SEQDETR: BRIDGING THE QUERY-FEATURE GAP IN DETR VIA DEFORMABLE STATE SPACE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Detection transformers refine object queries through a fixed three-stage pipeline: self-attention models relationships between queries, cross-attention aligns each query independently to image features, and feed-forward networks refine predictions locally. Despite their success, this pipeline has a structural gap, as no single component jointly models inter-query context and image features at the same time. Self-attention sees queries but not the image, while cross-attention sees the image but treats each query in isolation, remaining blind to what neighboring queries are attending to. We introduce **SeqDETR**, which fills this gap with a novel Deformable State Space Module (DSSM). For each object query, DSSM deformably samples K reference features from the encoder output. It then constructs an interleaved sequence of (image feature, query) pairs across all queries. A selective state space model processes this sequence and propagates a joint inter-query and image-feature representation through its recurrent state. This makes DSSM the first decoder operation in DETR-based models to simultaneously reason over both. Our analysis shows that the performance gain comes from two main factors: the interleaved sequence design that connects queries with their local image features, and the selective context propagation of the Mamba model. Component analysis further demonstrates that DSSM is highly efficient, as it approximates most of the full attention pipeline's performance on its own. It is also complementary, yielding additional gains when combined with standard modules. SeqDETR achieves a 59.8% relative improvement on LVIS rare categories (+5.5 AP) and +4.0 AP on the 13,204-category V3Det benchmark. It also improves performance on COCO across different backbones and baselines. Code will be available upon acceptance.

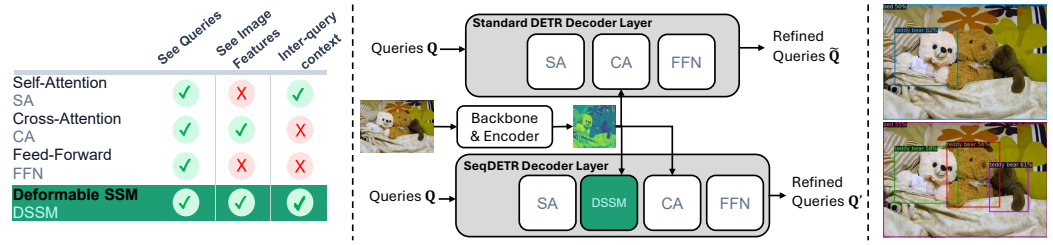
1 INTRODUCTION

Detection transformers revolutionized object detection by replacing hand-crafted pipelines with a fully learned, end-to-end framework Carion et al. (2020). Over five years, rapid progress has been driven by better query design Meng et al. (2021); Liu et al. (2022); Wang et al. (2022), more efficient attention Zhu et al. (2021); Li et al. (2023); Roh et al. (2021); Zhao et al. (2024b), and stronger training strategies Liu et al. (2023); Chen et al. (2023); Jia et al. (2023); Pu et al. (2023), enabling DETR-based models to reach state-of-the-art performance on standard benchmarks Zhang et al. (2022); Zong et al. (2023); Robinson et al. (2025). Yet, despite this progress, the internal structure of the decoder has remained essentially unchanged since the original paper.

Every DETR decoder layer applies the same three operations in the same order. Self-attention (SA) allows queries to communicate with each other, but it does so without seeing any image content. Cross-attention (CA) allows each query to probe the image feature map, but it operates independently. When query i samples its local image region, it has no knowledge of what query j is attending to at the same time. The feed-forward network (FFN) then refines each query in complete isolation. As illustrated in Figure 1, this creates a structural gap: *no component in the DETR decoder jointly models inter-query context and image features at the same time.* SA sees queries without the image, while CA sees the image without neighboring queries. The gap between them has gone unaddressed.

This gap has a concrete cost. During cross-attention, each query resolves spatial ambiguity using only its own local evidence, with no knowledge of what features neighboring queries are observing at the same moment. While self-attention later allows queries to communicate and update each other, this

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Component capabilities (b) Decoder comparison (c) Qualitative comparison

Figure 1: **The isolation gap in DETR decoders.** (a) No existing decoder component simultaneously models inter-query context and image features. (b) SeqDETR inserts DSSM between self-attention and cross-attention, constructing an interleaved (feature, query) sequence processed by a selective SSM. (c) This yields improved detection compared to the baseline [Dino (Top) vs SeqDINO (Bottom)]

exchange happens entirely in query space: by the time queries interact through SA, the feature-reading step is already over, and no query has ever seen the image evidence gathered by its neighbors. A query detecting a small pedestrian in a crowd has probed its local region independently, updated itself against other queries without knowing what those queries saw. The consequence is reduced precision, particularly in handling crowded scenes where overlapping regions create persistent ambiguity and in detecting rare categories where visual evidence is sparse.

We introduce **Sequential DETR (SeqDETR)**, which addresses the isolation gap by inserting the **Deformable State Space Module (DSSM)** between self-attention and cross-attention in each decoder layer. For each object query, DSSM deformably samples local image features from the encoder output. It then chains all queries with their features into a single interleaved sequence $(F_1, Q_1, F_2, Q_2, \ldots, F_N, Q_N)$, which is processed by a selective SSM. Each query, therefore enters cross-attention already carrying context from the image regions of its neighbors. Sorting queries by predicted confidence further strengthens this effect. High-certainty detections act as spatial anchors in the hidden state and help resolve more ambiguous queries later in the sequence. Systematic ablations confirm that the gain is structural rather than due to increased capacity. Replacing DSSM with an extra SA or CA layer yields only a negligible improvement. Replacing the selective SSM with an LSTM on the same interleaved sequence already outperforms both attention-based alternatives, showing that sequential joint modeling is the main factor. Component analysis shows that DSSM alone approximates most of the full SA+CA pipeline. Combining SA, DSSM, and CA produces additional gains that exceed any two-component combination, confirming that DSSM is complementary rather than redundant. SeqDETR improves DINO on diverse benchmarks, with the strongest gains on rare categories, crowded scenes, and large-vocabulary fine-grained detection.

Our contributions are as follows:

1. **Identifying the isolation gap.** We identify a structural limitation that has persisted across the entire DETR family since 2020: no decoder component jointly models inter-query context and image features simultaneously. This gap hurts detection where contextual reasoning matters most small objects, rare categories, and densely packed scenes.

2. **SeqDETR and the Deformable State Space Module.** We propose SeqDETR, whose core component DSSM constructs an interleaved sequence of deformably sampled image features and object queries processed by a selective state space model, making DSSM the first DETR operation to simultaneously reason over inter-query relationships and image-feature content.

3. **Consistent improvements and empirical validation.** SeqDETR consistently improves strong baselines across multiple backbones and generalizes across diverse benchmarks. Controlled ablations confirm that the gains are from the sequential joint modeling design rather than added capacity or depth.

2 Related Work

2.1 Detection Transformers

DETR Carion et al. (2020) introduced end-to-end object detection as a set-prediction problem, eliminating the need for the anchor generation and NMS mechanisms used in earlier models Girshick (2015); Ren et al. (2016); Cai & Vasconcelos (2018); Redmon et al. (2016); Liu et al. (2016). Deformable DETR Zhu et al. (2021) addressed slow convergence and weak performance on small objects by replacing dense attention with sparse deformable sampling over reference points. DAB-DETR Liu et al. (2022) and DN-DETR Li et al. (2022) improved query design through dynamic anchor boxes and denoising training, respectively. DINO Zhang et al. (2022) unified these ideas with contrastive denoising and mixed query selection, establishing a strong baseline for subsequent work. Recent methods have pushed further by improving supervision and training strategies. For instance, Co-DETR Zong et al. (2023) enriches encoder supervision with collaborative hybrid assignment, MI-DETR Nan et al. (2025) improves detection via a multi-time inquiries mechanism, and DiffuDETR Nawar et al. (2026) explores diffusion-based refinement. Despite these advances, all such methods retain the original SA→CA→FFN decoder structure. SeqDETR takes a different direction: instead of focusing on query design or training strategies, we introduce DSSM as a new decoder component that jointly models object queries and image features within a single operation.

2.2 State Space Models for Vision

S4 Gu et al. (2021) showed that structured state space models can capture long-range dependencies with linear complexity. Mamba Gu & Dao (2023) introduced input-dependent selective gating, which makes SSMs more expressive than fixed-transition models. Mamba2 Dao & Gu (2024) further unified SSMs and linear attention through the structured state space duality (SSD) framework, linking recurrent modeling to the attention mechanisms used in DETR. Several vision variants followed. VMamba Liu et al. (2024) applies selective SSMs over 2D image patches in a hierarchical backbone, while Vision Mamba Zhu et al. (2024) replaces vision transformers with bidirectional SSMs at competitive accuracy. All these methods apply sequential modeling at the backbone level, operating on image patch tokens. Other architectures Shi et al. (2024); Xie et al. (2024); Xiao et al. (2024); Yu et al. (2026) have further explored hierarchical scaling and specialized scanning patterns to optimize Mamba for general vision tasks. Beyond these, DefMamba Liu et al. (2025) applies deformable scanning over image pixels within a multi-scale backbone. This is an image-level operation and does not interact with the detection decoder. 3DET-Mamba Li et al. (2024) uses a prefix-concatenation SSM in a 3D point cloud decoder. This setting is fundamentally different, as it does not use deformable sampling and does not address the 2D query isolation problem. SeqDETR is the first to integrate an SSM into the DETR decoder at the query level, operating on an interleaved sequence of object queries and deformably sampled image features.

2.3 Component Analysis in Detection Transformer Decoders

Understanding the distinct roles of SA, CA, and FFN in the DETR decoder has become an active research direction. SMCA-DETR Gao et al. (2021) provided an early analysis of cross-attention convergence, showing that CA must learn from scratch to focus on small spatial regions around predicted boxes. This process is slow, but spatial modulation can accelerate it. DETA Ouyang-Zhang et al. (2022) found that self-attention is essential for one-to-one prediction but redundant for one-to-many prediction. DAC-DETR Hu et al. (2023) identified a conflict between self-attention and cross-attention, and proposed a divide-and-conquer strategy with an auxiliary decoder to address the trade-off between query dispersion and aggregation. Similarly, MS-DETR Zhao et al. (2024a) further analyzed the roles of SA and CA. It showed that SA mainly handles inter-query deduplication, while CA performs image-query spatial alignment, and that supervising them with different matching strategies improves performance. Mr. DETR Zhang et al. (2025) extended this analysis to a multi-task setting, showing that SA, CA, and FFN each contribute non-redundant information, even when trained jointly for one-to-one and one-to-many objectives. Collectively, these works improve our understanding of what existing components do. However, none of them introduces a new component to address what they cannot do. Their findings point to the same limitation that SeqDETR targets: no existing component provides joint inter-query and image-feature context. SeqDETR is directly motivated by this gap and introduces DSSM as the missing component.

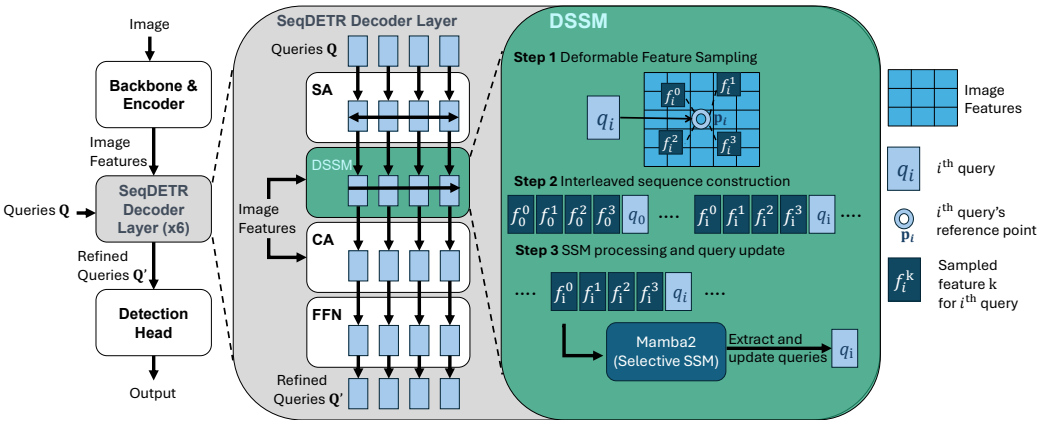


Figure 2: **Deformable State Space Module**. The DSSM integrates into the SeqDETR decoder layer to enable joint modeling of image features and object queries. Step 1: For each query q_i, K local image features f_i^k are sampled deformably around its reference point p_i. Step 2: Sampled features and queries are interleaved into a unified sequence. Step 3: A selective state-space model (Mamba2) processes this sequence, allowing queries to reason over both their local visual evidence and the context of neighboring queries through the recurrent state. The refined query representations Q' are then extracted to update the decoder's state

3 METHOD

3.1 PRELIMINARIES

Modern detection transformers operate in two stages. A backbone and transformer encoder first extract multi-scale features $\mathbf{F}$ from the input image. The encoder then selects the top-N spatial proposals ranked by confidence, which are used by the decoder as initial query positions and content Zhu et al. (2021). A standard DETR decoder transforms a set of N object queries $\mathbf{Q} = \{q_i\}_{i=1}^{N}$ by attending to $\mathbf{F}$. Each decoder layer applies three operations in sequence:

$$\hat{\mathbf{Q}} = \mathrm{SA}(\mathbf{Q}) + \mathbf{Q}, \quad \tilde{\mathbf{Q}} = \mathrm{CA}(\hat{\mathbf{Q}}, \mathbf{F}) + \hat{\mathbf{Q}}, \quad \mathbf{Q}' = \mathrm{FFN}(\tilde{\mathbf{Q}}) + \tilde{\mathbf{Q}}, \tag{1}$$

where SA is multi-head self-attention, CA is deformable cross-attention, and FFN is a feed-forward network. Self-attention allows queries to communicate with each other, but it does not see the image. Cross-attention lets each query probe the image, but it operates independently. When query q_i samples its local region, it has no knowledge of what q_j is attending to at the same time. As a result, no component jointly models inter-query context and image features. SeqDETR introduces a new module DSSM, to address this gap.

3.2 DEFORMABLE STATE SPACE MODULE

The Deformable State Space Module (DSSM) is inserted between self-attention and cross-attention in each decoder layer. This modifies the pipeline as follows:

$$\hat{\mathbf{Q}} = \mathrm{SA}(\mathbf{Q}) + \mathbf{Q}, \quad \tilde{\mathbf{Q}} = \mathrm{DSSM}(\hat{\mathbf{Q}}, \mathbf{F}) + \hat{\mathbf{Q}}, \quad \hat{\tilde{\mathbf{Q}}} = \mathrm{CA}(\tilde{\mathbf{Q}}, \mathbf{F}) + \tilde{\mathbf{Q}}, \quad \mathbf{Q}' = \mathrm{FFN}(\hat{\tilde{\mathbf{Q}}}) + \hat{\tilde{\mathbf{Q}}}. \tag{2}$$

DSSM takes the self-attention output $\hat{\mathbf{Q}}$ together with the encoder features $\mathbf{F}$, and produces enriched queries $\tilde{\mathbf{Q}}$. These queries already carry joint inter-query and image-feature context before cross-attention performs its final spatial alignment. It operates in three steps, as illustrated in Figure 2.

Step 1: Deformable feature sampling. For each query q_i, DSSM samples K image features from the encoder output by predicting offsets around the query's reference point p_i, following the same deformable sampling used in cross-attention:

$$f_i^k = \phi\big(\mathbf{F};\ p_i + \Delta p_i^k\big), \quad k = 1, \ldots, K, \tag{3}$$

where $\phi(\mathbf{F}; \cdot)$ denotes bilinear interpolation on the multi-scale feature maps, and Δp_i^k are learned offsets predicted from q_i. This gives each query a small set of spatially relevant image features, focused on its predicted object region. Using deformable sampling instead of attending to the full feature map keeps the sequence compact, with NK tokens rather than $N \cdot HW$. It also ensures that only spatially meaningful features are included in the sequence.

Step 2: Interleaved sequence construction. Rather than processing queries and features separately, DSSM chains them into a single interleaved sequence across all queries:

$$\mathbf{S} = \big(f_1^1, \ldots, f_1^K, \hat{q}_1,\ f_2^1, \ldots, f_2^K, \hat{q}_2,\ \ldots,\ f_N^1, \ldots, f_N^K, \hat{q}_N \big). \tag{4}$$

The ordering within each group is deliberate: the K sampled features for query i appear before query i itself. This means the SSM first reads the local image evidence for query i before updating its representation. As a result, each query is grounded in its own image features before this information is propagated forward as context to subsequent queries. Before constructing the sequence, we sort the queries by their encoder proposal objectness scores from highest to lowest, using the ranking produced by the two-stage encoder selection.

Step 3: SSM processing and query update. The sequence $\mathbf{S}$ is processed by a Mamba2 selective state space model Dao & Gu (2024). Unlike attention, which compares all positions simultaneously, Mamba2 processes the sequence recurrently, maintaining a hidden state h_t that accumulates context as it moves through the sequence:

$$h_t = \bar{A}_t\, h_{t-1} + \bar{B}_t\, \mathbf{S}_t, \qquad o_t = C_t\, h_t, \tag{5}$$

where $\bar{A}_t$, $\bar{B}_t$, and C_t are input-dependent parameters, which is the key property that makes Mamba *selective*. Selectivity is controlled by the step size Δ_t, which is predicted from the current input. It determines how much new information enters the hidden state at each position. A large Δ_t causes the state to absorb the current token strongly, while a small Δ_t carries the previous state forward with little change. When the model encounters the feature tokens $f_i^1, \ldots, f_i^K$, the hidden state absorbs the local image context of query i. When it then encounters the query token $\hat{q}_i$, it updates the query using a hidden state that already contains this image evidence, along with the accumulated context from all preceding query-feature pairs. After processing the full sequence, we extract only the outputs at the query positions and add them back to the queries through a residual connection:

$$\tilde{q}_i = \hat{q}_i + \mathrm{Proj}(o_{\hat{q}_i}), \quad i = 1, \ldots, N, \tag{6}$$

where $o_{\hat{q}_i}$ is the SSM output at the position of query i, and Proj is a linear projection. The feature tokens are discarded after this step, as they serve only as contextual carriers within the recurrence. The result is a set of enriched queries $\tilde{\mathbf{Q}}$, where each query has been updated with knowledge of what neighboring queries have observed in the image.

3.3 Complexity and Theoretical Connection

The object queries naturally form a set, making full self-attention over the queries and their sampled image features a natural way to jointly model their interactions. However, applying self-attention to the interleaved sequence of length $N(K+1)$ incurs quadratic complexity, $\mathcal{O}(N^2(K+1)^2)$, which is impractical for typical DETR settings. DSSM provides a more efficient alternative by processing the same joint query-feature sequence with linear complexity, $\mathcal{O}(N(K+1))$. With $N = 900$ and $K = 16$, the sequence contains 15,300 tokens, making quadratic all-pairs attention prohibitively expensive while remaining practical for the selective SSM.

Mamba2's structured state space duality (SSD) Dao & Gu (2024) further connects selective SSMs to linear attention. From this perspective, DSSM can be viewed as *deformable linear attention* over the joint query-feature sequence: deformable sampling selects sparse, spatially grounded image features, while the selective recurrence propagates context without quadratic all-pairs attention. This provides a scalable way to jointly model inter-query relationships and image features with linear complexity and input-dependent selectivity.

Table 1: Comparison of various object detectors on the COCO 2017 validation set using ResNet-50, ResNet-101 and Swin-Large backbones.

Model	Epochs	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
ResNet-50 He et al. (2016)							
Deformable DETR Zhu et al. (2021)	50	48.2	67.0	52.2	30.7	51.4	63.0
SeqDETR (Ours)	50	**49.3**	**68.0**	**53.7**	**31.8**	**52.8**	**63.9**
Dynamic DETR Dai et al. (2021)	12	47.2	65.9	51.1	28.6	49.3	59.1
DAB-Deformable DETR Liu et al. (2022)	12	46.9	66.0	50.8	30.1	50.4	62.5
DAC-DETR Hu et al. (2023)	12	50.0	67.6	54.7	32.9	53.1	64.2
MS-DETR Zhao et al. (2024a)	12	50.0	67.3	54.4	31.6	53.2	64.0
DINO Zhang et al. (2022)	12	49.9	68.0	54.6	32.6	53.5	63.9
SeqDINO (Ours)	12	50.8	68.8	55.4	33.9	53.8	65.1
MR-DETR Zhang et al. (2025)	12	50.7	68.2	55.4	33.6	54.3	64.6
MR-SeqDETR (Ours)	12	**52.5**	**69.8**	**57.2**	**36.6**	**56.4**	**66.8**
DINO Zhang et al. (2022)	24	51.0	68.2	55.4	34.0	54.2	66.4
SeqDINO (Ours)	24	51.7	69.9	56.5	35.6	55.2	66.2
MR-DETR Zhang et al. (2025)	24	51.4	69.0	56.2	34.9	54.8	66.0
MR-SeqDETR (Ours)	24	**53.2**	**70.6**	**57.9**	**36.8**	**57.4**	**67.8**
ResNet-101 He et al. (2016)							
DiffuDINO Nawar et al. (2026)	12	51.2	68.6	55.8	33.2	55.6	**67.2**
DINO Zhang et al. (2022)	12	50.3	68.2	54.9	34.1	53.5	64.9
SeqDINO (Ours)	12	**52.0**	**70.2**	**57.1**	**34.9**	**56.1**	66.3
Swin-Large Liu et al. (2021)							
H-DETR Jia et al. (2023)	12	55.9	75.2	61.0	39.1	59.9	72.2
Co-DETR Zong et al. (2023)	12	56.9	75.5	62.6	40.1	61.2	73.3
DAC-DETR Hu et al. (2023)	12	57.3	75.7	62.7	40.1	61.5	**74.4**
DINO Zhang et al. (2022)	12	56.8	75.4	62.3	**41.1**	60.6	73.5
SeqDINO (Ours)	12	**57.5**	**76.0**	**62.8**	40.6	**61.6**	74.3

Table 2: **Cross-dataset generalization results.** All models utilize a ResNet-50 backbone. Training schedules are 12, 50, and 24 epochs for LVIS, CrowdHuman, and V3Det, respectively.

(a) LVIS

Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	AP_r	AP_c	AP_f
DINO	26.5	35.9	27.8	20.0	35.2	40.9	9.2	24.6	36.2
SeqDINO	**29.0**	**38.7**	**30.6**	**21.5**	**37.4**	**43.6**	**14.7**	**26.8**	**37.8**

(b) CrowdHuman

Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
DINO	48.4	82.3	49.7	31.6	47.4	52.2
SeqDINO	**50.0**	**82.4**	**52.3**	**32.5**	**48.6**	**54.5**

(c) V3Det

Model	AP	AP_{50}	AP_{75}
DINO	33.5	37.7	35.0
SeqDINO	**37.5**	**42.6**	**39.3**

4 Experiments

4.1 Experimental Setup

We evaluate SeqDETR on four benchmarks. **COCO 2017** Lin et al. (2014) is our primary benchmark, with 118K training and 5K validation images across 80 categories. We report standard AP metrics at multiple IoU thresholds and object scales (AP_s, AP_m, AP_l). **LVIS** Gupta et al. (2019) contains 1,203 categories with a heavily long-tailed distribution. We focus on AP_r because it measures performance on rare categories with limited training examples, a setting in which additional contextual evidence may be particularly useful. **CrowdHuman** Shao et al. (2018) evaluates dense pedestrian detection, where per-query isolation is most harmful. **V3Det** Wang et al. (2023) includes 13,204 fine-grained categories and tests the ability to distinguish visually similar classes at scale. We instantiate DSSM on three different base detectors. **SeqDETR** builds on Deformable DETR Zhu et al. (2021), which uses standard bipartite matching without denoising. **SeqDINO** builds on DINO Zhang et al. (2022), which

Table 3: **Analysis of decoder components.** We evaluate the contribution of Self-Attention (SA), Cross-Attention (CA), and our proposed DSSM module.

SA	CA	DSSM	AP	AP_s	AP_m	AP_l
×	✓	×	45.2	30.6	49.1	57.2
×	×	✓	48.0	31.6	51.4	62.3
✓	×	✓	48.9	33.3	52.2	62.6
×	✓	✓	49.0	32.6	52.1	62.5
✓	✓	×	49.9	32.6	53.5	63.9
✓	✓	✓	**50.8**	**33.9**	**53.8**	**65.1**

Table 4: **Ablation of sequence modeling components.** We compare standard attention additions against LSTM and our proposed selective SSM (Mamba2). Params and FLOPs are reported for the full model.

Added component	AP	Params (M)	FLOPs (G)
Extra SA	50.1	49.2	248.4
Extra CA	50.1	49.0	252.6
LSTM	50.4	51.6	253.3
Mamba2	**50.8**	50.1	264.2

adds contrastive denoising and anchor-based queries. **MR-SeqDETR** builds on MR-DETR Zhang et al. (2025), which uses multi-route supervision with auxiliary branches. These three bases span four years of DETR development and represent different training paradigms. This allows us to evaluate DSSM across architectures rather than within a single design. In all variants, DSSM is inserted between self-attention and cross-attention in all six decoder layers. We use $K{=}16$ sampling points (4 points for each of the 4 feature levels) and apply confidence-based query ordering. All other hyperparameters follow the respective base detector settings.

4.2 Main Results on COCO

Table 1 reports results on COCO val2017 across three backbones and two training schedules. DSSM consistently improves every base detector it is applied to, and the magnitude of the improvement increases with stronger baselines. With ResNet-50 at 12 epochs, SeqDINO improves DINO from 49.9 to **50.8 AP** (+0.9), with gains across all six reported metrics. The improvement is more pronounced on MR-DETR. MR-SeqDETR reaches **52.5 AP** at 12 epochs, a **+1.8 AP** gain over its baseline. The same +1.8 AP gain is preserved at 24 epochs (53.2 vs 51.4), showing that the benefit of DSSM is structural and does not diminish with longer training. This amplification effect is consistent. Stronger base models produce better query representations, which provide richer inter-query context for DSSM to propagate, leading to larger gains. When applied to Deformable DETR at 50 epochs, SeqDETR improves performance from 48.2 to **49.3 AP** (+1.1). With ResNet-101, SeqDINO reaches **52.0 AP**, a +1.7 improvement over the DINO R101 baseline (50.3). With Swin-Large, SeqDINO achieves **57.5 AP**, outperforming Co-DETR (56.9) and H-DETR (55.9) under the same 12-epoch schedule.

4.3 Cross-Dataset Generalization

Table 2 evaluates SeqDINO on three benchmarks, each stressing a different aspect of the isolation gap. This tests whether DSSM's inter-query context provides consistent benefits beyond COCO. On **LVIS**, SeqDINO improves overall AP from 26.5 to **29.0** (+2.5). The largest gain is on rare categories: AP_r increases from 9.2 to **14.7**, a **+5.5 AP gain and a 59.8% relative improvement**. Rare LVIS categories have fewer than ten training instances, making contextual cues potentially more valuable when category-specific training evidence is limited, which DSSM provides. Gains on common (+2.1) and frequent (+1.6) categories are smaller but still consistent. On **CrowdHuman**, SeqDINO improves overall AP from 48.4 to **50.0** (+1.6). The key result is the difference across IoU thresholds. AP_{50} improves by only +0.1, while **AP_{75} improves by +2.6**. DINO already detects most people at loose thresholds (82.3 AP_{50}). DSSM instead improves precise localization in crowded scenes, where queries benefit from knowing what nearby queries have already localized, reducing spatial ambiguity at object boundaries. On **V3Det**, SeqDINO shows the largest absolute gain. AP improves from 33.5 to **37.5** (**+4.0 AP**). With 13,204 fine-grained categories, many visually similar classes appear in the same image. Knowing what neighboring queries see provides implicit category-level context, which independent per-query cross-attention cannot capture.

4.4 Ablation Studies

All ablations use SeqDINO with a ResNet-50 backbone trained for 12 epochs on COCO val2017.

Table 5: **Sequence design analysis.** Top: sequence construction format. Bottom: query ordering strategy within the sequence.

	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Sequence format						
Queries only: $[Q_1, \ldots, Q_N]$	50.1	68.2	54.8	34.4	53.7	63.6
Features then Queries: $[F_1 \ldots F_M, Q_1 \ldots Q_N]$	50.4	68.5	55.1	**34.9**	**53.9**	64.1
Interleaved: $[F_1, Q_1, \ldots, F_N, Q_N]$	**50.8**	**68.8**	**55.4**	33.9	53.8	**65.1**
Query ordering						
Random shuffled each iteration	50.2	68.3	54.9	34.2	53.5	63.3
Spatial position (Left $\rightarrow$ Right)	50.5	68.4	55.2	**35.2**	53.7	64.3
Predicted confidence (Low $\rightarrow$ High)	50.4	68.7	55.0	34.3	**53.9**	64.3
Predicted confidence (High $\rightarrow$ Low)	**50.8**	**68.8**	**55.4**	33.9	53.8	**65.1**

Component complementarity. Table 3 evaluates every subset of SA, CA, and DSSM. CA alone reaches only 45.2 AP, confirming that image grounding without inter-query reasoning is insufficient on its own. By contrast, SSM alone achieves **48.0 AP**, reaching 96.2% of the full SA+CA pipeline (49.9 AP), showing that DSSM is a general detection primitive that jointly incorporates both attention mechanisms. Adding DSSM to the full SA+CA pipeline produces additional gains: the complete SA+DSSM+CA model (50.8 AP) outperforms every two-component combination.

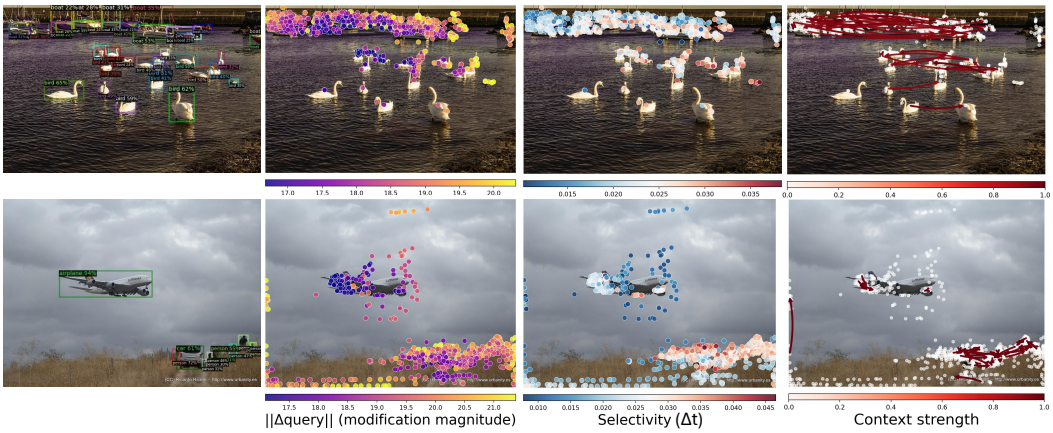
Sequential modeling drives the gain. Table 4 replaces DSSM with modules of comparable parameter count to isolate the source of improvement. Adding an extra self-attention or cross-attention layer yields only **+0.2 AP** each, ruling out model depth and capacity as explanations. Replacing the selective SSM with an LSTM yields **+0.5 AP** using the same interleaved sequence. This result is important: the LSTM already outperforms both attention alternatives using the same sequence design, confirming that *sequential joint modeling of queries and image features* is the primary driver of improvement, independent of the specific SSM architecture. Upgrading to Mamba's input-dependent selective gating adds a further **+0.4 AP**, showing that selectively deciding how much context to propagate at each step refines this gain. Mamba also uses fewer parameters than the LSTM (50.1M vs 51.6M) while achieving higher overall AP.

Sequence format. Table 5 (top) compares three sequence construction methods. Processing queries without image features results in a marginal gain of 0.2 AP over the baseline (49.9 AP vs. 50.1 AP), confirming that image features are essential for grounding the hidden state with visual evidence. Grouping all features before all queries improves performance to 50.4 AP. Our interleaved design achieves **50.8 AP**. This causal structure ensures each query is grounded in its own image evidence before propagating context to subsequent queries.

Query ordering. Table 5 (bottom) shows that the order in which queries appear in the sequence matters. The 0.6 AP gap between the worst case (random, 50.2) and the best (confident-first, 50.8) is itself an important finding. If the SSM were simply aggregating context without considering order, all strategies would perform similarly. The sensitivity to ordering confirms that the recurrent hidden state carries meaningful, order-dependent context. **Confident-first ordering** achieves the best overall AP. High-confidence detections establish reliable context early in the hidden state, which helps resolve ambiguous queries processed later. Spatial ordering (left-to-right) achieves the highest AP_s (35.2), suggesting that local adjacency in the sequence is particularly helpful for small object detection.

4.5 Analysis

DSSM routes sampled visual context between neighboring object queries. Figure 3 shows how DSSM controls which information is absorbed and propagated across queries. Query modification magnitude in (b) is strongest in ambiguous and densely packed regions, where queries benefit most from additional contextual information. Selectivity in (c) is higher for queries whose reference points lie on foreground objects, allowing DSSM to retain and propagate the sampled features of queries containing meaningful object evidence while suppressing less informative background



(a) (b) (c) (d)

Figure 3: **Visualization of DSSM internal behavior**. **(a)** SeqDINO's output. **(b)** Query modification magnitude $\|\tilde{q}_i - \hat{q}_i\|_2$ at each query's reference point. Warm colors indicate queries most strongly updated by DSSM. **(c)** DSSM selectivity (Δ_t): the input-dependent gate value controlling how much new information enters the hidden state at each query position. High selectivity concentrates on object regions. **(d)** Context propagation flow: arrows show the top-50 query pairs by hidden-state similarity. Dominant flows cluster within coherent object groups rather than across unrelated regions

signals. Context flow in (d) visualizes the query pairs with the highest hidden-state similarity, higher similarity indicates stronger information sharing after DSSM processing. The strongest interactions are concentrated among spatially neighboring queries with similar visual evidence, showing that DSSM routes feature context between related queries rather than propagating information indiscriminately. This provides a direct view of how neighboring queries can exchange complementary local evidence that is not captured by each query's own independently sampled features in CA.

DSSM provides a complementary role to self- and cross-attention. Prior studies such as DAC-DETR Hu et al. (2023) and DS-DET Cao et al. (2025) show that CA tends to gather multiple nearby queries around the same object, while SA subsequently encourages these queries to diversify and suppress duplicate predictions. As a result, neither component is designed to let neighboring object queries share their sampled visual evidence. DSSM fills this gap by explicitly propagating sampled image features together with inter-query context. This complementary behavior is also reflected in Table 3, where removing any of SA, DSSM, or CA leads to a clear performance degradation. It is also consistent with the largest gains on CrowdHuman, LVIS, and V3Det, where contextual reasoning is especially important: SeqDETR improves by $+4.0$ AP on V3Det, $+2.6$ AP_{75} on CrowdHuman, and $+5.5$ AP on LVIS rare categories, precisely in settings where neighboring queries' sampled visual evidence can help resolve fine-grained, crowded, or weakly supervised predictions.

5 Conclusion

In this work, we identify a structural limitation that has persisted across the DETR family since 2020: no decoder component within a layer jointly models inter-query relationships and image-feature content. SeqDETR addresses this gap with DSSM, which constructs an interleaved sequence of object queries and their deformably sampled image features. This sequence is then processed by a selective state space model. The result is consistent improvement across four benchmarks. SeqDETR improves the baseline by +5.5 AP on LVIS rare categories, +4.0 AP on V3Det, and +1.6 AP on CrowdHuman. Component analysis confirms that DSSM is complementary to self-attention and cross-attention, producing additional gains that no two-component subset can match. These results suggest that unified recurrent decoders are a promising direction for future work. They can be naturally extended to tasks such as panoptic segmentation and visual grounding, where multiple queries must reason jointly over shared image regions.

REFERENCES

Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162, 2018.

Guiping Cao, Xiangyuan Lan, Wenjian Huang, Jianguo Zhang, Dongmei Jiang, and Yaowei Wang. Ds-det: Single-query paradigm and attention disentangled learning for flexible object detection. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 286–295, 2025.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Qiang Chen, Xiaokang Chen, Jian Wang, Shan Zhang, Kun Yao, Haocheng Feng, Junyu Han, Errui Ding, Gang Zeng, and Jingdong Wang. Group detr: Fast detr training with group-wise one-to-many assignment. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6633–6642, 2023.

Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2988–2997, 2021.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3621–3630, 2021.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5356–5364, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Zhengdong Hu, Yifan Sun, Jingdong Wang, and Yi Yang. Dac-detr: Divide the attention layers and conquer. *Advances in Neural Information Processing Systems*, 36:75189–75200, 2023.

Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. Detrs with hybrid matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19702–19712, 2023.

Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13619–13627, 2022.

Feng Li, Ailing Zeng, Shilong Liu, Hao Zhang, Hongyang Li, Lei Zhang, and Lionel M Ni. Lite detr: An interleaved multi-scale encoder for efficient detr. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18558–18567, 2023.

Mingsheng Li, Jiakang Yuan, Sijin Chen, Lin Zhang, Anyu Zhu, Xin Chen, and Tao Chen. 3detmamba: Causal sequence modelling for end-to-end 3d object detection. *Advances in Neural Information Processing Systems*, 37:47242–47260, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pp. 740–755. Springer, 2014.

Leiye Liu, Miao Zhang, Jihao Yin, Tingwei Liu, Wei Ji, Yongri Piao, and Huchuan Lu. Defmamba: Deformable visual state space model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8838–8847, 2025.

Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. *arXiv preprint arXiv:2201.12329*, 2022.

Shilong Liu, Tianhe Ren, Jiayu Chen, Zhaoyang Zeng, Hao Zhang, Feng Li, Hongyang Li, Jun Huang, Hang Su, Jun Zhu, et al. Detection transformer with stable matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6491–6500, 2023.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37. Springer, 2016.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. *Advances in neural information processing systems*, 37:103031–103063, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3651–3660, 2021.

Zhixiong Nan, Xianghong Li, Jifeng Dai, and Tao Xiang. Mi-detr: an object detection model with multi-time inquiries mechanism. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 4703–4712, 2025.

Youssof Nawar, Mohamed Badran, and Marwan Torki. Diffudetr: Rethinking detection transformers with denoising diffusion process. In *International Conference on Learning Representations*, volume 2026, pp. 144095–144113, 2026.

Jeffrey Ouyang-Zhang, Jang Hyun Cho, Xingyi Zhou, and Philipp Krähenbühl. Nms strikes back. *arXiv preprint arXiv:2212.06137*, 2022.

Yifan Pu, Weicong Liang, Yiduo Hao, Yuhui Yuan, Yukang Yang, Chao Zhang, Han Hu, and Gao Huang. Rank-detr for high quality object detection. *Advances in Neural Information Processing Systems*, 36:16100–16113, 2023.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

Tianhe Ren, Shilong Liu, Feng Li, Hao Zhang, Ailing Zeng, Jie Yang, Xingyu Liao, Ding Jia, Hongyang Li, He Cao, Jianan Wang, Zhaoyang Zeng, Xianbiao Qi, Yuhui Yuan, Jianwei Yang, and Lei Zhang. detrex: Benchmarking detection transformers, 2023.

Isaac Robinson, Peter Robicheaux, Matvei Popov, Deva Ramanan, and Neehar Peri. Rf-detr: neural architecture search for real-time detection transformers. *arXiv preprint arXiv:2511.09554*, 2025.

Byungseok Roh, JaeWoong Shin, Wuhyun Shin, and Saehoon Kim. Sparse detr: Efficient end-to-end object detection with learnable sparsity. *arXiv preprint arXiv:2111.14330*, 2021.

Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

Yuheng Shi, Minjing Dong, and Chang Xu. Multi-scale vmamba: Hierarchy in hierarchy visual state space model. *Advances in Neural Information Processing Systems*, 37:25687–25708, 2024.

Jiaqi Wang, Pan Zhang, Tao Chu, Yuhang Cao, Yujie Zhou, Tong Wu, Bin Wang, Conghui He, and Dahua Lin. V3det: Vast vocabulary visual detection dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19844–19854, 2023.

Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 2567–2575, 2022.

Yicheng Xiao, Lin Song, Shaoli Huang, Jiangshan Wang, Siyu Song, Yixiao Ge, Xiu Li, and Ying Shan. Grootvl: Tree topology is all you need in state space model. *arXiv preprint arXiv:2406.02395*, 2024.

Fei Xie, Weijia Zhang, Zhongdao Wang, and Chao Ma. Quadmamba: Learning quadtree-based selective scan for visual state space model. *Advances in Neural Information Processing Systems*, 37:117682–117707, 2024.

Longlong Yu, Wenxi Li, Yaoqi Sun, Hang Xu, Chenggang Yan, and Yuchen Guo. 2d-crossscan mamba: Enhancing state space models with spatially consistent multi-path 2d information propagation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 17957–17965, 2026.

Chang-Bin Zhang, Yujie Zhong, and Kai Han. Mr. detr: Instructive multi-route training for detection transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9933–9943, 2025.

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

Chuyang Zhao, Yifan Sun, Wenhao Wang, Qiang Chen, Errui Ding, Yi Yang, and Jingdong Wang. Ms-detr: Efficient detr training with mixed supervision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17027–17036, 2024a.

Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. Detrs beat yolos on real-time object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16965–16974, 2024b.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6748–6758, 2023.

A Implementation Details

All experiments are conducted on 4 NVIDIA RTX 4090 GPUs using the DETREX framework Ren et al. (2023). We use a batch size of 16 for all experiments except those involving the Swin-Large backbone or MR-SeqDETR, where memory constraints require a reduced batch size of 4.

A.1 DSSM Hyperparameters

The DSSM module uses the following configuration across all experiments unless stated otherwise. The Mamba2 SSM is configured with a state dimension $d_{\text{state}} = 256$, expansion factor expand $= 1$, and head dimension headdim $= 32$. DSSM is inserted between self-attention and cross-attention in all six decoder layers. We sample 4 deformable points across 4 feature scales, giving $K = 16$ sampled features per query. For experiments with the Swin-Large backbone, we increase the sampling points to $K = 32$ (8 points per scale) to better exploit the richer features provided by the stronger backbone. All decoder variants use $N = 900$ object queries. For DINO and MR-DETR based models, we additionally apply contrastive denoising (CDN) with 300 denoising queries, described in detail in Section A.2.

A.2 Training with Denoising Queries

DINO and MR-DETR employ contrastive denoising (CDN), which augments the standard object queries with additional denoising groups containing noised ground-truth boxes. This creates a mixed sequence during decoder training consisting of two distinct query populations: denoising queries that reconstruct clean boxes from noised inputs, and standard detection queries that perform end-to-end object detection.

Naively concatenating denoising and detection queries into a single sequence would allow the SSM hidden state to propagate context from denoising queries into detection queries, contaminating the detection stream with supervision signal from ground-truth annotations. This would constitute a form of label leakage during training that is absent at inference, creating a harmful train/test discrepancy.

We resolve this using Mamba2's `seq_idx` mechanism, which assigns each token in the sequence to an independent subsequence. Tokens with the same `seq_idx` value share a recurrent state, tokens with different values start fresh states with no context propagation between them. Concretely, we assign each denoising group its own `seq_idx`, keeping denoising queries isolated from each other and from detection queries. Detection queries are assigned a separate shared index. The SSM therefore processes each denoising group and the detection stream as fully independent sequences that share no hidden state, eliminating any possibility of label leakage while preserving the inter-query context propagation within the detection stream.

The implementation constructs `seq_idx` as follows. For each denoising group, every token in that group's interleaved (feature, query) subsequence receives the same integer label. The detection queries receive a shared label distinct from all denoising labels. These per-token indices are assembled into a tensor of shape $(B, N_{\text{total}} \cdot (K+1))$ and passed directly to `mamba_split_conv1d_scan_combined` via the `seq_idx` argument, which handles variable-length independent subsequences natively within a single batched forward pass. This adds no measurable computational overhead while correctly isolating denoising supervision from detection context propagation.

B Dataset Details and Training Protocols

B.1 COCO 2017

Dataset. COCO 2017 Lin et al. (2014) contains 118,287 training images and 5,000 validation images spanning 80 object categories. Objects range from very small (insects, distant people) to very large (elephants, buses), covering diverse real-world scenes including indoor, outdoor, urban, and natural environments. The benchmark evaluates detection quality via mean average precision across IoU thresholds from 0.5 to 0.95 (AP), with additional breakdowns by object scale: AP_s for objects with area below 32^2 pixels, AP_m for 32^2–96^2 pixels, and AP_l for objects above 96^2

pixels. COCO is the primary benchmark for evaluating general-purpose object detection, and AP_s is particularly challenging because small objects provide minimal local visual evidence for per-query cross-attention.

Training protocol. We train SeqDINO and MR-SeqDETR under two schedules. The 12-epoch schedule uses an initial learning rate of 1×10^{-4}, with a step decay of factor 0.1 applied at epoch 11. The 24-epoch schedule applies the same decay factors at epoch 20. All other hyperparameters follow the respective base detector (DINO or MR-DETR) exactly. For SeqDETR based on Deformable DETR, we train for 50 epochs with a learning rate decay at epoch 40. Baseline DINO is trained under identical settings to ensure fair comparison.

B.2 LVIS

Dataset. LVIS v1 Gupta et al. (2019) contains 100,170 training images and 19,809 validation images with instance annotations across 1,203 categories exhibiting a long-tailed frequency distribution. Categories are divided into rare (fewer than 10 training instances), common (10–100 instances), and frequent (more than 100 instances). The standard evaluation metrics are AP computed at IoU thresholds 0.5–0.95, plus AP_r, AP_c, and AP_f for rare, common, and frequent categories respectively. LVIS tests a qualitatively different capability from COCO: the model must detect categories for which it has seen almost no training examples, forcing it to rely on contextual reasoning and visual similarity rather than memorised category-specific patterns. This is precisely the regime where inter-query context is most valuable, when a query cannot resolve its target from local evidence alone, the accumulated context from neighboring queries provides critical spatial and semantic grounding. AP_r is therefore the most sensitive metric for evaluating SeqDETR's contribution on this benchmark.

Training protocol. We train for 270,000 iterations total. The learning rate follows an iteration-based schedule, decaying by a factor of 0.1 at 210,000 iterations and by a further factor of 0.1 (total $\times 0.01$ relative to the initial rate) at 250,000 iterations, terminating at 270,000 iterations. This schedule is equivalent to approximately 12 epochs on the LVIS training set. To ensure a fair comparison, we train the DINO ResNet-50 baseline from scratch under the identical iteration schedule, optimizer settings, and data augmentation pipeline.

B.3 CrowdHuman

Dataset. CrowdHuman Shao et al. (2018) contains 15,000 training images, and 4,370 validation images collected specifically to test detection under extreme crowd conditions. Images contain an average of 22.6 persons per image, with many instances partially or heavily occluded by other persons. The benchmark reports AP at IoU thresholds 0.5 and 0.75. CrowdHuman tests the complementary aspect of our contribution relative to LVIS: rather than rare category recognition, it tests precise spatial localization when multiple queries attend to overlapping image regions. The disproportionate improvement in AP_{75} over AP_{50} that SeqDETR achieves on CrowdHuman directly measures localization precision improvement, which requires each query to resolve its spatial boundaries against those of its neighbors.

Training protocol. We train for 50 epochs with an initial learning rate of 2.5×10^{-5} and a step decay of factor 0.1 at epoch 40. We use the same data augmentation as COCO training. The DINO baseline is trained from scratch under the identical 50-epoch schedule for a fair comparison.

B.4 V3Det

Dataset. V3Det Wang et al. (2023) (Vast Vocabulary Visual Detection) contains 183,354 training images and 29,821 validation images spanning 13,204 object categories organized into a hierarchical taxonomy. Categories include both common objects and fine-grained subcategories, many of which are visually similar and co-occur in the same images. The benchmark evaluates AP at IoU thresholds 0.5–0.95. V3Det tests large-vocabulary fine-grained detection, a regime where disambiguation between similar categories requires understanding what neighboring queries detected in adjacent image regions. With 13,204 categories, the probability that two nearby queries attend to different

instances of similar subcategories is high, making inter-query context directly useful for category-level disambiguation.

Training protocol. We train for 24 epochs with a learning rate of 2.5×10^{-5}, decayed by a factor of 10 at epochs 16 and 22. A repeated factor sampler is used with repeat factor 10^{-3} to handle the long-tailed category distribution, following the protocol established by the V3Det dataset paper Wang et al. (2023). For the baseline comparison, we use the official DINO results reported in the V3Det paper and replicate their exact hyperparameters in our SeqDINO experiment to ensure a controlled comparison.

C Additional Ablation Studies

C.1 Module Placement: All Orderings of SA, DSSM, and CA

Table 6: Effect of different module orderings within the decoder layer. DSSM denotes the proposed sequential state-space module, SA self-attention, and CA cross-attention.

Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
DSSM-SA-CA	50.6	68.8	55.2	33.9	54.0	64.5
SA-DSSM-CA	**50.8**	68.8	55.4	33.9	53.8	**65.1**
SA-CA-DSSM	50.4	68.5	55.0	34.6	53.6	64.2
CA-DSSM-SA	50.7	**68.8**	55.4	34.3	54.2	64.5
CA-SA-DSSM	50.6	68.6	**55.6**	**35.0**	54.2	64.6
DSSM-CA-SA	50.6	68.6	55.4	34.4	**54.3**	64.7

Table 6 evaluates all six possible orderings of SA, DSSM, and CA within the decoder layer. The most important finding is that every ordering outperforms the SA+CA baseline (49.9 AP), confirming that DSSM provides consistent benefit regardless of where it is placed. The total spread across all six orderings is only 0.4 AP (50.4 to 50.8), which means DSSM is robust to placement, as it cannot be broken by choosing a suboptimal position. SA→DSSM→CA achieves the best overall AP (50.8), which starts with SA deduplicating queries producing clean input for the SSM, DSSM then enriching queries with inter-query context, and CA performing final precise spatial alignment on the enriched representations. The worst ordering is SA→CA→SSM (50.4 AP), which places DSSM after CA and denies it the opportunity to enrich queries before spatial alignment. Notably, orderings where SSM precedes CA achieve higher AP_l (64.5–65.1) while orderings where SSM follows CA achieve higher AP_s (34.3–35.0).

C.2 Decoder Layer Selection

Table 7: **Impact of SSM module integration depth.** We evaluate the performance by varying which decoder layers (out of 6) incorporate the inter-query sequential modeling.

SSM Integration	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Last layer only	50.4	68.2	55.2	34.5	53.9	64.0
First 3 layers	50.4	68.4	54.9	**34.8**	53.5	64.1
Last 3 layers	50.6	68.7	55.1	34.2	53.7	65.0
All 6 layers	**50.8**	**68.8**	**55.4**	33.9	**53.8**	**65.1**

Table 7 varies which of the six decoder layers include DSSM. Applying DSSM to only the final decoder layer achieves 50.4 AP, recovering large part of the full model's gain over the baseline with one-sixth the number of DSSM modules. This demonstrates that DSSM provides substantial benefit even as a minimal single-layer addition, which is practically important for deployment scenarios with strict computational constraints. Applying DSSM to only the last three layers achieves 50.6 AP, recovering most of the full gain. The first three layers alone also achieve 50.4 AP but with lower

AP_l (64.1 vs 65.0 for last three layers), suggesting that later decoder layers produce richer sequential context for the SSM to propagate, as these layers are where query representations are more precisely localized. The full model with all six layers achieves the best overall AP (50.8) and AP_m/AP_l, though at slightly lower AP_s (33.9) than the first-three-layers variant (34.8), consistent with the observation that applying DSSM to early layers when queries are not yet well-localized provides more diffuse context that benefits small object detection differently.

C.3 Number of Sampling Points Per Scale

Table 8: **Impact of sampling points per query.** We investigate the sensitivity of SeqDETR to the number of deformable sampling points K.

Sample points K	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
4	50.5	68.6	55.1	**34.6**	53.6	64.9
8	50.7	**68.8**	55.4	34.2	**54.1**	64.9
16	**50.8**	68.8	**55.4**	33.9	53.8	**65.1**
32	50.7	68.6	55.4	33.5	54.1	64.6

Table 8 evaluates sensitivity to K, the total number of deformable sampling points. Performance follows a concave curve peaking at $K = 16$ (50.8 AP), with graceful degradation in both directions. Even $K = 4$ achieves 50.5 AP, a +0.6 AP improvement over the baseline, demonstrating that the sequential inter-query context mechanism provides meaningful benefit even when each query contributes only a single sampled feature token to the sequence. This confirms that the sequential chaining structure is the primary driver of improvement. The degradation at $K = 32$ (50.7 AP) suggests that longer sequences dilute the inter-query context signal with redundant local feature detail, making the SSM's task of propagating meaningful context across the sequence more difficult. The optimal $K = 16$ matches the default sampling density in deformable cross-attention, validating that the spatial granularity appropriate for attention-based image querying transfers directly to SSM-based sequential processing.

C.4 Inference-Time Query Ordering

Table 9: **Effect of Inference-Time Query Ordering.** We evaluate the sensitivity of our sequential decoder to the input order of object queries. While training utilizes confidence-based sorting, we tested randomizing this order during inference, this leads to a notable performance degradation across all scales. This confirms that the DSSM effectively learns to utilize the ordered sequential relationship between queries.

Query Ordering Strategy	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Confidence-Sorted (Ours)	**50.8**	**68.8**	**55.4**	**33.9**	**53.8**	**65.1**
Randomly Shuffled	49.8	67.2	54.5	33.3	53.0	64.0
Absolute Drop	−1.0	−1.6	−0.9	−0.6	−0.8	−1.1

Table 9 isolates the contribution of confidence-based query ordering by randomizing query order at inference time while keeping the model trained with confidence-sorted sequences. Shuffling the query order at inference degrades AP by 1.0 points (50.8 to 49.8), with consistent drops across all scale metrics. This degradation has two implications. First, it confirms that the SSM hidden state carries order-dependent content, if the SSM were performing permutation-invariant aggregation, all orderings would produce identical outputs. Second, it validates that the encoder-proposal ordering used during both training and inference is not merely a convenient implementation choice but a learned structural prior: the model has internalized the assumption that high-confidence proposals appear early in the sequence and provide reliable spatial anchors for subsequent ambiguous queries. Disrupting this assumption at inference by shuffling forces the model to process queries in an order inconsistent with what it was trained on, degrading performance. This experiment also has practical

significance: it means SeqDETR requires consistent query ordering between training and inference, which is straightforwardly satisfied by always using encoder proposal scores for sorting.

C.5 Decoupled vs Shared Sampling Offsets

Table 10: **Decoupling of Sampling Offsets for DSSM and Cross-Attention.** We evaluate the impact of using independent vs. shared sampling offsets for the proposed DSSM and standard Cross-Attention (CA). Utilizing independent offsets allows each module to specialize in its respective spatial task.

Sampling Configuration	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Decoupled Offsets (Ours)	**50.8**	**68.8**	**55.4**	33.9	**53.8**	**65.1**
Shared DSSM-CA Offsets	50.4	68.5	54.8	**34.0**	53.7	64.7

Table 10 compares our design choice of using independent learned sampling offsets for DSSM and CA against a variant where both modules share the same offsets. Decoupled offsets yield +0.4 AP over shared offsets (50.8 vs 50.4 AP). The reason is functional specialization: DSSM and CA serve different purposes and benefit from attending to different spatial locations. DSSM samples features to build inter-query context, it benefits from attending to a query's broader neighborhood to capture the spatial relationship between nearby queries. CA performs precise spatial alignment to the query's predicted object region, it benefits from tight, accurate sampling around the object boundary. When offsets are shared, both modules must compromise on sampling locations that are suboptimal for both tasks. Decoupled offsets allow each module to independently learn its optimal sampling strategy, with DSSM developing broader contextual sampling patterns and CA maintaining tight localization sampling. The slight improvement in AP_s for shared offsets (34.0 vs 33.9) is due to shared offsets force more conservative, localized sampling that marginally benefits small objects at the cost of global inter-query reasoning.

C.6 Bidirectional SSM Ablation

Table 11: **Unidirectional vs. bidirectional DSSM processing.** We compare our confident-first unidirectional approach against a bidirectional variant. The bidirectional model processes the sequence in two passes: forward (low-to-high confidence), then backward (high-to-low confidence).

DSSM variant	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Unidirectional	**50.8**	**68.8**	**55.4**	**33.9**	53.8	**65.1**
Bidirectional	50.6	68.6	55.2	33.8	**54.0**	64.8

Table 11 compares our unidirectional approach against bidirectional variant that processes the sequence in two passes: a forward pass from lowest to highest confidence, then a backward pass from highest to lowest. CDN denoising groups are kept isolated via `seq_idx` in both passes. The bidirectional variant achieves 50.6 AP, below our unidirectional model (50.8 AP), confirming that the causal structure of our design is not a limitation but a deliberate and effective inductive bias.

The performance drop is explained by the interaction between bidirectional processing and our confident-first ordering. In our unidirectional model, the forward pass processes high-confidence queries first, allowing them to establish reliable spatial anchors in the hidden state before the SSM encounters more ambiguous queries. The causal structure ensures that this context flows in one direction only from certain to uncertain, which matches the intended information flow. In the bidirectional variant, the forward pass processes queries from lowest to highest confidence, meaning the hidden state is initialized with the most uncertain queries and only encounters reliable anchors late in the sequence. The backward pass then processes from highest to lowest, but by the time its output is combined with the forward pass, the two streams carry conflicting context about the same queries: one built on uncertain-first context, one on confident-first. Summing these streams dilutes the clean confident-first prior that our unidirectional model exploits. This interpretation is supported

by the inference-time ordering experiment (Table 9), which showed a -1.0 AP drop from shuffling query order, both results confirm that the direction and order of context propagation carry meaningful structure that the model has learned to rely on.

C.7 Performance Breakdown by Scene Density

Table 12: **Performance Breakdown by Scene Density.** We evaluate model robustness across varying object densities by splitting the COCO validation set into Sparse (≤ 10 objects) and Dense (> 10 objects) subsets. SeqDINO consistently outperforms the DINO baseline in both scenarios.

Subset	Model	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Sparse Scenes	DINO	56.2	73.0	61.7	34.5	**55.8**	64.9
	SeqDINO	**57.1**	**73.4**	**62.5**	**37.4**	55.7	**66.4**
Dense Scenes	DINO	43.2	62.2	46.5	32.3	51.0	57.9
	SeqDINO	**44.0**	**63.0**	**48.0**	**33.0**	**52.3**	**59.4**

Table 12 splits the COCO validation set into sparse scenes (≤ 10 objects per image, 3,217 images) and dense scenes (> 10 objects per image, 1,783 images) and evaluates SeqDINO and DINO on each subset separately. SeqDINO consistently outperforms DINO on both subsets, with gains of +0.9 AP on sparse scenes and +0.8 AP on dense scenes. The improvement on sparse scenes (57.1 vs 56.2 AP) is larger in absolute AP and is concentrated on AP_s (+2.9), reflecting that even in less crowded images, small objects benefit substantially from inter-query context. The improvement on dense scenes (44.0 vs 43.2 AP) spans all scales (AP_s +0.7, AP_m +1.3, AP_l +1.5), showing that DSSM provides broader gains when many queries must jointly reason about a complex scene. The fact that SeqDETR improves on both sparse and dense scenes demonstrates that the benefit of inter-query context is not limited to crowded scenarios but applies generally, with small object performance driving the sparse scene gains and medium/large object disambiguation driving the dense scene gains.

C.8 Computational Complexity

Table 13: **Comparison of Model Complexity and Computational Cost.** We report the number of parameters and FLOPs for SeqDETR and SeqDINO variants against several baseline architectures. Our sequential modeling approach adds minimal parameter overhead while maintaining competitive computational efficiency.

Model	Parameters (M)	FLOPs (G)
DETR	41.5	83.7
DN DETR	43.7	89.2
DN Deformable DETR	47.4	231.3
Align DETR	47.7	244.5
Deformable DETR	41.2	171.3
SeqDETR (Ours)	43.6	181.9
DINO	47.7	244.5
SeqDINO (Ours)	50.1	264.2
MR Deformable DETR	54.2	281.5
MR-SeqDETR (Ours)	56.6	337.5

Table 13 reports parameters and FLOPs for all model variants. SeqDINO adds 2.4M parameters over DINO (50.1M vs 47.7M, a 5% increase) and 19.7 GFLOPs (264.2 vs 244.5, an 8.1% increase). SeqDETR over Deformable DETR adds 2.4M parameters and 10.6 GFLOPs (6.2% increase). MR-SeqDETR adds 2.4M parameters and 56.0 GFLOPs over MR-DETR, a larger absolute FLOPs increase reflecting the higher base computational cost of the MR-DETR architecture but a comparable relative overhead. Across all three variants, the parameter overhead of DSSM is identical at 2.4M, confirming that the module has a fixed, architecture-independent parameter count. The FLOPs

overhead is proportional to the base model's decoder complexity because DSSM processes a sequence of $N(K+1)$ tokens per decoder layer, and the decoder is a fixed fraction of total model compute.

C.9 Inference efficiency

Table 14: **Practical inference efficiency**. Measured on the COCO validation set (5000 images) using a single GPU.

Model	Time / image	FPS
DINO	87 ms	11.5
SeqDINO	110 ms	9.1

Table 14 reports the practical inference efficiency of SeqDINO compared with the DINO baseline on the COCO validation set using a single GPU. DINO processes each image in 87 ms, corresponding to 11.5 FPS, while SeqDINO requires 110 ms per image and achieves 9.1 FPS. As shown in Table 13, SeqDINO introduces only a modest increase in theoretical computation, while the practical runtime overhead is larger. This gap is partly to the current hardware efficiency of Mamba-based state space operations, which are less mature than standard attention implementations that benefit from highly optimized CUDA kernels. We expect this gap to decrease as Mamba and SSM implementations mature and receive hardware-level optimization comparable to the highly optimized attention kernels available today.

C.10 Training complexity

Table 15: **Training complexity comparison on COCO.** Both models are trained for 12 epochs using four GPUs with a total batch size of 16.

Model	Training time	Max. training memory
DINO	19 h 20 min	23,029 MB
SeqDINO	22 h 08 min	26,250 MB

We further evaluate the training cost of DINO and SeqDINO on COCO. Both models are trained for 12 epochs using four GPUs with a total batch size of 16. As shown in Table 15, adding DSSM increases the total training time from 19 h 20 min to 22 h 08 min and the maximum training memory from 23,029 MB to 26,250 MB. Overall, this represents a moderate increase in training cost relative to the additional contextual modeling introduced by DSSM.

D Why Mamba Instead of LSTM?

Table 4 shows that both LSTM and Mamba outperform adding an extra self-attention or cross-attention layer. This is important because both recurrent variants operate on the same interleaved query-feature sequence, supporting our claim that the sequential design itself is a major source of the gain. We adopt Mamba rather than LSTM for three main reasons.

Selective memory updates. Mamba improves over the LSTM variant by $+0.4$ AP. Since both models use the same interleaved query-feature sequence, we attribute this additional gain to Mamba's selective state updates. As illustrated in Figure 3(c), the update strength is concentrated on informative object-related queries while less informative background signals are suppressed. We believe this selectivity helps DSSM retain more useful information over long interleaved sequences containing many sampled feature tokens.

Greater robustness to query ordering. We further compare the sensitivity of Mamba and LSTM to the confidence-based query ordering by randomly shuffling the query sequence during inference. As shown in Table 16, both models degrade when the ordering is disrupted, but the LSTM variant drops substantially more. Mamba decreases by only 1.0 AP, compared with a 2.5 AP drop for LSTM.

This suggests that selective state updates make DSSM less sensitive to imperfections in the encoder proposal ranking by prioritizing informative tokens while limiting the influence of less useful ones.

Table 16: **Robustness to query ordering.** Comparison of Mamba and LSTM under confidence-sorted and randomly shuffled query orders during inference.

Model	Confidence-sorted	Randomly shuffled	Δ AP
SeqDETR (SSM)	50.8	49.8	-1.0
SeqDETR (LSTM)	50.4	47.9	-2.5

Practical efficiency. Table 17 and Table 18 compare the practical efficiency of SeqDINO using Mamba and LSTM in the same interleaved DSSM design. The results show a substantial efficiency gap in both training and inference: the LSTM variant requires several times longer to train and is nearly an order of magnitude slower at inference. This difference is due to the parallel scan and optimized convolutional implementations used by modern selective state-space models, whereas LSTMs remain fundamentally sequential over the sequence dimension. Therefore Mamba provides a much more practical solution for processing the long interleaved query-feature sequences used by DSSM.

Table 17: **Training efficiency of SSM vs LSTM.** Comparison of DINO, SeqDINO with Mamba, and SeqDINO with LSTM on COCO for 12 epochs using four GPUs with a total batch size of 16.

Model	Training time
DINO	19 h 20 min
SeqDINO (SSM)	22 h 08 min
SeqDINO (LSTM)	6 days 3 h 35 min

Table 18: **Inference efficiency of SSM vs. LSTM.** Comparison of DINO, SeqDINO with Mamba, and SeqDINO with LSTM on COCO val (5000 images).

Model	Time / image	FPS
DINO	87 ms	11.5
SeqDINO (SSM)	110 ms	9.1
SeqDINO (LSTM)	981 ms	1.01

E Qualitative Results

Figures 4, 5, 6, and 7 provide qualitative comparisons between SeqDINO and the DINO baseline across the four evaluation benchmarks. For COCO, LVIS, and CrowdHuman, we show side-by-side comparisons, for V3Det we show SeqDINO detections only as baseline inference results are not available. On COCO, SeqDINO consistently improves detection across diverse scenarios including crowded scenes with many overlapping objects, small objects, where local evidence is insufficient, and ambiguous cases where similar nearby instances confuse per-query cross-attention. On LVIS, improvements are most visible in rare category detection, where the model has minimal training supervision and must rely on context propagated from neighboring queries attending to related image regions to ground its predictions. On CrowdHuman, SeqDINO produces tighter, more precisely localized bounding boxes in crowded and ambiguous pedestrian scenes, consistent with the disproportionate AP_{75} improvement reported in the main paper. On V3Det, inter-query context directly benefits category-level disambiguation: when two nearby queries attend to different instances of visually similar subcategories, which is a common occurrence across the 13,204-category vocabulary, the accumulated hidden state provides implicit context that helps each query commit to the correct fine-grained label.

F FAILURE CASES

Figure 8 shows representative failure cases of SeqDINO on COCO val2017 where the baseline DINO detects objects that SeqDINO misses or localizes poorly. The recurring failure case is the missed detection of large, visually prominent objects that occupy the center of the image in relative isolation such as the train in row 1 and the horse in row 3. This failure pattern is consistent with SeqDETR's design: DSSM provides benefit by propagating context from neighboring queries that attend to related image regions, but when a large object dominates the scene without nearby queries providing complementary spatial evidence, this mechanism offers no advantage and may introduce a mild perturbation that disrupts the precise localization the standard CA pipeline achieves independently.

REPRODUCIBILITY STATEMENT

The formulation and implementation of DSSM are described in Section 3 and the appendix. We provide model hyperparameters, training protocols for all datasets, denoising-query handling, additional ablations, computational-cost measurements, and failure-case analyses in the supplementary sections. These details are intended to enable reproduction of the reported experiments. Code will be made available as described in the paper.

AI USE STATEMENT

Large language models were used for minor edits and revisions during the writing of this paper, primarily to correct spelling and grammar and to improve clarity in certain passages. All AI-assisted text was reviewed and revised by the authors, who take full responsibility for the final content of the paper.

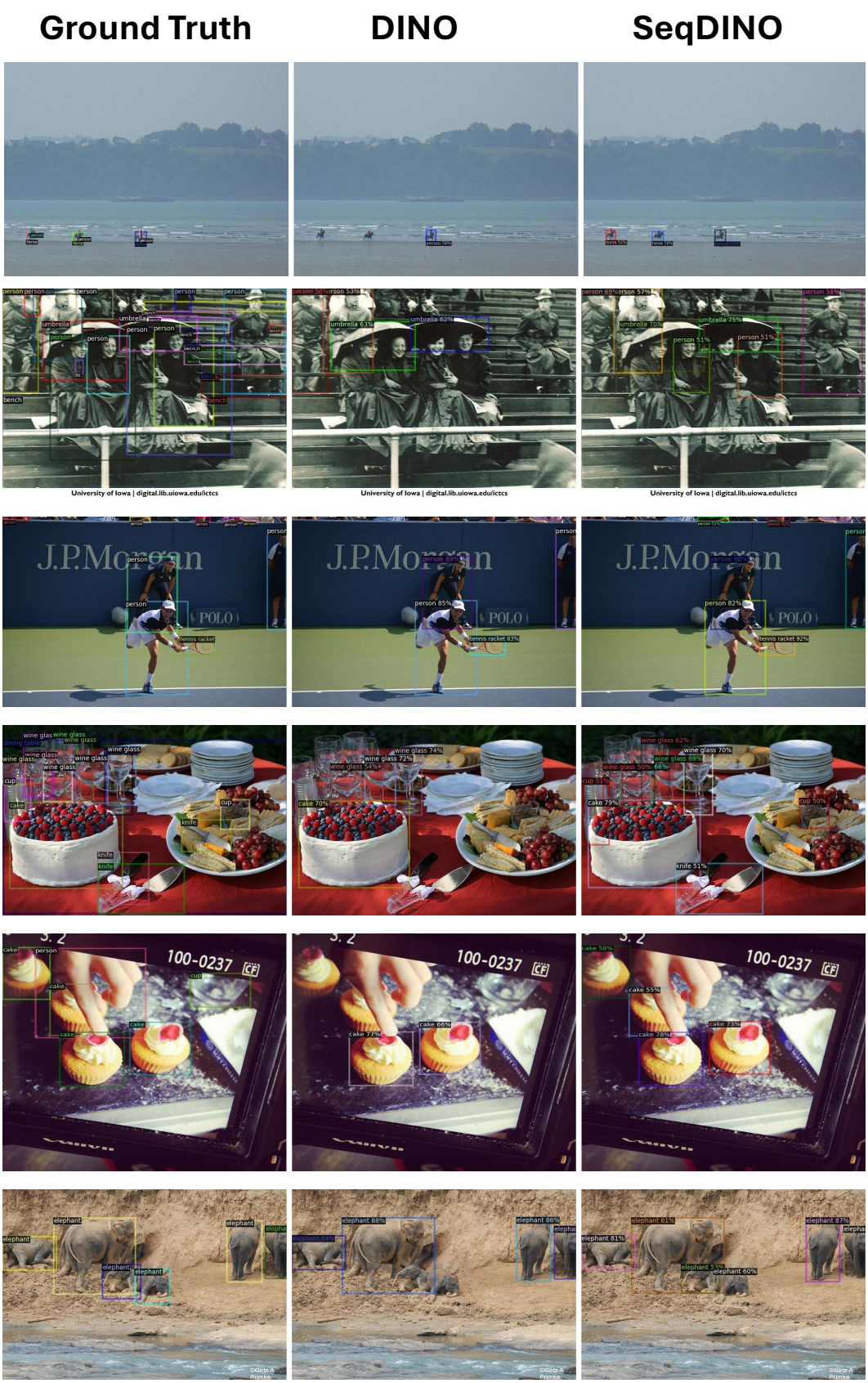


Figure 4: **Qualitative comparison on COCO val2017.**



Figure 5: **Qualitative comparison on LVIS val.**

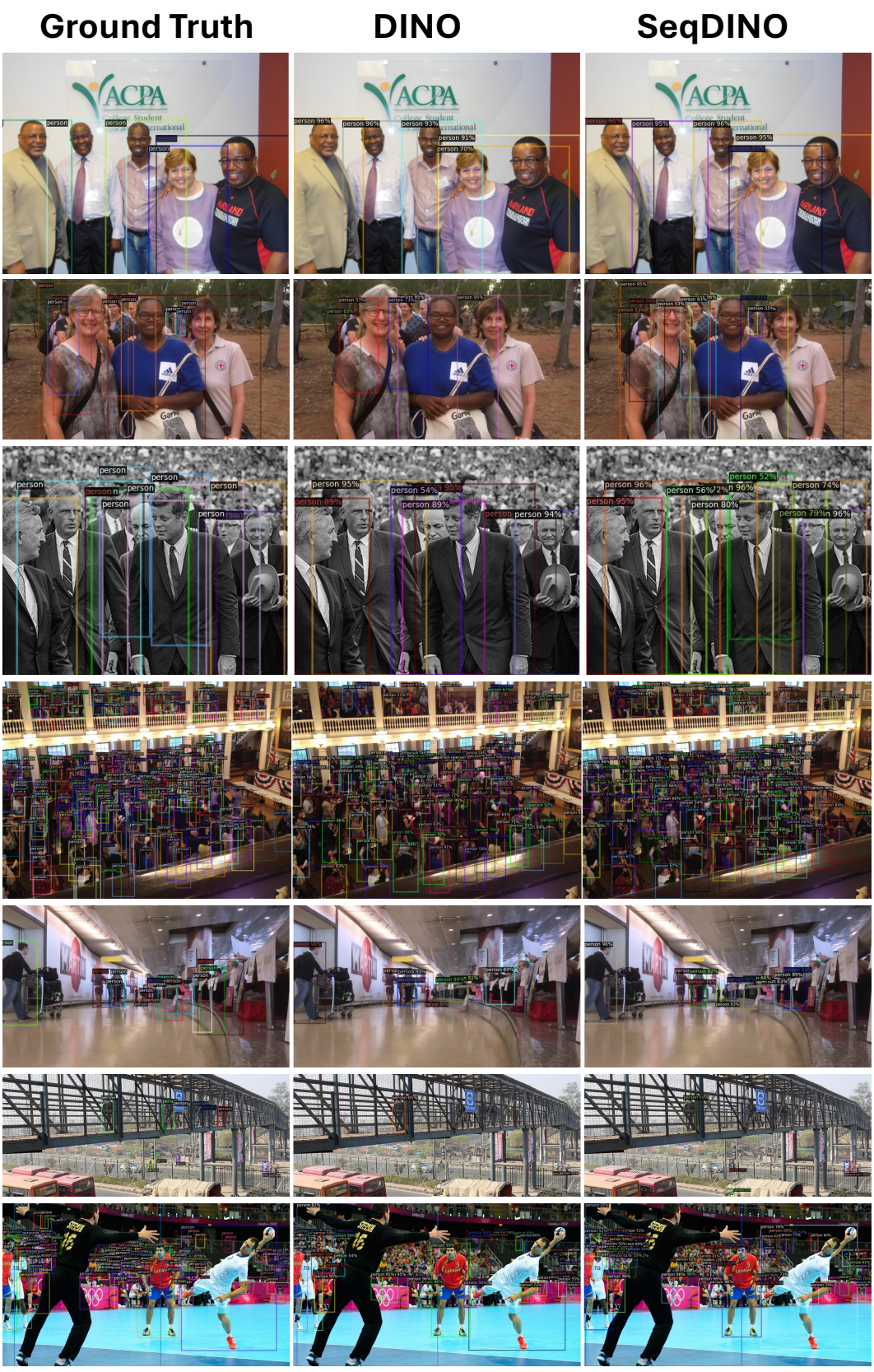


Figure 6: **Qualitative comparison on CrowdHuman val.**



Figure 7: **Qualitative results on V3Det val.**

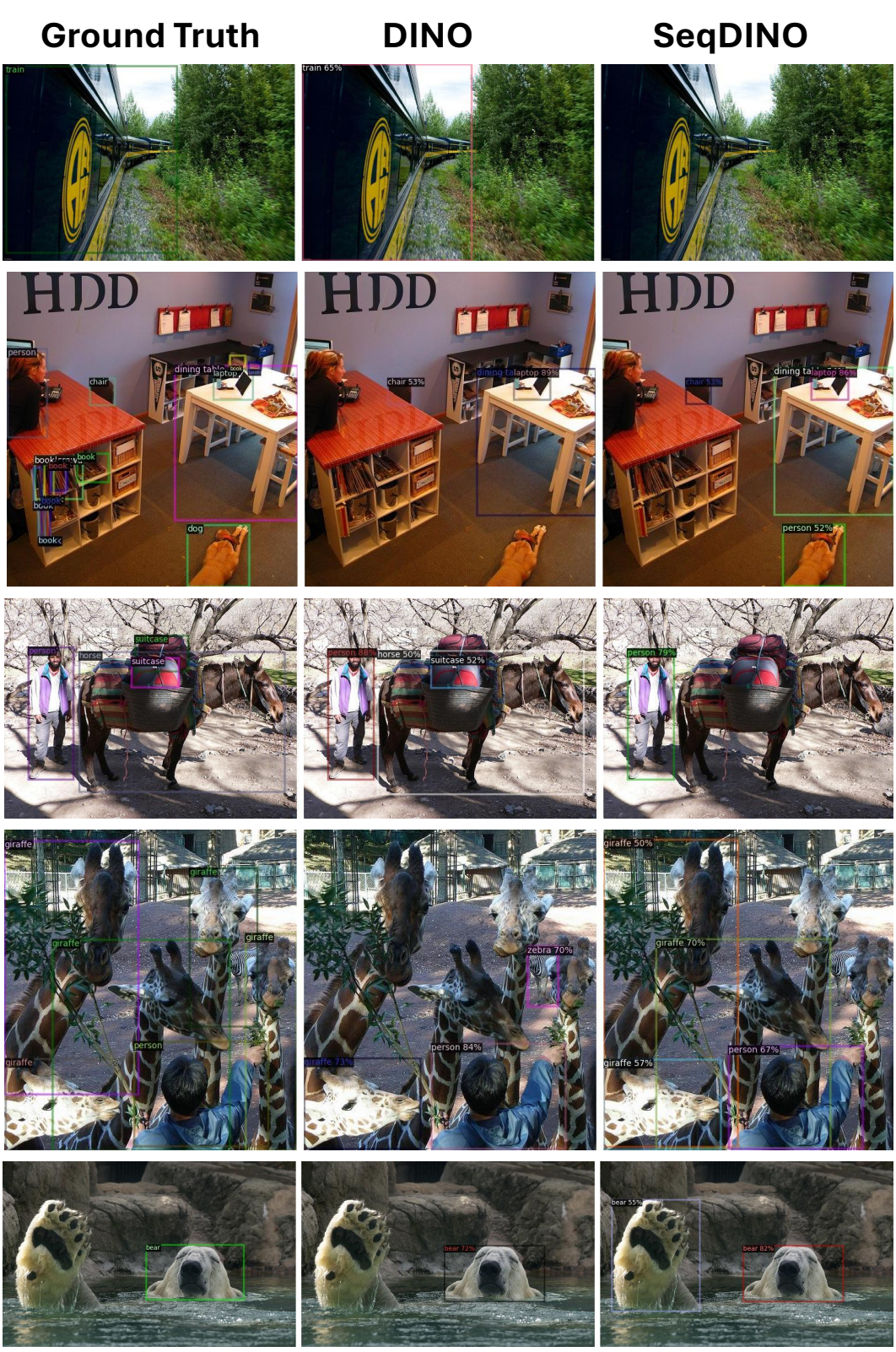


Figure 8: **Failure cases of SeqDINO.**