Ref-YOLO-World: Decoupling Text-Side Computation for Efficient Referring Expression Comprehension on Edge Devices

Yuki Aoki[1] Hiroo Bekku[2] Tadashi Okoshi[1] Jin Nakazawa[1]
[1]Keio University [2]Independent Researcher
aokiti@sfc.wide.ad.jp, bekku_squall@keio.jp, {slash, jin}@sfc.keio.ac.jp

Abstract

Many high-accuracy referring expression comprehension (REC) methods produce image-dependent multimodal representations. These representations must be recomputed for every video frame, even when the referring expression remains unchanged. Fast open-vocabulary detectors such as YOLO-World instead pre-encode text once and reuse it throughout the stream. However, encoding each referring expression as a single embedding prevents explicit token-level correspondence. We propose Ref-YOLO-World, which separates text-side computation that depends only on the referring expression from image processing and reuses its outputs across frames. Two lightweight modules inject cached per-token text features into YOLO-World's detection path. Ref-YOLO-World achieves 81.29% Acc@0.5 on RefCOCOg val, the highest among REC models with a decoupled text branch, and consistently outperforms a fine-tuned YOLO-World. Precomputation reduces per-image inference time by 18.5%, yielding 47.0 FPS. A Core ML version runs at 15.73 FPS on an iPhone 15 Pro, demonstrating efficient on-device REC. Code is available at https://github.com/sakusaku3939/Ref-YOLO-World.

1. Introduction

Referring expression comprehension (REC) identifies an object in an image from a natural-language expression that can combine attributes, locations, and inter-object relations, such as "the person wearing black and sunglasses." REC has primarily been formulated and evaluated on individual image–expression pairs [14, 23, 35]. In real-world use, the visual input may change over time while the user continues to search for the same target. For example, in an interactive camera application on a mobile device, the user may describe a target once, after which the system localizes it in each incoming frame as the camera viewpoint changes. In this situation, the visual input must be processed anew for every frame, whereas the referring expression remains unchanged across frames. Efficient per-frame inference is therefore important, particularly on resource-constrained devices.

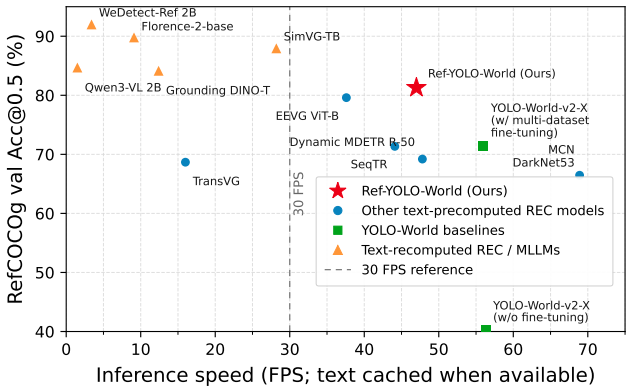


Figure 1. Accuracy–speed comparison on RefCOCOg val. FPS is measured under Stream inference for methods that reuse image-independent text representations and under Single inference otherwise.

Recent REC approaches achieve high accuracy through bidirectional vision-language fusion [7, 17] or multimodal large language models (MLLMs) [2, 11, 33]. Models with bidirectional fusion update text representations using image features, making them image-dependent and preventing their reuse across frames. Several REC models compute text representations independently of the image input, allowing the resulting representations to be reused across frames [5, 8, 21, 30, 38]. Prior work has not explicitly evaluated inference efficiency when these text representations are reused across frames.

Efficient open-vocabulary object detectors have also been developed for real-time and edge deployment [11, 28,

© 2027 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

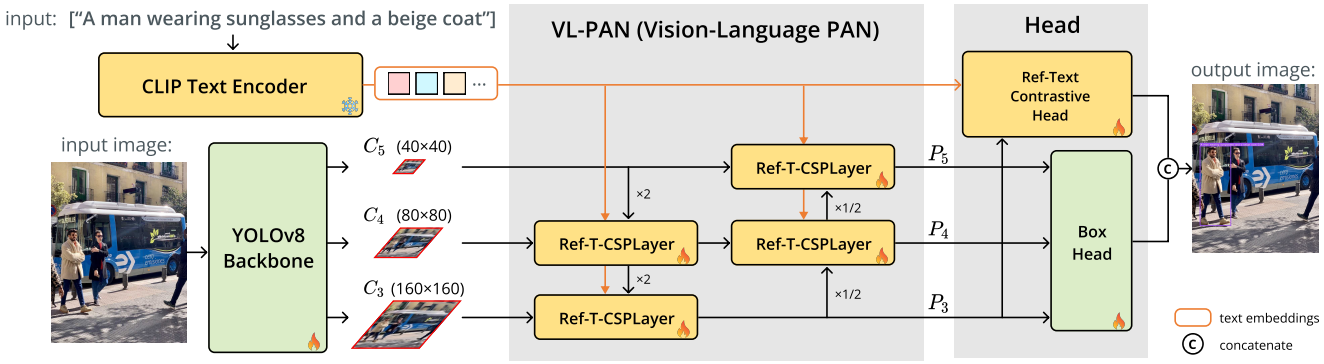


Figure 2. Overview of Ref-YOLO-World. Yellow components are introduced by our method. Orange paths depend only on the referring expression and can be precomputed.

31]. YOLO-World [6], one such detector, has an image-independent text branch whose outputs can be precomputed. Adapting this structure to REC requires preserving per-token language information for compositional referring expressions.

We therefore propose Ref-YOLO-World, which preserves per-token text features while reusing expression-dependent computation across frames. Ref-YOLO-World keeps text representations independent of image features and computes the text branch and its added projections once per expression. Ref-T-CSPLayer injects per-token text features into the neck, while Ref-Text Contrastive Head incorporates the full expression context into candidate-region scores. Both modules use unidirectional cross-attention, allowing text-side computation that depends only on the expression to be reused across frames. We refer to such REC models as having a decoupled text branch (Sec. 2.1).

As shown in Fig. 1, Ref-YOLO-World outperforms fine-tuned YOLO-World on all eight RefCOCO splits and, on RefCOCOg val, achieves the highest accuracy among REC models with a decoupled text branch at 81.29% Acc@0.5 and 47.0 FPS under streaming inference. After fine-tuning on TNL2K [32], Ref-YOLO-World attains AUC comparable to temporal trackers without using temporal state. Its Core ML model runs at 15.73 FPS on an iPhone 15 Pro, demonstrating efficient on-device REC.

Our contributions are threefold:

- We introduce Ref-YOLO-World, a one-stage REC model that performs text-side computation once per expression and reuses its outputs across frames.
- We introduce Ref-T-CSPLayer and Ref-Text Contrastive Head to pass per-token text features through the detection path and factor their text-side computation into quantities precomputed once per expression.
- We evaluate Ref-YOLO-World on standard REC benchmarks, under streaming inference, and on mobile hardware; it achieves the highest accuracy among decoupled models on RefCOCOg val and runs at 15.73 FPS on an iPhone 15 Pro.

2. Related work

2.1. Referring expression comprehension

Referring expression comprehension takes an image and a referring expression as input and predicts the bounding box of the described object. End-to-end Transformer-based methods, including MDETR [13] and TransVG [8], form a major line of visual grounding research. SimVG [7] further achieves high accuracy by exploiting large-scale vision-language pretraining. GLIP [15] and Grounding DINO [17] accept natural-language inputs, including referring expressions, through joint training for object detection and phrase grounding. Another line of work uses MLLMs, such as Florence-2 [33] and Qwen3-VL [2], to generate coordinates directly. WeDetect-Ref [11] instead applies an MLLM to classify candidate regions from a detector without fused features, avoiding autoregressive coordinate generation. When a referring expression is applied to each frame of a camera stream, bidirectional-fusion models must recompute their fused text representations for every frame since image features update these representations.

In contrast, other REC models can compute text representations independently of image features, including MCN [21], TransVG [8], SeqTR [38], Dynamic MDETR [30], EEVG [5], and SimREC [20]. Dynamic MDETR uses a dynamic Transformer decoder, and EEVG removes redundant visual tokens to accelerate inference. SimREC processes language and image inputs with separate encoders before multimodal fusion in a one-stage REC architecture. We refer to an REC model as having a *decoupled text branch* when it computes text representations without image input and allows them to be cached and reused across images without rerunning the text branch. Cross-modal operations may use these cached representations in the per-image path without updating them from image fea-

tures. These models also use per-token text features. However, they fuse them with image features in multimodal fusion layers that remain in the per-image path. Ref-YOLO-World instead injects per-token features into the neck and head of a one-stage detector through unidirectional cross-attention and precomputes the text-side projections of these modules once per expression (Sec. 3.5). We exploit this separability for streaming inference and evaluate the benefit of cross-frame reuse for computation that depends only on the referring expression.

2.2. Efficient open-vocabulary object detection

Open-vocabulary object detection uses vision-language representations learned through large-scale image-text contrastive training [27] to detect arbitrary categories specified by text. YOLO-World [6] combines CLIP text features with a one-stage detector from the YOLO family [12], enabling real-time open-vocabulary object detection. Grounding DINO 1.5 Edge [28] targets edge deployment by restricting vision-language fusion to high-level image features, reducing computation and demonstrating inference on an NVIDIA Orin NX. YOLOE [31], another YOLO-based model, introduces re-parameterizable region-text alignment to transfer text-feature adaptation into the detector with minimal inference overhead. WeDetect [11] formulates object detection as region-text retrieval and achieves real-time inference without deep vision-language fusion.

In YOLO-World's prompt-then-detect paradigm, text is pre-encoded into an offline vocabulary, and the resulting representations are used both in the neck to aggregate image features and in the head to match candidate regions. These two components provide a basis for introducing token-level correspondence and full-expression context required by REC into the detection path while keeping text-side computation separate from image processing. However, YOLO-World aggregates each expression into a single text feature and therefore cannot explicitly model token-level correspondences for attributes, locations, and inter-object relations. We address this limitation by introducing vision-language correspondence over all token features while retaining YOLO-World's one-stage detection architecture and image-independent, cacheable text representations.

2.3. Tracking by natural language

Tracking by Natural Language (TNL) specifies a target with a natural-language description. TNL2K [32] is a large-scale benchmark for this task and defines a tracking-by-natural-language-only setting, in which the target is specified without an initial ground-truth bounding box. Trackers such as JointNLT [37], UVLTrack [22], and MemVLT [10] maintain temporal state through templates or memory. This state exploits appearance continuity, although erroneous updates may affect subsequent predictions [32]. In contrast, our method performs stateless, frame-wise detection from the current image and the referring expression. Therefore, a prediction at one frame does not alter the inputs to later frames.

3. Ref-YOLO-World

3.1. Problem formulation

Given a fixed referring expression q and a sequence of images $\{I_t\}$ captured by a camera, our task is to estimate the bounding box $\hat{b}_t = f(I_t, q)$ of the object described by q in every frame. If the computation in f that depends only on q is performed once and reused for subsequent frames, per-frame inference executes only the detection path conditioned on cached expression features. We therefore adopt the decoupling property defined in Sec. 2.1 as a design requirement. This formulation includes conventional single-image REC as a special case, allowing direct evaluation on standard benchmarks.

Figure 2 presents the overall architecture of Ref-YOLO-World. The model builds on YOLO-World [6] and consists of a backbone that extracts image features, a Vision-Language Path Aggregation Network (VL-PAN, hereafter the neck) that combines multiscale features, and a head that scores the similarity between region and text features. We extend the two components that compute vision-language correspondence: T-CSPLayer in the neck and Text Contrastive Head for final scoring. The main inference path remains the feed-forward path of a one-stage detector, and the added operations are limited to lightweight cross-attention.

YOLO-World matches image regions only against an aggregated text feature taken from the end-of-sequence (EOS) token of the CLIP [27] text encoder (Sec. 2.2). Aggregating the expression into a single vector removes the token dimension, preventing downstream modules from explicitly aligning individual attributes, spatial terms, and relations with image regions. In contrast, Ref-YOLO-World retains the hidden states at all token positions, including the beginning-of-sequence (BOS) token, text tokens, and EOS token, and passes these per-token text features $\mathbf{T}^{\text{token}} \in \mathbb{R}^{L \times C}$ to the neck and head. Here, L is the number of token positions including padding and C is the feature dimension. For cross-attention, we add a padding mask $\mathbf{m} = (m_1, \ldots, m_L)$ to the logits, where $m_\ell = 0$ at valid token positions and $m_\ell = -\infty$ at padding positions.

3.2. Ref-T-CSPLayer

Ref-T-CSPLayer replaces T-CSPLayer in the neck (Fig. 3). The original T-CSPLayer weights the image feature $\mathbf{X}_l$ at resolution level l using Max-Sigmoid Attention, $\mathbf{G}_l^{\max} = \sigma(\max_j \mathbf{X}_l \mathbf{t}_j^\top / \sqrt{d} + b_{\text{attn}})$, which is based on the maximum similarity to an aggregated text feature $\mathbf{t}_j$. Here, σ is the sigmoid function, d is the feature dimension used

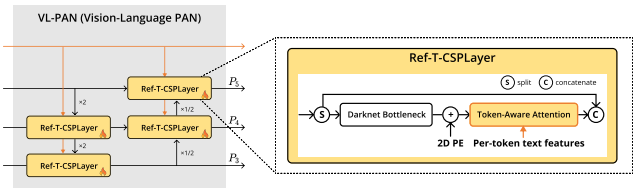


Figure 3. Structures of VL-PAN and Ref-T-CSPLayer.

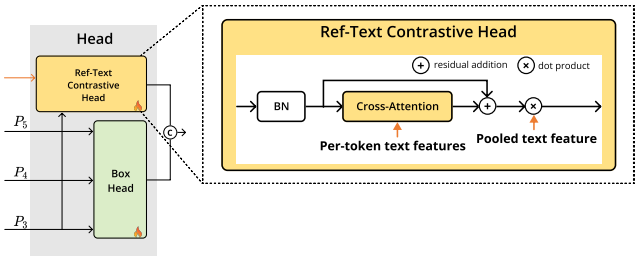


Figure 4. Structure of Ref-Text Contrastive Head.

for similarity computation, and b_{attn} is a learnable bias. Although efficient, this formulation weights each spatial position according to its strongest text response and provides only a limited mechanism for combining multiple constraints expressed by attribute and relation words.

Our module forms queries from image features and keys and values from all token features:

$$\begin{aligned} \mathbf{Q}_l &= \mathbf{X}_l \mathbf{W}_Q + \mathbf{b}_Q + \mathbf{P}_l, \\ \mathbf{K} = \mathbf{T}^{\text{token}} \mathbf{W}_K + \mathbf{b}_K, &\quad \mathbf{V} = \mathbf{T}^{\text{token}} \mathbf{W}_V + \mathbf{b}_V. \end{aligned} \tag{1}$$

The weights $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$ and biases $\mathbf{b}_Q, \mathbf{b}_K, \mathbf{b}_V$ are learnable projection parameters. $\mathbf{P}_l$ is the normalized 2D positional embedding described below. We use the query-key similarity averaged over all token positions to construct a gate:

$$\mathbf{G}_l = \sigma\left(\frac{1}{L}\sum_{\ell=1}^{L}\left(\frac{\mathbf{Q}_l \mathbf{K}_\ell^\top}{\sqrt{d}} + b_{\text{attn}}\right)\right), \tag{2}$$

where $\mathbf{K}_\ell$ is the ℓ-th row of $\mathbf{K}$. The gate averages over the fixed length L, including padding positions, to retain an output scale comparable to that of the pretrained block. We also compute a cross-attention context term

$$\mathbf{C}_l = \text{softmax}_\ell\left(\frac{\mathbf{Q}_l \mathbf{K}^\top}{\sqrt{d}} + \mathbf{m}\right)\mathbf{V} \tag{3}$$

and add it to the image features through a residual path. The output is

$$\mathbf{Y}_l = \text{Conv}_{3\times3}(\mathbf{X}_l) \odot \mathbf{G}_l + \beta\ \text{Conv}_{1\times1}(\mathbf{C}_l), \tag{4}$$

where $\odot$ denotes element-wise multiplication and β is a learnable scale factor for the added attention path. The gate $\mathbf{G}_l$ weights image features by query–key similarity averaged across tokens, whereas the context term $\mathbf{C}_l$ aggregates token features with token-wise attention. We refer to the per-token alignment mechanism formed by the gate in Eq. (2) and the context term in Eq. (3) as *Token-Aware Attention*.

Referring expressions frequently contain spatial words such as "left," "right," and "behind." To align these words with image coordinates, we map the normalized 2D coordinates of each feature map to the image-feature dimension with a small MLP:

$$\mathbf{P}_l(y, x) = \text{MLP}_{\text{PE}}\left(\frac{x}{W_l - 1}, \frac{y}{H_l - 1}\right). \tag{5}$$

We add this positional embedding to the image-side query in Eq. (1), where H_l and W_l are the height and width of the level-l feature map and (y, x) is a position on it.

3.3. Ref-Text Contrastive Head

Ref-T-CSPLayer updates each spatial location on a feature map. The head then determines whether each candidate region jointly satisfies the conditions in the referring expression. YOLO-World computes the score

$$s_i^{\text{base}} = \alpha\, \mathbf{x}_i \mathbf{t}^\top + b_s \tag{6}$$

from the inner product between the region feature $\mathbf{x}_i$ after BatchNorm and the L2-normalized aggregated text feature $\mathbf{t}$. Here, $\alpha = \exp(\tau)$, where τ is a learnable logit-scale parameter corresponding to the logarithm of the inverse temperature, and b_s is a learnable bias. This score is efficient, although it cannot incorporate the context of multiple expression tokens into a region feature immediately before scoring.

We insert a single cross-attention layer before the score computation (Fig. 4). The head uses projections independent of those in the neck, denoted by the superscript h. These projections form a query $\mathbf{q}_i = \mathbf{x}_i \mathbf{W}_Q^h + \mathbf{b}_Q^h$ from the region feature and keys $\mathbf{K} = \mathbf{T}^{\text{token}} \mathbf{W}_K^h + \mathbf{b}_K^h$ and values $\mathbf{V} = \mathbf{T}^{\text{token}} \mathbf{W}_V^h + \mathbf{b}_V^h$ from all token features. The resulting context term is

$$\mathbf{r}_i = \left[\text{softmax}_\ell\left(\frac{\mathbf{q}_i \mathbf{K}^\top}{\sqrt{d}} + \mathbf{m}\right)\mathbf{V}\right]\mathbf{W}_O^h + \mathbf{b}_O^h, \tag{7}$$

and the updated region feature is $\mathbf{x}_i' = \mathbf{x}_i + \lambda \mathbf{r}_i$. Here, $\mathbf{W}_O^h$ and $\mathbf{b}_O^h$ are the output-projection weight and bias, and λ is a learnable scale factor. The final score is the sum of the base score and a correction term:

$$s_i = s_i^{\text{base}} + \Delta s_i, \qquad \Delta s_i = \alpha\lambda\, \mathbf{r}_i \mathbf{t}^\top. \tag{8}$$

3.4. Zero initialization

Strong initial contributions from the added attention paths may disrupt pretrained image–text relations and destabilize fine-tuning. Following ControlNet [36] and ReZero [1], we therefore initialize the learnable scale factors β and λ to zero, so the added residual paths initially make no contribution. This guarantee applies only to the residual paths in Eqs. (4) and (8). Since the gate in Eq. (2) replaces Max-Sigmoid Attention, the initial neck output does not strictly match that of the original YOLO-World.

3.5. Text-side precomputation

Since the attention weights in our cross-attention modules change with the image, precomputation is limited to text-side projections. All queries come from image features, allowing key and value projections to be computed once per referring expression (orange paths in Fig. 2). For the neck, we precompute and cache $\mathbf{K}$ and $\mathbf{V}$ from Eq. (1).

For the head, we apply an equivalent transformation based on matrix associativity [34] through the final score computation. Substituting the query projection from Eq. (7) into the pre-Softmax similarity scores yields the expression-specific coefficients

$$\mathbf{A} = \frac{\mathbf{W}_Q^h \mathbf{K}^\top}{\sqrt{d}} \in \mathbb{R}^{C \times L}, \qquad \mathbf{c} = \frac{\mathbf{b}_Q^h \mathbf{K}^\top}{\sqrt{d}} \in \mathbb{R}^{L}. \tag{9}$$

This transformation combines query generation and the projected query-key product for each candidate region into one matrix product, $\mathbf{x}_i\mathbf{A} + \mathbf{c}$. Expanding the contribution of the value-side context feature to the score and using the fact that the Softmax outputs sum to one gives

$$\mathbf{u} = \lambda \left(\mathbf{V}\mathbf{W}_O^h \mathbf{t}^\top + \left(\mathbf{b}_O^h \mathbf{t}^\top\right) \mathbf{1}_L \right) \in \mathbb{R}^{L}. \tag{10}$$

The score correction becomes

$$\Delta s_i = \alpha \, \mathrm{softmax}_\ell \left(\mathbf{x}_i\mathbf{A} + \mathbf{c} + \mathbf{m} \right) \mathbf{u}. \tag{11}$$

This computation is mathematically equivalent to the original form and obtains the score correction directly from the product of the attention weights and $\mathbf{u}$, without explicitly constructing the C-dimensional context feature $\mathbf{r}_i$.

At inference time, we cache the text features, the neck-side $\mathbf{K}$ and $\mathbf{V}$, and the head-side coefficients $\mathbf{A}$, $\mathbf{c}$, and $\mathbf{u}$ for each referring expression. We also cache the normalized 2D positional embeddings for each resolution since they depend on neither the image nor the expression. Caching removes the text encoder, text-side projections, and positional-embedding MLP from per-frame inference.

4. Experiments

4.1. Experimental setup

Datasets and metrics. For single-image REC, we use RefCOCO [14], RefCOCO+ [35], and RefCOCOg [23], which provide referring expressions for images from MS COCO [16]. RefCOCO+ restricts the use of spatial words and therefore emphasizes the ability to distinguish candidates by appearance attributes. Expressions in RefCOCOg contain approximately 8.4 words on average and frequently combine attributes, locations, and inter-object relations. We report Acc@0.5, which counts a prediction as correct when its intersection over union (IoU) with the ground-truth box is at least 0.5. For streaming evaluation, we use all 700 test sequences from the TNL2K benchmark [32] under the tracking-by-natural-language-only setting. This setting specifies the target from language without providing the ground-truth box in the first frame for initialization. We report Success AUC, Precision (Prec), which measures the fraction of frames with a center-location error below 20 pixels, and Normalized Precision (N Prec), which normalizes the center-location error by the ground-truth box size. We measure inference efficiency on one NVIDIA L40S GPU with a batch size of one and report average end-to-end latency per image and FPS.

Latency measurement protocol. End-to-end latency includes image loading and preprocessing, the forward pass, and non-maximum suppression (NMS). *Stream* denotes inference that precomputes quantities depending only on the referring expression and reuses them for every frame, whereas *Single* denotes single-image inference. We apply Stream inference to MCN [21], TransVG [8], SeqTR [38], Dynamic MDETR [30], and EEVG [5] since their published architectures have a decoupled text branch (Sec. 2.1). Additional details on the latency measurement protocol are provided in Appendix A.

Implementation details. We implement our method with MMYOLO [24] and MMDetection [4]. Starting from pretrained YOLO-World-v2-X weights, we fine-tune Ref-YOLO-World on the combined training sets of RefCOCO, RefCOCO+, and RefCOCOg, excluding all images in their val/test splits [13]. The input resolution is 640×640. We train for 20 epochs with AdamW [19], FP16 mixed precision, a base learning rate of 2.0×10^{-4}, and a total batch size of 32. During horizontal-flip augmentation, we also replace directional words in each expression to keep the text consistent with the flipped image.

4.2. Comparison on RefCOCO, RefCOCO+, and RefCOCOg

Table 1 compares the methods on the RefCOCO benchmarks. Among REC models with a decoupled text branch, Ref-YOLO-World obtains the highest Acc@0.5 on four of the eight evaluation splits and the second-highest value on the remaining four. The largest improvements occur on RefCOCOg, which contains many long expressions combining attributes, locations, and inter-object relations. This pattern suggests that per-token features are particularly useful when

Method	FT	FPS (Stream / Single)	Params	Res.	RefCOCO			RefCOCO+			RefCOCOg	
					val	testA	testB	val	testA	testB	val	test
Decoupled text branch												
MCN DarkNet53 [21]	✓	68.9 / 44.9	81.8M	416	80.08	82.29	74.98	67.16	72.86	57.31	66.46	66.01
TransVG [8]	✓	16.0 / 11.1	168M	640	81.02	82.72	78.35	64.82	70.70	56.94	68.67	67.73
SeqTR [38]	✓	44.1 / 18.8	108M	640	81.23	85.00	76.08	68.82	75.37	58.78	71.35	71.58
SimREC CSPDarkNet53 [20]	✓	N/A	44.0M	416	82.45	85.91	77.98	70.58	76.75	61.12	72.59	72.86
Dynamic MDETR R-50 [30]	✓	47.8 / 31.0	155M	640	80.47	82.63	75.96	65.52	70.41	57.68	69.20	68.56
EEVG ViT-B [5]	✓	37.6 / 10.7	218M	448	**88.08**	90.33	**85.50**	**77.97**	82.44	**69.15**	79.60	80.24
YOLO-World-v2-X [6]	✗	56.3 / 45.4	136M	640	39.03	43.13	35.31	36.93	39.45	35.47	40.22	40.57
YOLO-World-v2-X [6]	✓	56.0 / 45.0	136M	640	63.56	70.73	55.15	65.79	73.02	56.35	71.49	71.86
Ref-YOLO-World (Ours)	✓	47.0 / 38.3	140M	640	88.04	**90.70**	83.38	76.36	**82.82**	67.52	**81.29**	**81.56**
Text recomputed for every frame												
SimVG-TB ViT-L/32 [7]	✓	– / 28.2	682M	640	92.99	94.86	90.12	87.43	91.02	82.10	87.95	88.96
Florence-2-base [33]	✓	– / 9.1	231M	768	92.6	94.8	91.5	86.8	91.7	82.2	89.8	82.2
Grounding DINO-T [17]	✓	– / 12.4	172M	800*	89.19	91.86	85.99	81.09	87.40	74.71	84.15	84.94
Qwen3-VL 2B [2, 11]	✗	– / 1.5	2B	Dynamic	88.2	91.0	83.1	78.6	85.2	70.4	84.7	85.0
WeDetect-Ref 2B [11]	✓	– / 3.4	2.4B	Dynamic	94.3	95.6	92.6	88.1	92.6	83.1	92.0	92.2

Table 1. Acc@0.5 and inference efficiency on the RefCOCO benchmarks. Acc@0.5 values are from the cited papers; FPS is measured on RefCOCOg val (Sec. 4.1). Bold and underlined values mark the best and second-best accuracies among REC models with a decoupled text branch. FT denotes fine-tuning on RefCOCO, – indicates that Stream inference is inapplicable, and N/A indicates no official checkpoint. Models marked with * use an 800-pixel short side and a long side of at most 1,333 pixels, while Dynamic refers to variable input resolution.

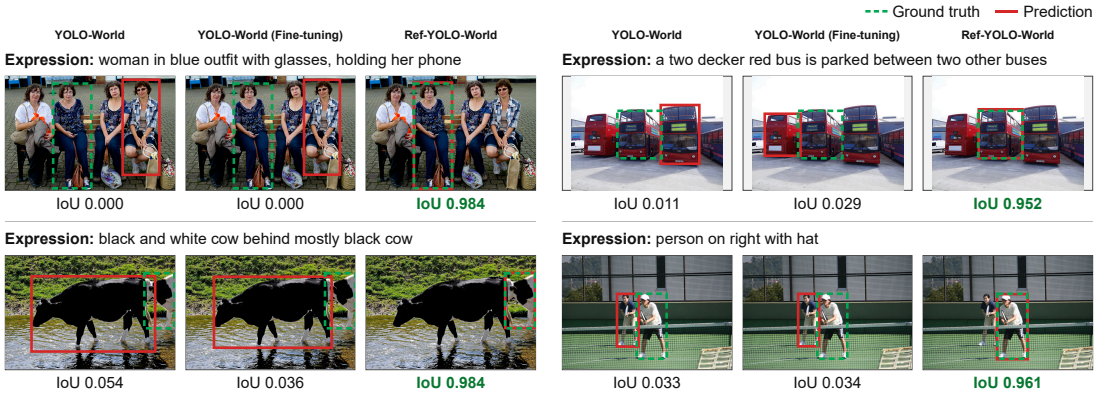


Figure 5. Qualitative comparison on RefCOCOg val. Green dashed boxes indicate ground truth, and red solid boxes indicate predictions.

an expression combines multiple visual cues. Ref-YOLO-World reaches 81.29% on RefCOCOg val and 81.56% on its test split, exceeding EEVG by 1.69 and 1.32 points, respectively. Its 38.3 FPS under Single inference is approximately 3.6× the 10.7 FPS of EEVG. Under Stream inference, it runs at 47.0 FPS, compared with 37.6 FPS for EEVG. Ref-YOLO-World is within 0.04 points of EEVG on RefCOCO val. It trails EEVG on the three splits that emphasize non-person targets or appearance descriptions: RefCOCO testB and RefCOCO+ val/testB. Compared with the fine-tuned YOLO-World-v2-X baseline, Ref-YOLO-World improves all eight splits, including a 9.80-point gain on RefCOCOg val. This gain costs 3.4 ms of per-image latency under Stream inference (47.0 vs. 56.0 FPS) due to the cross-attention added to the neck and head.

Several methods based on multimodal fusion architectures or MLLMs achieve higher Acc@0.5 than Ref-YOLO-World. SimVG-TB reaches 28.2 FPS under Single inference. Since it fuses image and text features inside the encoder, it repeats the same fusion computation for every image even when the expression is fixed. In contrast, Ref-YOLO-World reaches 38.3 FPS under the same Single inference, and reusing its text-side computation raises this to 47.0 FPS under Stream inference.

Figure 5 shows examples with multiple candidates from the same category that require appearance attributes, inter-

Method	Params	CoT	Acc@IoU		
			0.5	0.75	0.9
Qwen3-VL-2B [2, 9]	2B	✗	23.5	<u>19.2</u>	11.0
Qwen3-VL-2B [2, 9]	2B	✓	<u>25.2</u>	**20.6**	<u>11.4</u>
Qwen2.5-VL-3B [3, 9]	3B	✗	23.8	18.1	8.8
Qwen2.5-VL-3B [3, 9]	3B	✓	**25.3**	19.1	9.5
YOLO-World-v2-X [6]	136M	–	19.7	16.3	10.8
Ref-YOLO-World (Ours)	140M	–	23.6	<u>19.2</u>	**12.1**

Table 2. Zero-shot evaluation on Ref-Adv-s [9]. Values for MLLMs are reported for their Instruct variants. CoT indicates the use of a Chain-of-Thought prompt, and – indicates that CoT is not applicable to the detector. Bold and underlined values are the best and second-best values in each column.

Method	TNL2K Train	TNL2K Temp.	AUC	Prec	N Prec
TNL2K-I [32]	✓	✓	11.0	6.0	11.0
JointNLT [37]	✓	✓	54.6	55.0	70.6
QueryNLT [29]	✓	✓	53.3	53.0	70.4
UVLTrack-B [22]	✓	✓	55.7	57.2	–
MambaVLT [18]	✓	✓	**58.4**	58.9	**80.9**
MVLM [26]	✓	✓	57.8	60.9	–
Ref-YOLO-World (Ours)	✗	✗	32.8	29.2	34.7
Ref-YOLO-World (Ours)	✓	✗	56.4	**62.3**	61.4

Table 3. Comparison on the TNL2K test set under the natural-language-only setting. TNL2K Train and Temp. indicate use of the TNL2K training split and prior-frame information, respectively. The best value in each column is in bold. Comparison results are from the cited papers; – indicates unreported results.

object relations, or spatial words. YOLO-World and its fine-tuned variant select an incorrect candidate in these images. Ref-YOLO-World uses the relevant attributes, relations, and spatial words to localize the ground-truth region.

4.3. Zero-shot evaluation on Ref-Adv-s

Practical systems must distinguish a target from similar objects of the same category when given expressions that differ from those seen during training. We evaluate zero-shot generalization by applying Ref-YOLO-World trained on the RefCOCO benchmarks to Ref-Adv-s [9] without additional training. Ref-Adv-s is the public 1,142-example subset of Ref-Adv, with images from COCO val2017 and OpenImages. Its expressions contain 11.5 words on average, and each image contains an average of 4.0 distractors that resemble the target.

Table 2 reports the results. Ref-YOLO-World achieves 23.6% Acc@0.5, comparable to the 2–3B-parameter Qwen-VL models in the table. It also outperforms YOLO-World-v2-X by 3.9, 2.9, and 1.3 points at IoU thresholds of 0.5, 0.75, and 0.9, respectively. At IoU 0.9, Ref-YOLO-World achieves the highest accuracy in the table, 12.1%. The gains across all three thresholds indicate that per-token alignment also transfers to the expressions in Ref-Adv.

4.4. Streaming evaluation on TNL2K

Under TNL2K's natural-language-only setting, we evaluate target localization and precomputation reuse for a fixed expression on consecutive frames, using RefCOCO-trained weights both zero-shot and after fine-tuning on TNL2K. Both conditions process frames independently without temporal state and compute and cache text features and attention quantities once per sequence for reuse across all frames. Since all vision-language trackers used for comparison [18, 22, 26, 29, 32, 37] use prior-frame information, this experiment evaluates how stateless, frame-wise Ref-YOLO-World transfers to video sequences.

Table 3 presents the results. After fine-tuning, Ref-YOLO-World reaches 56.4 AUC, outperforming the temporal trackers JointNLT, QueryNLT, and UVLTrack-B. It remains within 2.0 points of MambaVLT at 58.4 and MVLM at 57.8. Its Precision of 62.3 is the highest among the compared methods. The zero-shot model, trained only on still images, operates without additional training and reaches 32.8 AUC.

Normalized Precision remains 61.4, which is 19.5 points below MambaVLT at 80.9. Since this metric normalizes center error by the ground-truth box size, the result indicates larger localization errors relative to target size. The bottom row of Fig. 6 shows a failure in which Ref-YOLO-World switches to another instance when several candidates from the same category are present. Once a temporal tracker identifies the target in the first frame, it can follow appearance continuity in later frames. Ref-YOLO-World instead re-identifies the target from the expression in every frame. It may therefore select another candidate when the expression describes an action or state that changes over time and does not uniquely identify the target in a single frame. Reusing the precomputed quantities also increases end-to-end inference speed on the TNL2K test set from 26.8 to 30.4 FPS. The per-frame saving of 4.5 ms is consistent with the 4.8 ms on RefCOCOg single images (Sec. 4.6), since the same text-side computation is reused over all frames of a sequence.

4.5. Ablation study

We ablate the modules that use all token features by fine-tuning each configuration only on RefCOCOg, whose long expressions frequently combine attributes, locations, and inter-object relations (Tab. 4). Adding Token-Aware Attention (Sec. 3.2) alone improves the baseline by 2.29 points, showing that per-token features in the neck account for the main gain. Adding the positional embedding or Ref-Text Contrastive Head to Token-Aware Attention yields 70.66% and 70.50%, respectively, within 0.7 points of the 71.16% obtained by Token-Aware Attention alone. Adding both



Figure 6. Predictions from Ref-YOLO-World on consecutive TNL2K frames under the natural-language-only setting. # denotes the frame number, and the quoted text is the referring expression. The top row shows successful continuous localization. In the bottom row, the model switches to another instance after multiple candidates from the same category appear.

Configuration	TAA	PE	Head	Acc@0.5
YOLO-World-v2-X FT	✗	✗	✗	68.87
+ TAA	✓	✗	✗	71.16
+ TAA + PE	✓	✓	✗	70.66
+ TAA + Head	✓	✗	✓	70.50
Ref-YOLO-World	✓	✓	✓	**72.06**

Table 4. Ablation on RefCOCOg val after fine-tuning each configuration only on RefCOCOg. TAA, PE, and Head denote Token-Aware Attention, the neck's normalized 2D positional embedding, and Ref-Text Contrastive Head, respectively.

raises accuracy to 72.06%. We attribute this to an interaction between the two components. The positional embedding adds explicit positional information to the neck features, and Ref-Text Contrastive Head, which scores each region against all tokens rather than a single aggregated text feature, is better suited to exploit this information. Consistent with this, setting λ to zero at inference to keep only the base score reduces the accuracy of Ref-YOLO-World from 72.06% to 12.9% on RefCOCOg val, whereas the same operation reduces the accuracy of the model trained without the positional embedding by only 0.9 points. This suggests that once the neck features carry explicit positional information, the base score alone is no longer sufficient to rank candidate regions.

4.6. Inference efficiency

We measure the effect of precomputing and reusing expression-only quantities on RefCOCOg val. Single inference, which includes text processing, runs at 38.3 FPS. Stream inference reuses the text features and precomputed attention quantities and reaches 47.0 FPS (Tab. 1), reducing per-image latency by 18.5%. The precomputed formulation is mathematically equivalent to per-image computation and produces the same predictions.

Device	Compute Unit	Time (ms)	FPS	Acc@0.5
iPhone 15 Pro (A17 Pro)	CPU/GPU	242.51	4.12	80.4
	CPU/ANE	**63.59**	**15.73**	80.8
iPad Pro (M4)	CPU/GPU	91.42	10.94	81.2
	CPU/ANE	**44.56**	**22.44**	80.8

Table 5. On-device inference performance. All configurations use the same 500 examples from RefCOCOg val, a batch size of one, and FP16 internal computation.

4.7. On-device deployment

To assess edge deployment, we convert the final per-frame detector to Core ML and feed precomputed text features to physical iOS devices (Tab. 5). Including preprocessing, CPU/ANE with FP16 is fastest on both devices: the iPhone 15 Pro processes an image in 63.59 ms at 15.73 FPS, and the iPad Pro in 44.56 ms at 22.44 FPS. Acc@0.5 remains near 80% across compute units without consistent accuracy loss. Together with precomputation, all per-frame computation runs locally on camera-equipped devices, enabling continuous REC without sending captured images to an external server.

5. Conclusion

We proposed Ref-YOLO-World, whose text representations remain independent of image features and can be reused across frames. Ref-YOLO-World passes per-token text features through the detection path under this design requirement and achieves the highest accuracy on RefCOCOg among REC models with a decoupled text branch. Reusing precomputed quantities reduces inference time without changing accuracy, and the Core ML model runs at 15.73 FPS on an iPhone 15 Pro. An accuracy gap remains relative to methods based on multimodal fusion architectures or MLLMs. Ref-YOLO-World processes each frame independently and may switch to another instance of the

same category matching the expression (Sec. 4.4). In exchange, this stateless design requires no first-frame bounding box and avoids error accumulation across frames. This decoupled design provides a foundation for continuously running REC in interactive mobile camera applications.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Number JP26K21939 and by a research grant from the Keio SFC Academic Society.

References

[1] Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. ReZero is all you need: Fast convergence at large depth. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 1352–1361, 2021.

[2] Shuai Bai et al. Qwen3-VL technical report. arXiv preprint arXiv:2511.21631, 2025.

[3] Shuai Bai et al. Qwen2.5-VL technical report. arXiv preprint arXiv:2502.13923, 2025.

[4] Kai Chen et al. MMDetection: Open MMLab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155, 2019.

[5] Wei Chen, Long Chen, and Yu Wu. An efficient and effective transformer decoder-based framework for multi-task visual grounding. In *European Conference on Computer Vision*, pages 125–141, 2024.

[6] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. YOLO-World: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16901–16911, 2024.

[7] Ming Dai, Lingfeng Yang, Yihao Xu, Zhenhua Feng, and Wankou Yang. SimVG: A simple framework for visual grounding with decoupled multi-modal fusion. In *Advances in Neural Information Processing Systems*, pages 121670–121698, 2024.

[8] Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. TransVG: End-to-end visual grounding with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1769–1779, 2021.

[9] Qihua Dong et al. Ref-Adv: Exploring MLLM visual reasoning in referring expression tasks. In *International Conference on Learning Representations*, 2026.

[10] Xiaokun Feng et al. MemVLT: Vision-language tracking with adaptive memory-based prompts. In *Advances in Neural Information Processing Systems*, pages 14903–14933, 2024.

[11] Shenghao Fu, Yukun Su, Fengyun Rao, Jing Lyu, Xiaohua Xie, and Wei-Shi Zheng. WeDetect: Fast open-vocabulary object detection as retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20377–20387, 2026.

[12] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8. https://github.com/ultralytics/ultralytics, 2023.

[13] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. MDETR – modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790, 2021.

[14] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 787–798, 2014.

[15] Liunian Harold Li et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022.

[16] Tsung-Yi Lin et al. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755, 2014.

[17] Shilong Liu et al. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55, 2024.

[18] Xinqi Liu, Li Zhou, Zikun Zhou, Jianqiu Chen, and Zhenyu He. MambaVLT: Time-evolving multimodal state space model for vision-language tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8731–8741, 2025.

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[20] Gen Luo, Yiyi Zhou, Jiamu Sun, Xiaoshuai Sun, and Rongrong Ji. A survivor in the era of large-scale pretraining: An empirical study of one-stage referring expression comprehension. *IEEE Transactions on Multimedia*, 26:3689–3700, 2024.

[21] Gen Luo et al. Multi-task collaborative network for joint referring expression comprehension and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10031–10040, 2020.

[22] Yinchao Ma, Yuyang Tang, Wenfei Yang, Tianzhu Zhang, Jinpeng Zhang, and Mengxue Kang. Unifying visual and vision-language tracking via contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4107–4116, 2024.

[23] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11–20, 2016.

[24] MMYOLO Contributors. MMYOLO: OpenMMLab YOLO series toolbox and benchmark. https://github.com/open-mmlab/mmyolo, 2022.

[25] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Modeling context between objects for referring expression understanding. In *European Conference on Computer Vision*, pages 792–807, 2016.

[26] Dae-Hyeon Park, Mina Baek, Jeong-Hun Ha, Chan-Seop Park, Jamshidjon Ganiev, and Seung-Hwan Bae. MVLM: Template-free tracking via vision-language margin confidence and memory-gated tracking. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 35156–35165, 2026.

[27] Alec Radford et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.

[28] Tianhe Ren et al. Grounding DINO 1.5: Advance the "edge" of open-set object detection. arXiv preprint arXiv:2405.10300, 2024.

[29] Yanyan Shao, Shuting He, Qi Ye, Yuchao Feng, Wenhan Luo, and Jiming Chen. Context-aware integration of language and visual references for natural language tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19208–19217, 2024.

[30] Fengyuan Shi, Ruopeng Gao, Weilin Huang, and Limin Wang. Dynamic MDETR: A dynamic multimodal transformer decoder for visual grounding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):1181–1198, 2024.

[31] Ao Wang, Lihao Liu, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. YOLOE: Real-time seeing anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 24591–24602, 2025.

[32] Xiao Wang et al. Towards more flexible and accurate object tracking with natural language: Algorithms and benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13763–13773, 2021.

[33] Bin Xiao et al. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4818–4829, 2024.

[34] Yu Yan et al. EL-Attention: Memory efficient lossless attention for generation. In *Proceedings of the 38th International Conference on Machine Learning*, pages 11648–11658, 2021.

[35] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85, 2016.

[36] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[37] Li Zhou, Zikun Zhou, Kaige Mao, and Zhenyu He. Joint visual grounding and tracking with natural language specification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23151–23160, 2023.

[38] Chaoyang Zhu et al. SeqTR: A simple yet universal network for visual grounding. In *European Conference on Computer Vision*, pages 598–615, 2022.

Supplementary Material

A. Latency Measurement Protocol

We measured all Stream and Single FPS values reported in the main paper on one NVIDIA L40S GPU with a batch size of one. Model construction and checkpoint loading occur before timing. We synchronize CUDA execution before reading each timestamp, so every interval includes all kernels launched within it. For each pass over the evaluation split, let N be the number of evaluated samples and T be their total timed inference duration in milliseconds. We compute

$$t_{\text{img}} = \frac{T}{N}, \qquad \text{FPS} = \frac{1000}{t_{\text{img}}}. \tag{12}$$

Each value in the main paper is the mean FPS over three passes. For TNL2K (Sec. 4.4), we apply the same protocol to all 700 test sequences in a single pass; PNG decoding of each frame is included in the timed interval. Four frames of one sequence with corrupt files are excluded.

Single inference follows the original evaluation path of each method and covers image loading, preprocessing, the model forward pass, and post-processing; the text encoder runs exactly once per image. Stream inference replaces the expression-only computation with cached tensors and does not run the text encoder. For each method, we implemented the Stream path on its official codebase (Appendix B).

MCN, TransVG, Dynamic MDETR, and EEVG use all 4,896 expressions in the RefCOCOg [23, 25] UMD validation split. SeqTR uses all 9,536 expressions in the Google validation split to match its official checkpoint and evaluation configuration.

B. Stream Paths for Comparison Methods

Table 6 summarizes where each implementation separates expression-only computation from the per-image path. For MCN, we cache the BiGRU hidden states and the text feature produced by self-guided attention. For TransVG, Dynamic MDETR, and EEVG, we run BERT once per expression and cache its token features. SeqTR caches the word features produced by its one-layer bidirectional GRU. The image encoder and every operation that combines image and text features remain in the timed per-image path.

Grounding DINO does not satisfy our Stream definition since its feature enhancer uses bidirectional fusion to recompute image-conditioned text representations for each image; in our measurements, caching only its BERT outputs preserved Acc@0.5 and yielded only a 1.04× end-to-end speedup, from 12.38 to 12.86 FPS.

We report the FPS of SimREC [20] as N/A because the corresponding official checkpoint is unavailable.

C. Cache Construction

We measured cache construction three times using the same GPU, batch size, and CUDA synchronization protocol as the FPS measurements, and report the mean. Timing begins after model construction and checkpoint loading. Each pass builds and retains the cache required for every expression in the listed split; expressions with the same cache key share one entry. Cache construction precedes timed inference and is excluded from the reported FPS. Stream FPS therefore measures the steady-state per-frame cost after construction, matching the target setting in which one expression is applied to consecutive frames.

YOLO-World [6] also provides an offline reparameterization path that embeds a fixed vocabulary into its neck and detection head. Applying this path to the complete RefCOCOg validation split would treat all referring expressions as simultaneous detection classes, whereas REC evaluates one expression at a time. We therefore retain the query-specific inference path and cache the CLIP text feature for each expression.

Ref-YOLO-World caches the pooled and per-token text features, the neck-side key and value projections, and the head-side coefficients $\mathbf{A}$, $\mathbf{c}$, and $\mathbf{u}$. Per-frame Stream latency begins after these quantities have been constructed once.

D. On-Device Evaluation

We evaluated the Core ML model on an iPhone 15 Pro with an A17 Pro and an iPad Pro with an M4. Both devices process the same 500 examples, corresponding to the 21st through 520th entries in the RefCOCOg UMD validation annotation order, with a batch size of one and FP16 internal computation. We run one warm-up inference before the 500 measured examples. The Core ML compute-unit settings are CPU/GPU and CPU/ANE.

The latency reported in the main paper is

$$T_{\text{device}} = T_{\text{pre}} + T_{\text{input}} + T_{\text{query}} + T_{\text{CoreML}}, \tag{13}$$

where image preprocessing includes resizing and normalization, Core ML input construction converts the preprocessed image into the model input, and query-state construction loads the precomputed expression tensors into the model state. The timer starts from an image already loaded in memory. File decoding, model loading, warm-up, text encoding, and the subsequent Acc@0.5 calculation run outside this interval. Text encoding is performed once per expression and is excluded from the per-frame latency.

Method	RefCOCOg split	Cached quantities	Build time (s)
MCN [21]	UMD	BiGRU hidden states and self-guided attention output	43.24
TransVG [8]	UMD	BERT token features	29.17
SeqTR [38]	Google	Bidirectional GRU word features	205.16
Dynamic MDETR [30]	UMD	BERT token features	29.29
EEVG [5]	UMD	BERT token features	27.66
YOLO-World-v2-X (pretrained) [6]	UMD	CLIP text features	19.52
YOLO-World-v2-X (fine-tuned) [6]	UMD	CLIP text features	19.36
Ref-YOLO-World (Ours)	UMD	Pooled/per-token CLIP features and attention-derived states	40.13

Table 6. Cached quantities used for Stream inference. Build time is the mean total time over three passes for the complete evaluation split and is reported separately from FPS.