MANIFOLD-DETR: LOCAL MANIFOLD-ALIGNED ATTENTION FOR OBJECT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse detection transformers reduce encoder computation by updating selected image tokens, but allocating updates according to aggregate attention demand can favor already well-represented regions. Effective sparse detection also depends on how queries read local features and determine object boundaries. We present Manifold-DETR, a detection framework that aligns local feature gathering and sparse update allocation with the detection task. Local Manifold-Aligned Attention (LMAA) uses a direction–magnitude descriptor and its induced local geometry to correct query-predicted sampling locations, then evaluates content compatibility to aggregate the sampled features. Joint Boundary Refinement combines localization-specific boundary features with the query state to predict four coordinated box-side residuals. A task-calibrated decoder attention map (TDAM) uses matched-instance demand to supervise encoder token selection, with restricted reranking that preserves a high-demand core. On COCO, Manifold-DETR improves over Sparse-DETR by 2.1–2.8 AP with ResNet-50 across encoder keeping ratios of 10%–50% after 50 training epochs. At a 10% keeping ratio, it achieves 48.9 AP with ResNet-50 and 50.8 AP with Swin-T, surpassing the corresponding Sparse-DETR models at 50% while updating one fifth as many encoder tokens. These results demonstrate the value of task-aligned local feature reading and update allocation under sparse encoder budgets.

1 INTRODUCTION

Detection transformers formulate object detection as set prediction, learning to recognize and localize objects through query-based feature aggregation (Carion et al., 2020). Their computational cost, however, depends heavily on how image features are processed. Dense encoder attention compares all token pairs and updates every spatial location, even though image regions differ in their relevance to detection. High-resolution, multi-scale features make this redundancy particularly costly. Efficient detection therefore requires deciding not only how to reduce attention computation, but also where that computation is most useful.

Deformable DETR addresses the first issue by replacing dense attention with a small set of query-predicted sampling locations, reducing attention complexity from quadratic to linear in the number of tokens (Zhu et al., 2021). Yet its encoder still updates all tokens across feature scales. Sparse-DETR further reduces this cost by selecting only a subset of tokens for encoder updates (Roh et al., 2022). Its selector learns from a decoder attention map (DAM), which aggregates where decoder queries sample and how strongly they weight the sampled features. This connects upstream computation to downstream feature demand.

However, current attention demand and the value of further feature updates need not coincide. We call this distinction the *attention–utility mismatch*. A region repeatedly sampled by many queries can accumulate strong demand even when its object is already well represented. Conversely, a small or weakly represented object may receive little attention and consequently fewer encoder updates. Supervising selection with aggregate demand can reinforce this imbalance: frequently attended regions receive more refinement, while weakly attended regions have fewer opportunities to improve. Reducing the number of updated tokens alone does not resolve how the retained computation should support detection.

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.

This motivates us to reflect on more sophisticated approaches which can calibrate encoder update allocation while improving query-conditioned feature gathering. The two decisions are complementary. Token selection determines where computation is invested, whereas sampling and aggregation determine what that computation reads. Local feature geometry connects the latter to the detection task: feature descriptors express compatibility with a query, and their spatial variation identifies directions in which this compatibility improves. Feature-based alignment provides a foundation for converting these local differences into geometric updates (Arvanitidis et al., 2018; Sarlin et al., 2021; Zhao et al., 2021).

We introduce Local Manifold-Aligned Attention (LMAA) to improve local feature reading. Starting from query-predicted sampling locations, LMAA uses a direction–magnitude descriptor to compare local features with the query target. The descriptor Jacobian and its induced local metric translate this discrepancy into a bounded sampling correction. Content compatibility at the corrected locations then guides feature aggregation. Sampling and weighting thus work together to gather features aligned with the query's recognition and localization task.

To calibrate update allocation, we introduce a task-calibrated decoder attention map (TDAM). Detection matching associates queries with individual objects, whose spatial attention distributions are normalized before being combined with aggregate DAM. This makes each matched instance an explicit contributor to token-selection supervision, rather than allowing aggregate attention alone to determine update priorities. Restricted reranking protects the high-demand core of the selection target while revising a limited number of positions according to instance-level demand.

Joint Boundary Refinement complements these feature updates by estimating object extent. It reads localization-specific features along a coarse box and combines them with the query state to jointly predict all four boundary residuals. The resulting refinement uses information across the boundary to determine the final box.

Manifold-DETR combines local feature alignment, instance-aware update allocation, and joint boundary estimation in a sparse detection framework. Our contributions are as follows:

- We propose Local Manifold-Aligned Attention to align sparse feature gathering with the detection task. A direction–magnitude descriptor induces a local metric for query-conditioned sampling correction, and content compatibility at the corrected locations guides aggregation. Together, these operations connect where a query samples with the features it needs to recognize and localize an object.
- We introduce Joint Boundary Refinement to complement feature sampling with explicit estimation of object extent. Localization-specific features gathered along a coarse box are combined with its query state to jointly predict all four boundary residuals, allowing the refinement of each side to account for information from the entire boundary.
- We develop restricted task-calibrated DAM supervision to connect object-level attention demand with sparse encoder computation. Matched-instance attention calibrates the token-selection target, while a protected core and a limited reranking budget bound target replacement and preserve aggregate demand. The calibration acts during training without adding an inference-time selection branch.

2 Related work

2.1 Spatial Sampling and Query Design in Detection Transformers

Spatial query design determines how detection transformers associate an object representation with an image region. DETR starts from learned object queries and predicts a set of detections (Carion et al., 2020). Conditional DETR introduces spatial queries that focus different heads on informative regions, while DAB-DETR uses dynamic anchor boxes to represent and refine query position and extent (Meng et al., 2021; Liu et al., 2022). DN-DETR and DINO further develop query training and anchor initialization through denoising (Li et al., 2022; Zhang et al., 2023a). These approaches improve the spatial state from which a query gathers features; LMAA uses that state to initialize a feature-aware local sampling update.

Sparse sampling makes the content gathered by each query an explicit design choice. Deformable DETR predicts independent offsets across feature scales (Zhu et al., 2021). AdaMixer adapts spatial and scale sampling together with the decoding of sampled features (Gao et al., 2022), and Lite DETR samples keys to condition deformable attention weights on image content (Li et al., 2023). LMAA connects these two decisions: local feature variation corrects the sampling locations, and the sampled features determine content-dependent aggregation weights.

Upstream computation allocation determines which image features are updated before decoding. Sparse-DETR supervises a sparse encoder selector with decoder attention demand (Roh et al., 2022); Focus-DETR uses foreground supervision and multi-scale selection, while Salience DETR addresses scale bias through normalized salience targets and hierarchical filtering (Zheng et al., 2023; Hou et al., 2024). TDAM links this allocation to matched-object attention demand and restricts how far the resulting supervision can depart from its high-demand core.

2.2 Local Feature Geometry and Geometric Learning

A learned representation can define geometry through the way its features vary with the underlying coordinates. Pullback metrics formalize this connection by relating a coordinate displacement to a change in feature space; Arvanitidis et al. (2018) study the resulting geometry in deep generative models. For spatial features, the descriptor Jacobian similarly specifies how motion along each image direction changes the representation. This provides a local metric for feature-based alignment.

Geometric learning couples such alignment procedures with representations suited to the task. Deep Lucas-Kanade learns feature maps for iterative homography estimation, and PixLoc refines camera pose through feature alignment (Zhao et al., 2021; Sarlin et al., 2021). PRISE shapes the loss landscape around known image transformations to support local optimization (Zhang et al., 2023b). These methods show that learned feature differences can drive geometric updates when the representation and optimization procedure are designed together.

LMAA applies this principle to individual attention sampling locations rather than to a global image transformation or camera pose. Each location is aligned locally to a learned query descriptor, and the feature projections are trained through the detection objective. A direction–magnitude descriptor guides coordinate correction, while normalized descriptors evaluate the sampled content. The geometric update thus acts directly on the spatial locations from which attention gathers features.

3 Method

Manifold-DETR connects three stages of detection: LMAA determines where a query samples and how it aggregates the sampled features; Joint Boundary Refinement estimates the object's extent; and TDAM supervises the allocation of encoder updates. Figure 1 introduces the feature and box paths. We describe the detection framework, LMAA, Joint Boundary Refinement, and TDAM before presenting their common training objective.

3.1 Detection framework

We build on Sparse-DETR (Roh et al., 2022). A backbone produces multi-scale feature maps, which are projected to a common dimension and flattened into S tokens. A selector predicts scores before encoding and chooses K_b valid tokens for image b at update ratio ρ. Only these tokens serve as encoder update queries, while the full feature sequence remains available for value sampling. Each encoder layer writes the updated features back into their original positions, producing the memory used by the decoder.

Encoder proposals initialize Q object queries and reference boxes. Decoder layers alternate between query updates and box refinement: the current reference box conditions attention sampling, and the updated query predicts the next box. The final query and coarse box feed Joint Boundary Refinement to produce the final localization.

LMAA operates on a query state, a spatial reference, and the multi-scale features available to that attention layer. A query may be an encoder token being updated or a decoder object query. We denote the input feature maps of this attention operation by $\{M_l\}_{l=1}^{L}$ and describe its computation below.

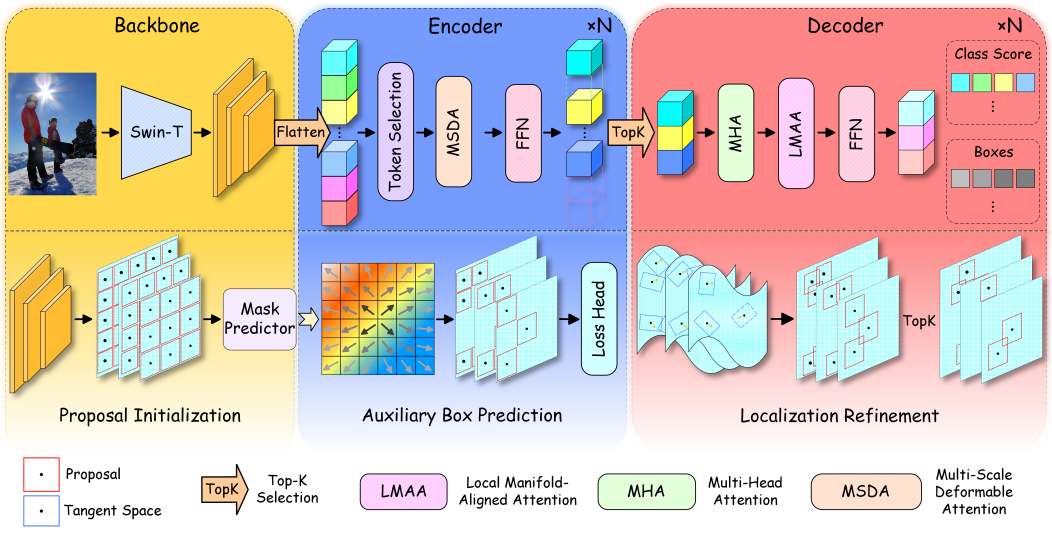


Figure 1: **Overview of Manifold-DETR.** The upper stream follows token and query updates; the lower stream follows proposal initialization and box refinement. Encoder proposals initialize decoder queries and reference boxes. LMAA corrects sampling locations and evaluates sampled content, illustrated here within decoder attention. The final query and coarse box feed Joint Boundary Refinement (Section 3.4), while restricted TDAM supervises encoder token selection during training (Section 3.5).

For a query q, head h, level l, and slot p, let $i = (q, h, l, p)$. The base module predicts an independent offset Δ_i and logit ℓ_i. For a box reference with center c_{ql} and size s_{ql}, its normalized sampling location is

$$p_i^0 = c_{ql} + \frac{\Delta_i}{P} \odot \frac{s_{ql}}{2}, \qquad x_i^0 = T_l p_i^0 - \tfrac{1}{2}\mathbf{1}, \quad T_l = \mathrm{diag}(W_l, H_l). \tag{1}$$

Here P is the number of slots per level, and x_i^0 uses feature cell coordinates. For a two-dimensional point reference, $p_i^0 = c_{ql} + T_l^{-1}\Delta_i$. Reference coordinates already include the valid region conversion.

3.2 Local Feature Geometry

Head specific linear projections produce a key field $k_{hl}(x) \in \mathbb{R}^d$ and a query target $t_{qh} \in \mathbb{R}^d$. Bilinear interpolation gives the key and its two spatial derivatives at x_i^0. The descriptor is normalized after interpolation, so the values and derivatives refer to the same continuous function.

For any projected vector k, define

$$\nu(k) = \sqrt{k^\top k + \varepsilon^2}, \qquad \phi(k) = \frac{k}{\nu(k)}, \qquad z_\gamma(k) = \begin{bmatrix} \phi(k) \\ \gamma \log \nu(k) \end{bmatrix}. \tag{2}$$

The normalized direction ϕ is used for content matching. The positional descriptor z_γ additionally retains magnitude, with the same transformation applied to the query target. This extra coordinate is computed from existing features and requires no new projection parameters.

The local residual and Jacobian are

$$r_i = z_\gamma\big(k_{hl}(x_i^0)\big) - z_\gamma(t_{qh}), \qquad J_i = \left.\frac{\partial z_\gamma(k_{hl}(x))}{\partial x}\right|_{x=x_i^0}. \tag{3}$$

The Jacobian is evaluated analytically through interpolation, normalization, and log magnitude. Appendix A gives the derivatives. The damped local metric is

$$C_i = J_i^\top J_i, \qquad H_i = C_i + \lambda I, \qquad \lambda > 0. \tag{4}$$

For fixed features, the augmented embedding $\Psi_{hl}(x) = [\sqrt{\lambda}\, x^\top, z_\gamma(k_{hl}(x))^\top]^\top$ has pullback metric $D\Psi_{hl}(x)^\top D\Psi_{hl}(x) = H_{hl}(x)$. This connects coordinate motion to feature variation (Arvanitidis

et al., 2018). On a configuration $X = (x_i)_i$, the local inner product is

$$\langle U, V \rangle_X = \sum_i U_i^\top H_i V_i. \tag{5}$$

The damping makes this metric positive definite even where the feature Jacobian is rank deficient. At a fixed sampling location, the metric describes feature variation, while the query defines the residual that guides the correction.

3.3 Sampling correction and content aggregation

Linearizing the residual yields a two dimensional regularized least squares problem,

$$v_i = \arg\min_{v \in \mathbb{R}^2} \frac{1}{2}\|r_i + J_i v\|_2^2 + \frac{\lambda}{2}\|v\|_2^2 = -H_i^{-1} J_i^\top r_i. \tag{6}$$

The feature field thus determines both the residual and the directional cost of reducing it. Related alignment methods optimize camera or image transformations (Sarlin et al., 2021; Zhao et al., 2021); here each attention slot receives its own local coordinate update.

Let $\mathrm{cap}_R(v) = v \min(1, R/\|v\|_2)$ for $v \neq 0$, and zero otherwise. The executed update is

$$d_i = \alpha_p(t)\mu_i \,\mathrm{cap}_R(v_i), \qquad x_i^1 = x_i^0 + d_i, \qquad p_i^1 = p_i^0 + T_l^{-1} d_i. \tag{7}$$

The indicator μ_i requires finite features and valid interpolation support at the initial and proposed locations, including padding. An unsupported correction leaves p_i^0 unchanged. The schedule α_p gradually enables the update and equals one at inference. The radius applies to the added correction in feature cells, not to the original offset.

We read the key again at x_i^1 and evaluate its directional content energy,

$$E_i^{\mathrm{c}} = \tfrac{1}{2}\|\phi(k_{hl}(x_i^1)) - \phi(t_{qh})\|_2^2, \qquad \widetilde{\ell}_i = \ell_i - \alpha_k(t)\beta E_i^{\mathrm{c}}. \tag{8}$$

Here $\beta = \beta_{\max}\sigma(\beta_{\mathrm{raw}})$ is a learned positive scale with upper bound $\beta_{\max} > 0$, and α_k enables the content branch. Sampled keys inform attention weights in Lite DETR (Li et al., 2023); LMAA also refines the coordinates from which both keys and values are read. Magnitude participates in the positional residual but does not directly rescale content scores.

Weights are normalized across all LP slots of each head. With V_{hl} denoting the base value field, the output is

$$\begin{aligned} a_{qhlp} &= \frac{\exp(\widetilde{\ell}_{qhlp})}{\sum_{l'=1}^{L}\sum_{p'=1}^{P}\exp(\widetilde{\ell}_{qhl'p'})}, \\ o_q &= W_O \,\mathrm{Concat}_{h=1}^{H}\left[\sum_{l=1}^{L}\sum_{p=1}^{P} a_{qhlp} V_{hl}(p_{qhlp}^1)\right]. \end{aligned} \tag{9}$$

This aggregation couples a bounded sampling correction with content evaluation at the corrected locations. Figure 2 summarizes the per-head computation.

3.4 Joint Boundary Refinement

LMAA improves the features gathered by a query; accurate localization additionally requires determining the object's extent. Joint Boundary Refinement (JBR) uses the final decoder state h_q and its coarse box $\bar{b}_q = (c_x, c_y, w, h)$ to estimate a coordinated update of all four sides. The coarse box is expressed in normalized image coordinates.

We place P_{b} sampling locations along each side of $\bar{b}_q$ and read them across the multi-scale encoder memory maps. Each level has a localization-specific projection g_l, from which bilinear interpolation produces boundary features f_{qlsj} for side $s \in \{\mathrm{L}, \mathrm{R}, \mathrm{T}, \mathrm{B}\}$ and position $j \in \{1, \ldots, P_{\mathrm{b}}\}$. The projected boundary features are not unit-normalized, retaining magnitude information for localization.

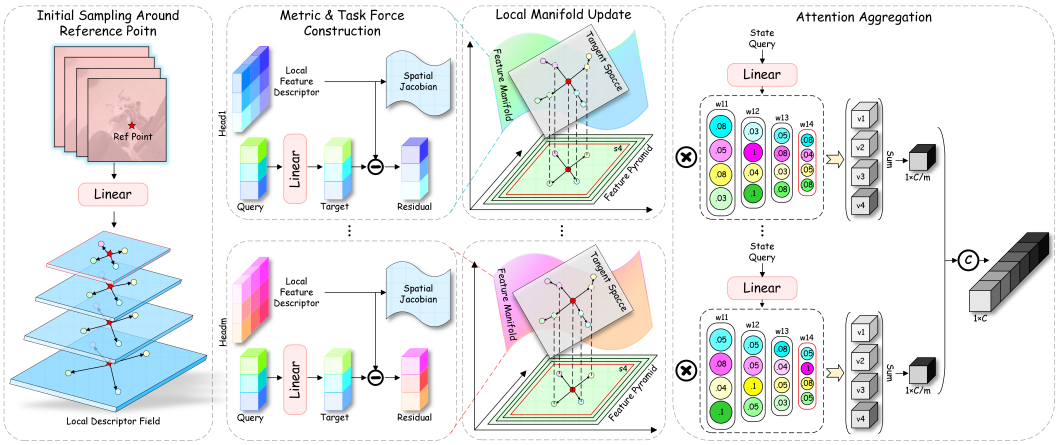


Figure 2: **Local Manifold-Aligned Attention (LMAA).** Independent query-predicted offsets initialize the sampling locations. For each head, the local feature descriptor and query target define a residual, and spatial feature derivatives induce a damped local metric for bounded sampling correction. Keys are resampled and values are read at the corrected locations. Normalized key–query compatibility adjusts the query logits before softmax across all levels and points within each head, followed by value aggregation. The manifolds and tangent planes visualize the local feature geometry underlying the sampling correction.

Concatenating all boundary features, their validity indicators v_{qlsj}, the projected final query, and the coarse box gives

$$z_q^{\text{box}} = \text{Concat}\big(\{f_{qlsj}\}_{l,s,j}, \{v_{qlsj}\}_{l,s,j}, W_{\text{box}} h_q, \bar{b}_q\big)\,. \tag{10}$$

A single prediction head jointly produces the four side residuals,

$$u_{qs} = \alpha_{\text{box}}(t)\,\tau\,\chi_{qs}\tanh\big([\text{MLP}(z_q^{\text{box}})]_s\big)\,, \tag{11}$$

where χ_{qs} indicates valid sampling support for side s, τ bounds the relative update, and α_{box} gradually activates box refinement. Because every side residual depends on the full boundary representation, the head can coordinate changes in box position and extent.

Let $(x_{\text{L}}, y_{\text{T}}, x_{\text{R}}, y_{\text{B}})$ be the coarse box corners. The updated corners are

$$\widehat{e}_q = (x_{\text{L}} - w u_{q\text{L}},\ y_{\text{T}} - h u_{q\text{T}},\ x_{\text{R}} + w u_{q\text{R}},\ y_{\text{B}} + h u_{q\text{B}})\,. \tag{12}$$

Positive residuals expand the corresponding side and negative residuals contract it. The corners are converted back to a valid center–size box $\widehat{b}_q$. JBR learns box refinement from boundary features through the detection loss, complementing LMAA's metric-based correction of attention sampling locations. Experimental settings are given in Section 4.1, with sampling coordinates and gradient paths in Appendix C.2.

3.5 Task Calibrated Supervision for Encoder Updates

DAM summarizes the decoder's spatial attention demand and supervises the upstream token selector. TDAM calibrates this demand using the correspondence between queries and object instances, then limits the change to the resulting selection target.

Let $\mathcal{V}_b$ be the valid token indices and $\mathcal{R}$ the operator that distributes attention weights from their sampling locations to the feature grids. The normalized aggregate demand is

$$D_{0b} = \mathcal{N}_b\Big(\mathcal{R}\Big(\{p_{qhlp}^{(n)}, a_{qhlp}^{(n)}\}_{n,q,h,l,p}\Big)\Big)\,, \tag{13}$$

where $\mathcal{N}_b$ masks invalid entries and normalizes nonnegative mass. Each layer contributes its actual sampling locations and weights, including LMAA corrections wherever the module is applied. Let $\pi_b(j)$ be the query assigned to object j by the standard Hungarian matching. Its final-layer demand is

$$D_{bj} = \mathcal{N}_b\Big(\mathcal{R}\Big(\{p_{\pi_b(j)hlp}^{(N)}, a_{\pi_b(j)hlp}^{(N)}\}_{h,l,p}\Big)\Big)\,. \tag{14}$$

Only matched instances with valid nonzero reading mass are included in $\mathcal{J}_b$. Their equal average and the candidate ranking score are

$$D_{Tb} = \frac{1}{|\mathcal{J}_b|} \sum_{j \in \mathcal{J}_b} D_{bj}, \qquad s_b = (1 - \eta(t)) D_{0b} + \eta(t) D_{Tb}, \quad 0 \le \eta(t) \le 1. \tag{15}$$

Each matched instance contributes equal total mass, distributed according to its actual decoder attention. Combining this instance demand with D_{0b} also accounts for attention assigned to contextual locations and other queries.

To bound the change in supervision, let $\pi_{0b} = (i_1, \ldots, i_{K_b})$ denote the original DAM target ranked by decreasing demand, with support A_{0b}. We protect its $K_b - m_b$ highest-demand tokens, where $m_b = \lfloor \delta K_b \rfloor$, and fill the remaining slots using s_b:

$$C_b = \{i_1, \ldots, i_{K_b - m_b}\}, \qquad E_b = \mathrm{Top}_{m_b}(s_b; \mathcal{V}_b \setminus C_b), \qquad A_b = C_b \cup E_b. \tag{16}$$

The original tail remains eligible for selection, so the rule limits the number of replacements without requiring every available slot to change. Ranking and tie-handling details are given in Appendix B.

For $K_b > 0$, the target satisfies

$$|A_b| = K_b, \qquad |A_b \setminus A_{0b}| \le m_b, \qquad \sum_{i \in A_b} D_{0b,i} \ge \left(1 - \frac{m_b}{K_b}\right) \sum_{i \in A_{0b}} D_{0b,i}. \tag{17}$$

These bounds protect the support and mass of the teacher target for a given forward pass. Their proofs and the corresponding Jaccard bound appear in Appendix B. If calibration is inactive, $m_b = 0$, or a valid demand map is unavailable, the original target is returned directly.

3.6 Learning and inference

The binary target $y_{bi} = \mathbb{1}[i \in A_b]$ supervises selector logits u_{bi} through a multilabel logistic objective. The overall training loss is

$$\begin{aligned} \mathcal{L}_{\mathrm{sel}} &= \mathrm{BCEWithLogits}(u, y), \\ \mathcal{L} &= \mathcal{L}_{\mathrm{det}} + \mathcal{L}_{\mathrm{aux}} + \lambda_{\mathrm{sel}} \mathcal{L}_{\mathrm{sel}}. \end{aligned} \tag{18}$$

The detection objective combines classification, ℓ_1 box regression, and generalized IoU (Rezatofighi et al., 2019). Hungarian matching uses the final classification scores and coarse boxes; the matched refined boxes receive the final regression loss. TDAM uses the same assignment to construct selector labels after decoding and supervises the selector predictions from that batch. Intermediate encoder and decoder predictions contribute $\mathcal{L}_{\mathrm{aux}}$. Loss reductions and padding conventions are detailed in Appendix C.5.

LMAA and JBR learn their feature projections through detection supervision, while TDAM supplies detached labels for the selector. At inference, LMAA corrects sampling and evaluates sampled content in the attention operations where it is applied. The decoder produces query states and coarse boxes, and JBR refines the boxes alongside the classification scores. TDAM target construction is used only during training.

4 Experiments

We evaluate Manifold-DETR on full COCO, comparing detection accuracy across backbones, training schedules, and encoder token keeping ratios.

4.1 Experimental setting

We train on COCO train2017 and evaluate bounding box AP, AP_{50}, AP_{75}, and AP for small, medium, and large objects on the 5,000-image val2017 set (Lin et al., 2014). ResNet-50 and Swin-T use ImageNet-1K pretraining (He et al., 2016; Liu et al., 2021). Training uses eight NVIDIA RTX 4090 GPUs with 24 GB memory each, two images per GPU, and AdamW. Our detector has six encoder and six decoder layers, 300 queries, and LMAA in the final decoder layer. Further settings appear in Appendix C.

Table 1: Detection results on COCO val2017. † denotes DETR results quoted from Carion et al. (2020); all other results are locally reproduced. Manifold-DETR uses LMAA, JBR, and restricted TDAM.

Method	Ep.	Keep	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50 backbone								
DETR†	500	100%	42.0	62.4	44.2	20.5	45.8	61.1
Deformable DETR	50	100%	46.9	66.4	50.1	29.7	50.5	61.8
Conditional DETR	50	100%	41.8	62.0	43.7	21.9	44.9	59.4
DAB-DETR	50	100%	42.7	64.1	44.9	21.8	47.0	61.6
Anchor DETR	50	100%	42.9	64.6	45.1	22.7	46.7	61.2
DN-DETR	50	100%	44.9	65.3	47.6	23.1	49.4	64.3
Lite-Deformable DETR	50	–	47.7	67.5	51.9	29.5	50.9	63.2
Lite-DINO	36	–	51.6	69.0	55.0	33.9	54.7	66.2
Focus-DETR	36	var.	51.6	69.5	55.9	34.4	54.0	65.4
Salience DETR	12	var.	49.6	68.4	54.7	32.9	53.4	63.2
Aug-DETR	12	–	50.8	68.4	55.2	32.4	53.3	65.1
Relation DETR	12	100%	52.6	69.6	57.7	36.3	56.2	67.4
Sparse-DETR	50	10%	46.8	66.3	50.4	28.8	50.1	60.9
		20%	47.0	66.6	50.5	28.9	50.4	61.8
		30%	47.3	67.0	50.6	29.0	50.4	62.4
		40%	47.5	67.1	51.0	29.2	50.7	62.7
		50%	47.6	67.5	51.3	29.3	50.7	63.2
Manifold-DETR	50	10%	48.9	67.9	52.7	29.9	52.0	63.9
		20%	49.4	68.1	53.3	30.0	52.2	64.1
		30%	50.1	68.1	53.4	30.1	52.5	64.4
		40%	50.1	68.5	53.7	30.1	52.7	64.5
		50%	50.4	69.0	54.0	30.2	52.8	64.8
Swin-T backbone								
Deformable DETR	50	100%	48.8	68.9	53.2	30.5	52.3	63.9
Sparse-DETR	50	10%	49.6	70.5	52.5	30.3	52.2	65.4
		20%	50.1	69.6	53.0	31.0	53.1	66.0
		30%	50.1	69.8	53.8	31.8	54.0	66.3
		40%	50.2	70.7	54.2	32.0	53.1	65.2
		50%	50.6	70.3	53.8	32.5	53.4	66.4
Manifold-DETR	50	10%	50.8	69.1	54.2	30.7	52.5	65.2
		20%	50.8	68.9	54.7	30.5	53.3	66.2
		30%	50.9	69.5	54.5	30.7	54.3	65.5
		40%	51.5	69.2	55.1	31.0	54.6	66.3
		50%	51.5	71.5	54.2	30.5	53.9	65.3

The keeping ratio ρ is the fraction of valid encoder tokens updated at each sparse layer; the full memory remains available for feature sampling. The main ratio sweeps use 50 epochs for both backbones; the 12-epoch Swin-T sweep is reported in Appendix E.2. In Table 1, DETR results are quoted from the original paper; all other results are locally reproduced with the training schedules shown.

4.2 Comparison with state-of-the-art detectors

Table 1 includes representative detectors such as Anchor DETR and Relation DETR (Wang et al., 2022; Hou et al., 2025). With ResNet-50 and $\rho = 10\%$, our 50-epoch model achieves 48.9 AP, exceeding Sparse-DETR, Deformable DETR, and Lite-Deformable DETR by 2.1, 2.0, and 1.2 points. At 50%, it reaches 50.4 AP versus Sparse-DETR's 47.6, with AP_{75} rising from 51.3 to 54.0. All six metrics improve over Sparse-DETR across the ratio sweep.

With Swin-T, our 50-epoch model reaches 50.8 AP at 10% and 51.5 at 50%, exceeding Sparse-DETR by 1.2 and 0.9 points. AP and AP_{75} improve at every ratio. The 12-epoch sweep appears in Appendix E.2.

4.3 Component ablations

Table 2 tests all eight component combinations. The complete model reaches 48.9 AP and 52.7 AP_{75}; removing LMAA, JBR, or TDAM reduces AP by 1.8, 0.2, or 0.4 points, respectively.

Internal designs (Appendix E.1) support combining position and content: LMAA achieves 47.9 AP versus 46.9 and 47.0 for either alone. Joint boundary prediction exceeds independent sides by 0.2 AP. Restricted TDAM improves over random refill and unrestricted instance calibration by 0.3 and 0.6 AP. Final-layer LMAA achieves 47.9 AP versus 47.3 for the final two layers and 46.9 for all six (Table 7), motivating our default placement.

Table 2: Component ablations with ResNet-50 on full COCO, 50 epochs, and $\rho = 10\%$. L/J/T denote LMAA/JBR/restricted TDAM.

L	J	T	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params (M)	FLOPs (G)	FPS	Latency (ms)
–	–	–	46.8	66.3	50.4	28.8	50.1	60.9	41.937	134.34	36.79	27.18
✓	–	–	47.9	67.4	52.0	29.6	51.3	62.6	41.970	134.73	17.04	58.67
–	✓	–	47.5	67.1	51.4	29.3	51.0	61.9	41.963	134.43	31.62	31.62
–	–	✓	47.3	66.6	51.1	29.1	50.5	61.8	41.937	134.34	36.79	27.18
✓	✓	–	48.5	67.5	52.1	29.6	51.4	63.0	41.996	134.83	15.88	62.97
✓	–	✓	48.7	67.7	52.6	29.7	51.8	63.6	41.970	134.73	17.04	58.67
–	✓	✓	47.1	66.5	51.3	29.1	50.2	62.0	41.963	134.43	31.62	31.62
✓	✓	✓	**48.9**	**67.9**	**52.7**	**29.9**	**52.0**	**63.9**	41.996	134.83	15.88	62.97

4.4 Accuracy under sparse encoder updates

At matched ratios, Manifold-DETR improves over Sparse-DETR by 2.1–2.8 AP with ResNet-50 and 0.7–1.3 with Swin-T. LMAA alone contributes 1.0–2.7 AP in the ResNet-50 sweep (Table 8). Our 10% model also exceeds Sparse-DETR at 50%: 48.9 versus 47.6 AP with ResNet-50, and 50.8 versus 50.6 with Swin-T, using one fifth as many updated encoder tokens. LMAA and JBR add only 0.059M parameters; TDAM adds none.

5 Conclusion

Manifold-DETR connects feature sampling, box refinement, and encoder computation. LMAA uses local feature geometry to correct sampling locations and evaluates the resulting sampled features for aggregation. Joint Boundary Refinement estimates object extent through coordinated updates of the four box sides. Restricted TDAM links matched-instance attention demand to encoder token selection while protecting the high-demand core of its supervision. Together, these components organize local feature use around the recognition and localization needs of detection queries.

REPRODUCIBILITY STATEMENT

The attention equations, descriptor derivatives, joint boundary updates, and target construction algorithm are specified in Section 3 and Appendices A–C. The support guarantees include complete proofs. Training schedules, evaluation conventions, and definitions of all mechanism statistics are provided in the implementation and evaluation appendices.

AI USE STATEMENT

Generative AI tools were used to assist with literature retrieval and code development.

ETHICS STATEMENT

The work concerns object detection on an established public benchmark and does not introduce new personal data or a deployment study. Object detection can support beneficial applications but can also be used for surveillance. Performance across object categories and sizes does not establish fairness across people or suitability for safety critical deployment. Such uses require assessment beyond the benchmark setting considered here.

REFERENCES

G. Arvanitidis, L. K. Hansen, and S. Hauberg. Latent Space Oddity: On the Curvature of Deep Generative Models. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1710.11379.

N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-End Object Detection with Transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. URL https://arxiv.org/abs/2005.12872.

Z. Gao, L. Wang, B. Han, and S. Guo. A Fast-Converging Query-Based Object Detector. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5364–5373, 2022. URL https://arxiv.org/abs/2203.16507.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. URL https://arxiv.org/abs/1512.03385.

X. Hou, M. Liu, S. Zhang, P. Wei, and B. Chen. Salience DETR: Enhancing Detection Transformer with Hierarchical Salience Filtering Refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17574–17583, 2024. URL https://arxiv.org/abs/2403.16131.

X. Hou, M. Liu, S. Zhang, P. Wei, B. Chen, and X. Lan. Relation DETR: Exploring Explicit Position Relation Prior for Object Detection. In *Computer Vision – ECCV 2024*, pages 89–105. Springer, 2025. doi: 10.1007/978-3-031-72973-7_6.

F. Li, H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang. DN-DETR: Accelerate DETR Training by Introducing Query DeNoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13619–13627, 2022. URL https://arxiv.org/abs/2203.01305.

F. Li, A. Zeng, S. Liu, H. Zhang, H. Li, L. Zhang, and L. 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*, pages 18558–18567, 2023. URL https://arxiv.org/abs/2303.07335.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, pages 740–755, 2014. doi: 10.1007/978-3-319-10602-1_48. URL https://doi.org/10.1007/978-3-319-10602-1_48.

Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. URL https://arxiv.org/abs/2103.14030.

S. Liu, F. Li, H. Zhang, X. Yang, X. Qi, H. Su, J. Zhu, and L. Zhang. DAB-DETR: Dynamic Anchor Boxes Are Better Queries for DETR. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2201.12329.

I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, and J. Wang. Conditional DETR for Fast Training Convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. URL https://arxiv.org/abs/2108.06152.

H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019. URL https://arxiv.org/abs/1902.09630.

B. Roh, J. Shin, W. Shin, and S. Kim. Sparse DETR: Efficient End-to-End Object Detection with Learnable Sparsity. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2111.14330.

P.-E. Sarlin, A. Unagar, M. Larsson, H. Germain, C. Toft, V. Larsson, M. Pollefeys, V. Lepetit, L. Hammarstrand, F. Kahl, and T. Sattler. Back to the Feature: Learning Robust Camera Localization from Pixels to Pose. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. URL https://arxiv.org/abs/2103.09213.

Y. Wang, X. Zhang, T. Yang, and J. Sun. Anchor DETR: Query Design for Transformer-Based Detector. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(3):2567–2575, 2022. doi: 10.1609/aaai.v36i3.20158.

H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum. DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection. In *International Conference on Learning Representations*, 2023a. URL https://arxiv.org/abs/2203.03605.

Y. Zhang, X. Huang, and Z. Zhang. PRISE: Demystifying Deep Lucas-Kanade with Strongly Star-Convex Constraints for Multimodel Image Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023b. URL https://arxiv.org/abs/2303.11526.

Y. Zhao, X. Huang, and Z. Zhang. Deep Lucas-Kanade Homography for Multimodal Image Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. URL https://arxiv.org/abs/2104.11693.

D. Zheng, W. Dong, H. Hu, X. Chen, and Y. Wang. Less is More: Focus Attention for Efficient DETR. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6674–6683, 2023. URL https://arxiv.org/abs/2307.12612.

X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.04159.

A NOTATION AND LOCAL GEOMETRY

A.1 NOTATION

Batch indices are omitted from the attention equations for readability. N denotes the number of decoder layers and L the number of feature levels. P counts attention slots per level, whereas K_b is the encoder update budget for image b.

Symbol	Definition
M_l	Input feature map at level l, with shape $C \times H_l \times W_l$.
q, h, l, p	Query, attention head, feature level, and sampling slot indices.
p_i^0, p_i^1	Original and corrected locations in normalized feature map coordinates; x_i^0, x_i^1 denote their feature-cell coordinates.
k_{hl}, t_{qh}	Projected key field and query target, each of dimension d.
z_γ, ϕ	Position descriptor and normalized content descriptor.
r_i, J_i, H_i	Position residual, its spatial Jacobian, and damped local metric.
$\bar{b}_q, \widehat{b}_q$	Coarse and refined boxes in normalized center–size coordinates.
u_{qs}, χ_{qs}	Relative side residual and boundary sampling validity indicator.
P_{b}	Number of JBR sampling locations per side and feature level.
D_{0b}, D_{Tb}	Normalized overall and matched-instance decoder demand.
A_{0b}, π_{0b}	Original binary target support and its stored ranking order.
K_b, m_b, C_b	Target cardinality, maximum replaceable slots, and preserved core.
$\widehat{A}_b, A_b$	Predicted encoder update subset and TDAM supervision subset.

A.2 BILINEAR VALUES AND SPATIAL DERIVATIVES

Let $x = (u, v)$ lie in a valid interpolation cell, with $a = \lfloor u \rfloor$, $b = \lfloor v \rfloor$, $\xi = u - a$, and $\zeta = v - b$. Write the four projected feature vectors as $f_{00}, f_{10}, f_{01}, f_{11}$. Bilinear interpolation is

$$k(x) = (1-\xi)(1-\zeta)f_{00} + \xi(1-\zeta)f_{10} + (1-\xi)\zeta f_{01} + \xi\zeta f_{11}. \tag{19}$$

Inside the cell, its analytic derivatives are

$$\begin{aligned} k_x &= (1-\zeta)(f_{10} - f_{00}) + \zeta(f_{11} - f_{01}), \\ k_y &= (1-\xi)(f_{01} - f_{00}) + \xi(f_{11} - f_{10}). \end{aligned} \tag{20}$$

No additional sampling stencil is needed beyond these four values. Let $D = [k_x, k_y]$ and $\nu = \sqrt{k^\top k + \varepsilon^2}$. The complete descriptor Jacobian is

$$J = \begin{bmatrix} (\nu^{-1} I - \nu^{-3} k k^\top) D \\ \gamma \nu^{-2} k^\top D \end{bmatrix}. \tag{21}$$

The first block differentiates normalization after interpolation. The final row differentiates the log magnitude. Normalizing the grid values before interpolation would define a different function and does not yield this Jacobian. Derivatives are in feature cell units. A normalized location is first converted by $x = T_l p - \frac{1}{2}\mathbf{1}$, so a cell displacement is converted back by T_l^{-1}.

A.3 METRIC AND UPDATE PROPERTIES

Proposition 1. *For $\lambda > 0$, $H = J^\top J + \lambda I$ is positive definite. On every differentiable interpolation region, it is the pullback metric of $\Psi(x) = [\sqrt{\lambda} x^\top, z_\gamma(k(x))^\top]^\top$.*

Proof. For any nonzero vector v, $v^\top H v = \|Jv\|_2^2 + \lambda \|v\|_2^2 > 0$. The derivative of Ψ is the vertical concatenation of $\sqrt{\lambda} I$ and J, whose Gram matrix is $\lambda I + J^\top J$. Thus the induced inner product equals H. □

For multiple slots, the product of these local embeddings has a block diagonal Gram matrix. This gives the inner product in Eq. (5). The construction is piecewise smooth because interpolation cells

and validity masks are discrete. It does not require a globally smooth image manifold or a geodesic computation. At a fixed feature location H is independent of the target query descriptor; the query changes the residual and the initial location at which the metric is evaluated.

The local objective in Eq. (6) has gradient $J^\top r + Hv$ and a unique minimizer $v = -H^{-1}J^\top r$. For $H = \left[\begin{smallmatrix} h_{11} & h_{12} \\ h_{12} & h_{22} \end{smallmatrix}\right]$ and $b = -J^\top r$, the solve is

$$v = \frac{1}{h_{11}h_{22} - h_{12}^2} \begin{bmatrix} h_{22}b_1 - h_{12}b_2 \\ h_{11}b_2 - h_{12}b_1 \end{bmatrix}. \tag{22}$$

The determinant is positive in exact arithmetic. The implementation also checks finite inputs and outputs.

For a fixed feature field and query, the gradient of the unlinearized feature energy at the current point is $g = J^\top r$. If $g \neq 0$, then $g^\top v = -g^\top H^{-1} g < 0$. Positive radial rescaling preserves this infinitesimal descent direction, and $\|d\|_2 \leq \alpha_p R$. This local statement does not imply that a finite step across interpolation cells reduces the nonlinear energy or the detection objective. The radial cap bounds the executed step; it is not claimed to solve an anisotropic trust region subproblem exactly.

A.4 Direction and magnitude contributions

Let r_{d} and J_{d} be the residual and Jacobian for the normalized direction. Define $e_m = \log \nu(k) - \log \nu(t)$ and $a = \nabla_x \log \nu(k)$. The position system can be written as

$$\begin{aligned} J^\top r &= J_{\mathrm{d}}^\top r_{\mathrm{d}} + \gamma^2 a e_m, \\ J^\top J &= J_{\mathrm{d}}^\top J_{\mathrm{d}} + \gamma^2 a a^\top. \end{aligned} \tag{23}$$

Thus magnitude changes both the residual driven update and its local cost, rather than multiplying the final step by γ. Setting $\gamma = 0$ recovers normalized geometry. It does not disable positional correction or content modulation. Query and key norms are responses of different learned projections, not calibrated object probabilities. Their comparison can favor either a larger or a smaller local magnitude.

B Restricted target construction

B.1 Demand maps and ranking semantics

The DAM accumulation operator $\mathcal{R}$ is inherited from Sparse-DETR. For a sampling location and weight it scatters nonnegative mass to neighboring encoder grid entries using the base grid indexing. TDAM uses this same operator for the aggregate and instance maps, rather than changing the discretization while changing the supervision. Only valid encoder entries contribute to normalization. A per instance map is normalized before instances are averaged, so valid instances have equal total contribution regardless of how many legal interpolation contributions they contain.

For each image, the baseline target is supplied together with its actual ordered top K indices. This stored order defines the protected core, including the baseline handling of equal scores; the core is not reconstructed by sorting a second copy of the DAM. Refill candidates are stably ranked by decreasing calibrated score. Ties prefer original positive indices in their supplied order, followed by other valid indices in token-index order. Inactive paths return the supplied labels exactly.

There is no rasterized ground truth support in this procedure, including its fallback paths. Matching identifies instances, but low confidence matched queries are not discarded by a new threshold. A query with no valid reading mass cannot supply an instance map. Empty instance collections return the supplied original target. Invalid budgets, duplicate indices, or nonfinite ranking scores are implementation errors rather than instructions to invent replacement labels.

Algorithm 1 Restricted TDAM target for one image

Require: Original binary target y_0, ordered positive indices π_0, nonnegative raw DAM $\overline{D}_0$, valid mask v, actual budget K, final layer reads, standard matching, α_p, $\eta_{\max}$, δ
Ensure: A binary target y with the original budget
Run target construction without gradients
$\eta \leftarrow \eta_{\max}\alpha_p$; $m \leftarrow \lfloor \delta K \rfloor$
if $\alpha_p = 0$ or $\eta = 0$ or $\delta = 0$ or $m = 0$ **then**
    return y_0
end if
Mask $\overline{D}_0$ with v
if the valid mass of $\overline{D}_0$ is zero **then**
    return y_0
end if
$D_0 \leftarrow \overline{D}_0 / \sum_i \overline{D}_{0i}$
Initialize an empty collection of instance maps
for each query associated with a valid matched instance **do**
    $D_j \leftarrow \mathcal{R}$(actual final layer locations and weights of the query)
    Mask D_j with v
    if D_j has positive valid mass **then**
        Append $D_j / \sum_i D_{ji}$ to the collection
    end if
end for
if the collection is empty **then**
    return y_0
end if
$D_T \leftarrow$ equal average of the collection
$s \leftarrow (1-\eta)D_0 + \eta D_T$
$C \leftarrow$ first $K - m$ indices in the supplied order π_0
$E \leftarrow \mathrm{Top}_m(s; \{i : v_i = 1\} \setminus C)$
$y_i \leftarrow \mathbb{1}[i \in C \cup E]$
return y

B.2 Cardinality, overlap, and demand preservation

Proposition 2. *Assume $D_0 \geq 0$ on the valid grid, the base support A_0 contains $K > 0$ indices ordered by nonincreasing D_0, and $0 \leq m \leq K$. The restricted set $A = C \cup E$ has cardinality K, introduces at most m indices outside A_0, and satisfies*

$$\frac{|A \cap A_0|}{|A \cup A_0|} \geq \frac{K-m}{K+m}, \qquad \sum_{i \in A} D_{0i} \geq \left(1 - \frac{m}{K}\right) \sum_{i \in A_0} D_{0i}. \tag{24}$$

Proof. The core has $K - m$ elements and the refill has m distinct elements outside the core, giving $|A| = K$. Since $C \subseteq A_0$, new indices can only come from E, so $|A \setminus A_0| \leq m$. Write $s = |A \cap A_0| \geq K - m$. Both sets have size K, and their Jaccard overlap is $s/(2K - s)$, which increases with s. Its minimum over the allowed range is $(K - m)/(K + m)$.

For the mass bound, let the original selected masses be $a_1 \geq \cdots \geq a_K \geq 0$. The average of the largest $K - m$ entries is at least the average of all K entries. Hence $\sum_{j=1}^{K-m} a_j \geq (K - m) \sum_{j=1}^{K} a_j / K$. The refill contributes nonnegative mass, so the same lower bound holds for A. The cases $m = 0$ and $m = K$ follow directly. □

For a fixed preserved core, choosing the top m remaining scores also maximizes $\sum_{i \in A} s_i$ over valid sets of size K containing C. The constraint is stronger than merely bounding the total number of replacements, since particular original indices are protected. These are properties of each target construction. They do not require the trained selector to reproduce the same overlap, nor do they impose temporal stability on the base DAM as training changes the detector.

At $\delta = 0.1$, the maximum allowed replacement is ten percent up to integer rounding. Actual replacement can be lower because the original tail remains eligible. If $r_b = |A_b \setminus A_{0b}|/K_b$, then for equal cardinalities the exact per image identity is

$$J_b = \frac{1 - r_b}{1 + r_b}, \qquad r_b = \frac{1 - J_b}{1 + J_b}. \tag{25}$$

Nonlinear conversion is applied per image when aggregating this statistic, not to an already averaged Jaccard value.

B.3 Training and inference dependency

The predicted subset $\widehat{A}_b$ is computed before the encoder runs. The decoder then produces the attention maps and detection assignment used by Algorithm 1. Its output supervises the selector logits from that same forward pass. The labels therefore influence parameter updates without retroactively changing which tokens were encoded. This distinguishes training supervision from ground truth selection in the forward graph.

The overall map can be accumulated using the base scatter operation. Instance maps can be normalized and accumulated one at a time, so target construction need not materialize a dense tensor for every query and spatial token simultaneously. Sorting and matching information are used only during training. Inference retains the same learned selector and top K update rule, with no target construction branch.

C Implementation and training details

C.1 Tensor layout and feature paths

After projection and concatenation, the encoder memory has shape $[B, S, 256]$, with level shapes and starting indices stored separately. Object queries have shape $[B, Q, 256]$. LMAA predicts no additional sampling slots. Its original and corrected locations have shape $[B, Q, H, L, P, 2]$, and attention weights have shape $[B, Q, H, L, P]$. Keys use eight channels per head, giving a nine dimensional positional descriptor after appending log magnitude. Values retain the original 32 channels per head when $H = 8$.

The key field is formed by projecting encoder memory before restoring its level grids. Projection outputs at padding are masked so that a projection bias does not create artificial features. Interpolation returns a value, two feature cell derivatives, and a support mask. The initial location, descriptor norm, solve, and proposed corrected neighborhood are checked before an update is accepted. Values use the existing deformable attention kernel. The added geometry does not alter the kernel's signature or the normal residual and feedforward operations of the decoder.

For a four dimensional reference, the base offset is scaled by its box size as in Eq. (1). A two dimensional reference uses $p_i^0 = c_{ql} + T_l^{-1}\Delta_i$. The reference has already been converted to the normalized coordinates of each padded feature map using its valid ratio. Applying that ratio again would change the sampling location. Feature cell displacements are divided by the current level's width and height only when converting the added correction back to normalized coordinates.

C.2 Joint Boundary Refinement implementation

JBR takes the final decoder states $[B, Q, 256]$, coarse boxes $[B, Q, 4]$, and the multi-scale encoder memory. Its boundary sampling locations are defined in normalized valid-image coordinates. For $t_j \in \{-1/4, 1/4\}$, the relative anchors on the left, right, top, and bottom sides are

$$a_{\mathrm{L}j} = (-1/2, t_j), \quad a_{\mathrm{R}j} = (1/2, t_j), \quad a_{\mathrm{T}j} = (t_j, -1/2), \quad a_{\mathrm{B}j} = (t_j, 1/2). \tag{26}$$

The image-coordinate location is $(c_x, c_y) + (w, h) \odot a_{sj}$. It is multiplied by the valid ratio of level l once to obtain the normalized coordinates used for interpolation on that padded feature map. Two locations on each of four sides, read at four levels, give 32 sampled features per query.

Each level has an independent linear projection from 256 to 16 localization channels. These projected features are not unit-normalized. Invalid or nonfinite samples are replaced by zero, and their

validity indicators are concatenated with the features. The final query is separately projected from 256 to 16 dimensions. Together with the four coarse box coordinates, the input in Eq. (10) has $32 \times 16 + 32 + 16 + 4 = 564$ entries. A joint MLP with dimensions $564 \to 9 \to 4$ and a GELU activation predicts the left, right, top, and bottom residuals. Its final linear layer is initialized to zero.

The side indicator χ_{qs} is one when at least one sample on that side has valid support across the levels. The relative residual bound is $\tau = 0.1$. After the corner update in Eq. (12), the box is converted to center–size form; centers are clamped to $[0, 1]$ and sizes to $[10^{-4}, 1]$, with finite-value safeguards. When all side residuals are zero, the implementation returns the coarse box exactly while retaining residual derivatives. The localization projections and joint head are separate from LMAA's key projections and local metric.

C.3 Gradients and numerical evaluation

The memory and query supplied to the key projections are detached. Projection weights and biases receive gradients through the local solve and content energy. The initial locations are not globally detached, so their original offset branch remains connected to interpolation and the update. The unchanged value path continues to train the detector. TDAM targets, matching indices, and attention maps used for supervision are detached; selector logits retain the original loss gradient.

For JBR, the memory and final query inputs to the localization projections are detached, as are the coarse box used to construct boundary locations and the box dimensions used to scale the side residuals. The localization projections and joint MLP receive gradients from the final box loss. The direct coarse-box term in Eq. (12) remains differentiable, so box regression also trains the preceding detector. Classification logits and intermediate box predictions follow their existing paths.

The geometry and selector target construction use floating point computations with finite checks. Query chunking can reduce the activation footprint without changing attention normalization, because the normalization is local to each query and head. Rematerialized forwards do not duplicate diagnostic accumulation. The reference configuration uses single precision, and any mixed precision comparison uses the same numerical treatment for all matched variants.

The descriptor has $\varepsilon = 10^{-6}$. The content scale is parameterized by $\beta = 4\sigma(\beta_{\mathrm{raw}})$, with $\beta_{\mathrm{raw}} = -\log 3$ at initialization. The added projections are initialized independently of the existing detector's random number stream, preserving its matched initial tensors. Setting both activation coefficients to zero recovers the original attention operation. Setting only $\gamma = 0$ recovers normalized positional geometry, not original attention.

C.4 Activation and optimization

For training progress $t \in [0, 1]$, define

$$\alpha(t; a, b) = \min\left(1, \max\left(0, \frac{t - a}{b - a}\right)\right). \tag{27}$$

The content coefficient is $\alpha_k(t) = \alpha(t; 0.05, 0.10)$, the position coefficient is $\alpha_p(t) = \alpha(t; 0.10, 0.20)$, and $\eta(t) = 0.5\alpha_p(t)$. JBR uses $\alpha_{\mathrm{box}}(t) = \alpha(t; 0.20, 0.30)$. In a TDAM only ablation, its numerical schedule is retained even though the attention module is absent. Inference uses the completed LMAA and JBR coefficients but never constructs a TDAM label.

The full-COCO configuration uses the following settings. Parameters controlling the base matcher, focal classification objective, and intermediate encoder and decoder losses follow the matched Sparse-DETR configuration.

Platform and batch. Eight NVIDIA RTX 4090 GPUs with 24 GB each, two images per GPU, no accumulation, and a global batch of 16. Other allocations preserve the same effective batch.

Architecture. Six encoder and six decoder layers, 300 queries, eight attention heads, four feature levels, and four sampling slots per level. Hidden and feedforward dimensions are 256 and 1024, with dropout 0.1. The default encoder update ratio is $\rho = 0.1$.

Local geometry. The position descriptor has eight direction entries and one log-magnitude entry, with $\gamma = 0.5$, damping $\lambda = 0.1$, and radius $R = 0.5$ feature cell.

Selection supervision. TDAM uses $\eta_{\max} = 0.5$ and replacement limit $\delta = 0.1$.

Boundary refinement. JBR uses 32 boundary samples and 16 localization channels, with relative side bound $\tau = 0.1$ and activation over 20%–30% of training.

Optimization. AdamW (Loshchilov and Hutter, 2019) uses weight decay 10^{-4}, detector learning rate 2×10^{-4}, backbone learning rate 2×10^{-5}, and gradient clipping at 0.1. Offset and reference projections use a learning-rate multiplier of 0.1.

Schedules and evaluation. Training lasts 12 or 50 epochs, with a learning-rate drop after epoch 10 or 40, respectively. Evaluation resizes the short side to 800 pixels and limits the long side to 1333 pixels.

Training samples use the base random resize, crop, and horizontal flip policy. Backbone classification pretraining is fixed within each comparison. ResNet-50 and Swin-T preserve their respective baseline feature extraction and parameter freezing settings. Swin-T uses its standard patch and window configuration (Liu et al., 2021). Both provide multi-scale features that are projected to the detector dimension. Evaluation uses the final checkpoint, without test-time augmentation or external detection data. LMAA is applied in the final decoder cross-attention layer unless placement is varied; the remaining attention layers use standard MSDA.

With data parallel training and accumulation, the effective batch equals the number of processes times the per process batch times the accumulation count. Learning rates and activation schedules are defined per optimizer update, not per microbatch. The actual optimizer group learning rates are recorded to prevent an automatic batch scaling rule from treating accumulation as a smaller effective batch. Global target counts are used consistently for losses normalized by object count. Clipping occurs once after the accumulated gradient is available. The geometry and target algorithm are identical across device counts.

C.5 Learning objective and matching

Hungarian matching uses the final classification logits and the coarse boxes before JBR. The resulting assignment is shared by final classification, regression of the refined boxes, and TDAM target construction. The ℓ_1 and generalized IoU losses are evaluated on the refined boxes, while intermediate encoder and decoder predictions retain their own detection objectives (Roh et al., 2022; Rezatofighi et al., 2019). Classification and box loss coefficients follow the baseline. Background and unmatched queries remain in the aggregate DAM even though they do not enter the matched instance average.

For selector logits and labels of shape $[B, S]$, the multilabel logistic loss in Eq. (18) is

$$\mathcal{L}_{\mathrm{sel}} = -\frac{1}{BS} \sum_{b=1}^{B} \sum_{i=1}^{S} \left[y_{bi} \log \sigma(u_{bi}) + (1 - y_{bi}) \log\bigl(1 - \sigma(u_{bi})\bigr) \right]. \tag{28}$$

This is evaluated with the baseline multilabel soft-margin implementation. Padded entries have zero target and remain in the tensor reduction, so both the denominator and padded-entry treatment are identical across supervision variants. TDAM changes the valid positive target indices, not the reduction or coefficient. No additional loss is applied to descriptor norms or to the entries of H. Matching identifies instance correspondence during supervision; no spatial ground truth raster or target-dependent update set enters the encoder forward pass.

D Evaluation and mechanism definitions

D.1 Target change and predictive selection

The original target A_{0b}, calibrated target A_b, and predicted encoder subset $\widehat{A}_b$ are three different sets. Target replacement is $r_b = |A_b \setminus A_{0b}|/K_b$. Target Jaccard is given in Eq. (25). Retained demand is

$$M_b = \frac{\sum_{i \in A_b} D_{0b,i}}{\sum_{i \in A_{0b}} D_{0b,i}}. \tag{29}$$

Zero denominator cases use the inactive return and are counted separately. This ratio concerns the original attention mass, not the probability of containing an object.

Predictive recalls are

$$R_{0b} = \frac{|\widehat{A}_b \cap A_{0b}|}{K_b}, \qquad R_{Tb} = \frac{|\widehat{A}_b \cap A_b|}{K_b}. \tag{30}$$

Their difference compares alignment to two teacher supports under the same model. It does not alone establish a selector capacity limit. Weighted summaries use total intersections or replacements divided by the corresponding summed budgets. Per image Jaccard is also retained, so its nonlinear relation to replacement is not applied after averaging.

The moved out and moved in sets are analyzed separately by feature level and original DAM rank. This identifies whether calibration preserves a broad reading pattern or mainly exchanges one scale for another. Demand maps are also compared on fixed images and fixed transforms at saved checkpoints. Changes caused by different random crops are not interpreted as teacher instability.

D.2 Object coverage and missed predictions

Coverage is measured independently of the supervision algorithm, including for models that do not use instance calibration. For each annotated box, feature cell regions are mapped to input coordinates and tested for positive overlap with the box. Target coverage records whether A_b contains at least one such cell; predictive coverage uses $\widehat{A}_b$. The number of supporting cells and their feature levels are also recorded. This is a spatial diagnostic, not a guarantee that the selected representation contains the object or that the detector recognizes it.

Coverage denominators count valid annotated instances. Main small, medium, and large reports follow COCO's original annotation area groups. Any separate grouping by transformed input area is labeled explicitly and is not substituted for official AP size categories. Target coverage, predicted coverage, candidate proposal recall, and final detection recall are distinct diagnostics. A token that is not updated is still readable from memory, so absence from the update set is not interpreted as physical deletion of its image content.

D.3 Local sampling and proposal trajectories

Displacement diagnostics record the norm before and after capping, cap frequency, valid support fraction, feature level, and changes in the positional and directional energies. Values are given in feature cell units, with conversion to image coordinates when visualized. Energy statistics exclude invalid comparisons and near zero denominators. The positional energy for one representation is not ranked against the numerical energy of a different representation as if they were a common detection measure.

A fixed-checkpoint intervention requires recomputing classification scores, queries, and affected decoder operations from the intervention point. A norm-matched random control preserves the sampled displacement norm; its valid-support rate is a separate diagnostic. A local detection-loss comparison uses a fixed reference assignment, whereas AP uses the normal evaluator. This distinguishes a local loss test from an end-to-end detector evaluation.

Query traces use a final layer assignment to follow the same query and object pair through decoder layers. Center and side errors are evaluated in image coordinates, with their normalization stated. Unmatched objects and missed candidates are recorded in addition to matched traces. A sampling displacement is not itself a box displacement, and its quality is not measured by assigning an IoU to the sampling point.

D.4 Computation

Inference timing uses one NVIDIA RTX 4090, FP32 with TF32 disabled, batch size one, and a fixed 800×1333 synthetic input. Cost measurements use deterministically initialized architectures without trained detection checkpoints, with all method schedules fully active. Each configuration uses at least 30 warmup forwards followed by three batches of 50 timed forwards; the four distinct component graphs at $\rho = 10\%$ are confirmed on the same GPU with 50 warmup and 250 timed forwards. Synchronized wall time covers model inference and GPU bounding-box postprocessing with the top 100 detections; file I/O and data transfers are excluded. Latency is the mean time per forward and FPS is its reciprocal. The measured path is the released eager implementation,

including auxiliary heads, rather than an optimized export. Nominal FLOPs use MAC=1 for matrix and convolution operations, count scalar arithmetic, and include deformable sampling and the local solve; indexing and comparisons are excluded. These architecture-level measurements are separate from the trained-model AP results.

E ADDITIONAL COMPARISONS AND ABLATIONS

Native encoder budgets. Focus-DETR and Salience DETR use layer-dependent update policies, denoted by *var.* in Table 1 (Zheng et al., 2023; Hou et al., 2024). Lite-DETR interleaves high- and low-level feature updates rather than assigning a uniform keeping ratio (Li et al., 2023). A dash indicates that no single ratio is reported for that reproduction. The explicit 10%–50% sweep applies to Sparse-DETR and Manifold-DETR.

E.1 INTERNAL COMPONENT DESIGN

Tables 3, 4, and 5 examine internal choices with ResNet-50, 50 epochs on full COCO, and $\rho = 10\%$. Attention variants are trained without JBR or TDAM. Boundary-refinement variants retain LMAA and original DAM. Supervision variants retain LMAA and standard decoder box regression, without JBR.

Table 3: Attention-design ablations with original DAM supervision and no JBR.

Attention	Position descriptor	Metric	Content	AP	AP_{75}
Original attention	None	None	Query logits	46.8	50.4
Content only	None	None	Key energy	46.9	50.4
Position only	Direction + magnitude	Full	Query logits	47.0	50.7
Normalized geometry	Direction	Full	Key energy	46.9	50.7
Raw geometry	Raw key	Full	Key energy	47.6	51.6
Isotropic update	Direction + magnitude	Isotropic	Key energy	46.8	50.3
LMAA	Direction + magnitude	Full	Key energy	**47.9**	**52.0**

Table 4: Boundary-refinement ablations with LMAA and original DAM supervision.

Final refinement	AP	AP_{75}
None	47.9	52.0
Query-only residual head	47.9	51.8
Boundary features, shared projection	48.1	52.0
Localization projection, independent sides	48.3	51.9
Localization projection, joint sides (JBR)	**48.5**	**52.1**

Table 5: Token-selection supervision ablations with LMAA, no JBR, and $\rho = 10\%$.

Target	Task source	Reranking	AP	AP_{75}
Original DAM	All layers, all queries	None	47.9	52.0
Final layer demand	Final layer, all queries	Restricted	48.0	52.0
Instance calibration	Final layer, matched queries	Unrestricted	48.1	52.2
Random refill	Uniform noncore candidates	Restricted	48.4	52.4
TDAM	Final layer, matched queries	Restricted	**48.7**	**52.6**

Attention geometry. Combining sampling correction and content weighting yields 47.9 AP, compared with 46.9 and 47.0 for either operation alone. The direction–magnitude descriptor improves over normalized and raw geometry by 1.0 and 0.3 AP, respectively. Replacing the anisotropic metric with an isotropic update reduces AP to 46.8.

Boundary features and joint prediction. A query-only residual head leaves AP at 47.9. Reading boundary features raises it to 48.1, and localization-specific projection with independent side heads reaches 48.3. Joint prediction of all four sides gives the highest AP and AP_{75}, at 48.5 and 52.1.

Task-calibrated supervision. Final-layer demand reaches 48.0 AP, compared with 47.9 for original DAM. Unrestricted instance calibration improves AP to 48.1, while restricted TDAM reaches 48.7. Random refill under the same protected-core constraint obtains 48.4 AP. Thus restricted TDAM exceeds both unrestricted calibration and random refill, supporting the combination of instance-aware supervision and controlled target replacement.

E.2 Short-schedule accuracy across token budgets

Table 6 reports the 12-epoch Swin-T sweep on full COCO, with 300 inference queries and six encoder and six decoder layers. Each model is trained with its own 12-epoch schedule rather than taken from an intermediate checkpoint of a 50-epoch run.

Manifold-DETR improves AP over Sparse-DETR at every keeping ratio, by 2.0, 2.4, 3.0, 3.2, and 3.2 points from 10% to 50%. AP_{75} improves by 3.1–4.1 points and AP_S by 2.6–3.1 points. At 10%, it achieves 41.4 AP, exceeding Sparse-DETR at 50% by 0.9 points with one fifth as many encoder tokens updated per sparse layer.

Table 6: Keeping-ratio ablation with Swin-T on full COCO after 12 training epochs.

Method	Keep	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Sparse-DETR	10%	39.4	58.2	41.6	22.4	42.3	51.7
	20%	39.9	58.9	42.0	22.7	43.0	52.2
	30%	40.1	59.5	42.3	22.9	43.5	52.7
	40%	40.3	60.0	42.6	23.1	43.6	53.1
	50%	40.5	60.3	42.7	23.1	43.7	53.3
Manifold-DETR	10%	41.4	60.8	44.7	25.0	44.8	55.1
	20%	42.3	61.6	45.4	25.5	45.5	55.5
	30%	43.1	62.1	46.0	25.9	45.9	55.8
	40%	43.5	62.4	46.5	26.1	46.2	56.1
	50%	43.7	62.6	46.8	26.2	46.5	56.3

E.3 Placement and update budget

Tables 7 and 8 vary decoder placement and encoder update ratio separately. A higher update ratio is a cost reference rather than an assumed accuracy upper bound.

Table 7: LMAA decoder placement with ResNet-50, 50 epochs, $\rho = 10\%$, and six encoder and six decoder layers.

Active decoder layers	LMAA layers	AP	AP_{75}	AP_S	FLOPs (G)	FPS	Latency (ms)
None	0	46.8	50.4	28.8	134.34	36.79	27.18
Final layer	1	47.9	52.0	29.6	134.73	17.04	58.67
Final two layers	2	47.3	51.8	29.4	135.12	11.12	89.95
All decoder layers	6	46.9	50.8	29.0	136.69	4.66	214.68

Table 8: Encoder keeping-ratio ablation with ResNet-50 on full COCO after 50 training epochs.

Method	Keep	AP	AP_S	FLOPs (G)	FPS	Latency (ms)
Sparse-DETR	10%	46.8	28.8	134.34	36.79	27.18
Sparse-DETR + LMAA	10%	47.9	29.6	134.73	17.04	58.67
Manifold-DETR	10%	48.9	29.9	134.83	15.88	62.97
Sparse-DETR	20%	47.0	28.9	146.44	35.52	28.15
Sparse-DETR + LMAA	20%	48.0	29.7	146.84	16.77	59.64
Manifold-DETR	20%	49.4	30.0	146.93	15.33	65.23
Sparse-DETR	30%	47.3	29.0	158.55	33.74	29.63
Sparse-DETR + LMAA	30%	49.2	29.7	158.94	16.37	61.09
Manifold-DETR	30%	50.1	30.1	159.04	15.14	66.05
Sparse-DETR	40%	47.5	29.2	170.66	32.01	31.24
Sparse-DETR + LMAA	40%	49.9	29.8	171.06	15.95	62.69
Manifold-DETR	40%	50.1	30.1	171.15	14.95	66.88
Sparse-DETR	50%	47.6	29.3	182.77	31.00	32.26
Sparse-DETR + LMAA	50%	50.3	30.1	183.16	15.58	64.19
Manifold-DETR	50%	50.4	30.2	183.26	14.70	68.02

E.4 Update-target reconstruction

Table 9 compares three target-construction rules using identical decoder outputs from a fixed checkpoint, without retraining. Restricted TDAM changes 8.40% of the target tokens and retains 98.95% of the original DAM mass, versus 15.52% and 97.65% for unrestricted instance demand. The comparison isolates the effect of restricting target replacement.

Table 9: Target-reconstruction diagnostics using the same fixed checkpoint. Replacement, Jaccard overlap, and retained DAM mass are measured relative to original DAM.

Target rule	Replaced (%)	Jaccard	DAM mass (%)
Original DAM	0.00	1.00	100.00
Unrestricted instance demand	15.52	0.73	97.65
Restricted TDAM	8.40	0.84	98.95

E.5 Computational cost

Architecture and measurement. Parameter counts include frozen parameters. LMAA adds 32,897 parameters and JBR adds 25,685; TDAM adds no inference parameters. The complete ResNet-50 model has 41,995,787 parameters at every keeping ratio. At $\rho = 10\%$, it uses 134.83 nominal GFLOPs versus 134.34 for the baseline, while latency increases from 27.18 to 62.97 ms. The small operation-count increase therefore does not imply a small runtime overhead in the current eager implementation. Tables 10 and 11 use the same hardware and timing protocol.

Table 11 compares Sparse-DETR, LMAA without JBR or TDAM, and the complete framework at the same $\rho = 10\%$. All models use ResNet-50, six encoder and six decoder layers, and the inference protocol above.

Table 10: ResNet-50 accuracy and inference cost. AP uses 50-epoch full-COCO training; costs follow Appendix E.5.

Method	Keep	AP	FLOPs (G)	Latency (ms)	FPS
Sparse-DETR	10%	46.8	134.34	27.18	36.79
Sparse-DETR	50%	47.6	182.77	32.26	31.00
Sparse-DETR + LMAA	10%	47.9	134.73	58.67	17.04
Sparse-DETR + LMAA	50%	50.3	183.16	64.19	15.58
Manifold-DETR	10%	48.9	134.83	62.97	15.88
Manifold-DETR	30%	50.1	159.04	66.05	15.14
Manifold-DETR	50%	50.4	183.26	68.02	14.70

Table 11: Component-level inference cost at $\rho = 10\%$: one RTX 4090, 800×1333 input, FP32, and batch size one.

Method	Params (M)	FLOPs (G)	FPS	Latency (ms)
Sparse-DETR	41.937	134.34	36.79	27.18
LMAA only	41.970	134.73	17.04	58.67
Manifold-DETR	41.996	134.83	15.88	62.97

F Scope and limitations

LMAA's bounded local correction cannot recover missing backbone detail or guarantee lower detection loss. TDAM calibrates existing demand; its guarantees concern target-set preservation rather than detection accuracy. Target construction adds training work without an inference branch, while LMAA adds runtime beyond its nominal FLOP increase. These limitations motivate reporting localization, selection, and measured cost separately.