedy matching and 3.54 points under Hungarian matching. Under an identical replay budget the resulting replay image sets overlap those selected under Eq. (1) with a Jaccard index of 0.9227 or above. The replay stream is the only channel through which the matching rule can affect training, and it is left largely intact.

Unless stated otherwise, τ_s is set equal to the confidence filtering configuration used by the detector wrapper to create the final prediction set. MissBank therefore applies its matching rule to the final predictions exposed by the wrapper, without searching raw logits, low-score proposals, or detector-internal candidates.

The score condition in Eq. (1) is retained so that the MissBank matching rule is explicit. Although the formulation permits a separate confidence threshold, this study ties τ_s to the wrapper score filter reported in the implementation details. Under this setting, the confidence-gap term defined later in Eq. (2) is primarily active when the final prediction set contains no same-class prediction. In that case, MissBank records $\hat{s}_g = 0$.

If $\text{detected}_e(g) = 1$, MissBank marks the current missed state of the instance as false and resets its consecutive miss count to zero. Otherwise, it marks the current missed state as true and updates the consecutive miss count, cumulative missed count, and historical maximum consecutive miss count. At the same time, MissBank selects a representative diagnostic candidate p_g^* among same-class final predictions that is closest to satisfying the joint score-IoU condition in Eq. (1). When a same-class final prediction exists, $p_g^* = \arg \max_{p \in P_i^e: c_p = c_g} \min(\text{IoU}(b_p, b_g)/\tau_m, s_p/\tau_s)$, and MissBank records $\hat{u}_g = \text{IoU}(b_{p_g^*}, b_g)$ and $\hat{s}_g = s_{p_g^*}$. If no same-class final prediction exists, MissBank records $\hat{u}_g = 0$ and $\hat{s}_g = 0$. These diagnostics distinguish instances with no same-class final candidate from instances for which a same-class final candidate exists but no single prediction simultaneously satisfies both the score and localization conditions.

3.3 Replay Target Eligibility and Priority

Hard Replay does not replay every instance that has ever been missed. A ground-truth instance g is an eligible replay target for the next epoch only if it is marked missed in the latest mining snapshot. Persistence affects priority, not eligibility: a newly missed instance may enter the replay pool with low priority, while an instance with a long previous miss history is excluded if it is detected in the latest snapshot. Rather than filtering recurrent misses with a separate fixed threshold, ReMiss directly incorporates consecutive, cumulative, and historical-maximum miss counts into Eq. (2). The priority score is therefore a scheduling heuristic over eligible replay targets, not a loss term or a separate detection objective.

The priority for an eligible target combines miss persistence and matching gaps. In our implementation, we use the following fixed heuristic:

$$\pi(g) = m_g + 0.25M_g + 0.1m_g^{\max} + \max(0, \tau_m - \hat{u}_g) + 0.25 \max(0, \tau_s - \hat{s}_g), \quad (2)$$

where m_g is the current consecutive miss count, M_g is the cumulative missed count, and $m_g^{\max}$ is the historical maximum of the consecutive miss count across mining stages. The values $\hat{u}_g$ and $\hat{s}_g$ are the IoU and confidence score of the representative same-class diagnostic candidate p_g^* selected in the latest mining pass. The two gap terms act as $[\tau_m - \hat{u}_g]_+$ and $[\tau_s - \hat{s}_g]_+$; if the representative candidate already exceeds the localization or confidence threshold, the corresponding gap does not increase the priority. Under the wrapper-filtered final prediction sets used in this study, the confidence gap is mainly active when no same-class final prediction is present; it is retained in the formulation for implementations that expose lower-score final candidates. Eq. (2) is not proposed as an optimal priority function. It is a fixed scheduling heuristic that adapts the role of importance signals used in prioritized replay and hard-sample selection to detector-output history [8,23,39]. The count terms intentionally emphasize recurrent misses, while the bounded gap terms provide auxiliary information about how close the best final same-class prediction is to the joint matching condition in Eq. (1).

The coefficients in Eq. (2) are not tuned values. They are chosen so that the count terms, unbounded non-negative integers weighted $1.0 > 0.25 > 0.1$, dominate the two gap terms. The gap terms are bounded by construction and, with $\tau_m = 0.5$ and $\tau_s = 0.1$ (Section 4.3), contribute at most 0.525 in total, below the increment of 1.0 produced by a single additional consecutive miss. They therefore act as tie-breakers among instances with identical miss statistics rather than reordering instances across consecutive-miss levels, a

structural property holding for any weights with $a_u \tau_m + a_s \tau_s < 1$. The same coefficient set is used for all four datasets and all three detectors, and the component-level ablation in [Section 4.6](#) measures what each term contributes when removed.

3.4 Hard Replay Scheduling

An image i becomes a replay candidate if it contains at least one eligible target. The image-level priority Π_i is computed by summing $\pi(g)$ over eligible targets in that image. This aggregation increases the chance of replaying an image that contains multiple currently missed targets, especially targets with larger missed-object history priorities. Images without eligible targets are not included in the Hard Replay candidate pool. The replay unit is therefore the full image, while eligible ground-truth instances determine the image-level priority. Summation is used rather than a maximum or an average because it is the aggregation consistent with this replay unit: a single replay exposure trains on every eligible target the image contains, so the value of replaying an image grows with the number of currently missed targets in it. A maximum would discard that multiplicity, and an average would penalize crowded scenes, which is the opposite of the intended behavior in dense driving imagery. The multiplicity is not marginal where it matters: the mean number of eligible missed targets per replayed image, obtained from the replay diagnostics reported in [Section 4.6](#), ranges from 1.00 on KITTI, where the three aggregations nearly coincide, to above 7 in every Cityscapes and BDD100K CNN configuration. A head-to-head retraining comparison of the three aggregations is left as future work ([Section 4.7](#)).

For a replay ratio ρ , ReMiss defines the requested epoch-level replay budget by using the number of base samples consumed by the base training loader in one epoch, denoted N_{load} :

$$B_\rho = \lfloor \rho N_{\text{load}} \rfloor. \quad (3)$$

Here, N_{load} is not necessarily the nominal number of images in the original dataset. It is the number of base samples actually consumed during one epoch by the base training loader before Hard Replay is added in a given dataset-detector run. In a standard one-pass loader, N_{load} equals the number of base training images, but detector-specific preprocessing, resolution, batch/device configuration, and loader construction can make the epoch sample count differ from the nominal image count. Thus, ρ and B_ρ define a replay budget relative to each run's base-loader epoch, not a raw dataset-size ratio across detectors.

The scheduler samples candidate images according to image-level priority and adds the selected full-image replay samples to the next epoch's data stream. These replay samples are additional image exposures relative to the base loader rather than replacements for base images: an HR epoch contains the base-loader samples plus up to B_ρ replay samples, subject to candidate availability and the exposure cap. With the same batch size, consuming these additional samples can increase the number of mini-batch optimizer updates relative to the training-only baseline under an epoch-based loader. Under a fixed-iteration schedule the total number of optimizer updates is unchanged, and replay instead alters which samples those updates consume.

For a candidate pool $\mathcal{C}$, we use the image-level priority distribution $q_i = \Pi_i / \sum_{j \in \mathcal{C}} \Pi_j$. We also apply an instance-level exposure cap so that a small number of high-priority missed objects or images do not dominate the replay stream; in all experiments, this cap is four replay exposures per ground-truth instance per epoch. During replay scheduling, the scheduler maintains an exposure counter for each eligible replay target. Candidate images are sampled from the image-level priority distribution, and an accepted image contributes one full-image replay sample. After acceptance, the exposure counters of the eligible replay targets contained in that image are incremented. A candidate is admissible only while accepting it does not violate the per-instance exposure cap for its eligible replay targets. Sampling stops when the requested budget

is filled or no admissible candidate remains, so the actual replay count B_{actual} can be smaller than B_{ρ} . We therefore report both the requested replay ratio ρ and the effective replay ratio $B_{\text{actual}}/N_{\text{load}}$.

Replay samples use the same annotations, preprocessing, and augmentation pipeline as base images; missed objects are neither cropped nor assigned separate labels. Full-image replay preserves context, scale, occlusion, and clutter around missed instances [4,49,50].

3.5 Detector Integration through a Common Prediction Interface

ReMiss is integrated outside the detector-specific forward path. Each detector wrapper provides a normalized final prediction set consisting of boxes, labels, and scores after final post-processing, and MissBank applies the matching rule in Eq. (1) to this common output format. Hard Replay uses only missed-object histories stored in MissBank and data loader scheduling, rather than detector-internal tensors or loss values. ReMiss is therefore applied to FCOS, Faster R-CNN, and DINO in this study without changing their detector architectures, standard inference outputs, or training losses [13,14,17].

This integration does not imply universal compatibility with every detector. ReMiss requires an interface that exposes final boxes, labels, and scores, and a training data stream into which full-image replay samples can be added. For detectors that satisfy these requirements, MissBank matching and the Hard Replay scheduler can be kept separate from detector internals. ReMiss is therefore positioned as a training procedure that converts missed-object history observed through the final prediction interface into a data-centric scheduling signal.

Fig. 2 combines the matching rule, MissBank state, priority scoring, replay sampling, and detector integration into a detailed architecture view.

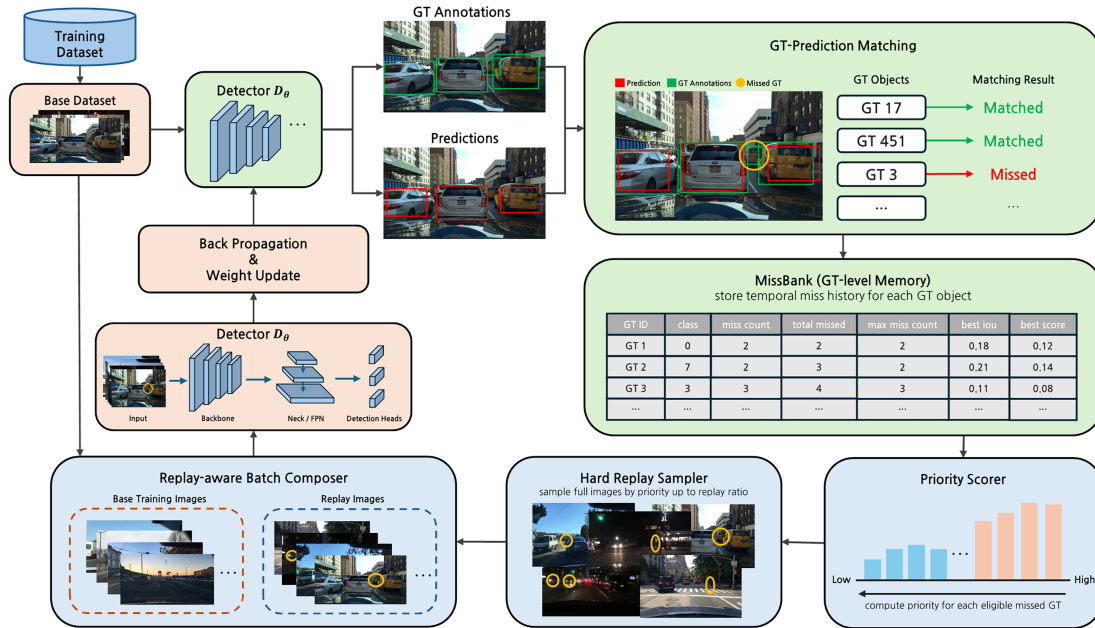


Figure 2: Detailed ReMiss architecture: training path, prediction-ground-truth matching, ground-truth-instance-level MissBank memory, priority scoring, and Hard Replay sampling. Missed-object history is passed to data-loader scheduling without changing the forward path or loss.

4 Experiments

4.1 Datasets

We evaluate ReMiss on four COCO-format driving-scene object detection datasets: KITTI, BDD100K, Cityscapes, and nuImages [1–3,5,11,12]. All experiments use only 2D image bounding-box annotations. For KITTI, we use a fixed internal train/validation split derived from the labeled training set and apply the same split to all detectors. BDD100K uses 69,863 training images and 10,000 validation images at 1280×720 resolution. Cityscapes uses 2975 training images and 500 validation images at 2048×1024 resolution; for DINO, the same split is preprocessed to 1024×512 . nuImages uses 67,279 training images and 16,445 validation images at 1600×900 resolution; for DINO, the same split is preprocessed to 1280×720 .

All detector configurations use an aspect-ratio-preserving transform with a minimum size of 800 and a maximum size of 1333. The resulting model inputs are approximately $1333 \times 402\text{--}404$ for KITTI, depending on the original frame size; approximately 1333×750 for BDD100K and nuImages; and approximately 1333×667 for Cityscapes. During COCO-format conversion, dataset-specific class mappings, image IDs, annotation IDs, bounding-box coordinates, areas, and category IDs are fixed so that MissBank instance keys and COCO-style evaluation use the same annotation ordering.

4.2 Experimental Setup

For each dataset, we evaluate FCOS, Faster R-CNN, and DINO [13,14,17]. FCOS represents a one-stage anchor-free detector, Faster R-CNN represents a two-stage detector, and DINO represents a transformer-based detector. The performance comparison baseline is the No-HR MissBank diagnostic baseline (hereafter, No-HR baseline), where offline MissBank mining is enabled but Hard Replay is disabled. In this setting, offline MissBank mining is used only to collect missed-object statistics: it evaluates the base training data without gradient updates and does not change the training stream, mini-batch contents, image exposure, or optimizer steps. The No-HR baseline is therefore designed to leave training updates unchanged and serves as the plain-baseline performance reference. Hard Replay variants keep the detector configuration fixed and use replay ratios of 2.5%, 5.0%, and 7.5%, where the ratio defines an epoch-level budget for additional replay image exposures. Runtime overhead is reported separately against a training-only baseline that excludes diagnostic MissBank mining. These three detectors are chosen to span distinct training regimes rather than to sample several members of one family. FCOS and Faster R-CNN are trained under fixed-iteration schedules with anchor-free and anchor-based assignment respectively and produce final boxes after score thresholding and non-maximum suppression, whereas DINO is a query-based set-prediction detector trained under an epoch-based schedule with Hungarian matching and without non-maximum suppression. Because ReMiss reads only the final prediction interface, the property under test is whether one shared configuration transfers across these differing training regimes.

All ReMiss experiments use only the offline mining path. After the configured epoch, MissBank is updated by evaluating the base training data through the detector's final prediction interface. Before the next training epoch begins, replay candidates are selected from the current MissBank snapshot. This protocol keeps the detector architecture, loss function, and inference procedure identical between the baseline and Hard Replay variants. For each dataset–detector pair, the No-HR baseline and all Hard Replay ratios use the same validation-checkpoint rule. The Hard Replay ratio with the highest validation $\text{mAP}_{50:95}$ is then used for the main comparison, MissBank diagnostic comparison, and runtime summary, while the full replay-ratio sweep is reported separately.

4.3 Implementation Details

FCOS and Faster R-CNN use ResNet-50 backbones pretrained on ImageNet-1K with three trainable backbone layers. Both CNN detectors are trained with SGD; the learning rate is 0.01 for FCOS and 0.02 for Faster R-CNN. On KITTI, BDD100K, and nuImages, the two CNN detectors are trained for 90K iterations with learning-rate milestones at 60K and 80K iterations. On Cityscapes, the CNN detectors use a compact 5K-iteration schedule with milestones at 3.5K and 4K iterations, reflecting the smaller training split and the validation-checkpoint protocol used for that dataset. The batch size for both CNN detectors is 16.

DINO is initialized from the public COCO-trained DINO-4scale R50 checkpoint released in the DINO repository and trained with AdamW at a learning rate of 1×10^{-4} . Training lasts 12 epochs on KITTI and Cityscapes and 24 epochs on BDD100K and nuImages, with a global batch size of 16 for all datasets and a single learning-rate drop after epoch 11 and epoch 20, respectively. All training is performed on an NVIDIA GeForce RTX 5090 GPU. Runtime is measured with the same GPU, batch size, data loader settings, and validation-selected checkpoint protocol. Checkpoint saving, logging, and validation evaluation time are excluded from per-epoch train/mining wall-clock measurements.

MissBank and Hard Replay use the same hyperparameters for all datasets. Offline-mining target selection uses a score threshold of 0.1 and an IoU threshold of 0.5, and the replay stage caps each ground-truth instance at four replay exposures per epoch. The score threshold of 0.1 and IoU threshold of 0.5 correspond to τ_s and τ_m in Eq. (1); they define whether a final same-class prediction explains the ground-truth instance, i.e., whether $\text{detected}_e(g) = 1$. The replay exposure cap is separate from this detection rule and only prevents a small number of hard ground-truth instances from occupying too much of the replay stream. The configured mining epochs are treated as part of the run configuration rather than as adaptive decisions made during training. For reproducibility, each run records the mining-epoch list, MissBank snapshot identifier, replay candidate pool, requested replay budget, actual replay count, selected replay image identifiers, best-validation checkpoint, evaluation log, and runtime log.

Every configuration reported in this paper is trained three times with the fixed random seeds 41, 42, and 43, and all reported metric values are means over those three independent runs, which differ only in the random seed. Best-checkpoint selection by validation $\text{mAP}_{50:95}$ is performed separately within each run rather than shared across runs, so the reported mean averages three independently selected checkpoints.

This shared configuration is not the result of per-dataset tuning, and most of its entries are not free hyperparameters. The matching threshold $\tau_m = 0.5$ follows the standard COCO AP_{50} convention, and τ_s is tied to the detector wrapper's score filter rather than varied independently. The four-exposure cap is read as a sensitivity result through the replay-budget diagnostics in Section 4.6. The mining schedule is fixed as part of the run configuration and is not adapted during training, so it cannot leak validation information into the replay stream; a sweep over mining frequency and placement is left as future work. A single shared configuration yields improvements in all 36 dataset–detector–ratio runs.

4.4 Evaluation Metrics

We use COCO-style bounding-box average precision [5]. $\text{mAP}_{50:95}$ denotes COCO bounding-box AP averaged over IoU thresholds from 0.50 to 0.95 in steps of 0.05, computed on the validation split with area = all and maxDets = 100. When this section uses mAP without qualification, it refers to $\text{mAP}_{50:95}$. AP_{50} , AP_{75} , and average recall at maxDets = 100 (AR_{100}) are used as auxiliary diagnostic metrics. All AP, AR, their differences, and missed ground-truth (missed-GT) ratios are reported on the percentage scale, i.e., raw metric values multiplied by 100. Unless stated otherwise, each reported run uses the best checkpoint selected by validation $\text{mAP}_{50:95}$; MissBank missed-GT statistics are reported as diagnostics of the replay signal, not as objectives

or model-selection criteria. Unless stated otherwise, all AP and AR values, and all differences computed from them, are means over the three fixed seeds of [Section 4.3](#). The replay and MissBank diagnostics that are instance or image counts are instead reported from the seed-42 run, so that they remain integers referring to one concrete schedule. Significance is assessed across dataset–detector pairs rather than within a pair, since three seeds give a per-pair test negligible power while twelve repetitions would introduce an uncontrolled multiple-comparison problem. Two-sided sign tests and paired Wilcoxon signed-rank tests are therefore reported over the 12 dataset–detector pairs, and over the 36 dataset–detector–ratio runs where the full sweep is in question.

4.5 Results

[Table 2](#) compares the No-HR baseline with the Hard Replay result selected by validation $mAP_{50:95}$ among the three replay ratios for each dataset–detector pair. Under this validation-selected protocol, the selected Hard Replay setting for every reported pair obtains higher validation $mAP_{50:95}$ than the baseline. The largest selected gain appears for DINO on BDD100K, where validation mAP increases from 25.06 to 27.29. All three KITTI detectors also improve under their selected settings, although the selected replay ratio differs by architecture. On Cityscapes, the selected gains are more moderate than the largest DINO gains, highlighting variation with the density and persistence of the mined missed-object pool. For DINO on nuImages, HR-2.5% is selected and validation mAP increases from 27.11 to 29.31. The replay-ratio sweep below analyzes how these validation-selected outcomes vary across replay budgets.

Table 2: Validation-selected main results. Baseline AP is the No-HR baseline; Best HR AP is selected by validation $mAP_{50:95}$ among the evaluated replay ratios for each dataset–detector pair. Both are three-seed mean $\pm$ standard deviation. The last two columns are selection-independent, read from the sweep in [Table 3](#): a ratio fixed at 5.0%, and the least favorable of the three evaluated ratios per pair.

Dataset	Detector	Baseline AP	Best HR AP	Δ AP	Best Ratio	Fixed-5.0% Δ AP	Worst-Ratio Δ AP
KITTI	FCOS	58.06 $\pm$ 0.21	59.55 $\pm$ 0.43	+1.49	2.5%	+1.01	+0.62
KITTI	Faster R-CNN	60.21 $\pm$ 0.28	62.19 $\pm$ 0.32	+1.98	5.0%	+1.98	+1.02
KITTI	DINO	70.99 $\pm$ 0.24	72.90 $\pm$ 0.38	+1.91	2.5%	+0.87	+0.24
BDD100K	FCOS	18.50 $\pm$ 0.18	19.86 $\pm$ 0.23	+1.36	2.5%	+1.33	+1.26
BDD100K	Faster R-CNN	16.85 $\pm$ 0.20	18.38 $\pm$ 0.21	+1.53	7.5%	+1.04	+0.99
BDD100K	DINO	25.06 $\pm$ 0.19	27.29 $\pm$ 0.24	+2.23	2.5%	+2.17	+1.54
Cityscapes	FCOS	24.50 $\pm$ 0.38	25.53 $\pm$ 0.41	+1.03	7.5%	+0.86	+0.22
Cityscapes	Faster R-CNN	30.10 $\pm$ 0.44	31.34 $\pm$ 0.48	+1.24	7.5%	+1.00	+0.61
Cityscapes	DINO	33.16 $\pm$ 0.37	34.52 $\pm$ 0.43	+1.36	7.5%	+0.99	+0.41
nuImages	FCOS	18.54 $\pm$ 0.25	19.86 $\pm$ 0.33	+1.32	7.5%	+1.21	+0.64
nuImages	Faster R-CNN	17.37 $\pm$ 0.22	18.48 $\pm$ 0.27	+1.11	7.5%	+0.64	+0.64
nuImages	DINO	27.11 $\pm$ 0.26	29.31 $\pm$ 0.29	+2.20	2.5%	+1.04	+0.60
Mean	–	–	–	+1.56	–	+1.18	+0.73

Across the 24 standard-deviation entries of [Table 2](#) the seed-to-seed dispersion ranges from 0.18 to 0.48 AP, and in every dataset–detector pair the selected gain exceeds the larger of the two standard deviations by a factor of at least 2.5. The Hard Replay arm is the more dispersed of the two in every pair, but by a margin at least six times smaller than the corresponding gain.

Because the replay ratio is chosen on the same validation split on which the result is reported, the validation-selected column is not an independent estimate of generalization, and the conclusion is stated below in a form that does not depend on that choice. In the complete sweep of [Table 3](#) all 36 configurations

improve over the No-HR baseline, with a minimum of +0.22 AP and a mean of +1.16 AP ($p \approx 2.9 \times 10^{-11}$ under both tests over the 36 runs). A ratio fixed in advance at any of the three evaluated values improves all 12 pairs, and 5.0% has the strongest worst case and is recommended as a single fixed rule. Taking for each pair the *least* favorable ratio still improves all 12. Ratio selection can therefore change the magnitude of the reported improvements but not their presence, and validation usage is otherwise symmetric between the arms (Section 4.2). A fully independent protocol would partition the labeled training data into training, ratio-selection, and evaluation subsets; since the test annotations of all four benchmarks are withheld, this would shrink the training split and render absolute AP values incomparable with published results, and it is left as future work.

Table 3: Hard Replay ratio ablation. AP is mAP_{50:95}, and Δ AP is relative to the No-HR baseline.

Dataset	Detector	HR Ratio	AP	AP ₅₀	AP ₇₅	AR ₁₀₀	Δ AP
KITTI	FCOS	2.5%	59.55	83.63	66.34	68.30	+1.49
KITTI	FCOS	5.0%	59.07	87.16	65.00	66.73	+1.01
KITTI	FCOS	7.5%	58.68	87.82	66.88	67.67	+0.62
KITTI	Faster R-CNN	2.5%	61.99	87.15	67.96	67.51	+1.78
KITTI	Faster R-CNN	5.0%	62.19	85.81	68.50	67.95	+1.98
KITTI	Faster R-CNN	7.5%	61.23	87.00	66.21	67.32	+1.02
KITTI	DINO	2.5%	72.90	93.23	84.05	83.03	+1.91
KITTI	DINO	5.0%	71.86	92.99	82.44	82.73	+0.87
KITTI	DINO	7.5%	71.23	93.17	83.55	82.88	+0.24
BDD100K	FCOS	2.5%	19.86	36.99	18.28	32.18	+1.36
BDD100K	FCOS	5.0%	19.83	36.98	18.12	32.12	+1.33
BDD100K	FCOS	7.5%	19.76	36.82	18.06	32.06	+1.26
BDD100K	Faster R-CNN	2.5%	17.84	35.36	16.28	28.42	+0.99
BDD100K	Faster R-CNN	5.0%	17.89	35.31	16.28	28.27	+1.04
BDD100K	Faster R-CNN	7.5%	18.38	35.38	16.14	28.28	+1.53
BDD100K	DINO	2.5%	27.29	46.42	26.98	53.07	+2.23
BDD100K	DINO	5.0%	27.23	46.81	26.97	50.82	+2.17
BDD100K	DINO	7.5%	26.60	45.95	26.43	52.43	+1.54
Cityscapes	FCOS	2.5%	24.72	50.41	22.59	39.07	+0.22
Cityscapes	FCOS	5.0%	25.36	49.92	23.13	38.80	+0.86
Cityscapes	FCOS	7.5%	25.53	49.86	22.63	39.37	+1.03
Cityscapes	Faster R-CNN	2.5%	30.71	56.94	28.24	41.15	+0.61
Cityscapes	Faster R-CNN	5.0%	31.10	56.99	28.54	41.80	+1.00
Cityscapes	Faster R-CNN	7.5%	31.34	57.58	29.24	42.00	+1.24
Cityscapes	DINO	2.5%	33.57	56.06	32.80	54.11	+0.41
Cityscapes	DINO	5.0%	34.15	56.74	32.41	54.89	+0.99
Cityscapes	DINO	7.5%	34.52	57.35	32.80	53.31	+1.36
nuImages	FCOS	2.5%	19.18	33.18	19.36	34.95	+0.64
nuImages	FCOS	5.0%	19.75	34.17	19.32	34.59	+1.21
nuImages	FCOS	7.5%	19.86	34.34	19.39	34.63	+1.32
nuImages	Faster R-CNN	2.5%	18.37	31.72	17.93	28.29	+1.00
nuImages	Faster R-CNN	5.0%	18.01	31.73	17.43	28.47	+0.64
nuImages	Faster R-CNN	7.5%	18.48	31.85	17.86	28.64	+1.11
nuImages	DINO	2.5%	29.31	45.31	30.81	58.05	+2.20
nuImages	DINO	5.0%	28.15	45.54	30.70	56.88	+1.04
nuImages	DINO	7.5%	27.71	44.92	30.23	58.87	+0.60

Table 4 reports replay controls matched by requested ratio, comparing ReMiss-HR with random image replay and uniform sampling from the missed-object candidate pool. It also defines a comparison setting matched by optimizer updates for DINO. The random replay control samples training images at the same requested replay ratio as the selected ReMiss-HR setting in [Table 2](#). The uniform-missed control uses the MissBank missed-object candidate pool but samples it uniformly, without the priority in [Eq. \(2\)](#). Because the controls in Panel A are matched by requested replay ratio, cases limited by the replay budget should be interpreted together with the effective-ratio diagnostics in [Table 5](#). Exposure-only denotes a DINO control with the same logged total number of added optimizer updates as ReMiss-HR but without MissBank-guided replay; in Panel B, Added Updates reports optimizer-update counts, while the ReMiss-HR and Exposure-only columns report validation Δ AP relative to the No-HR baseline.

Table 4: Mechanism decomposition of the replay signal. Δ AP values are relative to the No-HR baseline. Panel A separates the factors that could produce the gain, matched by requested replay ratio: added exposure alone (random), a difficulty signal without miss history (uniform-missed), and the full priority (ReMiss-HR). Panel B matches added optimizer updates for DINO. Δ AP values are three-seed means; Added Updates is from the seed-42 run ([Section 4.4](#)).

A. Decomposition by Replay Content, All Detectors					
Dataset	Detector	Selected HR Ratio	ReMiss-HR Δ AP	Random Δ AP	Uniform-Missed Δ AP
KITTI	FCOS	2.5%	+1.49	−0.06	+0.25
KITTI	Faster R-CNN	5.0%	+1.98	+0.23	+0.61
KITTI	DINO	2.5%	+1.91	+0.75	+1.22
BDD100K	FCOS	2.5%	+1.36	+0.07	+0.24
BDD100K	Faster R-CNN	7.5%	+1.53	+0.44	+1.02
BDD100K	DINO	2.5%	+2.23	+0.68	+1.99
Cityscapes	FCOS	7.5%	+1.03	−0.50	+0.31
Cityscapes	Faster R-CNN	7.5%	+1.24	+0.15	+0.48
Cityscapes	DINO	7.5%	+1.36	+0.11	+1.33
nuImages	FCOS	7.5%	+1.32	−0.09	+0.23
nuImages	Faster R-CNN	7.5%	+1.11	−0.02	+0.53
nuImages	DINO	2.5%	+2.20	+1.32	+2.01
Mean	–	–	+1.56	+0.26	+0.85
B. Optimizer-Update-Matched Control, DINO Only					
Dataset	Detector	Selected HR Ratio	Added Updates	ReMiss-HR Δ AP	Exposure-Only Δ AP
KITTI	DINO	2.5%	+72	+1.91	+0.49
BDD100K	DINO	2.5%	+2052	+2.23	+0.31
Cityscapes	DINO	7.5%	+72	+1.36	−0.20
nuImages	DINO	2.5%	+2014	+2.20	+0.16
Mean	–	–	–	+1.93	+0.19

Table 5: Replay-budget utilization across Hard Replay runs. Effective Ratio is Actual Replay divided by the base-loader epoch size; Budget Fill is Actual Replay divided by Requested Replay. All entries are counts and ratios from the seed-42 run, not seed averages ([Section 4.4](#)).

Dataset	Detector	HR Ratio	Effective Ratio	Requested Replay	Actual Replay	Budget Fill	Unique Images	Active Missed GT	Exposure/Missed GT
KITTI	FCOS	2.5%	2.5%	149	149	1.00	81	92	1.772
KITTI	FCOS	5.0%	5.0%	299	299	1.00	142	177	1.977
KITTI	FCOS	7.5%	7.5%	448	448	1.00	155	162	3.037

(Continued)

Table 5 (continued)

Dataset	Detector	HR Ratio	Effective Ratio	Requested Replay	Actual Replay	Budget Fill	Unique Images	Active Missed GT	Exposure/ Missed GT
KITTI	Faster R-CNN	2.5%	0.7%	149	40	0.27	10	10	4.000
KITTI	Faster R-CNN	5.0%	0.7%	299	40	0.13	10	10	4.000
KITTI	Faster R-CNN	7.5%	1.9%	448	112	0.25	27	28	4.000
KITTI	DINO	2.5%	2.5%	149	149	1.00	48	50	3.240
KITTI	DINO	5.0%	5.0%	299	299	1.00	97	104	3.298
KITTI	DINO	7.5%	1.9%	448	116	0.26	29	29	4.000
BDD100K	FCOS	2.5%	2.5%	1746	1746	1.00	1702	2,96,372	0.049
BDD100K	FCOS	5.0%	5.0%	3493	3493	1.00	3310	2,95,129	0.098
BDD100K	FCOS	7.5%	7.5%	5239	5239	1.00	4883	2,93,172	0.141
BDD100K	Faster R-CNN	2.5%	2.5%	1746	1746	1.00	1710	2,17,110	0.057
BDD100K	Faster R-CNN	5.0%	5.0%	3493	3493	1.00	3281	2,16,840	0.115
BDD100K	Faster R-CNN	7.5%	7.5%	5239	5239	1.00	4838	2,19,463	0.172
BDD100K	DINO	2.5%	2.5%	873	873	1.00	834	40818	0.085
BDD100K	DINO	5.0%	5.0%	1746	1746	1.00	1593	15314	0.222
BDD100K	DINO	7.5%	7.5%	2619	2619	1.00	2311	29860	0.240
Cityscapes	FCOS	2.5%	2.5%	74	74	1.00	68	12955	0.071
Cityscapes	FCOS	5.0%	5.0%	148	148	1.00	136	12596	0.140
Cityscapes	FCOS	7.5%	7.5%	223	223	1.00	210	12736	0.183
Cityscapes	Faster R-CNN	2.5%	2.5%	74	74	1.00	68	6977	0.095
Cityscapes	Faster R-CNN	5.0%	5.0%	148	148	1.00	137	6871	0.219
Cityscapes	Faster R-CNN	7.5%	7.5%	223	223	1.00	202	7823	0.270
Cityscapes	DINO	2.5%	2.5%	37	37	1.00	35	1821	0.091
Cityscapes	DINO	5.0%	5.0%	74	74	1.00	66	1797	0.198
Cityscapes	DINO	7.5%	7.5%	111	111	1.00	97	2246	0.252
nuImages	FCOS	2.5%	2.5%	1681	1681	1.00	1592	84942	0.109
nuImages	FCOS	5.0%	5.0%	3363	3363	1.00	3069	83514	0.212
nuImages	FCOS	7.5%	7.5%	5045	5045	1.00	4408	82908	0.310
nuImages	Faster R-CNN	2.5%	2.5%	1681	1681	1.00	1574	59072	0.137
nuImages	Faster R-CNN	5.0%	5.0%	3363	3363	1.00	3011	63046	0.240
nuImages	Faster R-CNN	7.5%	7.5%	5045	5045	1.00	4282	62697	0.373
nuImages	DINO	2.5%	2.5%	841	841	1.00	708	3018	0.414
nuImages	DINO	5.0%	5.0%	1682	1682	1.00	1209	2962	0.826
nuImages	DINO	7.5%	7.5%	2523	2523	1.00	1639	3123	1.239

Fig. 3 visualizes the changes relative to the baseline in **Table 2**. Under the validation-selected replay-ratio protocol, all 12 dataset-detector pairs show positive validation Δ AP under the selected settings. The largest selected improvements occur for transformer-based DINO on BDD100K (+2.23 AP) and nuImages (+2.20 AP); the smallest individual selected gain is Cityscapes FCOS (+1.03 AP), and Cityscapes has the smallest mean selected gain across detectors (+1.21 AP).

Since every configuration in the sweep improves, the validation-selected column quantifies how much a ratio choice adds rather than whether the effect is present. Replay-budget utilization helps interpret why the largest requested ratio is not always best. Except for the KITTI cases where Budget Fill is below 1.00, the effective replay ratio equals the requested ratio in **Table 5**; all BDD100K, Cityscapes, and nuImages runs have Budget Fill = 1.00. On KITTI Faster R-CNN, and on KITTI DINO at 7.5%, the small active missed-object pool together with the four-exposure cap limits the effective replay ratio, so an intermediate requested ratio can outperform the largest ratio.

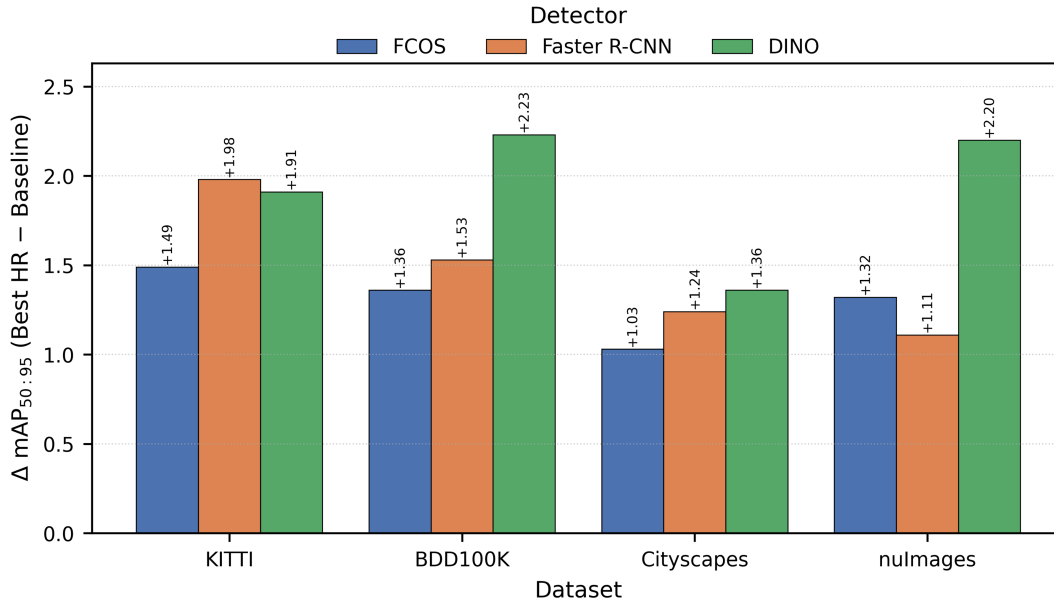


Figure 3: Validation-selected Δ AP for Hard Replay relative to the No-HR baseline.

These utilization figures also serve as a sensitivity analysis for the exposure cap, which is otherwise shared across all datasets without a dedicated sweep. The cap of four exposures per instance per epoch limits the total replay volume only in the four KITTI runs identified above. In the remaining 32 configurations the requested budget is filled exactly, so whatever the cap rejects is substituted by another eligible image, not lost. Realized exposures also stay below a quarter of the cap in 26 of the 36 runs. The most strongly constrained setting is nonetheless not the one that suffers: KITTI Faster R-CNN at 5.0% fills only 13% of its requested budget, yet attains the largest margin over the ratio-matched random control in Table 4. Relaxing the cap is therefore not indicated by these diagnostics.

Panel B of Table 4 shows that the exposure-only DINO Δ AP gains are much smaller than the ReMiss-HR gains on every dataset (mean +0.19 vs. +1.93 Δ AP, with Cityscapes exposure-only at -0.20), so for DINO the selected gains are not explained by additional optimizer exposure alone. Panel A separates the remaining factors.

Panel A decomposes the mechanism rather than serving only as a set of sanity controls. Random replay adds image exposures at the same requested ratio with no difficulty signal. Uniform sampling from the MissBank pool adds a difficulty signal but no miss history, and is therefore generic hard-sample resampling under a matched replay budget. ReMiss-HR additionally uses recurrence history and matching gaps. The three levels are strictly ordered as random < uniform-missed < ReMiss-HR in all 12 dataset-detector pairs, each step giving $p \approx 4.9 \times 10^{-4}$ under both tests. Averaged over the pairs, additional exposure accounts for +0.26 AP, the missed-object pool for a further +0.59 AP, and recurrent missed-object history for a further +0.71 AP, the last being the component specific to the contribution of this paper. That margin is smallest for DINO on BDD100K, Cityscapes, and nuImages and largest for the eight CNN pairs, so the priority ordering matters most where the candidate pool is least selective.

Panel B is reported for DINO alone because that is the setting in which additional optimizer updates can plausibly matter. The CNN detectors are trained to a fixed iteration budget, so replay enlarges each epoch and correspondingly reduces the number of epochs, leaving the total number of updates unchanged. DINO is instead trained to a fixed epoch budget. For the CNN detectors the matched comparison is instead the

random-replay condition of Panel A, which fills the same number of replay slots with uninformative content in seven of the eight selected settings. Aggregated over the eight pairs it yields +0.03 AP against +1.38 AP for ReMiss-HR, which isolates replay content rather than replay volume as the operative factor. Every run is additionally evaluated at its own best validation checkpoint (Section 4.4), so a longer schedule cannot raise the reported baseline. Figs. 4 and 5 plot the corresponding validation curves, sampled on a fixed evaluation grid, so a plotted maximum can fall slightly below the best-checkpoint value reported in Table 2.

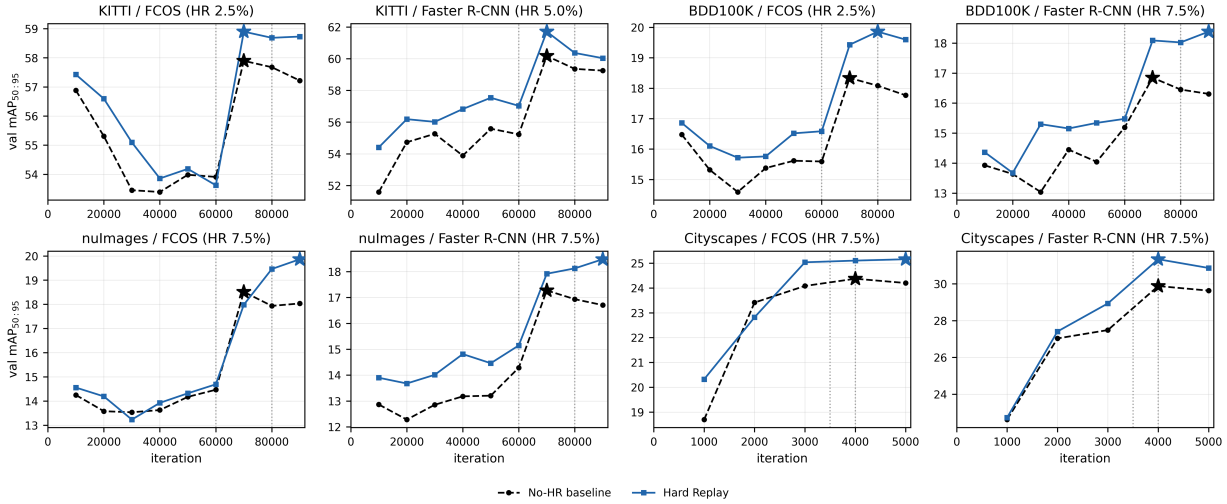


Figure 4: Validation $mAP_{50:95}$ vs. iteration for the CNN detectors: No-HR baseline (black, dashed) against validation-selected Hard Replay (blue, solid). Stars mark each curve's best checkpoint; dotted lines mark the learning-rate milestones.

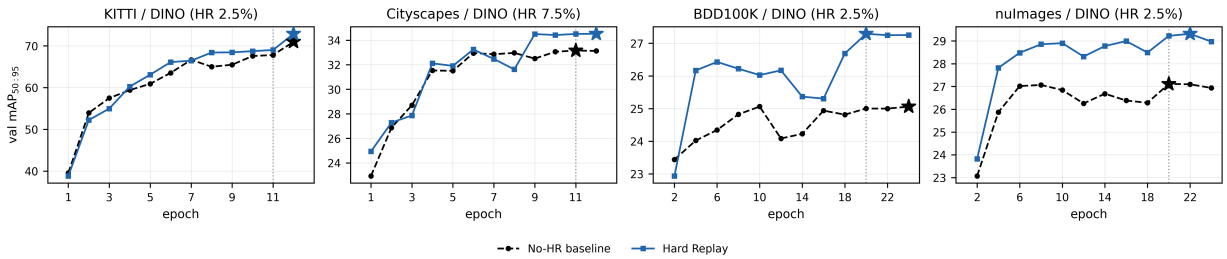


Figure 5: Validation $mAP_{50:95}$ vs. epoch for DINO, plotted as in Fig. 4; the dotted line marks the single learning-rate drop. KITTI and Cityscapes are sampled every epoch, BDD100K and nuImages every second epoch.

Fig. 6 qualitatively compares the detection outputs of the No-HR baseline and the validation-selected Hard Replay model on representative dataset-detector pairs. These examples complement the AP results by showing how replay-guided training can change final predictions in driving scenes. The rows are organized by failure mode rather than by dataset, and each row header names the mode that the panel is selected to illustrate: distant and small-scale targets (KITTI, Faster R-CNN), heavily occluded targets (nuImages, FCOS), recurrently missed targets (Cityscapes, DINO), and dense, mutually occluding traffic (BDD100K, DINO). Green boxes are ground truth, red boxes are baseline predictions, and blue boxes are Hard Replay predictions.



Figure 6: Qualitative detection comparison between the No-HR baseline (left) and validation-selected Hard Replay (right). Green boxes denote ground truth; red boxes are baseline predictions and blue boxes Hard Replay predictions, with confidence labels. Rows are organized by failure mode, named in each row header. Selection criterion and split are given in [Section 4.5](#).

The panels are not chosen by comparing the two prediction sets: in every row the selection was made from the No-HR baseline outputs alone. The recurrently missed row is drawn from the ground-truth instances the baseline failed to detect at every mining snapshot, and is necessarily a base-training-split image, since MissBank never observes the validation split. The recovery it shows is the intended consequence of having replayed that instance, not an independent result. Evidence on held-out data is carried by the other three rows, which are validation images, and by the validation AP of [Tables 2](#) and [3](#).

Across the four rows Hard Replay recovers annotated instances that the baseline leaves unmatched. The effect is not a uniform increase in the number of predictions: in the recurrently missed row Hard Replay also drops the two low-confidence false positives that the baseline places on the bollards in front of the targets. It is not free of false positives either, and in the dense row several instances of the distant vehicle cluster remain unmatched for Hard Replay as well, which shows the limit of the method.

A figure of this size illustrates the mechanism but cannot establish that it operates over a population; the corresponding population-level measurement is Table 6, which compares missed-GT ratios and persistent missed-GT counts against the No-HR baseline over every annotated instance of the mined split. That diagnostic is measured on the base training data by construction (Section 3.2), so it is not interchangeable with the validation AP that carries the generalization evidence, nor is it a restatement of the replay schedule, since persistent missed-GT counts increase in several settings in which AP nevertheless improves (Section 4.6). Object size is the one attribute the four benchmarks define identically, and Table 7 decomposes the gain along it. All three COCO area buckets improve in every pair ($p \approx 4.9 \times 10^{-4}$ each), but small objects gain least in all 12, at roughly half the medium and large gains. Replay therefore converts near-misses on larger objects more readily than it recovers small ones. Occlusion and distance are not decomposed, since these benchmarks do not record them in a uniform, comparable form.

Table 6: MissBank diagnostic snapshot for validation-selected Hard Replay. Missed-GT ratios and persistent missed-GT counts are computed over the base training split scanned by offline mining and are compared with the No-HR baseline; persistent counts are MissBank history diagnostics and should not be interpreted as subsets of the Active Missed GT replay pool in Table 5. Δ AP is relative to that baseline. Missed-GT ratios and persistent counts are from the seed-42 run; Δ AP is the three-seed mean of Table 2.

Dataset	Detector	HR	Base Missed GT (%)	HR Missed GT (%)	Δ Missed GT (pp)	Base Persistent Missed GT	HR Persistent Missed GT	Δ Persistent Missed GT	Δ AP
KITTI	FCOS	HR-2.5%	0.27	0.28	+0.01	75	45	-30	+1.49
KITTI	Faster R-CNN	HR-5.0%	0.08	0.03	-0.05	11	5	-6	+1.98
KITTI	DINO	HR-2.5%	0.14	0.15	+0.01	35	16	-19	+1.91
BDD100K	FCOS	HR-2.5%	22.87	22.66	-0.21	2,72,662	2,68,614	-4048	+1.36
BDD100K	Faster R-CNN	HR-7.5%	16.85	17.22	+0.37	1,87,927	1,81,742	-6185	+1.53
BDD100K	DINO	HR-2.5%	2.87	4.80	+1.93	29942	40943	+11001	+2.23
Cityscapes	FCOS	HR-7.5%	19.50	22.69	+3.19	10239	11581	+1342	+1.03
Cityscapes	Faster R-CNN	HR-7.5%	13.19	13.94	+0.75	6273	6349	+76	+1.24
Cityscapes	DINO	HR-7.5%	9.04	7.97	-1.07	2101	2496	+395	+1.36
nuImages	FCOS	HR-7.5%	14.64	14.87	+0.23	74288	73183	-1105	+1.32
nuImages	Faster R-CNN	HR-7.5%	10.96	11.24	+0.28	54117	51812	-2305	+1.11
nuImages	DINO	HR-2.5%	1.38	1.05	-0.33	3800	2834	-966	+2.20

Table 7: Scale-stratified validation AP for the validation-selected Hard Replay setting. AP_S , AP_M , and AP_L are the COCO area buckets ($<32^2$, 32^2-96^2 , and $>96^2$ pixels), from the same COCOeval pass that produces Table 2. The first three columns are the No-HR baseline; the last three are differences relative to it. All values are three-seed means.

Dataset	Detector	AP_S	AP_M	AP_L	ΔAP_S	ΔAP_M	ΔAP_L
KITTI	FCOS	55.89	60.42	55.70	+1.26	+1.50	+3.15
KITTI	Faster R-CNN	59.45	61.08	60.27	+1.03	+2.29	+2.17
KITTI	DINO	63.93	70.45	75.80	+0.32	+2.71	+1.28
BDD100K	FCOS	9.66	27.39	33.34	+0.45	+1.56	+1.24
BDD100K	Faster R-CNN	9.04	24.83	31.58	+0.68	+1.88	+1.68
BDD100K	DINO	13.80	34.63	48.02	+1.78	+2.69	+2.87
Cityscapes	FCOS	6.68	22.94	39.18	+0.36	+1.23	+1.17
Cityscapes	Faster R-CNN	10.25	27.95	45.22	+0.87	+1.25	+0.94
Cityscapes	DINO	16.93	44.18	61.34	+1.23	+1.39	+1.29

(Continued)

Table 7 (continued)

Dataset	Detector	AP _S	AP _M	AP _L	Δ AP _S	Δ AP _M	Δ AP _L
nuImages	FCOS	9.24	19.53	25.03	+0.81	+1.12	+1.36
nuImages	Faster R-CNN	9.79	17.61	23.79	+0.72	+1.15	+1.04
nuImages	DINO	17.59	30.65	46.16	+1.26	+1.99	+2.64
Mean	–	–	–	–	+0.90	+1.73	+1.74
Pairs improved	–	–	–	–	12/12	12/12	12/12

4.6 Ablation, Diagnostics and Runtime

[Table 3](#) reports the full Hard Replay ratio ablation with AP, AP₅₀, AP₇₅, AR₁₀₀, and Δ AP relative to the No-HR baseline for each dataset–detector pair. [Table 5](#) reports replay-budget utilization at the best-validation checkpoint for each Hard Replay run. Budget Fill is the actual number of replay samples divided by the requested replay budget; when the active missed-object pool is not large enough to fill the requested budget, the effective ratio can be lower than the requested ratio. Because replay is performed at the image level, a replayed image can expose multiple active missed-GT instances at once. Exposure/Missed GT is therefore an average object-exposure statistic: it divides the total replay object exposures induced by selected images by the number of active missed-GT instances, rather than counting one isolated object crop per replay sample.

[Table 8](#) reports the component-level ablation of the priority function, in which each term of [Eq. \(2\)](#) is deleted in turn and the run repeated. [Table 6](#) aligns MissBank diagnostics for the selected Hard Replay settings against the No-HR baseline and reports how missed-GT ratio and persistent missed-GT counts change with AP. [Table 9](#) summarizes runtime overhead for the same Hard Replay settings used in the MissBank diagnostic comparison, relative to the training-only baseline. In [Table 9](#), Base Train is the training-only baseline epoch time, Mining Pass is the no-gradient base-loader scan time, and HR Total includes HR training plus amortized mining cost.

Table 8: Priority-component ablation. Entries are three-seed mean Δ AP relative to the No-HR baseline, and Drop is the full priority minus the variant. Uniform-missed samples the same eligible pool without any priority.

Priority Variant	KITTI	Cityscapes	nuImages	BDD100K	Mean	Drop	Share
	Faster R-CNN 5.0%	FCOS 7.5%	Faster R-CNN 7.5%	DINO 2.5%			
Full priority, Eq. (2)	+1.98	+1.03	+1.11	+2.23	+1.59	–	–
Uniform-missed (no priority)	+0.61	+0.31	+0.53	+1.99	+0.86	0.73	100%
<i>Single-term deletion</i>							
π without m_g	+0.73	+0.45	+0.60	+2.03	+0.95	0.64	87%
π without $0.25M_g$	+1.63	+0.89	+0.91	+2.16	+1.40	0.19	26%
π without $0.1m_g^{\max}$	+1.89	+0.92	+0.91	+2.15	+1.47	0.12	16%
π without IoU gap	+1.97	+1.02	+1.09	+2.22	+1.58	0.01	2%
π without confidence gap	+1.96	+1.04	+1.10	+2.20	+1.58	0.01	2%
<i>Block deletion</i>							
Gap terms only (count block removed)	+0.63	+0.40	+0.52	+1.97	+0.88	0.71	97%

(Continued)

Table 8 (continued)

Priority Variant	KITTI Faster R-CNN 5.0%	Cityscapes FCOS 7.5%	nuImages Faster R-CNN 7.5%	BDD100K DINO 2.5%	Mean	Drop	Share
Count terms only (gap block removed)	+1.95	+1.01	+1.13	+2.25	+1.59	0.00	0%

Table 9: Runtime overhead for validation-selected hard replay, with the column definitions given in the text. The last column reports the validation Δ AP of [Table 2](#) per 10% of added training time.

Dataset	Detector	HR Ratio	Base Train (s)	HR Train (s)	Mining Pass (s)	HR Total (s)	Overhead (%)	Δ AP/+10%
KITTI	FCOS	2.5%	71.28	72.75	48.65	82.27	+15.42	0.97
KITTI	Faster R-CNN	5.0%	83.53	85.23	51.54	95.32	+14.11	1.40
KITTI	DINO	2.5%	70.23	71.69	26.32	83.54	+18.96	1.01
BDD100K	FCOS	2.5%	1381.71	1401.18	1171.77	1752.71	+26.85	0.51
BDD100K	Faster R-CNN	7.5%	1546.04	1587.76	1199.62	1887.66	+22.10	0.69
BDD100K	DINO	2.5%	1532.37	1556.98	760.86	1774.36	+15.79	1.41
Cityscapes	FCOS	7.5%	63.93	64.53	51.53	72.45	+13.34	0.77
Cityscapes	Faster R-CNN	7.5%	69.49	72.87	52.92	81.02	+16.59	0.75
Cityscapes	DINO	7.5%	46.13	48.34	23.75	58.23	+26.24	0.52
nuImages	FCOS	7.5%	1308.01	1346.40	970.89	1623.80	+24.14	0.55
nuImages	Faster R-CNN	7.5%	1422.66	1502.58	935.40	1769.84	+24.40	0.45
nuImages	DINO	2.5%	1425.00	1451.02	593.35	1624.08	+13.97	1.57

[Fig. 7](#) and [Table 3](#) show that Hard Replay is sensitive to replay ratio. In several settings, AP does not increase monotonically with ratio. For example, KITTI Faster R-CNN peaks at the intermediate ratio (5.0%) and drops at the largest ratio (7.5%), while BDD100K DINO decreases as the ratio grows and moves back toward the baseline at 7.5%.

Other settings behave differently. BDD100K FCOS is nearly insensitive to ratio changes (AP 19.86 $\rightarrow$ 19.83 $\rightarrow$ 19.76), whereas the FCOS series for Cityscapes (24.72 $\rightarrow$ 25.36 $\rightarrow$ 25.53) and nuImages (19.18 $\rightarrow$ 19.75 $\rightarrow$ 19.86) increase with replay ratio. [Table 3](#) confirms these trends quantitatively and includes several non-selected ratios with smaller Δ AP than the selected setting. These patterns indicate that replay-ratio effects are setting-dependent. Lower gains at non-selected ratios are consistent with interactions among replay budget, active missed-object pool size, candidate-image diversity, and detector-specific optimization behavior. The direction of the effect is nevertheless not setting-dependent: every ratio improves every pair. What varies is the magnitude, so the replay ratio tunes the size of an already-positive effect; it does not determine whether Hard Replay helps.

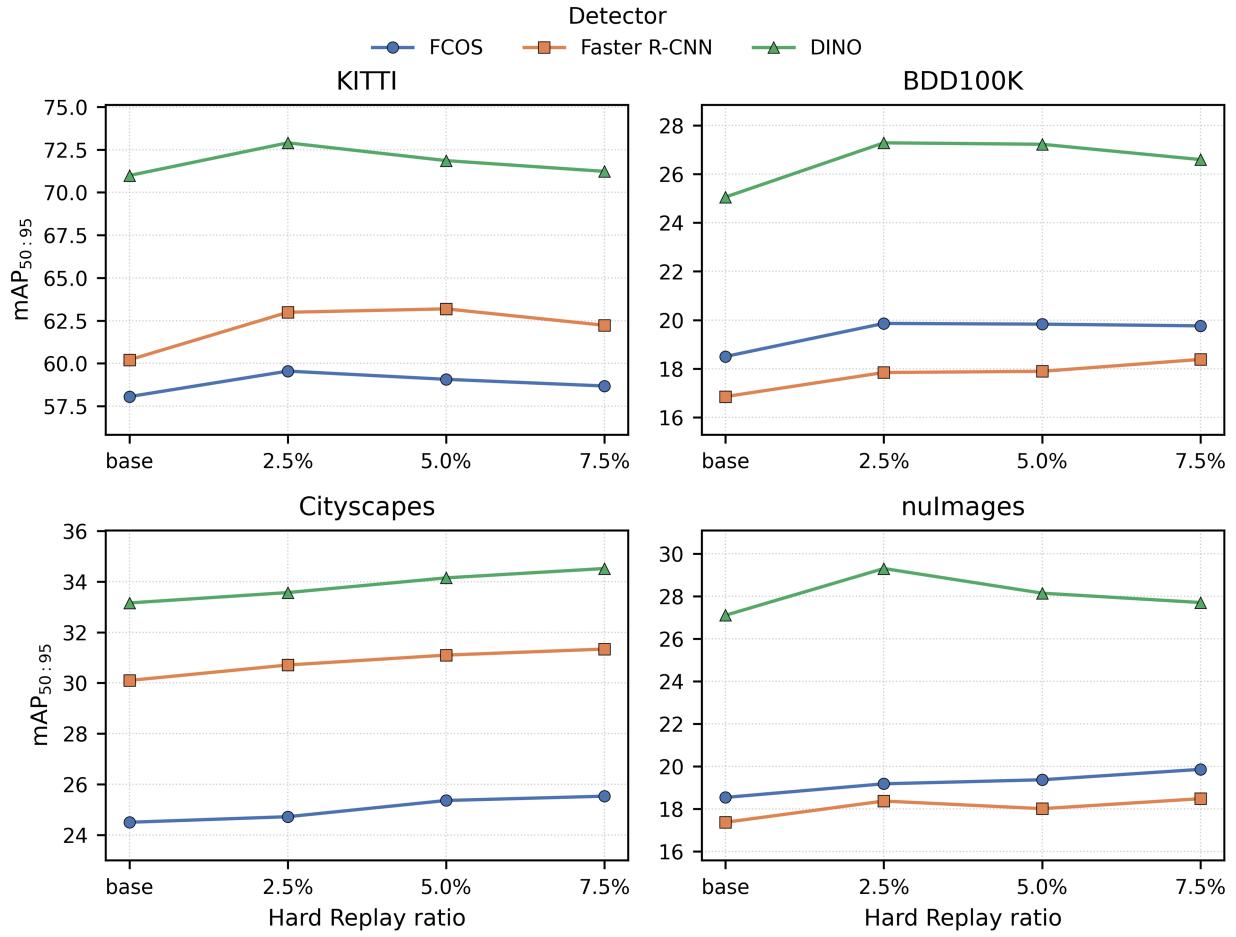


Figure 7: Validation $mAP_{50:95}$ across replay ratios. The baseline point denotes the No-HR baseline.

Because B_p is computed from the logged base-loader epoch size N_{load} , the DINO rows on BDD100K, Cityscapes, and nuImages can have smaller requested replay counts than the nominal dataset image counts would imply.

Table 8 decomposes the priority function itself. For each variant we delete one term, or one block of terms, from Eq. (2) and retrain from scratch under the identical shared configuration, the same validation-selected replay ratio, and the same three seeds. No rescaling is applied after a deletion, since replay selection depends on the priority only through the normalized image-level distribution q_i of Section 3.4, which a global rescaling would leave unchanged. The four pairs used are KITTI Faster R-CNN, Cityscapes FCOS, nuImages Faster R-CNN, and BDD100K DINO, which cover all four datasets and all three detector families. They also span the quantity being decomposed, including both the largest margin of ReMiss-HR over uniform-missed in Table 4 and the low-margin DINO regime, at an average matching that over all twelve pairs.

Three results follow. First, the count block carries the effect: removing all three count terms costs 0.71 AP of the 0.73 AP that separates the full priority from uniform sampling of the same pool, and the consecutive miss count alone accounts for 0.64 AP. Second, the two gap terms are inert, which confirms by measurement what Section 3.3 establishes in closed form, namely that with $a_u \tau_m + a_s \tau_s = 0.525 < 1$ they cannot invert an ordering fixed by the count terms. Third, the decomposition is stable: although the total priority contribution

varies by a factor of 5.7 across the four pairs, the share attributable to the consecutive miss count remains within 81%–91%. Two limits qualify the table. The individual contributions of M_g and $m_g^{\max}$ lie inside the seed-to-seed spread of Table 2 for every pair taken separately, so no separate contribution is claimed for either. Deleting m_g also removes the term that sets the scale of the priority, making its single-term figure an upper bound on its isolated contribution. The block rows are free of this effect, and the claim that the count block carries the effect rests on them.

The MissBank diagnostics in Table 6 are taken from the mining snapshot associated with each selected Hard Replay run and its corresponding No-HR diagnostic baseline. These missed-GT statistics describe the state of the replay signal rather than quantities minimized by the training loss or used for model selection. Missed-GT ratio, persistent missed-GT count, and Active Missed GT in Table 5 are related but distinct diagnostics: Active Missed GT denotes the latest-snapshot replay-eligible missed pool used for scheduling, whereas persistent missed-GT count is computed from MissBank miss history at the diagnostic snapshot. AP can increase even when the missed-GT ratio or persistent count increases if confidence ranking or localization quality improves [6,7].

Fig. 8 extends Table 6 by showing how persistent missed-GT ratio changes at each mining snapshot throughout training. Each panel compares the No-HR baseline (dashed) and selected Hard Replay (solid) for a dataset–detector pair. At each snapshot, persistent missed-GT ratio is computed conditionally over missed ground-truth instances: the numerator is the number of currently missed MissBank-tracked instances that satisfy the persistent-miss criterion, and the denominator is the number of currently missed MissBank-tracked instances in that snapshot. It is therefore a persistence-within-misses diagnostic, not a missed-GT ratio over all annotated instances. The relative position of the two curves at the same epoch is more informative than their absolute level alone.

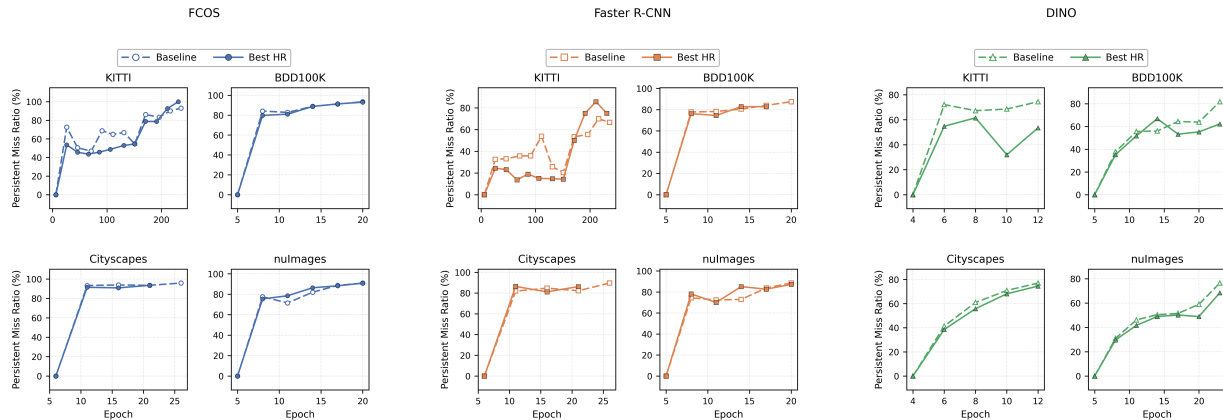


Figure 8: Persistent missed-GT ratio among currently missed MissBank-tracked instances at mining snapshots. Dashed curves show the No-HR baseline; solid curves show validation-selected Hard Replay.

For Faster R-CNN and DINO on KITTI, the Hard Replay curve moves below the baseline in subsequent training, indicating fewer persistent misses under this conditional diagnostic. In BDD100K DINO and the Cityscapes settings, the selected Hard Replay run does not consistently stay below the baseline even though AP improves. This reflects that Hard Replay changes image exposure using missed-object history without directly optimizing persistent missed-GT ratio.

Fig. 9 and Table 9 report runtime cost relative to the training-only baseline. Base Train is the per-epoch training time without diagnostic mining. HR Train is the per-epoch training time with replay exposures.

Mining Pass is the no-gradient scan of the base training loader used to update MissBank at a configured mining stage. HR Total adds the amortized mining cost to HR Train, and Overhead reports the relative increase of HR Total over Base Train.

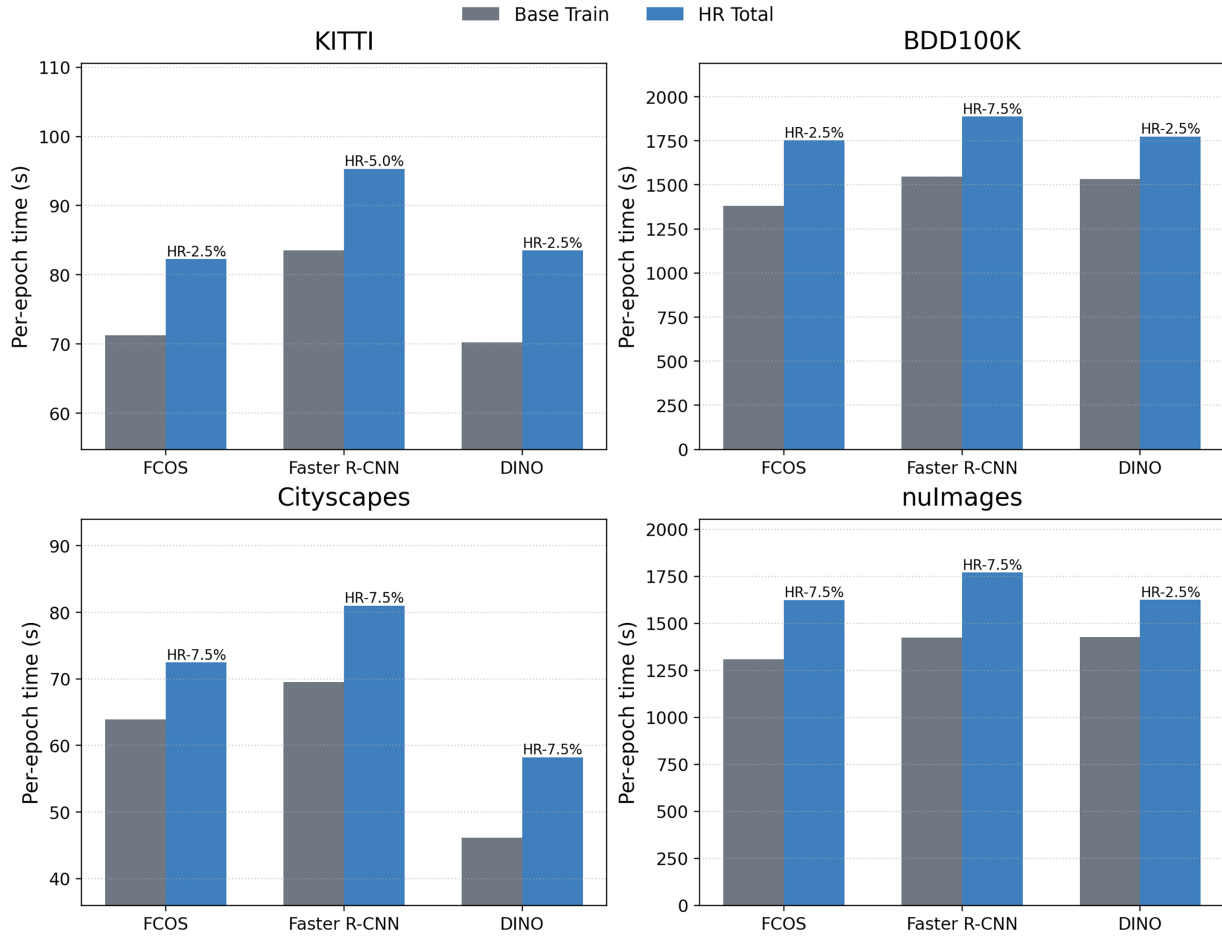


Figure 9: Per-epoch wall-clock cost for validation-selected Hard Replay. Base Train is training-only time; HR Total includes HR training plus amortized mining cost.

The absolute added time is $\text{HR Total} - \text{Base Train}$. The reported percentage overhead divides this quantity by Base Train. The amortized mining term is included in HR Total rather than shown as a separate table column. This runtime overhead should therefore be interpreted as the relative difference between HR Total and Base Train, not as the diagnostic mining cost of the No-HR baseline used for performance comparison.

Comparing HR Total with the training-only baseline runtime, the selected BDD100K settings incur +15.8% to +26.9% runtime overhead. Across all selected settings, the overhead range is +13.3% to +26.9%. The largest value occurs for BDD100K FCOS, where the no-gradient mining pass is close to the cost of a base training epoch; Cityscapes DINO is close behind because the base training epoch is short relative to the amortized mining pass.

Relating this cost to the accuracy it buys, the validation-selected ΔAP per 10% of added training time averages 0.88 across the 12 pairs. It is most favorable for DINO on the two large datasets, where the mining pass is cheap relative to a base training epoch and the AP gain is largest. This cost is incurred entirely at training time: because ReMiss modifies neither the architecture, the loss, nor the inference path, and

MissBank is discarded once training ends, the deployed detector is a standard detector and its inference cost is unchanged.

4.7 Discussion and Future Work

The experiments clarify both the utility and the operating scope of ReMiss. Missed-object history provides a detector-agnostic scheduling signal that improves validation $mAP_{50:95}$ in every configuration evaluated, so obtaining an improvement does not require tuning the replay ratio. The sweep in [Table 3](#) nevertheless shows that the size of the improvement varies with the active missed-object pool, candidate-image diversity, detector, and dataset. The replay budget is therefore worth tuning on validation data when the last few tenths of AP matter, while a ratio fixed in advance suffices to obtain the effect.

The MissBank diagnostics reinforce this operating scope. In some settings, persistent missed-GT counts decrease together with AP improvements; in others, AP improves while missed-GT ratios or persistent missed-GT counts increase. This pattern is consistent with using missed-object history to shape image exposure rather than treating missed-GT counts as the training objective. Future replay policies could therefore incorporate persistent misses, localization quality, confidence calibration, and AP-oriented ranking effects more explicitly.

Because ReMiss operates on the data schedule and leaves the loss unchanged, loss-level difficulty methods such as online hard example mining and Focal Loss are composable with it rather than alternatives to it, and are therefore not included as competing baselines in [Section 4.5](#). The evaluation carries indirect evidence for this composability. FCOS and DINO classify with a focal or focal-style loss natively, whereas Faster R-CNN uses standard cross-entropy, so the gains in [Table 2](#) are obtained on top of loss-level difficulty weighting in eight of the twelve dataset–detector pairs. The mean over those eight pairs is comparable to that over the four cross-entropy pairs, and every pair in both groups improves. The effect is thus neither confined to detectors that lack loss-level reweighting nor made redundant by its presence. A controlled combination experiment, together with direct comparison against named loss-based, uncertainty-based, and forgetting-event-based schedulers, remains future work.

To keep comparisons controlled across datasets and detectors, this study uses one shared ReMiss configuration. The priority-score weights in [Eq. \(2\)](#), exposure cap, mining schedule, sampling policy, and score and IoU thresholds in [Eq. \(1\)](#) are unchanged across experiments. MissBank updates occur at configured offline mining stages, making the replay signal a stagewise snapshot of missed-object history rather than a continuously refreshed state. Natural extensions include intra-epoch mining, a sweep over mining frequency and placement, which this study fixes rather than varies, a head-to-head retraining comparison of sum, maximum, and average image-level aggregation, for which [Section 3.4](#) gives the design argument but not a measurement, learned or adaptive replay priorities, and validation-aware replay-budget selection.

ReMiss uses stable annotation-instance keys to compare the same annotated object across mining snapshots. When converted datasets do not provide persistent annotation identifiers, this study constructs keys from image identifier, class label, and file-order index ([Section 3.2](#)); conversion procedures that preserve stable annotation identities provide the cleanest interface for this tracking step.

Hard Replay also adds replay exposures and offline-mining passes, producing the +13.3% to +26.9% runtime overhead reported in [Table 9](#). The current evaluation is limited to 2D bounding-box detection on driving-scene images. Extending ReMiss to 3D or multimodal perception, video-level temporal consistency, multi-task perception, downstream planning, and safety-oriented evaluation is a promising direction because missed-object histories can be defined for any prediction interface with stable instance-level targets [[3,12,51](#)].

5 Conclusion

We introduced ReMiss, a training-time framework that represents recurrent missed detections as missed-object histories of ground-truth instances. ReMiss uses these histories as a data scheduling signal without changing detector architectures or losses: MissBank records missed-object states during offline mining, and Hard Replay prioritizes full images containing currently missed objects for later training exposure.

Across 36 dataset–detector–ratio configurations spanning KITTI, BDD100K, Cityscapes, and nuImages with FCOS, Faster R-CNN, and DINO, every configuration improves over the No-HR baseline, with a minimum gain of +0.22 AP and a mean of +1.16 AP (two-sided sign test, $p \approx 2.9 \times 10^{-11}$). The conclusion therefore does not depend on choosing a replay ratio: a ratio fixed in advance at 5.0% improves all 12 dataset–detector pairs by +0.64–+2.17 AP, while selecting the ratio by validation mAP_{50:95} raises the range to +1.03–+2.23 AP and is reported as a secondary quantity. These gains incur a measured 13%–27% increase in training time over the training-only baseline. Reported values are means over three fixed random seeds, with per-seed standard deviations of at most 0.48 AP (Table 2), and the 12-pair improvement is significant under paired sign and Wilcoxon signed-rank tests ($p \approx 4.9 \times 10^{-4}$). These results support missed-object history as a detector-agnostic scheduling signal across the evaluated dataset–detector pairs, independently of the replay-ratio selection.

This cost profile is asymmetric in a useful way: ReMiss acts only on the training data stream, and MissBank is discarded once training ends, so the deployed model adds no inference-time latency, memory, or engineering surface relative to the same detector trained conventionally.

The operating scope set out in Section 4.7 delimits these conclusions: 2D bounding-box detection on driving-scene imagery, one shared configuration throughout, and a miss history refreshed only at discrete mining stages. The gain is also characterized by object size but not by other attributes: every size bucket improves in all 12 pairs while small objects gain least (Table 7), whereas occlusion and distance are not exposed in a uniform form by the evaluated benchmarks.

Within this operating scope, ReMiss turns recurrent missed-object histories into a practical detector-agnostic training signal. The ratio sensitivity and MissBank diagnostics discussed in Section 4.7 motivate adaptive replay policies, while the current results establish a baseline for applying missed-object-history scheduling to broader perception settings.

Acknowledgement: The authors thank the members of their research group for helpful discussions during the preparation of this manuscript.

Funding Statement: This work was supported by National Research Foundation of Korea (NRF) grant funded by the Korea government, Ministry of Science and ICT (MSIT) (RS-2026-25473871) and by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, Forestry and Fisheries (IPET) through Technology Commercialization Support Program, funded by Ministry of Agriculture, Food and Rural Affairs (MAFRA) (RS-2025-02218444) and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2020R1A6A1A03038540).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Seongwook Lee and L. Minh Dang; Methodology, Seongwook Lee; Software, Seongwook Lee, Seungyeong Kim and Jaemin Oh; Validation, Seongwook Lee, Seungyeong Kim and Jaemin Oh; Formal analysis, Seongwook Lee; Investigation, Seongwook Lee, Seungyeong Kim and Jaemin Oh; Resources, Hyeonjoon Moon; Data Curation, Seongwook Lee, Seungyeong Kim and Jaemin Oh; Writing—Original Draft, Seongwook Lee; Writing—Review & Editing, Seongwook Lee and Hyeonjoon Moon; Visualization, Seongwook Lee; Supervision, Hyeonjoon Moon and L. Minh

Dang; Project administration, Hyeonjoon Moon; Funding acquisition, not applicable. All authors reviewed and approved the final manuscript.

Availability of Data and Materials: This study uses four public driving-scene object detection datasets [1–3,11,12]. KITTI is available at <https://www.cvlibs.net/datasets/kitti/>, BDD100K at <http://bdd-data.berkeley.edu/>, Cityscapes at <https://www.cityscapes-dataset.com/>, and nuImages at <https://www.nuscenes.org/nuimages>. No new dataset was created for this study, and all datasets were converted to a common COCO-format annotation before training. Reproducibility artifacts include COCO-format conversion scripts, class mappings, split lists, detector and ReMiss configuration files, random-seed records, mining-epoch schedules, MissBank snapshot logs, replay candidate and selection logs, checkpoint-selection records, AP evaluation logs, runtime logs, and command-line/run metadata. These artifacts are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable. This study does not involve human or animal subjects and uses only public object detection datasets.

Conflicts of Interest: Given their roles as Lead Guest Editor (L. Minh Dang) and Co-Guest Editor (Hyeonjoon Moon) of this journal, these authors had no involvement in the peer review of this article and had no access to information regarding its peer review. Full responsibility for the editorial process for this article was delegated to another journal editor. The authors declare no other conflicts of interest.

References

1. Geiger A, Lenz P, Urtasun R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition; 2012 Jun 16–21; Providence, RI, USA. p. 3354–61.
2. Yu F, Chen H, Wang X, Xian W, Chen Y, Liu F, et al. BDD100K: a diverse driving dataset for heterogeneous multitask learning. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 2636–45.
3. Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, et al. nuScenes: a multimodal dataset for autonomous driving. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 11621–31.
4. Saeedizadeh N, Jalali SMJ, Khan B, Mohamed S. Cutting-edge deep learning methods for image-based object detection in autonomous driving: in-depth survey. *Exp Syst.* 2025;42(4):e70020. doi:10.1111/exsy.70020.
5. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft COCO: common objects in context. In: Computer Vision–ECCV 2014. Vol. 8693 of Lecture Notes in Computer Science. Cham, Switzerland: Springer; 2014. p. 740–55.
6. Hoiem D, Chodpathumwan Y, Dai Q. Diagnosing error in object detectors. In: Computer Vision–ECCV 2012. Vol. 7574 of Lecture Notes in Computer Science. Berlin/Heidelberg, Germany: Springer; 2012. p. 340–53.
7. Bolya D, Foley S, Hays J, Hoffman J. TIDE: a general toolbox for identifying object detection errors. In: Computer Vision–ECCV 2020. Vol. 12348 of Lecture Notes in Computer Science. Cham, Switzerland: Springer; 2020. p. 558–73.
8. Shrivastava A, Gupta A, Girshick R. Training region-based object detectors with online hard example mining. In: Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 761–9.
9. Lin TY, Goyal P, Girshick R, He K, Dollar P. Focal loss for dense object detection. In: Proceedings of the 2017 IEEE International Conference on Computer Vision; 2017 Oct 22–29; Venice, Italy. p. 2999–3007.
10. Cui Y, Jia M, Lin TY, Song Y, Belongie S. Class-balanced loss based on effective number of samples. In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA; 2019. p. 9268–77.

11. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, et al. The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 3213–23.
12. Motional. nuImages [Internet]. 2020 [cited 2026 Jun 9]. Available from: <https://www.nuscenes.org/nuimages>.
13. Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell*. 2017;39(6):1137–49. doi:10.1109/tpami.2016.2577031.
14. Tian Z, Shen C, Chen H, He T. FCOS: fully convolutional one-stage object detection. In: Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision; 2019 Oct 27–Nov 2; Seoul, Republic of Korea. p. 9627–36.
15. Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S. End-to-end object detection with transformers. In: *Computer Vision–ECCV 2020*. Cham, Switzerland: Springer; 2020. p. 213–29.
16. Zhu X, Su W, Lu L, Li B, Wang X, Deformable DJ, et al. Deformable transformers for end-to-end object detection. In: Proceedings of the 2021 International Conference on Learning Representations; 2021 May 4; Vienna, Austria.
17. Zhang H, Li F, Liu S, Zhang L, Su H, Zhu J, et al. DINO: DETR with improved DeNoising anchor boxes for end-to-end object detection. In: Proceedings of the Eleventh International Conference on Learning Representations; 2023 May 1–5; Kigali, Rwanda.
18. Zong Z, Song G, Liu Y. DETRs with collaborative hybrid assignments training. In: Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision; 2023 Oct 1–6; Paris, France. p. 6725–35.
19. Zhao Y, Lv W, Xu S, Wei J, Wang G, Dang Q, et al. DETRs beat YOLOs on real-time object detection. In: Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2024 Jun 16–22; Seattle, WA, USA. p. 16965–74.
20. Gupta A, Dollar P, Girshick R. LVIS: a dataset for large vocabulary instance segmentation. In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA. p. 5356–64.
21. Li B, Liu Y, Wang X. Gradient harmonized single-stage detector. *Proc AAAI Conf Artif Intell*. 2019;33(1):8577–84. doi:10.1609/aaai.v33i01.33018577.
22. Pang J, Chen K, Shi J, Feng H, Ouyang W, Lin D. Libra R-CNN: towards balanced learning for object detection. In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA. p. 821–30.
23. Cao Y, Chen K, Loy CC, Lin D. Prime sample attention in object detection. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 11583–91.
24. Bengio Y, Louradour J, Collobert R, Weston J. Curriculum learning. In: Proceedings of the 26th Annual International Conference on Machine Learning; 2009 Jun 14–18; Montreal, QC, Canada. p. 41–8.
25. Jiang L, Meng D, Zhao Q, Shan S, Hauptmann AG. Self-paced curriculum learning. *Proc AAAI Conf Artif Intell*. 2015;29(1):2694–700. doi:10.1609/aaai.v29i1.9608.
26. Toneva M, Sordoni A, Tachet des Combes R, Trischler A, Bengio Y, Gordon GJ. An empirical study of example forgetting during deep neural network learning. In: Proceedings of the 2019 International Conference on Learning Representations; 2019 May 6–9; New Orleans, LA, USA.
27. Swayamdipta S, Schwartz R, Lourie N, Wang Y, Hajishirzi H, Smith NA, et al. Dataset cartography: mapping and diagnosing datasets with training dynamics. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2020 Nov 16–20; Online. p. 9275–93.
28. Sorscher B, Geirhos R, Shekhar S, Ganguli S, Morcos AS. Beyond neural scaling laws: beating power law scaling via data pruning. In: Proceedings of the 36th International Conference on Neural Information Processing Systems; 2022 Nov 28–Dec 9; New Orleans, LA, USA. p. 19523–36.
29. Zheng H, Liu R, Lai F, Prakash A. Coverage-centric coreset selection for high pruning rates. In: Proceedings of the 2023 International Conference on Learning Representations; 2023 May 1–5; Kigali, Rwanda.
30. Maharana A, Yadav P, Bansal M. D2 pruning: message passing for balancing diversity and difficulty in data pruning. In: Proceedings of the 2024 International Conference on Learning Representations; 2024 May 7–11; Vienna, Austria.

31. Moser BB, Shanbhag AS, Frolov S, Raue F, Folz J, Dengel A. A coreset selection of coreset selection literature: introduction and recent advances. *arXiv:2505.17799*. 2025.
32. Zheng H, Tsai E, Lu Y, Sun J, Bartoldson BR, Kaikhura B, et al. ELFS: label-free coreset selection with proxy training dynamics. In: *Proceedings of the 2025 International Conference on Learning Representations*; 2025 Apr 24–28; Singapore.
33. Lee H, Kim S, Lee J, Yoo J, Kwak N. Coreset selection for object detection. In: *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*; 2024 Jun 17–18; Seattle, WA, USA. p. 7682–91.
34. Schubert M, Riedlinger T, Kahl K, Rottmann M. Deep active learning with noisy oracle in object detection. In: *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*; 2024 Feb 27–29; Rome, Italy. p. 375–84.
35. Sener O, Savarese S. Active learning for convolutional neural networks: a core-set approach. In: *Proceedings of the 2018 International Conference on Learning Representations*; 2018 Apr 30–May 3; Vancouver, BC, Canada.
36. Yoo D, Kweon IS. Learning loss for active learning. In: *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; 2019 Jun 15–20; Long Beach, CA, USA. p. 93–102.
37. Zha D, Bhat ZP, Lai KH, Yang F, Jiang Z, Zhong S, et al. Data-centric artificial intelligence: a survey. *ACM Comput Surv*. 2025;57(5):1–42. doi:10.1145/3711118.
38. Yuan T, Wan F, Fu M, Liu J, Xu S, Ji X, et al. Multiple instance active learning for object detection. In: *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; 2021 Jun 20–25; Nashville, TN, USA. p. 5330–9.
39. Schaul T, Quan J, Antonoglou I, Silver D. Prioritized experience replay. In: *Proceedings of the 2016 International Conference on Learning Representations*; 2016 May 2–4; San Juan, Puerto Rico.
40. Rebuffi SA, Kolesnikov A, Sperl G, Lampert CH. iCaRL: incremental classifier and representation learning. In: *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*; 2017 Jul 21–26; Honolulu, HI, USA. p. 5533–42.
41. Shmelkov K, Schmid C, Alahari K. Incremental learning of object detectors without catastrophic forgetting. In: *Proceedings of the 2017 IEEE International Conference on Computer Vision*; 2017 Oct 22–29; Venice, Italy. p. 3420–9.
42. Cermelli F, Geraci A, Fontanel D, Caputo B. Modeling missing annotations for incremental learning in object detection. In: *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*; 2022 Jun 19–20; New Orleans, LA, USA. p. 3700–10.
43. Liu Y, Schiele B, Vedaldi A, Rupprecht C. Continual detection transformer for incremental object detection. In: *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; 2023 Jun 17–24; Vancouver, BC, Canada. p. 23799–808.
44. Menezes AG, de Moura G, Alves C, de Carvalho ACPLF. Continual object detection: a review of definitions, strategies, and challenges. *Neural Netw*. 2023;161:476–93.
45. Acharya M, Hayes TL, Kanan C. RODEO: Replay for Online Object Detection. In: *Proceedings of the 31th British Machine Vision Conference*; 2020 Sep 7–10; Virtual.
46. Liu Y, Cong Y, Goswami D, Liu X, van de Weijer J. Augmented box replay: overcoming foreground shift for incremental object detection. In: *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision*; 2023 Oct 1–6; Paris, France. p. 11367–77.
47. Yang D, Zhou Y, Hong X, Zhang A, Wei X, Zeng L, et al. Pseudo object replay and mining for incremental object detection. In: *Proceedings of the 31st ACM International Conference on Multimedia*; 2023 Oct 29–Nov 3; Ottawa, ON, Canada. p. 153–62.
48. Kim J, Cho H, Kim J, Tiruneh YY, Baek S. Stable diffusion-based deep generative replay for class incremental object detection. In: *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; 2024 Jun 16–22; Seattle, WA, USA. p. 28772–81.

49. Bell S, Zitnick CL, Bala K, Girshick R. Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 2874–83.
50. Hu H, Gu J, Zhang Z, Dai J, Wei Y. Relation networks for object detection. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 3588–97.
51. Philion J, Kar A, Fidler S. Learning to evaluate perception models using planner-centric metrics. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 14055–64.